On the Status of Word Embeddings as Implementations of the Distributional Hypothesis

Timothee Mickus

IAEM Doctoral School
ATILF UMR 7118 (CNRS-UL)

On the Status of Word Embeddings as Implementations of the Distributional Hypothesis

A dissertation submitted to the
Université de Lorraine

by Timothee Mickus

in partial fulfillment of the requirements for the degree of
Docteur en Informatique de l'Université de Lorraine

31/03/2022

Jury Members

Supervisor	Mathieu Constant, Université de Lorraine
Co-supervisor	Denis Paperno, Universiteit Utrecht
Reviewers	Benoit Crabbé, Université de Paris, *Jury President*
	Nabil Hathout, CNRS / Université de Toulouse Jean Jaurès
Examiners	Gemma Boleda, Universitat Pompeu Fabra
	Vera Demberg, Universität des Saarlandes
	Claire Gardent, CNRS / Université de Lorraine
	Alessandro Lenci, Università di Pisa
Guest member	Kees Van Deemter, Universiteit Utrecht

Acknowledgments

So pour the beer for thirsty men
A drink that they have earned
And pour drink for those who fell
For those who did not return

— Amon Amarth, *Raise your horns*

This work would not have been possible without the support of my two supervisors, Mathieu Constant and Denis Paperno. Their thorough feedback throughout my PhD significantly bettered the present dissertation. Thanks for gladly following along in this mad ride through fish puns and squirrel drawings.

I would also like to thank the jury members who attended my defense: Benoit Crabbé, Nabil Hathout, Gemma Boleda, Vera Demberg, Claire Gardent and Alessandro Lenci. I am extremely grateful that you took the time to weigh in on my work.

My thanks also go to Kees van Deemter, whose guidance and input I could not have done without. Likewise, many thanks to Timothée Bernard, Takamura-san and all the NLP team at AIST-AIRC for the internship opportunity: I'm sure this is the start of a fruitful collaboration. I also wish to thank my previous teachers

at Paris Diderot in general and Olivier Bonami in particular, without whom I would certainly not be writing this today.

A good deal many people also helped me go through the grind of a three year research program: in no particular order, I'd like to thank a wrangler of beavers, a connoisseuse of plastic chicken music, four rabbits and a cat, mi hermano bandido, a cinephile commie turned historian and a translator of lame duck jokes, the very limited clientele of my non-existent T-shirt business, a man who lost his farm to boars, friends and family who went to Beaubourg with me, colleagues at ATILF and Synalp who listened to my linear algebraic rants, and many more.

Lastly, this work was supported by a public grant overseen by the French National Research Agency (ANR) as part of the "Investissements d'Avenir" program: Idex *Lorraine Université d'Excellence* (reference: ANR-15-IDEX-0004).

Table of Contents

Acknowledgments iii

Table of Contents vii

List of Acronyms xv

List of Equations xix

List of Figures xxiii

List of Tables xxvii

Abstract xxxi

Résumés en français xxxv

Introduction 3

I A Tale of Two Theories 9

1 Distributional Semantics 11

1.1 Chronological Overview 12
1.1.1 Inception 13
1.1.2 From a methodology to a lexical semantic theory 14
1.1.3 Co-occurrence count matrices 16
1.1.4 The birth and rise of neural word embeddings 20
1.2 Major examples of DSM architectures 25
1.2.1 The word2vec model 25
1.2.2 The BERT model 32
1.2.3 The ELECTRA model 40
1.2.4 Comparing the three architectures 43
1.3 Heterogeneity of DSM Evaluation Protocols 44
1.3.1 Vector Space Structure 46
1.3.2 Classifier Probes 52
1.3.3 Attention Head Analysis 54
1.4 Vector Size and Concentration 55
1.5 Conclusions 62

2 Dictionaries in NLP 67

2.1 What are dictionaries 69
2.1.1 Terminology 69
2.1.2 Defining the concept of a dictionary 70
2.1.3 Dictionaries as semantic theories 73
2.2 Dictionaries and Semantic Grounding 75
2.2.1 What is semantic grounding? 75
2.2.2 What dictionaries show of NLP systems and grounding . 78

2.3 Dictionaries as NLP meaning inventories 82
2.4 Definition Modeling and Reverse Dictionary 85
2.4.1 Reverse Dictionary 86
2.4.2 Definition Modeling 88
2.5 Conclusions 92

II Distributional Semantics vs. Dictionaries 95

3 Topographic Similarity 97
3.1 Topographic Similarity 99
3.1.1 Measuring topographic similarity with Mantel tests . . . 100
3.1.2 Relevant Applications 103
3.2 Topographic similarity and Artificial Languages 105
3.2.1 Methodology 105
3.2.2 Results 110
3.2.3 Discussion & Conclusions 112
3.3 Topographic similarity and Sentence Encoders 114
3.3.1 Methodology 114
3.3.2 Results 117
3.3.3 Discussion & Conclusions 118
3.4 Topographic similarity and definitions 119
3.4.1 Dataset 120
3.4.2 Methodology 129
3.4.3 Results 130
3.4.4 Discussion 133

3.5 Replication study on definitions 134
3.5.1 Methodology 135
3.5.2 Results 138
3.5.3 Discussion 140
3.6 Conclusions 143

4 Inverse functions **147**
4.1 Aligning word definitions and word embeddings 149
4.2 Definition Modeling and Examples of Usage 151
4.2.1 Formalization 154
4.2.2 Experimental Protocol 159
4.2.3 Results 163
4.2.4 Qualitative Analysis 166
4.2.5 Conclusions 170
4.3 Can Definition Modeling discriminate DSMs? 170
4.3.1 Dataset 171
4.3.2 Embeddings and Definition Modeling system 172
4.3.3 Results 175
4.3.4 Manual analysis 178
4.3.5 Conclusions 182
4.4 The CoDWoE Shared task 183
4.4.1 Metrics 185
4.4.2 Baseline Architectures 189
4.4.3 Results 191
4.5 Conclusions 203

III Word Embeddings are not Distributional Semantics Models 207

5 Limits of the distributional hypothesis 209

5.1 Distributional substitution 210

5.2 Pilot Study 218

5.2.1 Word type vs. word sense judgments 219

5.2.2 Word Pairs Selection 221

5.2.3 Results 227

5.2.4 Conclusions of the pilot study 229

5.3 Implementing an interface 231

5.3.1 Dataset Construction 233

5.3.2 Player Engagement 235

5.3.3 Implementation details 237

5.4 Analyzing the collected data 240

5.4.1 Contents Overview 240

5.4.2 Success rates 243

5.4.3 Comparing human and model behaviors 246

5.4.4 Manipulating the distributional hypothesis 252

5.5 Conclusions 260

6 The Structure of Transformer Embedding Spaces 265

6.1 Is BERT a vector-space model of meaning? 267

6.1.1 Word type cohesion 267

6.1.2 Cross-sentence coherence 273

6.1.3 Sentence-level structure 278

6.2 The Linear Structure of Transformers 283
6.2.1 Mathematical re-framing 283
6.2.2 Step-by-step derivation of Equation (6.8) 287
6.3 Intrinsic analyses 292
6.3.1 Visualizing the contents of embeddings 292
6.3.2 Quantifying non-linearity 297
6.4 Extrinsic analyses 299
6.4.1 The MLM objective 299
6.4.2 Lexical contents & WSD 302
6.4.3 Effects of finetuning & NER 305
6.5 Conclusions 310

Conclusions 315

Bibliography 325

IV Appendices 361

A Bayesian Optimization 363

B Analogy Dataset translated from BATS 371

C Instructions Provided to Annotators 379

C.1 BlankCrack Pilot Study, Adversarial Word Pair Submissions . . . 379
C.2 BlankCrack Pilot Study, On-Screen Instructions 381
C.3 BlankCrack Online Game, On-Screen Instructions 382

C.3.1 (a)-annotation instructions 382
C.3.2 (b)-annotation instructions 382
C.4 BlankCrack Online Game, Instructions for Re-Annotation of BERT-Selected Contexts 382

D Illustrations, pictures, visual supports 385

List of Acronyms

BART	Bidirectional and Auto-Regressive Transformers
BATS	Balanced Analogy Test Set
BERT	Bidirectional Encoder Representations from Transformers
BLEU	BiLingual Evaluation Understudy
BPE	Byte Pair Encoding
CBOW	Continuous Bag Of Words
CoDWoE	Comparing Dictionaries and Word Embeddings
CONLL	Computational Natural Language Learning
DSM	Distributional Semantics Models
ELECTRA	Efficiently Learning an Encoder that Classifies Token Replacements Accurately
ELMo	Embeddings from Language Modeling
EuroParl	European Parliament
GAN	Generative Adversarial Networks
GCIDE	GNU Collaborative International Dictionary of English
GELU	Gaussian Error Linear Unit
GLAWI	GLÀFF & Wiktionnaire
GloVe	Global Vectors

GPT	Generative pre-trained Transformer
HAL	Hyperspace Analogue to Language
idf	Inverted Document Frequency
iid	independent and identically distributed
IR	Information Retrieval
KNN	k Nearest Neighbors
LayerNorm	Layer Normalization
LDA	Latent Dirichlet Allocation
LSA	Latent Semantic Analysis
LSTM	Long Short-Term Memory units
MCC	Matthews Correlation Coefficient
MEN	Marco, Elia and Nam
METEOR	Metric for Evaluation of Translation with Explicit Ordering
MFS	Most Frequent Sense
MHA	Multi-head attention
MLM	Masked Language Model
MLP	Multi-layer perceptron
MSA	Mean squared Error
NER	Named Entity Recognition
NLG	Natural Language Generation
NLI	Natural Language Inference
NLP	Natural Language Processing
NSP	Next Sentence Prediction
OED	Oxford English Dictionary

PCA	Principal Component Analysis
PMI	Pointwise Mutual Information
POS	Part of Speech
PPMI	Positive Pointwise Mutual Information
PPL	Perplexity
QA	Question Answering
ReLU	Rectified Linear Unit
RDF	Resource Description Framework
ROUGE	Recall-Oriented Understudy for Gisting Evaluation
SGNS	Skip-gram with Negative Sample
SICK	Sentences Involving Compositional Knowledge
SQL	Structured Query Language
STS	Semantic Text Similarity
SVD	Singular Value Decomposition
TED	Tree Edit Distance
tf	Term Frequency
tf.idf	Term Frequency–Inverted Document Frequency
UML	Unified Modeling Language
USE	Universal Sentence Encoder
VQA	Visual Question Answering
WN	Wordnet
WNUT	Workshop of Noisy User-generated Text
WSD	Word Sense Disambiguation
XML	Extended Markup Language

List of Equations

1.2 Term frequency–inverse document frequency (tf.idf) 18

1.3 Pointwise mutual information 19

1.4 Positive pointwise mutual information 19

1.5 One-hot vectors 27

1.6 Softmax function 28

1.7 Negative log-likelihood loss 28

1.8 Skip-gram architecture 29

1.9 Joint negative log-likelihood objective 29

1.10 Sigmoid function 31

1.11 Negative sampling in word2vec 31

1.12 Hierarchical softmax in word2vec 31

1.13 Subsampling rate in word2vec 32

1.14 Feed-forward sub-modules in Transformers 32

1.16 Multi-head attention mechanisms in Transformers 33

1.17 Residual connections & LayerNorms in Transformers 35

1.18 Positional encodings in Transformers 35

1.19 GAN objective 40

1.20 Equivalence between word2vec and count matrices 47

1.21 Euclidean norm 56
1.22 Euclidean distance 57
1.23 Cosine similarity 57
1.24 Scalar product 57
1.25 χ-distribution 57
1.26 Expected value and variance of Euclidean norm (χ-distribution) as a function of dimension 57
1.27 Distribution of Euclidean distance as a χ-distribution 58
1.28 Expected value and variance of Euclidean distance as a function of dimension 58
1.29 Variance of cosine as a function of dimension 60

3.1 Distance matrices 100
3.2 Distance matrices as parallel sequences of observations 101
3.3 Pearson Correlation Coefficient (Pearson's r) 101
3.4 Ranking function 102
3.5 Spearman Correlation Coefficient (Spearman's ρ) 103
3.6 Topographic Similarity 103
3.7 Hamming distance 109
3.8 Levenshtein distance 109
3.9 Normalized Levenshtein distance 109
3.10 t-statistic 124
3.11 t-distribution 124
3.12 Jaccard index 129

4.1 Marking strategy to highlight definienda in context 155

4.3 Additive and multiplicative marking 155
4.5 Post-encoder and pre-encoder marking 155
4.6 Scalar multiplication based marking 156
4.7 Vector addition based marking 158
4.8 Minimal pairwise edit distance in GLAWI 179
4.9 Ranking of reconstructed embedding 186

5.1 Formal definition of distributional substitutability 213
5.3 Distributional substitution for skip-gram type models 214
5.4 Distributional substitution for negative sampling models 215
5.5 Distributional substitution for generative models 216
5.6 Distributional substitution using log-probability evidence 244
5.7 Matthews Correlation Coefficient 247
5.8 Mann–Whitney U test 250
5.9 Quantifying human uncertainty using time and annotation correctness 251
5.10 Coefficient of determination r^2 256
5.11 r^2 as a sum over model predictors 256
5.12 Importance of predictor p_i to parameters $P_{M'}$ 257
5.13 Overall importance of predictor p_i 257

6.1 Silhouette scores 268
6.2 Cohen's d effect size 271
6.3 Pooled standard deviation for Cohen's d 271
6.4 Propagation of input through a single Transformer layer 275
6.5 Segment encodings trace in BERT embeddings 275

6.6 Mean-squared error 277
6.7 Sentence-wise cosine set 279
6.8 Linear decomposition of Transformer embeddings 284
6.9 Input trace in Transformer embeddings 284
6.10 Feed-forward trace in Transformer embeddings 285
6.11 Multi-head attention trace in Transformer embeddings 285
6.12 Biases trace in Transformer embeddings 286
6.13 Normalized dot product 292

A.1 Gaussian Process 366
A.2 Power exponential covariance function 366
A.3 Màtern kernel 367
A.4 Modified Bessel function of the second kind 367
A.5 Conditional distribution under a Gaussian process derived using Bayes' rule 368
A.6 Expected Improvement acquisition function 368

List of Figures

1.1 CBOW and Skip-gram architectures; taken from Mikolov, K. Chen, et al. (2013) 26

1.2 Linguistic Regularity in distributional vector spaces 47

1.3 Vector space metrics, as a variable of dimensionality 61

2.1 Thought experiment of Searle (1980) 76

2.2 Thought experiment of Bender and Koller (2020) 78

3.1 Overview of topographic similarity computations 99

3.2 Artificial languages: basic setup 106

3.3 Artificial languages: holisticity 106

3.4 Artificial languages: synonymy 107

3.5 Artificial languages: semantically ungrounded elements 107

3.6 Artificial languages: paraphrases 108

3.7 Topographic similarity for artificial languages, grouped by parameter (significant items only, avg. of 50 runs) 111

3.8 Meaning distance metrics evaluated on the SICK dataset. (Spearman correlation) 116

3.9 Topographic similarity scores for natural language sentences. . 117

3.10 Hyperparameter selection: Results on BATS and translations . . 123
3.11 Topographic similarity scores for definitions (Common words). 131
3.12 Topographic similarity scores for definitions (Rare words). . . . 132
3.13 Meaning distance metrics evaluated on the MEN dataset. (Spearman correlation) 136
3.14 Topographic similarity scores for natural language definitions. . 138

4.1 Inverse tasks of Definition Modeling and Reverse Dictionary . . 148
4.2 SELECT: Selecting from encoded items; items are contextualized and the definiendum is singled out from them 157
4.3 ADD: Additive marking in encoder; context items and definiendum are marked by adding dedicated embeddings 158
4.4 Overview of Definition Modeling for French data 173
4.5 Logo for the CoDWoE Shared task 184
4.6 Baseline architectures for the CoDWoE shared task 190

5.1 Game Annotation Interface 232
5.2 BlankCrack SQL model: UML diagram 238
5.3 Success rates (in %), groups with fewer than 100 items not included 241
5.4 Word pair difficulty compared to BERT scores 254

6.1 Visualization of silhouette score 269
6.2 Distribution of token silhouette scores 270
6.3 Toy example for segment encoding bias 275
6.4 Log-scaled MSE per reference 277
6.5 Mann–Whitney U tests, 1^{st} vs. 2^{nd} sentences 280

6.6 Relative importance of main terms 294
6.7 Fitting the $\vec{F}t$ term: r^2 across layers 298
6.8 Prediction agreement for WSD models 306
6.9 NER prediction agreement (macro-average) 309

D.1 BlankCrack website banner picture 386
D.2 Member of the United Riddle Solving Squirrels, used to represent cracker playstyle ((a)-annotators) 387
D.3 Minion squid of Tippesk, used to represent blanker playstyle ((b)-annotators) 388
D.4 Displayed for correct (a)-annotations 389
D.5 Displayed for incorrect (a)-annotations 390
D.6 Displayed for highly efficient (b)-annotations 391
D.7 Displayed for mildly efficient (b)-annotations 392
D.8 Displayed for poor (b)-annotations 393
D.9 Displayed during contests ((a)-annotators ranking) 394
D.10 Displayed during contests ((b)-annotators ranking) 395
D.11 Advertisement leaflet (English) 396
D.12 Advertisement leaflet (French) 397

List of Tables

3.1 Linear model of correlation with parameters as predictors. Intercept: $h = 1, s = 1, u = 0, p = 1$. 113

3.2 T-tests between tuned and untuned models 125

3.3 DBnary: number of items per language 126

3.4 Embeddings: corpus statistics 127

4.1 Results (perplexity) 164

4.2 Examples of common errors (ADD model trained on D_{Nor}) 167

4.3 Items retrieved from GLAWI 172

4.4 Analogy results for embedding sets (in %) 174

4.5 General results on test 175

4.6 Supplementary baselines on validation data (BLEU scores) 177

4.7 Production errors typical of neural NLG systems 180

4.8 Production errors of semantic nature 181

4.9 Reverse Dictionary track: results on development set (baseline models) 192

4.10 Definition Modeling track: results on development set (baseline models) 193

4.11 Definition Modeling track results 195
4.12 SGNS Reverse Dictionary track results 196
4.13 Char Reverse Dictionary track results 197
4.14 ELECTRA Reverse Dictionary track results 198

5.1 Distributional pairs produced 223
5.2 WordNet-based pairs 224
5.3 Hand-crafted pairs 225
5.4 Adversarially submitted word pairs 226
5.5 Control word pairs 227
5.6 Results of pilot experiment 228
5.7 Example annotation item 235
5.8 Number of items collected 240
5.9 Effects of filtering on dataset size 245
5.10 Success rates (in %) 245
5.11 Matthews' correlation coefficient 248
5.12 Common language effect size from Mann–Whitney U tests for log time taken when answering correctly vs. incorrectly 249
5.13 Spearman correlations of model scores and time-weighted human judgments 251
5.14 Proportion of r^2 explained by type of predictor (in %) 258

6.1 Correlation (Spearman ρ) of cosine similarity and relatedness ratings on the STS and SICK-R benchmarks 282
6.2 Masked language model accuracy, in % 301
6.3 Accuracy on SemCor WSD (in %) 304

6.4 Macro-f_1 on WNUT 2016 (in %) 308

B.1 BATS translations: examples and numbers per category 375

B.2 BATS translations: examples and numbers per category (continued) 376

Abstract

To be or not be—who are we? That is the question
You might us ask: to bleed or not to bleed, for we are metal

— The Metal Shakespeare Company, *To Bleed or not to Bleed*

This dissertation studies the status of word embeddings, i.e, vectors produced by NLP systems, insofar they are relevant to linguistic studies. We more specifically focus on the relation between word embeddings and distributional semantics—the field of study based on the assumption that context correlates to meaning. We question whether word embeddings can be seen as a practical implementation of distributional semantics.

Our first approach to this inquiry consists in comparing word embeddings to some other representation of meaning, namely dictionary definitions. The assumption underlying this approach is that semantic representations from distinct formalisms should be equivalent, and therefore the information encoded in distributional semantics representations should be equivalent to that of definitions. We test this assumption using two distinct experimental protocols: the first is based on overall metric space similarity, the second relies on neural networks. In both cases, we find limited success, suggesting that either distributional se-

mantics and dictionaries encode different information, or that word embeddings are not linguistically coherent representations of distributional semantics.

The second angle we adopt to study the relation between word embeddings and distributional semantics consists in formalizing our expectations for distributional semantics representations, and comparing these expectations to what we observe for word embeddings. We construct a dataset of human judgments on the distributional hypothesis, which we use to elicit predictions on distributional substitutability from word embeddings. While word embeddings attain some degree of performance on this task, their behavior and that of our human annotators are found to drastically differ. Strengthening these results, we observe that a large family of broadly successful embedding models all exhibit artifacts imputable to the neural network architecture they use, rather than to any semantically meaningful factor.

Our experiments suggest that, while we can formally delineate criteria we expect of distributional semantics models, the linguistic validity of word embeddings is not a solved problem. Three main conclusions emerge from our experiments. First, the diversity of studies in distributional semantics do not entail that no formal statements regarding this theory can be made: we saw that distributional substitutability provides a very convenient handle for the linguist to grasp. Second, that we cannot easily relate distributional semantics to another lexical semantic theory questions whether the distributional hypothesis actually provides an alternative account of meaning, or whether it deals with a very distinct set of facts altogether. Third, while the gap in quality between practical implementations of distributional semantics and our expectations necessarily adds on to the confusion, that we can make quantitative statements about this gap should

be taken as a very encouraging sign for future research.

Résumés en français

Vive Henri IV, vive ce Roi vaillant!
Ce diable à quatre a le triple talent
De boire et de battre, et d'être un vert-galant.
De boire et de battre, et d'être un vert-galant.

— Boisson Divine, *Vive Henri IV*

Résumé court

Cette thèse s'intéresse au statut des plongements lexicaux (ou « *word embeddings* »), c'est-à-dire des vecteurs de mots issus de modèles de Traitement Automatique des Langues. Plus particulièrement, notre intérêt se porte sur leur valeur linguistique et la relation qu'ils entretiennent avec la sémantique distributionnnelle, le champ d'étude fondé sur l'hypothèse que le contexte est corrélé au sens. L'objet de notre recherche est d'établir si ces plongements lexicaux peuvent être considérés comme une implémentation concrète de la sémantique distributionnelle.

Notre première approche dans cette étude consiste à comparer les plongements lexicaux à d'autres représentations du sens, en particulier aux définitions

telles qu'on en trouve dans des dictionnaires. Cette démarche se fonde sur l'hypothèse que des représentations sémantiques de deux formalismes distincts devraient être équivalentes, et que par conséquent l'information encodée dans les représentations sémantiques distributionnelles devrait être équivalente à celle encodée dans les définitions. Nous mettons cette idée à l'épreuve à travers deux protocoles expérimentaux distincts : le premier est basé sur la similarité globale des espaces métrisables décrits par les vecteurs de mots et les définitions, le second repose sur des réseaux de neurones profonds. Dans les deux cas, nous n'obtenons qu'un succès limité, ce qui suggère soit que la sémantique distributionnelle et les dictionnaires encodent des informations différentes, soit que les plongements lexicaux ne sont pas motivés d'un point de vue linguistique.

Le second angle que nous adoptons ici pour étudier le rapport entre sémantique distributionnelle et plongements lexicaux consiste à formellement définir ce que nous attendons des représentations sémantiques distributionnelles, puis à comparer nos attentes à ce que nous observons effectivement dans les plongements lexicaux. Nous construisons un jeu de données de jugements humains sur l'hypothèse distributionnelle. Nous utilisons ensuite ce jeu pour obtenir des prédictions sur une tâche de substituabilité distributionnelle à partir de modèles de plongements lexicaux. Bien que nous observions un certain degré de performance en utilisant les modèles en question, leur comportement se démarque très clairement de celui de nos annotateurs humains. Venant renforcer ces résultats, nous remarquons qu'une large famille de modèles de plongements qui ont rencontré un franc succès, ceux basés sur l'architecture Transformer, présente des artéfacts directement imputables à l'architecture qu'elle emploie plutôt qu'à des facteurs d'ordre sémantique.

Nos expériences suggèrent que la validité linguistique des plongements lexicaux n'est aujourd'hui pas un problème résolu. Trois grandes conclusions se dégagent de nos expériences. Premièrement, la diversité des approches en sémantique distributionnelles n'implique pas que ce champ d'étude est voué aux approches informelles: nous avons vu que le linguiste peut s'appuyer sur la substituabilité distributionnelle. Deuxièmement, comme on ne peut pas aisément comparer la sémantique distributionnelle à une autre théorie lexicale, il devient nécessaire d'étudier si la sémantique distributionnelle s'intéresse bien au sens, ou bien si elle porte sur une série de faits entièrement distincte. Troisièmement, bien que l'on puisse souligner une différence entre la qualité des plongements lexicaux et ce qu'on attend qu'ils puissent faire, la possibilité d'étudier cette différence sous un angle quantitatif est de très bon augure pour les travaux à venir.

Résumé long

Lorsque les auteurs qui s'intéressent aux plongements lexicaux établissent un parallèle avec un cadre théorique linguistique, il s'agit invariablement d'études tirées du domaine de la sémantique distributionnelle. Ce domaine est issu des travaux de Harris (1954), et se fonde sur l'hypothèse que le contexte linguistique suffit pour caractériser le sens d'un mot. Ce champ d'étude n'est cependant pas défini de manière formelle, et il est courant que les chercheurs adoptent une définition pratique : par exemple Boleda (2020) spécifie qu'un modèle de sémantique distributionnelle doit correspondre à un ensemble de vecteurs dans un espace continu calculés à partir de co-occurrences linguistiques. Malgré cela, les études en sémantique distributionnelle font preuve d'une extrême diversité

dans leurs outils, entre réseaux de neurones artificiels et matrices de compte de co-occurrence. Cette hétérogénéité est si impressionnante qu'il est discutable de considérer toutes ces études comme faisant partie d'une même théorie.

En somme, le statut *non-formel* de la sémantique distributionnelle complique la tâche à qui veut y voir un cadre théorique cohérent. Si l'on cherche à établir ce qu'est la sémantique distributionnelle, on ne saurait aisément dire de quels faits elle relève, quelles prédictions elle fait, ou à quel autre cadre on peut la comparer. Les applications du réseau profond de plongements contextualisés BERT (Devlin et al., 2019) sont par exemple très distinctes de ce que l'on peut étudier et apprendre du modèle LSA (Landauer et Dumais, 1997), calculé à partir de tabulations d'occurrences dans des documents.

Pour surmonter ces limites, nous nous intéresserons plus particulièrement à déterminer si la sémantique distributionnelle peut être considérée comme ce que nous appelons ici une théorie de sémantique lexicale. Par ce terme de « théorie de sémantique lexicale » (ou plus simplement « théorie lexicale »), nous entendons deux caractéristiques :

(i) que le cadre en question fasse l'hypothèse d'un nombre de propositions, qui, prises ensemble, produisent une définition générale du concept de sens tel qu'il s'applique aux mots du lexiques ;

(ii) qu'à cette définition générale du sens corresponde une méthode concrète pour assigner un sens à tout mot en particulier.

Dans cette perspective, la sémantique distributionnelle serait la théorie lexicale (i) qui suppose que le sens d'un mot est dérivé du contexte linguistique dans lequel il apparaît, et (ii) qui pour assigner un sens à un mot emploie généralement

des plongements lexicaux.

Ce concept de théorie de sémantique lexicale a deux conséquences importantes. Premièrement, si l'on peut assigner un sens à chaque mot, alors on doit pouvoir répéter cette procédure pour la totalité du lexique. En d'autres termes, à toute théorie lexicale doit correspondre une ou plusieurs implémentations concrètes ; pour ce qui est de la sémantique distributionnelle, ces implémentations seront les modèles de plongements lexicaux.

Secondement, ce concept de théorie lexicale peut s'appliquer à d'autres cadres non-formels. En particulier, la lexicographie (que nous définissons ici comme l'étude et la pratique de l'élaboration de dictionnaires) correspond aussi à notre concept. Dans cette grille d'analyse, l'hypothèse fondamentale de la lexicographie est que le sens d'un mot peut être décrit à l'aide de la langue, et plus particulièrement à l'aide de définitions. En pratique, on peut implémenter cette théorie à l'aide de dictionnaires. Dans cette thèse, nous utilisons la lexicographie comme un étalon pour jauger la sémantique distributionnelle. Ces deux définitions implicites de ce que doit être le sens du mot peuvent ne pas être compatibles ; étudier si et comment elles entrent en conflit est l'un des enjeux majeurs de notre étude.

Nous ne pouvons évidemment pas comparer ces deux théories lexicales de manière directe : puisqu'il ne s'agit pas de cadres formels, nous ne pouvons comparer leurs portées en les plaçant toutes deux dans le champ abstrait des mathématiques. Il nous faut à la place comparer expérimentalement leurs implémentations — d'où notre insistance à ce qu'une théorie lexicale possède une manière concrète d'assigner un sens à un mot. Même cette comparaison d'implémentations s'avère difficile, puisque les définitions lexicographiques et les plongements lexicaux sont deux types d'objets différents, les premiers étant des séquences de

mots, et les seconds des vecteurs de nombres réels.

En somme, cette thèse a pour but d'étudier si la sémantique distributionnelle peut être conçue comme une théorie de sémantique lexicale. Quels éléments à charge ou à décharge pouvons-nous établir ? Y a-t-il des caractéristiques communes aux représentations distributionnelles et aux implémentations d'autres théories lexicales, tels que les dictionnaires ? Que nous révèle l'examen des plongements lexicaux eux-mêmes ?

Dans la première partie de cette thèse, nous donnons une vue d'ensemble de l'état de l'art pertinent pour notre objet de recherche. Le Chapitre 1 contient une présentation générale de la sémantique distributionnelle. Nous y discutons des développements historiques de la sémantique distributionnelle et des modèles qui y sont associés, depuis la création de ce champ d'études jusqu'aux évolutions les plus récentes. Nous soulignons aussi la grande diversité de ce domaine, marqué à la fois par la variété des modèles et par l'hétérogénéité des méthodologies proposées pour leur évaluation. Dans le Chapitre 2, nous nous penchons sur la lexicographie. Nous définissons brièvement ce qu'est un dictionnaire, avant de présenter les usages qu'ils ont trouvés dans le TAL.

La deuxième partie de cette thèse aborde de front notre problématique. Nous y comparons dictionnaire et plongements lexicaux. Dans le Chapitre 3, nous essayons de les comparer en nous basant sur les notions de distance qu'ils encodent : distance d'édition entre deux définitions, et distance vectorielle entre deux plongements. Il ressort de cette première série d'expérience que de nombreux facteurs peuvent perturber nos analyses. Ceci nous amène à adopter une méthodologie différente dans le Chapitre 4, où nous étudions si les réseaux de neurones artificiels nous permettent d'implémenter des fonctions inverses pour

convertir un plongement en une définition équivalente et vice-versa.

Nous ne rencontrons qu'un succès limité dans ces deux approches. Ceci nous amène, dans la troisième et dernière partie, à questionner notre hypothèse de départ : les plongements lexicaux correspondent-ils à une implémentation de la sémantique distributionnelle ? Dans le Chapitre 5, nous étudions si les plongements lexicaux correspondent aux intuitions des êtres humains quant à l'hypothèse distributionnelle. Nous construisons à cette fin un jeu en ligne afin de collecter des jugements humains, et proposons une formalisation pour la sémantique distributionnelle. La comparaison des plongements lexicaux et des jugements humains démontre cependant que les premiers semblent très éloignés des seconds, ce qui remet en cause la validité linguistique qu'on leur avait jusque-là attribuée. Le Chapitre 6 adopte une approche complémentaire ; nous y étudions les artéfacts imputables à l'architecture Transformer qui sont perceptibles dans les plongements lexicaux contextualisés.

À l'issue de ces expériences, nous disposons de multiples élements pour répondre à notre problématique. Dans le Chapitre 1, nous avons pu voir la diversité de la sémantique distributionnelle en tant que domaine d'étude. Cette diversité n'implique pas qu'aucune formalisation n'est possible, et nous en avons proposée une qui se base sur la substituabilité distributionnelle dans la Section 5.1. Cependant, si l'on admet que cette formalisation est appropriée, il faut alors aussi admettre que les plongements lexicaux tels qu'ils existent aujourd'hui semblent de piètres modèles linguistiques. S'ils atteignent un certain degré de performance sur la tâche de substitution distributionnelle (Section 5.4.2), la manière dont ils produisent ces résultats ne correspond pas à ce que nous observons chez les êtres humains (Section 5.4.3). Ceci est d'autant plus troublant qu'on peut proposer une

description beaucoup plus fine de certains modèles de plongements parmi les plus populaires à partir de leur seule architecture neuronale (Chapitre 6). Somme toute, il est raisonnable de conclure que les plongements lexicaux modernes ne répondent pas aux critères que l'on peut attendre des modèles de sémantique distributionnelle.

Ceci est à l'opposé de ce que l'on peut voir de la lexicographie. Bien qu'il y ait un certain flottement dans la définition d'un dictionnaire (Section 2.1), la communauté TAL fait confiance aux dictionnaires et les utilise comme inventaires sémantiques fiables (Sections 2.3 et 2.4). Cette confiance n'est toutefois qu'un des facteurs qui entrent en compte lorsque l'on veut comparer dictionnaires et plongements lexicaux. L'alignement entre mot et sens (Sections 4.1 et 4.2), la qualité des représentations vectorielles (Sections 3.3 et 4.3) ou même la structure de la langue, avec ses synonymes et ses mots grammaticaux (Section 3.2) pèsent également sur ces comparaisons.

Nous ne pouvons pas établir que la sémantique distributionnelle et la lexicographie sont des théories de sémantique lexicale équivalentes. Il est difficile de tirer une conclusion qui porte sur la sémantique distributionnelle à partir des plongements lexicaux, puisque que leur lien reste à déterminer. Mais si nous supposons que les plongements lexicaux sont des modèles de sémantique distributionnelle (quoique imparfaits), alors nos résultats suggèrent que le sens tel qu'il est compris en lexicographie est une notion distincte du sens tel que la sémantique distributionnelle le conçoit. Peut-être que nous pourions trouver une théorie lexicale davantage comparable aux plongements lexicaux. Peut-être qu'un véritable modèle de sémantique distributionnelle serait à même de résoudre la tâche de substitution distributionnelle de la même manière que les

humains. Peut-être que les problèmes que nous avons soulignés se résument à une question de calibrage de nos modèles. Peut-être que résoudre toutes ces difficultés nous permettrait d'obtenir des représentations distributionnelles riches et aisément comparables à des répresentations sémantiques, définitions ou autres. Nous n'avons pour l'heure pas de preuve qui indique que ces obstacles seront surmontables.

Ces problèmes sont en partie issus de la culture en TAL, qui se focalise principalement sur l'ingénierie d'applications pratiques. Les considérations théoriques qui intéressent la linguistique (et la linguistique computationnelle) n'arrivent qu'après, si jamais elles entrent en compte. Il semble que *ce qu'un modèle peut faire* prime souvent sur *ce qu'il modélise*. S'il est bon de reconnaître que l'approche qui correspond au premier de ces deux objets d'études à ses avantages, notre intérêt dans cette thèse s'est portée plutôt sur le second. Dans un contexte où les modèles neuronaux deviennent de plus en plus larges et omniprésents et de moins en moins compréhensibles, il nous semble crucial de ne pas abandonner nos liens aux sciences du langage. Un des buts que nous poursuivons dans cette thèse est d'explorer la variété des outils à la disposition du chercheur en TAL, de l'algèbre fonctionnelle et linéaire (Sections 1.4 et 6.2) aux réseaux de neurones (Sections 4.2 et 4.3), des expériences de pensées (Section 2.2) à la collecte de jugements linguistiques (Section 5.3). Cet éventail n'est bien sûr pas exhaustif, mais nous espérons avoir démontré ce qu'apporte une approche ancrée dans la linguistique à l'étude des systèmes de TAL.

On the Status of Word Embeddings as Implementations of the Distributional Hypothesis

Introduction

There's no time to waste
There's so much more to do
We built it all backwards so let's fix that too

— Rivers of Nihil, *The Tower*

When authors interested in word embeddings draw an explicit connection to a linguistic framework, they invariably point towards the field of distributional semantics. This field stems from the seminal work of Harris (1954), and is based on the assumption that linguistic context suffices to characterize the meaning of a word. This field of study is however not formally defined; instead, researchers often adopt a practical definition, such as the one from Boleda (2020), which states that Distributional Semantics Models correspond to graded high-dimensional vectors learnt from natural language data. It remains that distributional semantic studies vary so wildly in the setup they use—from deep neural networks to count matrices—that is debatable, if not doubtful, that they can be considered as parts of a coherent and systematic theory.

The *non-formal* status of distributional semantics makes it complex to conceptualize this field as a coherent framework. When characterizing what distri-

butional semantics are, we cannot easily delineate which facts are relevant, what is predicted, or which framework to compare to. Applications of the deep contextualized embedding model BERT (Devlin et al., 2019), for instance, are very different from what we can study through and learn from the count-based document model of LSA (Landauer and Dumais, 1997).

To overcome these limitations, we will study whether distributional semantics can be conceptualized as what we here call a lexical semantic theory. By "lexical semantic theory" (or indifferently "lexical theory"), we mean two things:

(i) that the field of study at hand proposes or assumes a number of statements that, taken together, provides a general definition of word meaning;

(ii) that this general definition of word meaning corresponds to a practical way of ascribing a meaning to any specific word.

In such a view, distributional semantics would be the lexical theory (i) which argues that the meaning of a word is ultimately derived from the linguistic contexts in which it occurs; and (ii) where to ascribe meaning to words we generally employ word embeddings, i.e., vector representations computed from large numbers of word collocations, as observed in text corpora.

This working definition of a lexical semantic theory allows to tease apart two types of studies that could be construed as distributional semantics. On the one hand, we find research that attempts to describe the meaning of every lexicon item by relying solely on linguistic contexts. This is for instance what we find in Harris (1954), who argues that word distribution correlates with, and suffices to characterize word meaning. On the other hand, we have works that happen at some point or another to use word distribution for some practical purpose,

such as characterizing the polysemy of a word. An example of this second type would be Firth (1957), whose take on meaning is that it ought to be informed by all manners of linguistic analyses, including, but not limited to, those based on word distribution. Only the former of these two types can be properly thought of as a lexical semantic theory, and we will almost exclusively consider the former type of studies in this dissertation.

There are two important consequences to this concept of a lexical semantic theory. First, if we can assign a meaning to any word, then we should be able to extend this procedure to the full lexicon. In other words, any lexical semantic theory should have a corresponding practical implementation. In the case of distributional semantics, these implementations will correspond to word embedding models.

Second, the concept of a lexical semantic theory can be applied to other non-formal frameworks. In particular, lexicography—that we define here as the study and practice of dictionary-making—also corresponds to our concept of lexical semantic theory. In such a view, the fundamental assumption that underlies this lexical theory is that word meaning can be described through language and more specifically definitions; in practice, it can be implemented as dictionaries. Throughout this dissertation, we will use lexicography as a standard against which to compare distributional semantics. These two views on what is word meaning may or may not conflict, and studying whether—and how—they are incompatible will be one of the major topics that we will discuss.

Obviously we cannot compare these two lexical semantic theories directly—as they are not purely formal models, comparing their predictions by setting them in the abstract neutral ground of mathematics is outside our reach. In-

stead, we will have to compare their concrete implementations—hence our insistence on providing a practical way of ascribing meaning to words. Even comparing their implementations can prove to be difficult, as dictionary definitions and word embeddings correspond to very different objects: vectors on the one hand, and sequences of text on the other.

In sum, this dissertation sets about to study whether distributional semantics can be construed as a lexical semantic theory. The diversity of approaches that have been labelled as distributional semantics models may entail that this field of study does not correspond to a coherent and consistent theoretical framework of analysis. What evidence can we find that would confirm or infirm this view? Are there common characteristics between distributional semantics models and implementations of other lexical semantic theories, such as dictionaries? Are there any immediate arguments to be made by studying word embeddings themselves, or by comparing their behavior to that of human speakers? We will structure our argumentation in three parts.

In the first part, we will set the scene, so to speak, by giving an overview of the state of the art. Chapter 1 will provide a general overview of distributional semantics. In this first chapter, we will discuss the history of distributional semantics and distributional semantics models, from their inception up till the most recent developments. We will also provide some elements to underscore how diverse the field of distributional semantics is, both in terms of how word embedding models are formally defined and studied in practice. In Chapter 2 we will then turn to lexicography. After a brief review of what a dictionary is, we will discuss some of the applications they have been found useful for in NLP.

The second part will tackle our research question head on, and attempt to

compare dictionaries and word embeddings. In Chapter 3, we will attempt to compare the two using the notion of distance: edit distance between definitions and vector distance between word embeddings. As we will see, many caveats apply when adopting such a methodology. These issues will lead us to adopt a different experimental protocol. Namely, in Chapter 4, we will study whether we can use neural networks to implement inverse functions that convert word embeddings into definitions and back.

In the third and final part, we will take stock of the limitations of our comparisons, and reflect on the initial hypothesis we based our experiments on: are word embeddings an implementation of a lexical semantic theory? Chapter 5 will investigate whether word embeddings match human judgments with respect to the distributional hypothesis. To that end, we will describe an online game implemented to collect human judgments, and propose a tentative formalization for distributional semantics models. This will allow us to compare the behavior of word embeddings to that of human annotators. In our final chapter Chapter 6, we will adopt a complementary approach. Namely, we will look for artifacts in Transformer embedding spaces imputable to the design of such networks, i.e., biases due to their objective functions or their formal structure.

I

A Tale of Two Theories

1

Distributional Semantics as a Lexical Theory

Has he lost his mind?
Can he see or is he blind?
Can he walk at all
Or if he moves will he fall?

Is he alive or dead?
Has he thoughts within his head?
We'll just pass him there
Why should we even care?

— Black Sabbath, *Iron Man*

This chapter will study whether distributional semantics can be construed as a lexical semantic theory. The notion of lexical semantic theory, as presented in the introduction, requires of a field of study that it yields a corresponding implementation. In distributional semantics, these implementations are generally known as "distributional semantics models" (DSM) or, almost interchangeably in the literature, "word embeddings." We will nonetheless introduce a terminolog-

ical distinction. We will reserve the term "distributional semantics models" to items we consider as implementations of distributional semantics. On the other hand, "word embeddings" will refer to algorithms, systems and softwares that convert words into vectors—regardless of whether these algorithms, systems or softwares are related to distributional semantics. The term of "word embedding," more precisely, was coined specifically for word vector representations drawn from neural networks.

While the position that distributional semantics could be considered a lexical theory on its own was more or less self-evident in early works, the chronological overview we conduct in Section 1.1 stresses that a more nuanced position is required to account for the important variations across implementations of distributional semantics. The major point that will emerge from our study is that this variation in implementations is in fact a thoroughly central characteristic of the field of distributional semantics as it exists today. We will focus on a few DSM architectures, word2vec, BERT and ELECTRA, in Section 1.2, underscoring their important differences. Differences in model architectures, as we will see in Section 1.3, also entail differences in experimental protocols designed to study and evaluate distributional semantics models. Lastly, we will focus on how vector space dimensionality itself affects comparisons of distributional semantics models in Section 1.4.

1.1 Chronological Overview

Our starting point will be to provide a summary of what is addressed in the field of distributional semantics, mostly through a chronological perspective.

1.1.1 Inception

Distributional semantics was founded by the seminal work of Harris (1954), which proposed the *distributional hypothesis*. In short, word distribution should correlate with word meaning. The idea is that the meaning of a word should constrain the sort of contexts it can appear in. Let us take a referential view of meaning: more precisely, let us consider that "*cat*" refers to the feline animal, and focus on the sort of situation in which cats are likely to be involved. It is more probable that these situations involve purring and whiskers than foreign politics or churches. If we now think of how we would describe these situations through language, we see that the words we will use are more likely to be related to cats—in other words, because of what they mean, words like "*purring*" and "*whiskers*" are intuitively more likely to occur in the context of the word "*cat*" than words like "*ecclesiastical*" or "*foreign*". Harris (1954) stresses that it is unlikely that specific words or morphemes can be ascribed to specific meanings in a one-to-one relation. Nonetheless, his distributional hypothesis claims that semantic statements—statements about the meaning of utterances and words—have distributional counterparts. One can say things about word meaning from word distribution alone, and the two are therefore correlated, to some extent. To quote Harris (1954) directly:

> if one wishes to speak of language as existing in some sense on two planes—of form and of meaning—we can at least say that the structures of the two are not identical, though they will be found similar in various respects.

1.1.2 From a methodology to a lexical semantic theory

What sort of distributional statement can and should be made is not entirely transparent in the original proposal of Harris (1954). Harris himself considers contrasting the valid contexts of words. For instance, if "*eye-doctor*" and "*oculist*" can systematically occur in the same contexts, we can say they mean the same thing. For word pairs that do not share this property, like "*oculist*" and "*lawyer*", we can still approximate how similar their meanings are, simply by considering the "amount of difference in their environments." If two words share no context, like "*oculist*" and "*of*", we can highlight that they belong to two different classes of words, or parts of speech. Context here is to be understood as linguistic context: Harris himself gave as example the phonemes or the words in the immediate vicinity of the target word of interest.

To test pairs of words in similar contexts, the linguist can rely on informants' judgments. While this method sheds a light as to how exactly we can leverage distributional facts for semantic analysis, it is mostly intended to be applied to controlled situations and small corpora. This methodology therefore has the disadvantage of being partial: it does not cover all the facts at the linguist's disposal. Kay (2011) argues that the complex phenomenon that is language in fact demands that we adopt techniques spanning the entirety of the lexicon and encompassing as much data as possible. Much as there is little that we can learn about the weather system from a few well chosen drops of rain, linguists should focus on understanding human languages as the complex systems they are rather than focus specifically on how some well chosen elements interact within them. The argument that Kay (2011) develops is centered on the computational nature of

language processes. He further stresses that machine learning and statistics are central to the tool set of the computational linguist, as they greatly expand the scope of the work undertaken by automating the tedious, tiresome work of manually studying every single data point. If we adopt a position similar to that of Kay (2011), the proposal of Harris (1954) seems inadequate: eliciting judgments from informants for word pairs of interest prevents us from studying the system as a whole. Instead, we are forced to focus specifically on pre-selected items. This characteristic is also necessary if we want to study distributional models as semantic theories: we require a procedure that yields the meaning representation of any possible word.

We can see this play out, for instance, in the study of Rubenstein and Goodenough (1965). The study sets on to assess the validity of the distributional hypothesis. The authors focus on a small subset of words: 65 pairs, all nouns, with varying degrees of semantic similarity (from "*cord*" vs. "*smile*" to "*gem*" vs. "*jewel*"). They then contrast semantic similarity judgments from informants to the overlap in attested contexts. Their methodology is convincing, but the strength of their conclusion is somewhat undermined by the limited size and diversity of their sampled paired words. We must not forget that the computational means available to Rubenstein and Goodenough (1965) are not those available fifty years later to Kay (2011), and modern computing power does allow us to overcome this limitation.

In fact, setting aside any notion of computational power, a very similar criticism is expressed by Miller (1967). Miller's position is that the study of Rubenstein and Goodenough (1965) cannot be converted into a systematic description of meaning: assuming we restrict ourselves to a vocabulary of 100 words, there

are already 4950 different word pairs for which similarity judgments are required. The problem only worsens when we consider a larger, more realistic vocabulary.

1.1.3 Co-occurrence count matrices

The first accounts of distributional semantics approaches for the study of large amounts of data rely on a second branch of research, that of vector-based models derived from co-occurrence count. This approach can be seen as a natural development for distributional semantics, in that it is based on the tabulation of the contexts of each word.

Let us use a simple example, and assume we have the following corpus of three documents D_1, D_2 and D_3, whose contents would be:

D_1: Fat cat sat on the mat.

D_2: The cat is chasing the mouse on the mat.

D_3: My dog is a pooch.

From this raw data, we can construct a list of contexts for any word. Two possibilities have been explored in the literature: we can either consider the set of documents where a given word occurs or the words which co-occur with our target words. The former would correspond to:

$$\text{contexts}_D(w) = \#\{D_i | w \in D_i\}$$

and the latter to:

$$\text{contexts}_W(w) = \#\{w' | \{w, w\} \subset D_i\} \tag{1.1}$$

To take a concrete case, if we consider the word "*cat*", modeling contexts as doc-

ument would yield:

$$\text{contexts}_D(cat) = \{D_1, D_2\}$$

whereas adopting a word-based approach would yield:

$$\text{contexts}_W(cat) = \{Fat, cat, sat, on, the, mat, .,$$
$$The, is, chasing, mouse\}$$

A natural refinement of this approach consists in keep track not only of the contexts, but also of the number of times our target word co-occurs with this context. It is rather handy to represent this information as a count matrix, where each cell $C_{t,c}$ tracks the number of times the target word t co-occurs with the context c. Continuing with our previous example, if we model contexts through word co-occurrence, then we would have $C_{cat,\ mat} = 2$ and $C_{cat,\ dog} = 0$.

Perhaps one of the best-known pioneering works for this approach is that of Salton et al. (1975), who proposed to represent documents as a vectors. In their proposal, components of document-vectors corresponds to the presence or absence of a given term in the corresponding document. It should be noted that the work of Salton et al. (1975) does not refer to the distributional hypothesis. The motivations underpinning this application are in fact purely practical and stem from the authors' interests in the field of Information Retrieval. The intuition behind this layout is to ensure that related documents containing similar terms are placed close to one another, and unrelated documents would are placed far apart in the vector space, so as to be easily separable.

Another point to stress in the work of Salton et al. (1975) is that it pro-

poses to construct document representations, rather than word representations. Nonetheless, converting these into word vectors is straightforward. As specific components correspond to individual terms, we can consider the whole vector set as a count matrix. Column vectors therefore summarize the documents in which a specific term can be found. This is essentially the starting point of the 'Latent Semantic Analysis' of Landauer and Dumais (1997). Count matrices of the sort, and LSA in particular, were one of the first type of DSMs to attract a certain level of attention among the scientific community. Another important proposal to mention here that also belongs to this category of models is the 'Hyperspace Analogue to Language' of Lund and Burgess (1996).

Count-based models can be classified using characteristics such as how vector component values are computed. Aside from simple raw counts, we find tf-idf weighting:

$$\text{tf.idf}(w,d) = \frac{f_{w,d}}{\sum\limits_{w' \in T} f_{w',d}} \cdot \log \frac{\#D}{1 + \#\{d' | w \in d' \wedge d' in D\}} \tag{1.2}$$

where $f_{w,d}$ corresponds to the number of occurrences of the word or term w in the context d. This metric was born from the field of Information Retrieval (IR), where contexts are mostly documents. As such, its purpose is to quantify how strong and exclusive is the association between the context document d and the word w. It is therefore constructed from two components: first, the term frequency (tf, formally, $f_{w,d} / \sum\limits_{w' \in T} f_{w',d}$) measures the frequency of the word w in the context document d, as compared to all other words w' attested in the context; second, the inverted document frequency (idf, formally $\log(\#D / (1 + \#\{d' \| w \in d' \wedge d' \in D\}))$) measures how exclusive this association is,

by counting the number of contexts that contain the word w. Another frequently encountered scheme is that of pointwise-mutual information:

$$\text{PMI}(w, d) = \log \frac{p(w, d)}{p(w)p(d)} \tag{1.3}$$

where w is the word of interest, and d is the context, often the sentence where w occurs. The idea is to compare the likelihood of $p(w, d)$, observing the word w and the context d jointly, to $p(w)p(d)$, what we would expect if the two items were independent. Closely related to the latter is the positive pointwise mutual information, where values below 0 are capped:

$$\text{PPMI}(w, d) = \max(0, \text{PMI}(w, d)) \tag{1.4}$$

Note that PMI yields negative values only if $p(w)p(d)$, the baseline expectations for independent events, is greater than $p(w, d)$, the probability of jointly observing the word w and the context d. As such, capping PMI values to 0 corresponds to disregarding observations where the joint probability of a word and a context is less likely than a random accident.

Another axis of variation consists in the algorithm used to down-project the very high dimensional count vectors (with dimensions in the tens of thousands) to manageable lower-dimensional spaces (with dimensions in the hundreds). A noteworthy mention here is truncated SVD, which was specifically introduced to the community by Landauer and Dumais (1997). Under this algorithm, word vectors are rotated and stretched such that vector components are all ordered by the variance in the dimension they represent: hence the first dimension after a

truncated SVD algorithm varies the most, followed by the second, and so on. As such, the last components tend to not vary much, and can be ignored in order to limit the number of dimensions necessary to describe the embedding space.

Another important remark is that, unlike the LSA model of Landauer and Dumais (1997), the HAL model of Lund and Burgess (1996) uses word co-occurrences as contexts. Vector components do not represent the documents where a word can be found: instead, they represent other words in the vocabulary; the scalar value for a given component measures how frequently the word being represented and the word for that component occur near one another. These two definitions of word context—the documents wherein the word of interest can be found, or the other words attested right next to the word of interest—would remain a major point of distinction between various DSM architectures.

1.1.4 The birth and rise of neural word embeddings

Independently to these works, neural approaches to computing word representations were developed with the intent of solving the "curse of dimensionality." The longer a specific sequence of words is, the more unlikely it is to occur in any sample. In the case of statistics-based models such as neural networks trained on linguistic data, this entails that it is virtually impossible to gather numerous examples of every possible n-gram. Models will therefore be tested on items that are in likelihood very different from all training items. To address this, Bengio et al. (2003) suggest to use vector representations of words such that representations of semantically similar words end up near one another.

This approach is very reminiscent of what Salton et al. (1975) tried to achieve.

Salton et al. (1975) suggest document vectors designed such that any neighboring vectors should containing related keywords and address related topics. Bengio et al. (2003) propose word representations computed such that any neighboring vectors correspond to related concepts. In both cases, the basis for creating consistent vector neighborhoods is to facilitate the software exploitation of the items—retrieve documents more easily in some IR systems for Salton et al. (1975), and model language more efficiently for Bengio et al. (2003).

In detail, Bengio et al. (2003) propose to use a neural network to compute word representations. Their proposed architecture is a feed-forward language model architecture. The model receives as input the concatenation of learned representations of the previous words. Its objective is then to predict what word comes next. These learned representations, which were coined "word embeddings," were found to significantly bring down perplexity scores on language modeling benchmarks, as compared to n-gram approaches. This seminal work proved to be inspirational. For instance, Collobert and Weston (2008) point out that architectures of this sort can be trained to solve a broad variety of tasks at once through multitask learning.

These approaches led to the work of Mikolov, K. Chen, et al. (2013). The word2vec model they propose consists in a feed-forward neural network rather similar to the proposal of Bengio et al. (2003). Two training procedures are considered, depending on how the context words $c_1 ... c_n$ and the word of interest w are wired. The first, CBOW, consists in summing the learned representations of the context words $c_1 ... c_n$, and using this summed vector as the input of a log-linear classifier to predict the word of interest w. The second approach is conceptually the reverse: using the learned representation of the word of interest w

as an input to predict the context words $c_1 ... c_n$ one after the other, and taking the sum of the losses to train the model.

Until then, there had been no sudden transition from count-based approaches to neural approaches to distributional semantics: for instance, count-based approaches such as that of Griffiths et al. (2007) were contemporaneous to the neural proposal of Collobert and Weston (2008). Yet the approach of Mikolov, K. Chen, et al. (2013) turned out to be extremely popular. One reason for this enthusiasm was that word2vec vectors were suggested to describe a manifold—that is to say, basic vector operations over word2vec vectors encode analogy relations (Mikolov, Yih, et al., 2013). Another reason was that they were shown to be roughly equivalent: Levy and Goldberg (2014b) formally demonstrated how the word2vec model and the earlier count-based matrices were related, as the loss function used in the skip-gram training procedure was shown to implicitly factorize a PMI-weighted occurrence count matrix. A third reason, perhaps even more important, was software optimizations and hardware improvement. The proposal of Mikolov, K. Chen, et al. (2013) cut down the necessary training time from months to days, if not hours.

One research avenue focused on how to improve the word2vec architecture, of which we can cite two major examples. The first is the GloVe model of Pennington et al. (2014). The name, a contraction of 'global' and 'vector', reflects their intention to take into account global, document-level co-occurrences when computing a vector. The second example is the FastText model of Bojanowski et al. (2017). They remarked that models like word2vec were better adapted to morphologically poor languages like English than morphologically rich languages like French or Russian: these models do not take into account sub-word informa-

tion, which is crucial in inflectional languages. The proposal of Bojanowski et al. (2017) was therefore to add vector terms corresponding to character n-grams of the word of interest w. These character n-grams would model the orthographic regularities that correlate with the inflection patterns of words.

In recent years, yet another development has occurred in the field of neural word embeddings. An important conceptual shift was the introduction of contextual word embeddings. The idea was perhaps most clearly expressed in the work of Peters et al. (2018), who stress that contextual embeddings like those yielded by their ELMo model are "a function of the entire input sentence." That is to say the vector representation associated with a given word depends not only on what that word is, but also on what other words occur in this sentence. Another way to frame the difference is to stress that, if models like word2vec yield vector representations for word *types*, contextual word embeddings like ELMo yield vector representations for word *tokens*.

The idea of taking context into account in word representation was not entirely new, and is to be found in count-based approaches, such as Erk and Padó (2010). The authors start by computing token count-based representations—i.e., for each token t, they produce a sparse vector $\vec{t}$ where each component t_d corresponds to the number of times a given word w_d type is attested in the sentential context of the token t. Each of these token vectors is taken as an exemplar, and a similarity metric such as cosine similarity or Jaccard index allows us to restrict token exemplars on the basis of context. Another example is the work of Reisinger and Mooney (2010), where token vectors are computed from 10-word windows instead of sentential contexts. Unlike Erk and Padó (2010), who produce context-specific token representation, Reisinger and Mooney (2010) then

cluster token representations and use cluster centroids as the representation for one prototypical usage of the word.

Perhaps the most successful modern neural contextual architecture is the BERT model of Devlin et al. (2019). The BERT model is based on the Transformer architecture of Vaswani et al. (2017), and has proven to be highly efficient on a wide variety of NLP tasks. We will look at this model at greater length in Section 1.2.2. Simply put, BERT is trained to solve a "fill-in-the-blank" sequence denoising task. Variations on the architecture of BERT have since blossomed. Among these, we can cite the BART model of Lewis et al. (2020), which consists in an encoder-decoder extension of BERT, allowing it to be trained on a greater variety of sequence denoising sub-tasks, such as adding in missing tokens and removing superfluous ones. Another is the ELECTRA architecture of Clark, Luong, et al. (2020): this model is derived from the Generative Adversarial Net (GAN) architecture (Goodfellow et al., 2014), where two sub-modules compete against one another; we will discuss this architecture in Section 1.2.2.

In parallel to works on contextual word embeddings, another closely related focus of research has been the development of large pre-trained language models. The flagship here is represented by the various iterations of the General Pre-trained Transformer of the OpenAI team (Radford, 2018; Radford et al., 2019), although larger language models have been developed by other players. In many cases, comparisons have been made between large language models like GPT and contextual embeddings like BERT or ELMo. While the work of Bengio et al. (2003) provides historical reasons for this, it is worth pointing out that these models are intrinsically different in how they are designed and trained: they are designed as models that predict the next word in a continuous span of text;

that is to say, language models are primarily models for text generation, rather than models of word tokens in specific contexts. In practice, large pre-trained language models are commonly used to derive embeddings of words in context, and are tested on the same benchmarks and datasets as contextual embedding models like BERT or ELMo. It therefore stands to reason that large pre-trained language models are also relevant to distributional semantics studies.

1.2 Major examples of Distributional Semantics Model architectures

We have sketched out a chronological overview of the developments of the theory of distributional semantics in the previous section. One of the elements that we highlighted is their great diversity: from count matrices to deep neural networks, the variety of approaches that have been deployed is dizzying. In the present section, we provide tangible elements on what this variation concretely entails, by describing in depth three famous DSM architectures: word2vec in Section 1.2.1, BERT in Section 1.2.2 and ELECTRA in Section 1.2.3.

1.2.1 The word2vec model

The word2vec model of Mikolov, K. Chen, et al. (2013) is well-known in the NLP community. Its fame is due to its efficiency—both in terms of computation and downstream applications. As we have pointed out earlier, the term of "word2vec model" is misleading, as it conflates two distinct but related shallow neural network architectures, which are illustrated in Figure 1.1.

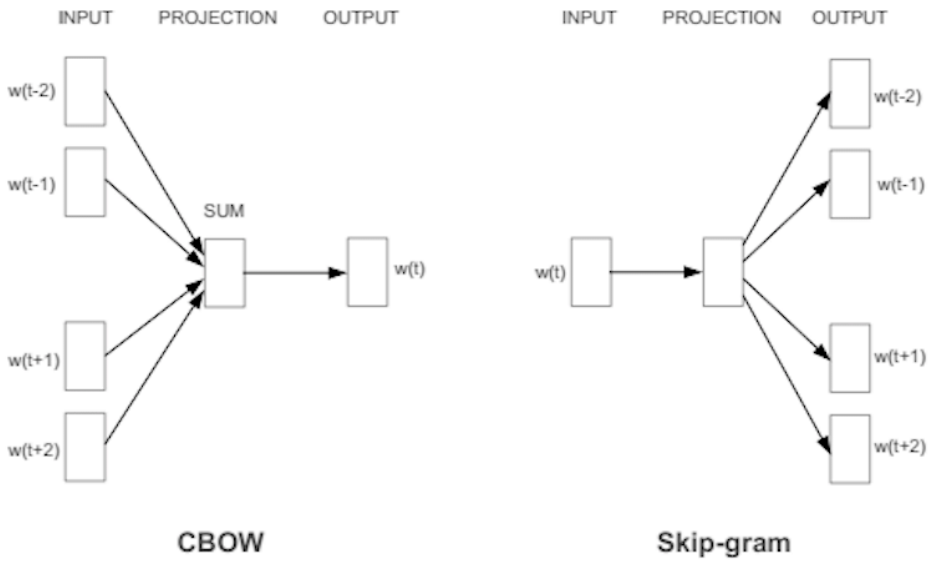


Figure 1.1: CBOW and Skip-gram architectures; taken from Mikolov, K. Chen, et al. (2013)

The first of these, dubbed CBOW (short for "Continuous Bag of Words") consists in using the context around a word as the input, and trying to predict the target word for this context. The second architecture is known as skip-gram, and can be thought of as the mirror image of CBOW: using the target word as sole input, the model will be tasked with predicting each word of the context one after the other.

CBOW is comprised of one linear projection W^P of shape $[V \times d]$ and a log-linear classifier W^C of shape $[d \times V]$, Here, V is the size of the vocabulary and d is the number of dimensions, typically ranging from 50 to 300 or more. All context words are first transformed as one-hot vectors, then down-projected in a vector space $\mathbb{R}^d$ using the projection W^P, as the use of one-hot vectors allows us to transform a vocabulary index in a vector. Given a word w_i, and its index i

in the vocabulary, we define

$$\vec{w}_i = (c_1, \ \ldots, \ c_d)$$
$$c_j = \begin{cases} 1 & \text{if } j = i \\ 0 & \text{otherwise} \end{cases} \tag{1.5}$$

Therefore the down-projection using W^P corresponds to selecting the i$^{\text{th}}$ row of W^P and the row-vectors of the W^P matrix therefore are the actual word2vec embeddings used in downstream applications.

The average of all projected vectors is then used as input for the log-linear classifier W^C itself. The context is determined by a *window size*: a window of size t will correspond to selecting the t tokens before the target and the t tokens after it. In a more formal and succinct manner, we can describe the entire algorithm of CBOW as:

$$\vec{h}_i = \frac{1}{2t}\left(\sum_{j=i-1-t}^{i-1} W^P \cdot \vec{w}_j + \sum_{j=i+1}^{i+1+t} W^P \cdot \vec{w}_j\right)$$
$$\hat{y}_i = \text{softmax}\left(W^C \cdot \vec{h}_i\right)$$

It is worth noting that traditionally the classifier W^C only serves for training, and is discarded afterwards.

In terms of actual training, the use of a log-classifier entails that the objective to optimize is a log-likelihood maximization. More precisely, the training objective is to maximize the log-likelihood of the probability of predicting the target word based on its context, $\log p(w_t|c)$. The probability distribution is explicitly

derived by the softmax function:

$$\begin{aligned}\hat{y} &= \text{softmax}\left(W^C \cdot \vec{h}\right) \\ &= \frac{\exp\left(W^C_j \cdot \vec{h}\right)}{\sum_{j'} \exp\left(W^C_{j'} \cdot \vec{h}\right)}\end{aligned} \tag{1.6}$$

where W^C_j is the j$^{\text{th}}$ column vector of the matrix W^C. The components of $\hat{y}$ sum to 1, and therefore define a probability distribution for each element of the vocabulary; in other words, $\hat{y}$ is a vector of dimension V. In practice, as maximizing the probability of predicting the current word knowing the context is equivalent to minimizing the negative log-likelihood for that word, models are trained to minimize the negative log-likelihood instead:

$$\mathscr{L}(\hat{y}, w_i) = -\log \hat{y}_i \tag{1.7}$$

The skip-gram architecture, as we have already pointed out, can be thought of as a "reversed" CBOW architecture, since the aim is to predict the context based on the target word. The parameters used in skip-gram models and their shape are therefore very reminiscent of what we see in CBOW models. To train a skip-gram model over a datapoint, we first project the target word using a linear projection W^P of shape $[V \times d]$, and use a classifier to predict each word in the context W^C of shape $[d \times V]$. As with CBOW, we derive vectors from the W^P matrix; likewise, a probability distribution is inferred by applying a softmax after

the classifier's output. Or more formally:

$$\begin{aligned}\vec{h}_i &= W^P \cdot \vec{w}_i \\ \hat{y}_i &= \text{softmax}\left(W^C \cdot \vec{h}_i\right)\end{aligned} \tag{1.8}$$

where $\vec{w}_i$ is the one-hot vector for the target word w_i.

The next major difference between skip-gram and CBOW is to be found in their loss functions. As all context words are to be predicted using the same input word, we aim to maximize the joint probability of all context words knowing the target word:

$$p(w_{i-t},\ \ldots,\ w_{i+t}|w_i)$$

In practice, we estimate this probability using the chain rule:

$$\prod_{j=i-t}^{i-1} p(w_j|w_i) \times \prod_{j=i+1}^{i+t} p(w_j|w_i)$$

For efficiency considerations, it is more efficient to perform these computations in log-space, as we can transform the product into a sum by maximizing a log-likelihood instead. Hence the model is trained by minimizing the joint negative log-likelihood of each context word:

$$\mathscr{L}(\hat{y}, \langle w_{i-t}, \ldots, w_{i+t}\rangle) = -\left(\sum_{j=i-t}^{i-1} \log \hat{y}_j + \sum_{j=i+1}^{i+t} \log \hat{y}_j\right) \tag{1.9}$$

Mikolov, K. Chen, et al. (2013) observe that obtaining the multinomial distribution of the skip-gram model is computationally inefficient. One may instead

consider training the classifier to distinguish whether a given context is attested for a given word. Goldberg and Levy (2014) provide a formal approach which we retrace here. Let D^+ the set of all pairs of words w and contexts c that occurs in our dataset, and let D^- a set of *negative examples* (also pairs of words and contexts), such that $D^+ \cap D^- = \emptyset$, i.e., that no negative example pair of word and context is attested in our base dataset. Let $p(X = 1|w, c)$ the probability that the pair $\langle w,\ c\rangle$ is present in the base dataset D^+. We can redefine the classifier's objective as maximizing:

$$p(X = 1|w, c)\ \forall \langle w,\ c\rangle \in D^+$$

and minimizing:

$$p(X = 1|w, c)\ \forall \langle w,\ c\rangle \in D^-$$

As we are dealing with a binomial variable X, minimizing $p(X = 1|w, c)$ is equivalent to maximizing $1 - p(X = 1|w, c)$. The objective can therefore be re-framed as maximizing:

$$\prod_{\langle w,\ c\rangle \in D^+} p(X = 1|w, c) \prod_{\langle w,\ c\rangle \in D^-} \big(1 - p(X = 1|w, c)\big)$$

To perform this re-framing, one needs to amend the network architecture. Unlike previously, an explicit probability distribution over the full vocabulary is no longer required; hence the computationally costly softmax function can be replaced with a sigmoid function:

$$\sigma(y) = \frac{1}{1 + \exp(-y)} \tag{1.10}$$

Since $1-\sigma(y)=\sigma(-y)$, we can compute the score for $\langle w_j,\ w_i\rangle$ simply using:

$$\sigma\left(W_j^C\cdot\left(W^P w_i\right)\right)$$

for pairs drawn from D^+, and:

$$\sigma\left(-W_j^C\cdot\left(W^P w_i\right)\right)$$

for pairs drawn from D^-. To limit computation complexity, the latter term is estimated using only k negative examples. This yields the following loss function to minimize:

$$-\log p(w_j|w_i)=-\log\sigma\left(W_j^C\cdot\left(W^P w_i\right)\right)+\sum_{w_n\in N}\sigma\left(-W_n^C\cdot\left(W^P w_i\right)\right) \tag{1.11}$$

where $N=\{\langle w_i,c_m\rangle,\cdots\langle w_i,c_{m+k}\rangle\};N\subset D^-$ is a set of k negative examples sampled for w_i.

As an alternative to negative sampling, Mikolov, K. Chen, et al. (2013) introduce a *hierarchical softmax* which encodes probabilities using a binary tree structure. Leaves correspond to words in the vocabulary, and each node n stores the relative probabilities of its children using a dedicated weight vector $\vec{v}_n$. More precisely, let $\mathscr{P}(w_i)=\{n_0,\ \ldots\ ,n_{w_i}\}$ be the path from the root node n_0 to the leaf node n_{w_i} for word w_i. One can redefine the output probability as

$$p(w_j|\vec{h}_i)=\prod_{n\in\mathscr{P}(w_j)}\sigma\left(\vec{v}_n\cdot\vec{h}_i\right) \tag{1.12}$$

Lastly, Mikolov, K. Chen, et al. (2013) also proposed to avoid issues arising with class imbalance (also known as "Zipf's law") by dropping words from the training set based on their frequency. They define the *subsampling* rate:

$$P(w_i) = 1 - \sqrt{\frac{t}{f(w_i)}} \tag{1.13}$$

where t is a "temperature" hyperparameter (typically 10^{-5}) and $f(w)$ is the frequency of word w. This subsampling rate defines the probability that any token will be discarded based on the frequency of its type w_i.

1.2.2 The BERT model

We now turn to a second embedding architecture, namely BERT.

BERT is a deep neural network based on the Transformer architecture (Vaswani et al., 2017).[1] Formally, a Transformer model is a stack of *sublayers*. Each sublayer is centered around a specific sublayer function. Sublayer functions can either be *feed-forward* sub-modules or *multi-head attention* sub-modules.

The former are perceptrons of the form:

$$\vec{y} = W_O^{\mathscr{F}} \cdot \left(\Phi\left(W_I^{\mathscr{F}} \cdot \vec{x} + b_I^{\mathscr{F}}\right)\right) + b_O^{\mathscr{F}} \tag{1.14}$$

where Φ is some non-linear function such as ReLU or GELU (Hendrycks and Gimpel, 2016). The input and output dimensions are equal, whereas the inner layer dimension is larger.

[1] The original architecture of Vaswani et al. (2017) is a sequence-to-sequence model, comprising both an encoder and a decoder. In practice, a "Transformer" architecture is often understood as the encoder from the sequence-to-sequence architecture of Vaswani et al. (2017), and it is this acception we adopt throughout this dissertation.

Multi-head attention mechanisms (MHA) are concatenations of individual scaled-dot *attention heads*:

$$\vec{y}_t = W_O^H \cdot \left(\bigoplus_h A_t^h \right) + b_O^H \tag{1.15}$$

$$A^h = \text{softmax}\left(\frac{Q \cdot K^T}{\sqrt{d_v}} \right) \cdot \left(W_V^h \cdot \begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_n \end{bmatrix} + b_V^h \right) \tag{1.16}$$

The *attention weights* a_t^h are computed by means of a softmax dot-product between keys K and queries Q projections of all the input layer representations for the full sequence. In other words, the product $\text{softmax}(Q \cdot K^T / \sqrt{d_v})$ can be thought of as weights in an average over the transformations $W_V^h \cdot \begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_n \end{bmatrix} + b_V^h$. As such, we can provide a step-by-step derivation of multi-head attention outputs as a weighted sum of value vectors. Here, we heavily rely on the presentation by Kobayashi et al. (2020).

For simplicity, let:

$$V = \left(W_V^h \begin{bmatrix} \vec{x}_1 \\ \vdots \\ \vec{x}_n \end{bmatrix} + b_V^h \right)$$

$$\mathscr{W} = \text{softmax}\left(\frac{Q \cdot K^T}{\sqrt{d_v}} \right)$$

with $\oplus$ representing vector concatenation. The attention weights $\mathscr{W}$ define a single matrix of shape $[S_Q \times S_V]$, where S_Q is the number of query vectors and S_V

is the number of vectors to which we pay attention.[2] Each cell of this $\mathscr{W}$ matrix corresponds to one attention weight, i.e., the similarity between a query vector and a key vector, computed as a scaled dot-product smoothed with a softmax $\text{softmax}\left(\langle Q_q \cdot K_k \rangle / \sqrt{d_v}\right)$. Since by definition of the matrix product:

$$(M \cdot N)_{i,j} = \sum_k M_{i,k} \times N_{k,j}$$

we get that the cell at row q and column c of the attention head output $A^h_{q,c}$ will be equal to the weighted sum of the c^{th} component of the k linearly mapped vectors:

$$A^h_{q,c} = \sum_k \mathscr{W}_{q,k} \times V_{k,c}$$

Hence the q^{th} row of the attention head output matrix A^h can be rewritten as a weighted sum of linearly mapped vectors:

$$A^h_q = \sum_k \mathscr{W}_{q,k} \times V_k$$

To arrive at our unbiased output $\vec{\mathscr{H}}_l$ and the corresponding bias term $b_{\mathscr{H}_l}$, note that due to the softmax, the attention weights sum to one: $\sum_k \mathscr{W}_{q,k} = 1$, hence we can rewrite the weighted sum A^h_q to extract the head specific bias:

$$\begin{aligned} A^h_q &= \sum_k \mathscr{W}_{q,k} \times \left(W^h_V \vec{x}_k + b^h_V\right) \\ &= \sum_k \left(\mathscr{W}_{q,k} \times W^h_V \vec{x}_k\right) + \sum_k \mathscr{W}_{q,k} \times b^h_V \\ &= b^h_V + \sum_k \mathscr{W}_{q,k} \times W^h_V \vec{x}_k \end{aligned}$$

[2] For BERT and self-attention in general, $S_Q = S_V$.

From there on, one can pass the head bias b_V^h and the weighted average $\sum_k \mathcal{W}_{q,k} \times W_V^h \vec{x}_k$ through the output projection matrix of the MHA module $W_O^{H_l}$ to match the expressions above.

Going back to the definition of a Transformer architecture, after each sublayer function S (either MHA or feed-forward), a *residual connection* (i.e., adding the input to the output) and a *layer normalization* (Ba et al., 2016, henceforth "LayerNorm") are applied:

$$\vec{y} = \gamma_\lambda \odot \frac{(S(\vec{x}) + \vec{x}) - \vec{\mu}_\lambda}{\sigma_\lambda} + \beta_\lambda \tag{1.17}$$

with $\odot$ representing element-wise multiplication. The gain γ_λ and bias β_λ correspond to learned parameters. $\vec{\mu}_\lambda = \mu_\lambda \cdot \vec{1}$ is the vector $\vec{1} = (1 \ \ldots \ 1)$ scaled by the mean component value μ_λ of the input vector $(\vec{x}) + \vec{x}$, and σ_λ is the standard deviation of the component values of the input vector.

Two sublayers are stacked into a single Transformer *layer*: the first corresponds to a multi-head attention, and the second to a feed-forward. To kick-start the propagation through the layers, a static representation is fed into the first layer. This initial input corresponds to the sum of a simple word lookup static embedding and a sinusoidal positional encoding vector, where the components are defined using the sine and cosine functions:

$$\begin{aligned} \mathrm{p}(t)_{2i} &= \sin\left(\frac{t}{10000^{2i/d}}\right) \\ \mathrm{p}(t)_{2i+1} &= \cos\left(\frac{t}{10000^{2i/d}}\right) \end{aligned} \tag{1.18}$$

where d is a hyperparameter defining the output and hidden representations

vector sizes, and t corresponds to the timestep, i.e., the index of the token in the input. These positional encodings are introduced so as to inform Transformer models of word order, as it would otherwise not be retrievable from the set of vectors they receive as inputs. Note that the use of parameters dedicated to specific positions entails that Transformers are in principle incapable of modeling sentences past a certain length. In variants such as the BERT model of Devlin et al. (2019), there are additional terms to this static input to encode the segment the current token belongs to, as well as an additional LayerNorm before the very first sublayer. Other variants also encode positions by means of an offset in the computation of attention weights (C.-Z. A. Huang et al., 2018; P. Shaw et al., 2018).

Lastly, Transformer models generally use *word-pieces*, rather than raw word types, to convert text into lookup indices in an embedding table. Word-pieces correspond to the output of sub-word tokenization algorithms, i.e., algorithms that chunk sequences of texts in tokens that do not correspond to word types. One of the simplest such algorithms is Byte Pair Encoding (BPE). To initialize it, we define a set of known pieces P as the set of all characters[3] attested in the corpus we wish to tokenize. We then repeat the following three instructions until the set P reaches some predetermined size (often 30 000 word-pieces):

1. tokenize the full dataset, using the largest word pieces available from the current set of tokens P

2. find the most frequent pair of word-pieces $\langle m \in P, n \in P \rangle$ to appear sequentially in the current tokenized dataset

[3]Or bytes, in some variants.

3. add the merged pair mn to the current set of word pieces: $P \leftarrow P \cup \{mn\}$

As such, the BPE algorithm preferentially provides distinct token representations for frequent sequences of characters: hence common words ("*is*", "*the*", "*that*", etc.) will receive distinct token representations that match their word types, whereas commonplace orthographic realizations of morphological exponents ("*-s*", "*-ing*", etc.) will likely correspond to sub-word token representations.

Aside from its Transformer architecture, the BERT model of Devlin et al. (2019) also distinguished itself by its novel training objective. BERT is trained to solve two tasks simultaneously: a word-level objective called "masked language model" (MLM) and sentence-level objective dubbed "next sentence prediction" (NSP).

The MLM word-level objective for BERT is tied to earlier experiments from psychology, and especially to a methodology referred to as the Cloze test (Taylor, 1953). This experimental protocol, also known as "Gap-Fill," "Cloze deletion test," "Fill in the blanks" and by many other names, consists in blanking out a token (or group of tokens) in a given sentence and requiring subjects to fill in said blanks with plausible words. This task has mostly been used as a learning exercise to assess reading proficiency and mastery of grammar. BERT is a Transformer architecture trained to perform well on the Cloze test—the intuition being that parameters able to correctly solve a reading exercise are likely to be decent textual representations. To reformulate the Cloze test as the MLM training objective, it is framed as a prediction task, where the model must correctly predict the token that has been blanked out. As such, the prediction can be done with a simple log-linear classifier, defined as a softmax over a vocabulary projection to which the embedding of the blanked-out item is fed. This process is quite similar

to what we described for word2vec in the previous section, which is why we do not detail the prediction itself further.

Concretely, Devlin et al. (2019) devise the following procedure. The model first randomly selects 15% of the word-pieces, which will be fed to the softmax prediction layer. 80% of the randomly selected items (12% of the word-pieces in total) will be replaced by a special token `[MASK]`, representing a blank to be filled. 10% of the randomly selected word-pieces (1.5%) are replaced by a word at random; this is done to mitigate the mismatch between pre-training and usage of the model further down the line, since the special token `[MASK]` will likely never be encountered during downstream applications or fine-tuning. Lastly, 10% of the randomly selected word-pieces (1.5%) are left as is, in order to "bias the representation towards the actual observed word."

The second training objective of BERT, NSP, consists in predicting whether a sentence immediately follows another in the corpus. This objective entails that BERT can only be trained on a corpus of coherent documents, and not on corpora composed of shuffled sentences, and that inputs to the BERT model during training are comprised of two sentences. This second objective has been said to be helpful in Question Answering (QA) and Natural Language Inference (NLI) downstream tasks.

The NSP objective can be naturally implemented as a binary classification, using paired sentences $\langle S_A, S_B \rangle$ as inputs. The model needs to discriminate cases where S_A is immediately followed by S_B in the training corpus from cases where S_A and S_B were randomly selected. This can be implemented as a sigmoid-based classifier quite similar to the negative-sampling reformulation we discussed earlier for word2vec. In practice, sentences are presented as a contiguous span of

tokens to the system, using two special tokens [CLS] and [SEP] as separators. More concretely, if $S_A = w_1^A, ..., w_n^A$ and $S_B = w_1^B, ..., w_m^B$, the system will receive the following sequence as input:

$$\texttt{[CLS]}, w_1^A, \ldots, w_n^A, \texttt{[SEP]}, w_1^B, \ldots, w_m^B, \texttt{[SEP]}$$

To further facilitate the model's ability to distinguish two sentences, learned *segment encodings* for S_A and S_B are added respectively to the two sentences involved. As a concrete example, if the initial training example was "*My dog barks. It is a pooch.*," the actual input would correspond to the following sequence of vectors:

$$\begin{aligned}
\vec{\texttt{[CLS]}} + \vec{p}(0) + \vec{\texttt{seg}}_A,\ \vec{My} + \vec{p}(1) + \vec{\texttt{seg}}_A,\\
\vec{dog} + \vec{p}(2) + \vec{\texttt{seg}}_A,\ \vec{barks} + \vec{p}(3) + \vec{\texttt{seg}}_A,\\
\vec{.} + \vec{p}(4) + \vec{\texttt{seg}}_A,\ \vec{\texttt{[SEP]}} + \vec{p}(5) + \vec{\texttt{seg}}_A,\\
\vec{It} + \vec{p}(6) + \vec{\texttt{seg}}_B,\ \vec{is} + \vec{p}(7) + \vec{\texttt{seg}}_B,\\
\vec{a} + \vec{p}(8) + \vec{\texttt{seg}}_B,\ \vec{pooch} + \vec{p}(9) + \vec{\texttt{seg}}_B,\\
\vec{.} + \vec{p}(10) + \vec{\texttt{seg}}_B,\ \vec{\texttt{[SEP]}} + \vec{p}(11) + \vec{\texttt{seg}}_B
\end{aligned}$$

where $\vec{p}(i)$ are the positional encodings, and $\vec{\texttt{seg}}_A$ and $\vec{\texttt{seg}}_B$ are the segment encodings. Lastly, although not specified in the paper, the sentence prediction only uses the [CLS] token for its prediction.

1.2.3 The ELECTRA model

The third and last architecture we describe has not encountered the same degree of fame as BERT and word2vec. It nonetheless both combines some of the features we have discussed above as well as introduced a number of design choices that starkly contrast with the two architectures we have just reviewed.

Conceptually, ELECTRA is a Transformer-based contextual embedding model with an architecture inspired from GANs (Goodfellow et al., 2014). A GAN consists in two modules that are pitted one against the other. The *Discriminator* module (a.k.a the "Critic") is trained on a binary classification task: distinguish real datapoints, attested in some training set, from fake ones. These fake datapoints are those created by the *Generator* module (also called the "Actor").[4] The losses of these two modules are adversarial: maximizing the loss of either module entails minimizing the loss of the other. More formally, the objective for the discriminator D and the generator D can be described as:

$$\min_{\theta_G} \max_{\theta_D} V(G_{\theta_G}, D_{\theta_D}) = \mathbb{E}_{x\sim I}\left(D_{\theta_D}(x)\right) + \mathbb{E}_{x\sim R}\left(D_{\theta_D}\left(G_{\theta_G}(z)\right)\right) \tag{1.19}$$

where I is the input dataset, G and D refer to the Generator and Discriminator modules, respectively parameterized with θ_G and θ_D, and R is a random variable (generally multivariate standard Gaussian) that guarantees the Generator is not deterministic and can produce multiple fake datapoints.

This objective essentially relies on each module forcing the other to improve itself. At the beginning of the training, the Generator's outputs are more or less

[4] Hence why GANs are also called "Actor–Critic models."

random, hence the Discriminator can easily learn to distinguish a fake output $G_{\theta_G}(z)$ from an actual datapoint x, which leads the entire GAN to minimize the first term of the objective function, $\mathbb{E}_{x\sim I}\left(D_{\theta_D}(x)\right)$. Once the Discriminator is able to discriminate the Generator's output consistently, the only way to improve on the objective function is to maximize the second term $\mathbb{E}_{x\sim R}\left(D_{\theta_D}\left(G_{\theta_G}(z)\right)\right)$, i.e., to have the Generator produce more plausible outputs that are more likely to fool the Discriminator. As the Generator learns to produce more convincing outputs, the GAN is incentivized to also update the Discriminator, so that it remains able to distinguish real from fake datapoints despite the fake datapoints looking more similar to the real ones. In all, this leads to an "arms race" where both modules are forced to reach higher performances on their respective sub-objectives.

The practical application of this GAN architecture is that the Generator is trained to transform some random noise $z \sim R$ into likely datapoints. At their inception, GANs were presented as an algorithm to generate images resembling those listed in the input dataset I. The objective in Equation (1.19) is however more broadly applicable, and only requires that some gradient can be computed from the Discriminator and passed on to the Generator. In practice, this can be done by having the Generator produce continuous outputs, or by using a Reinforcement Learning algorithm. As such, GANs have also been applied to generate text (Gulrajani et al., 2017; Nie et al., 2019, e.g.).

The Generator and Discriminator modules in ELECTRA are both implemented as Transformers. Unlike classical GANs, the Generator in ELECTRA is trained on a simplified MLM objective rather than on a noise-transformation task as we saw above. The Discriminator, on the other hand, has to distinguish tokens that have been unmasked by the generator using a sigmoid-based classifier, much like

what we saw with the negative sampling in word2vec. As such, ELECTRA is not exactly a GAN. The authors' reasoning is that while methods to propagate gradient through a discrete sampling exist, they tend to be noisy. As such, they may under-perform in the case of a purely adversarial setup, where it is crucial that the Generator correctly factors in the response of the Discriminator. Note that the objective of the Discriminator nonetheless depends on not being fooled by the Generator: any unmasked token that is not flagged as such penalizes the Discriminator. As such, a similar dynamic can be found in ELECTRA and in GANs: at the beginning of the training, the Discriminator's task is relatively easy as the Generator's unmasking outputs are more or less random, and as the Generator reaches higher performance, it incentivizes the Discriminator to do as well. The key difference is that the Generator gets its gradient directly from the input dataset rather than through the generator.

The ELECTRA architecture distinguishes itself from most applications of GANs in that its main by-product is intended to be the Discriminator. In practice, the Generator is much smaller than the Discriminator, in terms of number of parameters. Another key point to note is that the Discriminator's objective, classifying tokens according to whether or not they have been filled in by the Generator, is conceptually very similar to the negative sampling performed in some word2vec models. In essence, the task at hand is to determine whether or not the input word-type is attested in the given context.[5] Given that the Discriminator embeddings are those intended for use on downstream applications, it makes sense to consider ELECTRA as an approach to perform negative sampling with a BERT-

[5]Consistent with this analysis, tokens correctly retrieved by the Generator—i.e., tokens produced by the Generator that correspond to the actual input token prior to masking—are associated with the same label as tokens that were not masked in the first place.

like model.

1.2.4 Comparing the three architectures

Our brief technical review of the DSM architectures of word2vec, BERT and ELECTRA has stressed these three models are formally quite distinct from one another.

At one end of the spectrum, word2vec is a shallow neural network, that can be thought of as a 2-layer perceptron. Most of the technical baggage associated with it consists in computational optimizations: how to avoid using the softmax function, for instance, gave rise to two distinct strategies: negative sampling and hierarchical softmax.

BERT's approach is the opposite. It employs an impressive number of parameters and sub-modules, computes many hidden representations, and relies on two distinct gradient computation mechanisms for its training. If word2vec is a lightweight DSM, then the design of BERT is very much geared towards making its performances on downstream application as impressive as possible. In particular, the finetuning approach suggested by Devlin et al. (2019) entails that downstream applications benefit from the numerous weight parameters of the model.

At the other end of the spectrum, we find ELECTRA. This model is literally twice as complex as BERT: in terms of layout, it contains two stacks of Transformer layers, instead of one. It relies on a complex training dynamic, pitting two sub-modules one against the other. Nonetheless, despite its even greater complexity, the ELECTRA architecture employs concepts similar to those we find in

the static word2vec architectures, although the means it uses vastly differ.

In all, these three DSMs illustrate the variety of approaches that can be construed as implementations of distributional semantics. One therefore needs to ponder whether such distinct models should be conceived as similar linguistic objects. Lastly, it is worth pointing out that the three models we have considered here also have a number of similarities: all use gradient descent to estimate their parameters. The burning issue of the heterogeneity of distributional semantics models is therefore even more significant if we consider models derived from completely distinct approaches, such as SVD-based models like the Latent Semantic Analysis of Landauer and Dumais (1997).

1.3 Heterogeneity of Distributional Semantics Models Evaluation Protocols

The chronological survey conducted in Section 1.1 and the characterization of three DSM architectures in Section 1.2 both highlighted how the models that are considered as pertaining to distributional semantics are numerous and varied. This stems from the relatively flexible theoretical ground of these models: although they are historically connected, there are very few testable propositions we can derive from the general formulation of the theoretical framework of Harris (1954). In fact, although a clear parallel is well established between static embeddings (e.g., word2vec) and the theory of distributional semantics (e.g., see the overviews by Lenci (2018) or Boleda (2020)), the same has not been held clearly for contextual embeddings. For instance Westera and Boleda (2019)

explicitly consider only "context-invariant" representations as distributional semantics. This may be because of the original remark by Peters et al. (2018) that contextual embeddings are "functions of the entire input sentence"—whereas static embeddings map words to vectors. Intuitively, the setups required to handle static and contextual embeddings differ, and thus it is legitimate to consider them as distinct, though related, theoretical constructs. This discrepancy warrants that we study more precisely how different these models are in practice, if we wish to understand the characteristics of distributional semantics as a semantic theory.

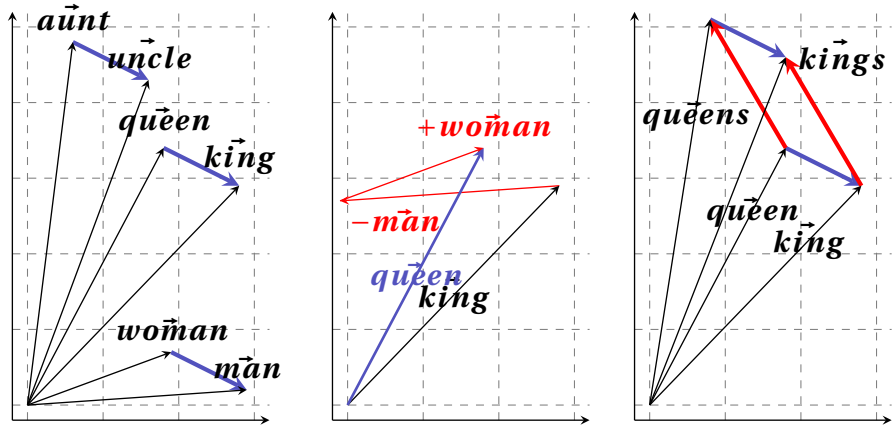
Different types of investigative methodology have been proposed in the literature: statistical studies on the structure of the vector space (see Section 1.3.1), classifier probe-based studies (see Section 1.3.2), or even attention visualization techniques (see Section 1.3.3). While we will focus primarily on these three groups of methodologies, other approaches exist: one such example would be the comparisons of model variants. For instance, Peters et al. (2018) analyzed through an extensive ablation study of ELMo what information is captured by each layer of their architecture. Likewise, Devlin et al. (2019) discussed what part of their BERT architecture is critical to the performances they obtained, comparing pre-training objectives, number of layers and training duration. A similar trend was pursued by Mikolov, K. Chen, et al. (2013), as they compared the time complexities of various word embedding algorithms. Another related methodological approach consists in benchmarking: for instance, BERT-based models have significantly increased state-of-the-art over the GLUE benchmark for natural language understanding (A. Wang et al., 2019) and most of the best scoring models for this benchmark include or elaborate on BERT. Such methodologies

are not especially tied to word embeddings, and have been applied in numerous neural network applications besides embedding architectures. We therefore set them aside and focus on more directly relevant experimental protocols.

1.3.1 Vector Space Structure

Broadly speaking, the literature on distributional semantics has put forth and discussed many mathematical properties of embeddings. An important element to take into account here is the work of Levy and Goldberg (2014b), which highlights that word vector spaces such as those derived from word2vec are equivalent to those we can infer from count-based matrices. We have briefly discussed this work in Section 1.1.4: the loss function of the word2vec skip-gram architecture with negative sampling was shown to implicitly factorize a PMI-weighted occurrence count matrix. In detail, their argument goes as follows: the word2vec architecture (cf. Section 1.2.1) is comprised of two matrices W^P and W^C, from which word and context representations are sampled. It therefore makes sense to consider what their product $M = W^P \cdot W^C$ corresponds to, i.e., what matrix M they implicitly factorize. Given that each cell of M_{ij} corresponds to the dot product between a word representation $\vec{w}_i$ and a context representation $\vec{c}_j$, one can consider what constraints are placed by the loss being optimized on this dot product term $\vec{w}_i \cdot \vec{c}_j$. They do so by comparing the partial derivative of the loss function with respect to this dot product and studying how it can be made to equate 0. They arrive at the reformulation:

$$\vec{w}_i \cdot \vec{c}_j = \text{PMI}(w_i, c_j) - \log k \tag{1.20}$$



(a) Stability of semantic contrasts

(b) Semantic nature of vector addition

(c) Paradigm encoding

Figure 1.2: Linguistic Regularity in distributional vector spaces

where k is the number of negative examples for the negative sampling, and $\text{PMI}(w_i, c_j)$ is the pointwise mutual information between a word w_i and a context c_j. As we noted above in our chronological overview, such weighting, or variants thereof such as the positive PMI, were until word2vec rather frequent in the NLP community, particularly in studies focusing on word similarity modeling. This entails that there is a strong connection between the vector spaces described by models similar to word2vec and those described by count-based methods.

One trait of traditional DSMs that is very often encountered, discussed and exploited in the literature is the fact that the relative positions of embeddings are not random. Early vector space models, by design, required that word with similar meanings lie near one another, as in the works of Landauer and Dumais (1997) or Bengio et al. (2003); as a consequence, regions of the vectors space are expected to describe coherent semantic fields.

Vectors encoding contrasts between words are furthermore expected to be

coherent, as shown by Mikolov, Yih, et al. (2013). A visual illustration of this fact is depicted in Figure 1.2. The idea is that word vectors should be a linear composition of their relevant semantic aspects: in short, we expect that the vectors for "*king*", "*queen*", "*prince*" and "*princess*" are such that $\vec{princess} \approx \vec{king} - \vec{queen} + \vec{prince}$. This core characteristic has led to the rise of analogy as an evaluation methodology for word embeddings and distributional semantic models. For instance, Gladkova et al. (2016) have proposed a balanced analogy dataset called BATS. This dataset is composed of four general categories of analogies (inflectional morphology, derivational morphology, lexicographic semantics and encyclopedic semantics), each of which contains 10 different specific analogical relations (e.g., noun singular–plural). Each relation contains 50 different pairs; any two pairs from a given relation can therefore be used as an evaluation datapoint.

Datasets like that of Gladkova et al. (2016) have contributed to establishing formal analogy, and vector offsets by extension, as a popular method for investigating distributional semantics model. Studies have therefore set to exploit this property for different goals. We can quote, for instance, the work of Bolukbasi et al. (2016), which looks into whether linear offsets that encode social stereotypes can be found in word2vec models. Their research stresses that in the widely distributed Google-News word2vec model, the difference between $\vec{man}$ and $\vec{computer\_programmer}$ is very close to the difference between $\vec{woman}$ and $\vec{homemaker}$. In other words, social stereotypes—in particular gender biases—are also encoded as linear offsets in word embedding models. Bolukbasi et al. (2016) also attempt to identify subspaces: not only offsets, but hyperplanes that would encode specific unwanted contrasts.

Another example of a work using vector offsets would be Bonami and Paperno (2018). These authors reason that, if semantic differences are encoded in vector offsets, it makes sense to assume that the systematicity of said differences should impact the systematicity of this offset—relations that are more systematic, such as the alternation between bare English verbs and their gerunds (*sing–singing* or *dance–dancing*), should be expressed more consistently than relations which are less systematic, such as that between a verb and an associated location noun (*hospitalize–hospital* or *dine–diner*). On the other hand, morphologists such as Stump (1998) have stressed that derivational morphology is less semantically systematic than inflectional morphology. That is to say, in the case of inflection (e.g., *sing–singing*), we should expect the offsets between related vectors to vary less; on the other hand the less systematical relations in derivational morphology (e.g., *dine–diner*) should lead to a wider variation of vector offsets.

The expectation that semantic contrasts should be reflected in vector offsets can be re-framed as a problem of linear dependence, as was done by Arora et al. (2016). The general idea underlying this work is that each word type vector $\vec{w}$ computed by neural models can be expressed as a linear combination of vectors representing its senses $\sigma_{w1} \ldots \sigma_{wn}$, or more formally: $\vec{w} \approx \alpha_1 \cdot \vec{\sigma_{w1}} + \cdots + \alpha_n \cdot \vec{\sigma_{wn}}$. Their argument focuses on log-linear architectures, like word2vec (without negative sampling) or GloVe. They start with the assumption that the context vector encodes a probability over the whole vocabulary describing which words are likely to occur. By applying Bayes' rule and plugging the equations for a log-linear classifier, Arora et al. (2016) identify a linear transformation between context words and target word. This linear assumption justifies their treatment of word type vectors as linearly decomposable along any partition of their contexts

of occurrences, as summation is commutative with respect to linear transformation. In other words, one can partition contexts of a word w in various sets, accord to the sense of w they describe; each of these sets of contexts will corresponds to a linear factor of the total embedding $\vec{w}$.

As a consequence, one can expect that there exist a set of basis vectors that describe the entire word embedding space, i.e., any word2vec-like word embedding can be expressed by a weighted sum of these basis vectors. This furthermore entails that vectors can be expressed using sparse coding techniques, as the weights of the weighted sums of basis vectors should suffice to describe any embedding. Hence, as nothing prevents some or most of these weights to be set to zero, there is an equivalent sparse vector that encodes the same information as the initial word embedding. Interestingly, as the basis vectors are constructed from groups of contexts, one can expect that the components of a sparse representation should correspond to individual senses of the initial word represented by the embedding.

It is worth noting that to arrive at this conclusion, Arora et al. (2016) make a number of non-trivial assumptions. They assume that the softmax denominator of a log-classifier can be expressed as a product of the context vector's norm and a constant factor; they inherit the commonly held "bag-of-word" independence assumption that words in a sequence are probabilistically independent from one another; lastly, as noted previously, the demonstration only holds for a specific type of word-embedding architectures. Also of note is that the method is not unique to word2vec models, for instance, a related approach has been suggested for the Transformer architectures by Yun et al. (2021).

More broadly, the assumption that linear structure of vector spaces is mean-

ingful has however been subjected to criticism by numerous authors, with scrutiny mostly focusing on the linear offset approach of Mikolov, Yih, et al. (2013). For instance, Linzen (2016) notes how the terms in an analogy relation tend to be very close to one another—so much so that the three cue terms in an analogy (e.g., *king*, *queen* and *prince* in the analogy *king* : *prince* :: *queen* : *x*), if not removed from the potential answers, are likely to be retrieved by the algorithm. Similar concerns are raised by Rogers, Drozd, et al. (2017), who point out that results are generally impacted by the distance of the target vector: further items are much less likely to be retrieved. An equivalent conclusion is reached by Schluter (2018): she furthers previous remarks on the non-inclusion of cue terms in potential targets, and points how normalizing word embeddings before performing vector addition distorts results.

It should be noted that most of these studies focus on word type semantic properties: hence they are not directly applicable to contextual embedding models. Some approaches have nonetheless been proposed. In particular, Vulić et al. (2020) propose a comprehensive test of pre-trained BERT models on six languages (English, German, Russian, Finnish, Chinese, Turkish) and five lexical semantic tasks: lexical semantic similarity (where they measure the correlation between embedding similarity and human judgments of similarity), word analogy using BATS (Gladkova et al., 2016), bilingual lexicon induction (where a mapping is learned across languages), cross-lingual information retrieval (where the cross-lingual mapping is tested on a document-level IR task) and lexical relation prediction (where they attempt to predict the semantic relation—e.g., synonymy, antonymy, etc.—between two words, based on their embeddings). They conclude that such models can perform well on such static tasks, sometimes out-

performing baseline models, but that results are highly dependent on the exact methodology employed. Such approaches to applying word type intrinsic evaluation procedures to contextual embeddings are overall uncommon in the present NLP research landscape.

1.3.2 Classifier Probes

Overall, despite the importance of the literature on the relation between semantic spaces and word embeddings, whether contextual embeddings depict a coherent semantic space on their own has been left mostly unexplored. While a trend of research has focused on characteristics of the vector space described by the embeddings, noting how many Transformer-based models tend to be anisotropic (Cai et al., 2021, e.g.), the focus here is chiefly on the geometry.

Instead, a prominent methodology to investigate attention-based networks and contextual embeddings is that of "probes": simple learned models such as classifiers designed to extract information from the embeddings. The general idea has been traced back to Linzen et al. (2016). In this work, the authors are interested in finding out whether LSTM-based language models are able to learn number agreement between subject and verb. They focus on whether they are perturbed by attractors.[6] To that end, Linzen et al. (2016) learn a classifier to predict the number of the verb based on the corresponding LSTM state, and vary experimental conditions by selecting sentences with different numbers of attractors.

The idea to train a simpler model to investigate the contents of a more com-

[6]Nouns in a different number occurring between the subject noun and its corresponding verb; also known as "distractors" in the literature.

plex one caught on. Peters et al. (2018), when presenting their ELMo embeddings, also took this framework as an opportunity to showcase the performances and capabilities of their architecture. They address multiple tasks, from Question Answering to Semantic Role Labeling and to Natural Language Inference, and highlight how different representations from their model—either the sole embedding, or that embedding with the addition of BiLSTM hidden representations—lead to different performances on the tasks.

This methodology has been criticized as potentially conflicting with the intended purpose of studying the representations themselves. Wieting and Kiela (2019) even stress how probes can achieve high performance *in spite of* the input representations being probed: the margin between probing randomly initialized neural networks and probing trained models can be rather thin. Wieting and Kiela link this to a question of inherent model expressivity—i.e., more complex models ought to produce more complex outputs, not by virtue of their output but by virtue of their more complex structure. Hence expressivity would follow the number of components in hidden representations, as outlined by Cover (1965).

To overcome this, Hewitt and Liang (2019) propose an interesting take on this problem: their core argument is that probe expressivity ought to be taken into account when defining a task. To neutralize the expressivity of a given probe, they compare the results obtained by the probe on the task of interest, and on a slight reformulation where target labels are randomly permuted. This allowed Hewitt and Liang to quantify a baseline expectation of how rich the outputs of a model are simply by virtue of its inherent complexity.

1.3.3 Attention Head Analysis

While classifiers are technically broadly applicable to any vector representation, they have gained traction specifically in studying Transformer-based representations. This field in question is now known as "BERTology" studies; Rogers, Kovaleva, et al. (2020) have written a comprehensive introduction to some of the main results in this line of research. One reason why classifiers are so useful to the BERTologist lie in that they are well-suited to study Transformers in particular: they consist in easily learned models that can gather information from specific parts of the neural network under scrutiny. Given the very high number of attention heads in Transformers, classifier probes allow researchers to run diagnostic probing tasks on all heads, and identify individual attention heads that behave differently from the others.

This wide-spread use of probes can be thought of as a consequence of the popularity of attention mechanisms. Works interested in analyzing the behavior of attention heads (Raganato and Tiedemann, 2018; Hewitt and Manning, 2019; Clark, Khandelwal, et al., 2019; Coenen et al., 2019; Jawahar et al., 2019, a.o.) have each introduced specific procedures. General trends and findings nonetheless emerge from these different methodologies. Researchers frequently focus on the attention weights to determine which value vector is the most dominant in a head's output. A consequent body of research also reports that syntactic structures can be derived from these attention weights.

Recent research has however questioned the pertinence of these attention-based analyses. Serrano and Smith (2019) argues that attention weights can be meaningless on their own and should instead be studied along the directionality

of the value embeddings (to which the models pays attention) as well as the task at hand. Brunner et al. (2019) highlight how the down-projection before each attention head limits the ability of a model to keep track of long sequences, and stress that self-attention weight distributions are not directly interpretable. Even more problematic, Pruthi et al. (2020) highlight that attention heads can be trained to be deceptive—i.e., assign low weights to a set of "impermissible tokens" while still relying on these features for prediction.

Overall, analyses of attention heads tend to focus more on the inner workings of the networks than on their adequacy with theories of meaning. A very clear example of this trend is exhibited by works such as Voita et al. (2019) or Michel et al. (2019), which look into which attention heads can be removed without being detrimental to the overall performances of the network.

All in all, what this review of DSM evaluation protocols reveals is that the variety of architectures that can be framed as DSM translates into a variation of how they are investigated. Testing word token contextual embeddings on word type benchmarks is a nascent field of inquiry; and methodologies developed to study the hidden representations of contextual embeddings are often tied to specific characteristics of the Transformer architecture, such as the existence of a multi-head attention mechanism in the model.

1.4 Vector Size and Concentration

One aspect that we have yet to discuss is that modern DSMs are invariably implemented as vector spaces, but that the exact dimensionality of the vector space is left as an hyperparameter to set. Dimensionality is perhaps one of the best

studied aspects of vector spaces in machine-learning, as it has an immediate relationship with the expressivity of a model: hidden representations with more dimensions are in theory able to make finer distinctions. In the case of distributional representations and word embeddings, it stands to reason that this hyperparameter is a crucial factor to set.

There is worth in considering the effects of vector size on some well known metrics, so as to develop an intuition of the sort effect that stems from varying the dimensionality of embeddings. We construct a small-scale experiment to review the effects of vector size on vector metrics. We will focus our observations on a pre-trained embedding skip-gram model available on the NLPL vector repository[7]. In order to define our baseline expectations, we will also consider standard Gaussian vectors of dimension d: $\vec{y} \sim \mathcal{N}(\vec{0}, I_d)$—i.e., independent random vectors whose components are independent and identically distributed (iid) and sampled from a Gaussian standard distribution with mean 0 and standard deviation of 1.

Let us focus our analysis on three metrics: Euclidean norm, Euclidean distance and cosine similarity, as these are fairly common metrics when dealing with distributional semantics models and vector spaces in general. For reference, the Euclidean norm $\|\vec{x}\|_2$ of a d-dimensional vector $\vec{x}$ is defined as:

$$\|\vec{x}\|_2 = \sqrt{\sum_i^d x_i^2} \tag{1.21}$$

This norm is related to the Euclidean distance $d(\vec{x}, \vec{y})$ between two vectors $\vec{x}$ and

[7] Available here: `http://vectors.nlpl.eu/repository/6.zip`.

$\vec{y}$:

$$d\left(\vec{x},\vec{y}\right) = \|\vec{x}-\vec{y}\|_2 = \sqrt{\sum_i^d (x_i - y_i)^2} \tag{1.22}$$

The last metric we mentioned is cosine similarity, which can be defined as:

$$\cos\left(\vec{x},\vec{y}\right) = \frac{\langle \vec{x}\cdot\vec{y}\rangle}{\|\vec{x}\|_2 \cdot \|\vec{y}\|_2} \tag{1.23}$$

which relies on the scalar product $\langle \vec{x}\cdot\vec{y}\rangle$ between $\vec{x}$ and $\vec{y}$:

$$\langle \vec{x}\cdot\vec{y}\rangle = \sum_i^d x_i \cdot y_i \tag{1.24}$$

As the components y_i of a standard Gaussian vector $\vec{y}$ are iid, the Euclidean norm of standard Gaussian vectors follows a χ-distribution with d degrees of freedom:

$$\|\vec{y}\|_2 = \sqrt{\sum_i^d {y_i}^2} \qquad \text{with} \quad y_i \sim \mathcal{N}(0,1) \tag{1.25}$$

We can therefore compute expected value and variance as function of the number of dimensions (Abell et al., 1999):

$$\begin{aligned} \mathbb{E}(\|\vec{y}\|_2) &= \frac{\sqrt{2}\Gamma(\frac{d+1}{2})}{\Gamma(\frac{d}{2})} \\ \mathbb{V}(\|\vec{y}\|_2) &= d - \mathbb{E}(\|\vec{y}\|_2)^2 \end{aligned} \tag{1.26}$$

where Γ is the gamma function: $\Gamma(r) = \int_0^\infty x^{r-1}e^{-x}dx$ for any real value r; hence there is a straightforward analytical solution.

A similar remark can be made for Euclidean distance: recall that by definition,

the Euclidean distance between two vectors is the Euclidean norm of their difference, Note that $X - Y$, the difference of two normal variables $X = \mathcal{N}(\mu_X, \sigma_X)$ and $Y = \mathcal{N}(\mu_Y, \sigma_Y)$, is itself a normal variable with mean $\mu_X - \mu_Y$ and standard deviation $\sqrt{\sigma_X^2 + \sigma_Y^2}$. Given that our components are drawn iid from a standard normal distribution, $a_i, b_i \sim \mathcal{N}(0,1)$, we get that their difference $a_i - b_i$ is drawn from a normal distribution with mean 0 and standard deviation $\sqrt{2}$. Since $\mathcal{N}(0,\sqrt{2}) = \sqrt{2} \cdot \mathcal{N}(0,1)$, this entails that the distribution of Euclidean distance in a d-dimensional space can be re-framed as a scaled chi distribution. Let two independent standard Gaussian vectors $Z = (z_1, \ldots, z_d), Z' = (z'_1, \ldots, z'_d) \sim \mathcal{N}(\vec{0}, I_d)$:

$$\begin{aligned} Y &= \sqrt{\sum_k^d \left(z_k - z'_k\right)^2} && \text{with} \quad z_k, z'_k \text{ iid standard Gaussian} \\ &= \sqrt{\sum_k^d \delta_k^2} && \text{with} \quad \delta_k := z_k - z'_k \text{ iid } \sim \mathcal{N}(0,\sqrt{2}) \\ &= \sqrt{2} \cdot \sqrt{\sum_k^d \delta_k^2} && \text{with} \quad \delta_k \text{ iid standard Gaussian} \end{aligned} \tag{1.27}$$

Hence the distribution of the Euclidean distance between two independent standard Gaussian vectors is equal to the distribution of the Euclidean norm multiplied by $\sqrt{2}$, from which we can retrieve the variance and expected value:

$$\begin{aligned} \mathbb{E}(\|\vec{y} - \vec{x}\|_2) &= \frac{2\Gamma(\frac{d+1}{2})}{\Gamma(\frac{d}{2})} \\ \mathbb{V}(\|\vec{y} - \vec{x}\|_2) &= 2d - 2 \cdot \mathbb{E}(\|\vec{y}\|_2)^2 \end{aligned} \tag{1.28}$$

As for cosine, we should note that its expected value is always 0, for any dimension d. We can nonetheless inspect what variance we observe as a function of dimension.[8] Let $\vec{x}$ and $\vec{y}$ be two standard Gaussian independent vectors. Denote R the random rotation such that $R(\vec{y})/\|R(\vec{y})\|_2 = (1,\ 0,\ \ldots,\ 0)$, i.e., the unit vector corresponding to the first axis. As a measure of angle, cosine is insensitive to global rotations of the vector space—that is to say, $\cos\left(\vec{y},\ \vec{x}\right) = \cos\left(R(\vec{y}),\ R(\vec{x})\right)$. It is also insensitive to individual vector scaling, that it is to say $\cos\left(\vec{y},\ \vec{x}\right) = \cos\left(k \cdot \vec{y},\ \vec{x}\right)$. From this, we can rewrite without loss of generality:

$$\cos\left(\vec{y},\ \vec{x}\right) = \cos\left(\frac{R(\vec{y})}{\|R(\vec{y})\|_2},\ \frac{R(\vec{x})}{\|R(\vec{x})\|_2}\right)$$

For simplicity, let us denote $\vec{z} = R(\vec{x})/\|R(\vec{x})\|_2$ and $\vec{b} = R(\vec{y})/\|R(\vec{y})\|_2$—i.e., $\vec{b} = (1,\ 0,\ \cdots,\ 0)$. As both vectors $\vec{z}$ and $\vec{b}$ are normalized (i.e., their norms is equal to 1), the cosine between the two is equal to their scalar product.

Note that R is a random rotation that only depends on $\vec{y}$, and is therefore independent from $\vec{x}$. As such, R is a linear application and $R(\vec{x})$ is iid sampled from a Gaussian vector. More precisely, $R(\vec{x})$ is iid sampled from a Gaussian vector with mean $R(\vec{0}) = \vec{0}$ and covariance matrix $RI_dR^T = I_d$, since R is a rotation and thus $RR^T = I_d$. Simply put, this entails that $R(\vec{x})$ is a standard Gaussian vector.

As such, the components $R(\vec{x})_i$ are symmetric. This entails that the two vectors $U = (R(\vec{x})_1, \cdots, R(\vec{x})_i, \cdots, R(\vec{x})_d)$ and $V = (R(\vec{x})_1, \cdots, -R(\vec{x})_i, \cdots, R(\vec{x})_d)$ have identical distributions. From this, we gather that $z_i = U_i/\|U\|_2$ and $-z_i = V_i/\|V\|_2$ are identically distributed. In other words the random variable z_i is symmetric, thus centered, and we therefore have $\mathbb{V}(z_i) = \mathbb{E}(z_i^2)$. Moreover, the

[8] We are highly indebted to Marianne Clausel for the following demonstration.

components z_i are all independent, and by linearity of expectation we have:

$$\sum_{i=1}^{d} \mathbb{V}(z_i) = \sum_{i=1}^{d} \mathbb{E}\left(z_i^2\right) = \mathbb{E}\left(\sum_{i=1}^{d} z_i^2\right)$$

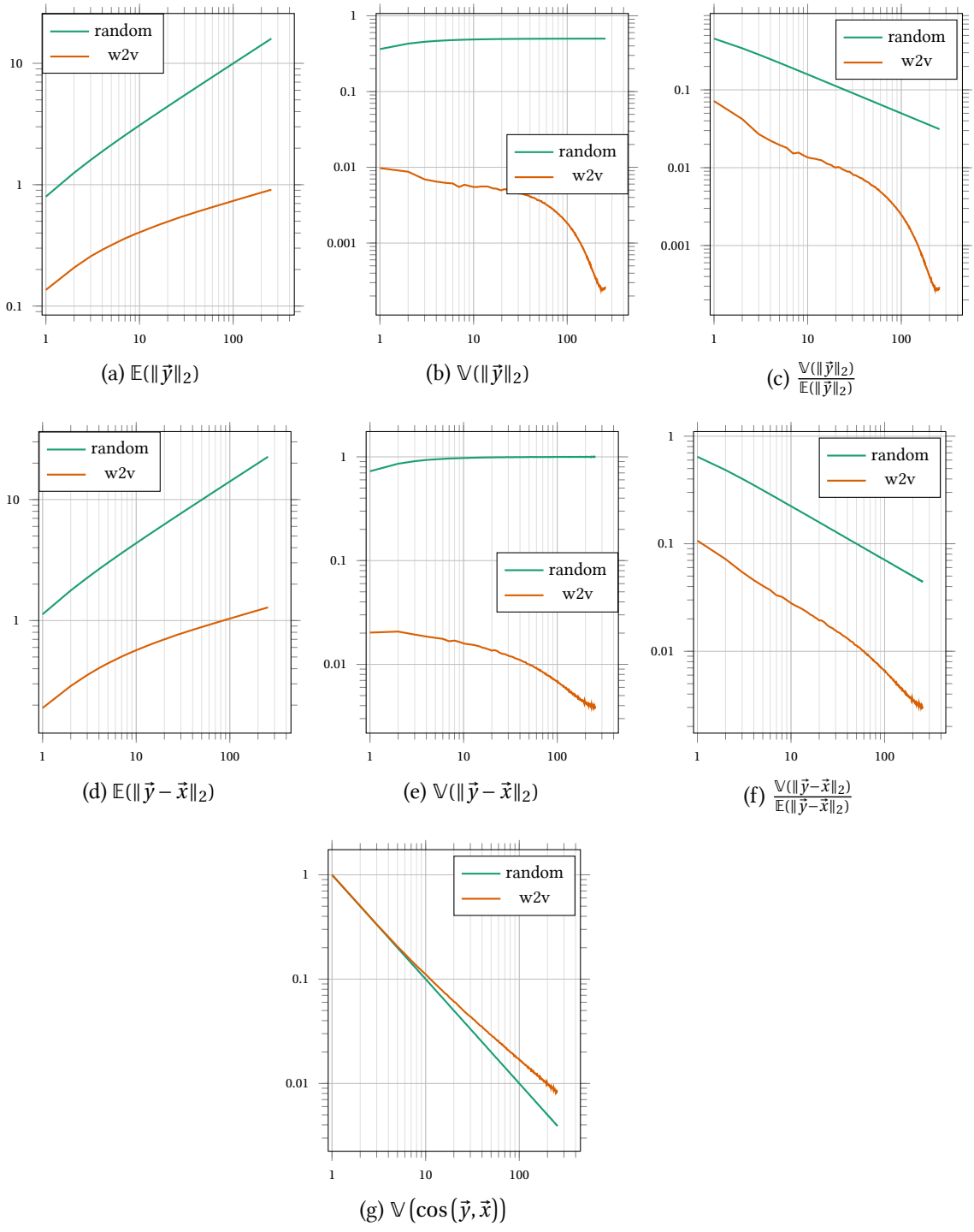
Note however that $\sum_{i=1}^{d} z_i^2$ is equal to the norm of $\vec{z}$ squared, which by construction is equal to 1. The components z_i are identically distributed and their variances are therefore all equal: $\forall i, j \quad \mathbb{V}(z_i) = \mathbb{V}(z_j)$; as all d components' variances sum to 1, we get that the variance for the distribution of each individual component is equal to $1/d$.

Returning to the cosine between $\vec{z}$ and $\vec{b}$, we have noted that the denominator $\|\vec{z}\| \cdot \|\vec{b}\|$ is equal to 1, and the cosine is entirely defined by the scalar product $\langle \vec{z} \cdot \vec{b} \rangle$. The definition of the scalar product in Equation (1.24) and our construction for $\vec{b}$ entail that $\cos(\vec{x}, \vec{y}) = \sum_{i=1}^{d} b_i \times z_i = 1 \times z_1 + \sum_{i=2}^{d} 0 \times z_i = z_1$. From this, we can derive the variance of cosine as:

$$\mathbb{V}(\cos(\vec{x}, \vec{y})) = \mathbb{V}(z_1) = \frac{1}{d} \qquad \text{with } \vec{x}, \vec{y} \text{ iid} \sim \mathcal{N}(\vec{0}, I_d) \tag{1.29}$$

To compare our random baselines in Equations (1.26), (1.28) and (1.29) to what we observe for word embeddings, we adopt the following approach: for each d' between 1 and 256, we make a random sample S of 100 000 embeddings drawn from the embedding model, and then apply a dimensionality reduction over S using PCA with d' components.

Plotting the expected value and the variance of Euclidean norm and Euclidean distance, as well as the variance of cosine similarity against dimensionality (as its



(a) $\mathbb{E}(\|\vec{y}\|_2)$ (b) $\mathbb{V}(\|\vec{y}\|_2)$ (c) $\frac{\mathbb{V}(\|\vec{y}\|_2)}{\mathbb{E}(\|\vec{y}\|_2)}$

(d) $\mathbb{E}(\|\vec{y}-\vec{x}\|_2)$ (e) $\mathbb{V}(\|\vec{y}-\vec{x}\|_2)$ (f) $\frac{\mathbb{V}(\|\vec{y}-\vec{x}\|_2)}{\mathbb{E}(\|\vec{y}-\vec{x}\|_2)}$

(g) $\mathbb{V}\left(\cos\left(\vec{y},\vec{x}\right)\right)$

Figure 1.3: Vector space metrics, as a variable of dimensionality

expected value is systematically 0) will therefore produce Figure 1.3. To better understand the degree to which the evolution of the two moments (expected value and variance) is commensurate, we can also look at their ratio in Figures 1.3c and 1.3f). Looking at the distance and norm of standard Gaussian vectors reveals that the norm appears bounded, whereas the expected value keeps rising. As a consequence, there is a concentration phenomenon: there is less and less variation around the mean. Turning to the norm and distance word2vec models, we find that the variance even decreases in higher dimensions, while the expected values rise. This mechanically leads to a similar effect of concentration around a mean value. Lastly, cosine variance can also be seen to sharply decrease, whereas the expected value is constant at 0: again, this entails that the distribution is more tightly distributed around a central value in higher dimensions.

Simply put, this experiment shows that vector length, distance and angle do not mean the same thing, *quantitatively* speaking, in higher dimensions. Observing two highly diverging values is all the less likely in higher dimensional spaces. Crucially, this holds for both random and distributional vectors. As such, even in the case of two sets of embeddings computed through the same embedding algorithm, we may in principle observe distinct behaviors owing to differences in dimensionality.

1.5 Conclusions

In this chapter, we have focused on what distributional semantics are, as a theory of semantics. The core aspect that we have attempted to highlight is the high

variability of what can be construed as a model of distributional semantics.

We have shown that the coherence of distributional semantics as a field of study is more historical than theoretical, as we saw in Section 1.1. Recent models have claimed themselves as distributional, but a historical perspective on this field of study stresses the progressive modification of what fits under this term. What used to be an emanation of the structuralist theories of Harris (1954) was turned into a psycholinguistic research paradigm; and only under the impulse of vector space model inherited from the field of Information Retrieval did large-coverage, systematic sets of distributional representations emerge. Recent developments in this area of research have introduced new characteristics at a fast pace: the emergence of neural models, as well as the predominance of contextual word-token vectors are both innovations from the last decade, whose implications have yet to be fully understood.

This mostly chronological coherence has the practical of outcome of fostering a very diverse field of study. Our in-depth review of selected architectures in Section 1.2 underscores very clearly the practical differences between existing models. Different implementations of distributional semantics, such as the Transformer of Devlin et al. (2019) or the log-linear classifiers of Mikolov, K. Chen, et al. (2013), will likely produce structurally different vector representations. The immediate consequence of this variation is that the approaches proposed to study DSMs must also be adapted to specifically match the models of interest, as we have surveyed in Section 1.3. Even when considering two sets of embeddings drawn from the same algorithm, differences in dimension may influence the behavior of well-known metrics, as we saw in Section 1.4.

In all, while this chapter has underscored the theoretical difficulties that come

with construing distributional semantics as a lexical semantic theory, it is still possible to consider the wide variety of models developed in the literature as a consistent and coherent group. There are nonetheless implications entailed by this overview: any work claiming to study distributional semantics as a coherent lexical semantic theory will have to embrace the multiplicity of existing approaches.

2

Dictionaries in NLP

The sentient is sent to seek out all the truth
A flight to earth that is a given from his birth
To rise from ashes of the dead
Out of the fire is sent to fulfill man's desire

— Iron Maiden, *The Book of Souls*

Dictionary-making us an age-old practice. 18th century Europe dictionary-makers that the general public is most familiar with, such as Harris and Diderot, were building upon a long and well established tradition. The Chinese Er-ya (尔雅), which is generally agreed upon as the oldest surviving dictionary, dates back between the 6th century BC to the 3rd century BC. Looking at other related lexical resources, we find glossaries written in cuneiform from the early 2nd millennium BC, as well as lexica, such as the *Átaktoi glôssai* (or "Disorderly words"), written in the 4th century BC by Philitas of Cos, which listed rare, archaic, dialectal or technical words.

Elements of this chapter were adapted from a previous publication (Mickus, Constant, et al. 2021b, "About Neural Networks and Writing Definitions").

Modern lexicographers are very much invested in using a descriptive approach, rather than a prescriptive one. They document what usages exist "in the wild" when writing dictionary definitions. Neither are lexicographers opposed to bringing in new technologies in the art of writing definitions. Lexicographers frequently use large corpora of texts to see whether their definitions describe actual word usage: this is made possible by the existence of technology to process and explore these large corpora such as the SketchEngine concordancer. Another domain where dictionaries make use of modern technology is for data storage: the widely used electronic document format XML was developed with the Oxford English Dictionary in mind.[1] There is also a rather long-standing tradition of linguists working with—or as—lexicographers. John Rupert Firth worked on the Oxford English Dictionary and discussed at length the proper methodology for writing definitions (Firth, 1952). Natalia Shvedova both succeeded Sergei Ozhegov in maintaining the Russian Ozhegov dictionary, and wrote multiple monographs and essays on Russian syntax.

Dictionaries are both lexical resources at scales where computer science tools become relevant, and objects of linguistic study in their own right. A large body of work in NLP is devoted to using dictionaries, which we will try to summarily review in this chapter. First, in Section 2.1, we will have a look at what are dictionaries. Second, we will discuss how dictionaries can help in studying problems of semantic grounding in Section 2.2. Third, in Section 2.3, we will consider NLP approaches that attempt to use dictionaries as meaning inventories. Lastly in Section 2.4, we will focus on two tasks based on dictionaries: the reverse

[1] See the notice from the OED on this topic: `https://public.oed.com/blog/the-oed-and-innovation/`.

dictionary task in Section 2.4.1 and the definition modeling in Section 2.4.2.

2.1 What are dictionaries

To provide a reductive definition of what a dictionary is, one could say it is, at its very core, a set of definitions. This very crude characterization of dictionaries is however too limited to properly account the entirety of the scientific work that goes into making a dictionary. We will first start by laying out the terminology we will employ (Section 2.1.1). We then review what different resources have been called dictionaries in Section 2.1.2, and make a few remarks on the relation between dictionaries and lexical semantic theories in Section 2.1.3.

2.1.1 Terminology

Throughout this dissertation, we will adopt the following terminology: A *definition* is an entry as found in a dictionary. It links a word to be defined—a *definiendum* (plural: definienda)—to a *gloss* which explains its meaning in natural language. A gloss is generally made up of multiple tokens, which we call *definientia* (singular: definiens).

realm:	An abstract sphere of influence, real or imagined.	(1)

We can take as an example definition 1. Here, the definiendum that we wish to define is "*realm*". The associated gloss is "*An abstract sphere of influence, real or imagined*". This gloss can also be analyzed as a sequence of definientia: ⟨`'An'`, `'abstract'`, `'sphere'`, `'of'`, `'influence'`, `','`, `'real'`, `'or'`, `'imagined'`, `'.'`⟩. We will specifically mention whenever we depart from this terminology.

2.1.2 Defining the concept of a dictionary

Paradoxically, dictionaries are familiar everyday objects as well as poorly delineated from a theoretical standpoint. On the one hand, the general public is familiar with the dictionary: it is both an enshrined arbiter of knowledge of all things linguistic, as well as the book that sits on their shelf and that they consult when playing Scrabble. As Durkin (2016) remarks at the very introduction of his introduction to the Oxford Handbook of Lexicography, "few people need to be told, in the broadest terms, what a dictionary is."

On the other hand, coming up with a satisfactory definition of what a dictionary is has been the object of many papers (Sterkenburg, 2003; Bergenholtz, 2012; Tarp, 2017, e.g.). To take a concrete example, Tarp (2017) reviews definitions proposed in the lexicography literature, discusses the limitations of each, and finally arrives at the following definition for a dictionary (p. 246):

> A dictionary is a utility tool, which is conceived for consultation with the genuine purpose of meeting punctual information needs experienced by specific types of potential user in specific types of extra-lexicographical context, and which is designed to assist its users by providing manual or automatic access to lexicographically prepared data, which can either be used directly by the users in order to retrieve the required information which they can subsequently use to solve specific problems in the context where the needs originally occurred, or by a digital tool in order to make automatic corrections in texts or translations produced by the users of this tool.

Needless to say, the technicality and complexity of this definition is a clear tes-

timony of the hardships encountered by the author.

Part of the hardship stems from the fact that "dictionary" is a rather loose term. To start with, lexicographers often note how the general public understands '*the* dictionary' as some kind of institution that records the 'true meaning' of words—whereas lexicographers themselves stress that no two dictionaries are quite the same in all respects, and much of the differences are due to the conscious choices and thoughts of editors.

Another point to take note of is that there are many types of dictionaries. Etymological dictionaries yield the etymology of words. Bilingual dictionaries propose glosses in some other target languages, and are therefore geared towards translation purposes. Learners' dictionaries attempt to cater more specifically to the needs of foreign learners of a given language—they therefore include detailed instruction about the grammar and usage of words, on top of definitions in the simplest style possible. A number of reference works can also be construed as dictionaries: works ranging from encyclopedia to dictionaries of place names, proper names, and the like, as well as technical lexica (e.g., the Oxford Dictionary of Music). Last, but not least, is the most familiar general-use dictionary.

The actual format of dictionaries and other reference works can vary as well. Dictionaries have been recorded on probably every medium—from clay tablets to books and to electronic formats. Even the general structure (a.k.a. the "macrostructure") of dictionaries is not fixed: while ordering definitions alphabetically by definienda is the most familiar practice, it is by no means the only one. For instance, thematic dictionaries group definienda according to their general meaning, whereas electronic dictionaries do not require any overt ordering of their

definitions.[2] Reverse dictionaries (a.k.a. retrograde dictionaries) flip the usual structure, and allow users to query definienda based on their glosses.[3] Finally, how definitions contained in these dictionaries are written is yet again a domain where variation abound, with practices ranging from coming up to paraphrase that can be substituted for the target definiendum, to displaying and explaining the definiendum in the context a typical sentence (Hanks, 2016).

In sum, the contents, format, structure and medium of dictionaries are not fixed. Attempting to subsume all these different reference works in a single concept is therefore rife with caveats. In this dissertation, we adopt two basic guidelines to sidestep this issue. First, on a practical level, we will focus solely on monolingual general-use dictionaries. While this does not solve every problem we have mentioned so far—since different linguistic traditions correspond to different lexicographic traditions—it does reduce the degree of variation we will have to juggle with.

Second, on a theoretical level, we consider the concept of a dictionary as a fuzzy one, much as what Wittgenstein (1953) sketches for games: they display criss-crossing similarities, but there is no set of sufficient and necessary properties to delineate dictionaries to be found. Some of these similarities were delineated in the definition from Tarp (2017) above, but we can stress two traits which directly influence our work. Foremost is that dictionaries are, at their core, sets

[2] In practice, a electronic dictionary created from a database of definitions will have an explicit order or index. These are purely technical implementation details: users of such a dictionary will in most cases not be aware of this order.

[3] The term "reverse dictionary" has also been used to characterized common dictionaries with entries ordered by the reverse spelling. Such dictionaries start by listing all words that end in "*-aa*", followed by all words that end in "*-ba*", etc. Throughout this thesis, we will ignore these dictionaries ordered by the suffix of the definienda. We strictly reserve the term "reverse dictionary" to describe dictionaries where users look up glosses to find corresponding definienda.

of definitions, and therefore they link words to glosses. Another point to consider is that dictionaries are tools, built for human users in mind—this aspect will necessarily impact any automated approach we consider.

2.1.3 Dictionaries as semantic theories

One can also point out that dictionaries cannot simply be practical tools, made for users to manipulate and find information. The fact that general-use dictionaries frequently include definitions for function words is a strong indicator that looking up the meaning or spelling of words is not likely the sole *raison d'être* of a dictionary: basic linguistic competence should rule out the need to look up the meaning of words such as "*of*", "*a*", "*this*", and so on. It should be stressed that including entries for such words is defensible in most dictionaries that do not target fully linguistically competent readers; moreover not all general-use dictionaries will define function words: for instance, Bergenholtz (2012) stresses that the early editions of the Nudansk Ordbog did not include definitions for words that were deemed common.

If the inclusion of common word definitions is not motivated by the utilitarian nature of a dictionary as a lexical resource, why are they included? Béjoint (2016) relates this to the social dimension of dictionaries, especially which were developed during the construction of European nation-states. He notes (p. 12):

> In some European countries, the motivation was to sing the praises of the language, at a time when nations were taking shape and found themselves competing for riches, for territories, for prestige, and for influence. General dictionaries were compiled to show how venera-

> ble, how rich, how harmonious, how regular the language was, how superior it was to all other languages.

Furthermore, he stresses this interest in the richness of the vocabulary is still very present to modern-day dictionary users (p. 19):

> the users want their [dictionary for general users] to represent the whole language, and this emblematic function of the dictionary is as important for them as its more practical functions.

That all manners of words, including the most common ones, are defined in dictionaries is not without its interest. This entails that dictionaries can be viewed as lexical semantic theories, as we had defined earlier in Chapter 1. They attempt to describe the semantic content of the entire lexicon, much as DSMs ascribe a vector representation to every word attested in a corpus. The difference is that the meaning of words is not described by means of numerical components. Instead, the fundamental hypothesis espoused by dictionaries is that words can be described by means of natural language. This connection between products of lexicography and theories of meaning is not just a mere happenstance: as Geeraerts (2016) notes, lexicography is applied lexicology—hence new developments of lexical semantics often influence how definitions are written.

It should also be noted that lexicographers have adopted corpus-based investigations as their primary methodology when writing definitions. This new trend was crystallized at the time in articles such as Kilgarriff (2000); modern lexicography handbooks will almost invariably contain some materials about corpus construction and use (Kupietz, 2016; Kosem, 2016, e.g.). This practice entails that the lexical semantic theory to which a given dictionary can be equated is often

rooted in a descriptive approach— modern dictionaries do not adopt a purely normative position: they characterize the normal usage as they observe it.

2.2 Dictionaries and Semantic Grounding

As we have just discussed, dictionaries are informed by corpus studies and developments in lexicology. This make dictionaries a very practical standard of comparison for NLP systems: as a lexical resource, they can be used to frame and investigate some of the limitations we expect to encounter in NLP. One domain where dictionaries have shown great usefulness is that of semantic grounding, and we will take this topic as an example of how NLP systems can make use of dictionaries as inventories of meanings.

2.2.1 What is semantic grounding?

In the thought experiment of Harnad (1990), we are asked to picture an English speaker who doesn't speak Chinese. We give them a Chinese monolingual dictionary and ask them to learn to speak Chinese from that dictionary alone. The task seems strictly impossible, and Harnad (1990) therefore concludes that external information is required. It is necessary to identify the real world objects that Chinese characters refer to: without this information, our English reader can only memorize strings of symbols that they will be unable to use in a conversation.

We focus here on a referential take on meaning. While this approach leaves out many crucial aspects of meaning, we do so for simplification purposes. The field of NLP is centered on applications and therefore values first and foremost

factual accuracy. To take a concrete example, if we design a software that generates image captions, our interest will lie in whether the caption actually describes items present in the image. Another point to take into account is that NLP models are not social agents in the same sense that competent speakers are (Bender and Koller, 2020): the text they produce does not correspond to a specific communicative intent. The pragmatic and social dimensions of language are virtually absent in artificial text, or are found only in the eye of the beholder.

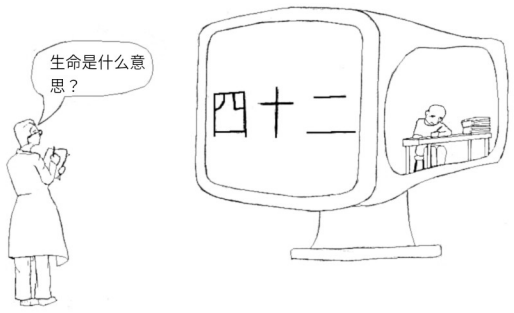


Figure 2.1: Thought experiment of Searle (1980)

Another metaphor describing the conundrum of semantic grounding is the Chinese Room Argument of Searle (1980), which we illustrate in Figure 2.1. The thought experiment goes as follows: suppose that we construct some piece of software capable of answering any Chinese question in flawless Chinese. If we take someone who doesn't speak Chinese and isolate them in a room with the source code of that software, then slip in a question in Chinese, the person inside the room will be able to perform all the computations described in the source code by hand and produce some sort of coherent answer. But nothing in the room

speaks Chinese. Searle (1980) takes this as a proof by absurdity that no such Chinese-speaking software can be written. To most, it sounds like an intriguing paradox, laying out a question that is not straightforward to answer: how can a machine learn to talk? Is talking merely about producing appropriate symbols, without any form of understanding?

A very similar argument was developed by Jackson (1982). He asks us to imagine Mary, a brilliant scientist, locked in a black and white room. From within this room, she has to study how human vision works. To that end, she has a black-and-white screen, which can display any black-and-white image she needs, as well as access to all the information she might require. She can gather all the physical evidence to establish how different wavelengths of light affect the retina, and learn that this is what humans outside her black-and-white room call color. Would Mary discover something about color by leaving her room? Jackson (1982)'s position is that there is something about colors—qualia, in the terms of Jackson (1982)—that cannot be conveyed through words and a black-and-white screen alone. Colors have to be perceived, and Mary would therefore learn something by seeing red for the first time. This qualia argument would equally apply to neural networks: we cannot expect word embedding models to encode the experience of color, if that experience is inferred from text alone.

The hardships inherent to this question are made even clearer in the Octopus thought experiment, proposed by Bender and Koller (2020), and illustrated in Figure 2.2. Two English speakers, Alice and Bob, are stranded on two islands connected by a telegraph wire running along the seafloor. Unbeknownst to them, a seafloor-dwelling super-intelligent octopus that does not speak English has tapped into the telegraph wire and listens in on their conversation. At some

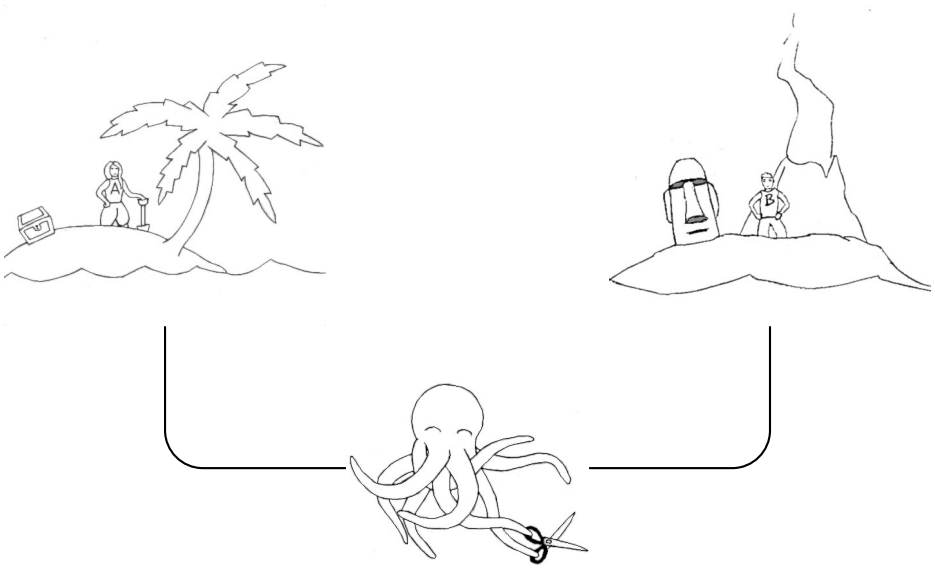


Figure 2.2: Thought experiment of Bender and Koller (2020)

point, it decides to alleviate its boredom by cutting the cable and impersonating Bob. Bender and Koller (2020) ask: will the super-intelligent octopus be able to successfully deceive Alice, or will she find out? Their answer is that it depends on the task: the Octopus should be able to reproduce mundane greetings, but won't be able to respond properly if Alice talks about something fundamentally new that requires the speaker to understand and know what it is they are talking about—for instance, if she discusses how to build something brand new, such as a coconut catapult.

2.2.2 What dictionaries show of NLP systems and grounding

These thought experiments highlight how relating words to real-world objects is a crucial goal for NLP systems: without this step, no neural network can be guaranteed to produce meaningful utterances—that is, we can't build the Chinese Room of Searle (1980) unless we are able to guarantee that the Octopus of

Bender and Koller (2020) understands what it is talking about. One of the more commonly accepted solutions to this difficulty is to provide our models with some sort of interaction with the real world, from which they will be able to build informed representations of words. Systems that rely on text alone therefore can't be guaranteed to produce coherent, meaningful outputs. Even if they do so, it's likely they do so by accident, by relying on clever heuristics rather than by actually manipulating information in a meaningful fashion.

Despite these limits, there are things that can be learned even without grounding. The setup described by Harnad (1990), where an English speaker has to learn Chinese from a dictionary, is especially useful here since it allows us to construe the problem as a task to solve. In the setup of Harnad (1990), there are things that the English speaker can learn: the meaning of the word being defined will influence what sorts of words appear in its definition, and said meaning is loosely correlated with the way the defined word is written. Much as how we can make educated guesses from a word's morphology in English, Chinese orthography is not entirely random. We can see this at play in the selected sample of Chinese definitions displayed in definitions 2, 3 and 4: the definienda all end with the character 星, whereas the glosses in the second column all start with the sentence "行星名。". Other sequences of characters are also found in all these glosses.

土星:	行星名。距离太阳第六近的行星，目前已知有六十余颗卫星，有明显行星环。属于类木行星，外观呈黄棕色，大气成分主要为氢和氦。古代称为「镇星」、「填星」、「信星」。	(2)

火星: 行星名。距离太阳第四近的行星，有两颗小卫星。属于类地行星，外观呈现红棕色，大气稀薄。表面的奥林帕斯山是太阳系的最高山峰。古称「荧惑」。 (3)

金星: 行星名。距离太阳第二近的行星，较地球略小。属于类地行星，外观呈现淡黄色，拥有浓厚大气层，温室效应剧烈，是太阳系中最热的行星。在古代，金星于日出前出现在东方称为「启明」，傍晚出现在西方则称为「长庚」 (4)

Neural networks attempting to generate definitions ought to be able to pick up on such regularities: if the word to be defined ends in 星, then start by generating 行星名。. To a Chinese speaker, this seems somewhat reasonable: words that end in 星 often denote planets, whereas 行星名 literally means 'name of a planet'. Nonetheless, such formal similarity could also in principle be purely coincidental: this sort of heuristic is not at all driven by meaning and relies solely on some surface property that happens to be expressed more or less regularly in a corpus of definitions. A patient enough reader who does not speak Chinese would be able to list all the sequences of characters common across the three glosses 2, 3 and 4, despite not knowing the meaning of any of these. This sort of heuristic is, in short, an educated guess: models that exhibit this sort of behavior have learned to apply certain patterns. They do not produce a definition grounded in their understanding of the word to be defined. As such, these patterns are bound to be brittle: in our example, we can see that many Chinese words that end with 星 do not cor-

respond to planet names. The character 星 itself is generally translated as 'star', and therefore appears in words such as 流星 ('shooting star') or 彗星 ('comet'). Consequently, their definitions 5 and 6 do not start with the sequence "行星名。".

流星: 太阳系中无数小微粒之一, 只有当它碰巧落到地球大气层内时才能直接观测到, 在大气中由于受到运动的阻力会引起暂时的灼热, 此时若在夜间, 天空中就出现一道亮光 (5)

彗星: 是明亮彗核周围通常包着的朦胧彗头组成的云雾状天体, 当它运行的轨道部分靠近太阳时经常出现长长的彗尾, 由于辐射压力使彗尾指向远离太阳的方向, 彗星的运行轨道随着从近似圆形到抛物线而具有不同的偏心率, 轨道倾角从 0° 到 180°, 运行周期从 3 年到几千年。通称扫帚星 (6)

The core argument of all the thought experiments we have surveyed is that text alone is not sufficient to posit that neural networks manipulate information in a meaningful fashion: words also have to be linked to the real-world objects they can refer to. Dictionaries, again, are found to be useful in this area of study. Vincent-Lamarre et al. (2016) have shown that in order to infer the referents for all the words in an entire dictionary, it suffices to know the referents for a small subset of its vocabulary. More precisely, they study how definitions are linked to one another: as a definiendum can appear as the definiens in another definition, we can establish a graph of all words listed in a dictionary by linking definienda

to their definientia. Vincent-Lamarre et al. (2016) further provide a method to reliably extract from this graph the minimum number of words to ground. If we can describe for every word in this minimal set what it refers to, we can iteratively ground definitions one by one. We start with definitions that only rely on words from this minimal set: these grounded definitions allow us to use their definienda to ground even more definitions, until the entire dictionary is grounded.

2.3 Dictionaries as NLP meaning inventories

Dictionaries are therefore useful to delineate and study issues that we expect of text-based NLP models. As such, they are highly relevant lexical resources to the NLP scientist. We now turn to review more closely the link between dictionaries and NLP applications.

Throughout the history of NLP, dictionaries have been invaluable sources of data because they provide semi-structured data in high volumes. Automated approaches are well suited to parse, re-format and enrich the semantic information contained in dictionaries.

To take a concrete example, Chodorow et al. (1985) suggested using the structure of a dictionary to extract semantic hierarchies. Their starting point is the observation that the Aristotelian model of definition—whereby a definiendum is defined by means of a genus (its broad semantic category) and a differentia (its specific attributes within this category)—is a prevalent style of definition in lexicography. This in turn justifies a simple heuristic: the syntactic head of a definition should correspond to a hypernym of the word being defined—because this

head is likely to be the genus of the word being defined, and the genus should be a hypernym of the definiendum. Any automated extraction of a syntactic head can therefore be used to establish hypernymy relations, and, by extension, a semantic hierarchy of words.

This heuristic, while certainly most useful to NLP applications, is not the only way element that NLP scientist have used from dictionaries. Lesk (1986), for instance, focused on the fact that dictionaries are also inventories of meaning and thereby relevant to a central problem in NLP: the task of word-sense disambiguation (WSD), and proposed to leverage dictionaries definitions to perform word disambiguation. This task consists in mapping ambiguous words to unambiguous senses, given the context they appear in: e.g., deciding whether the word *tie* as it appears in the sentence "*The game ended in a tie*" be mapped to the sense 'tie: article of clothing worn around the neck' or 'tie: draw, outcome with no clear winner'. The proposal of Lesk (1986) relies on two assumptions. The first assumption is that generally speaking, we can expect words semantically related to the relevant sense to appear in the context of the word to disambiguate; that is to say, if words such as *game*, *score*, or *match* occur near the noun *tie*, then it is more likely to be used in the sense of a 'draw', rather than 'a piece of clothing'. The second assumption is that the definientia are also semantically related to the specific sense they are defining: clothing is less likely to occur than *match* when trying to define the 'draw' sense of *tie*. Therefore one can expect that the overlap between words in a definition and word in a context can cue a machine into what sense matches a given word token. Further improvements on this method have been suggested throughout the years; for instance Gaume et al. (2004) also rely on the transitive nature of definitions—that is to say, that the words in a

definition of a definiens should also be semantically related to the definiendum, and so on—to compute this overlap.[4] This trend of research is still very relevant today, as attest approaches such as the GlossBERT model of L. Huang et al. (2019), which proposes to employ both contextual embeddings to model words to be disambiguated as well as existing dictionaries such as WordNet to represent target word senses.

Assuming that dictionaries correctly, consistently and exhaustively describe the meanings of words allows NLP researchers to use dictionaries to study all aspects of semantics. Hill et al. (2016) look at how dictionaries equate single words (the definiendum, e.g., "*giraffe*") to sequences of words (the definientia, e.g., "*a tall, long-necked mammal of Africa*"). This equation can be used to study and model semantic composition. The composed meaning of a gloss can be taken to be roughly equal to the meaning of the word being defined, i.e., dictionaries are a natural benchmark for compositional semantics. By using the definiendum as a target and the definition gloss as an input source, NLP researchers can train models to infer the meaning of a phrase from the meaning of its components. As a concrete use-case, Hill et al. (2016) showed how this could be exploited to implement a crossword solver.

Dictionaries have also been used in NLP to compute computer-friendly semantic representations. One such example is the work of Tissier et al. (2017), also known as dict2vec. Their key idea is that dictionary definitions provide all the information necessary to derive a consistent semantic representation of the

[4]The formulation of the WSD task itself is subject to criticism; in particular, it presupposes that there is a fixed, discrete set of senses that can be assigned to each word. Kilgarriff (1997) strongly argues that sense inventories only make sense with respect to a task. This trait is not exclusive to WSD. NLP applications derived from dictionaries rarely, if ever, question the sense inventory described by the dictionaries they use.

definiendum. Concretely, the proposal of Tissier et al. (2017) consists in computing word embeddings not from distributional contexts drawn from large corpora, but from the definientia associated to a given definiendum. Following a similar idea, Bosc and Vincent (2018) look to convert dictionary glosses into consistent embeddings by using an auto-encoder neural network. In their work, Chang and Y.-N. Chen (2019) suggest to transform definition glosses into embeddings using sentence encoders, and to learn a mapping from contextualized definienda embedding to gloss embedding as an explanation tool.

In all, these works share a common feature: dictionaries are used primarily as inventories of meaning, which can then be mined and exploited to yield more refined semantic information, and exploit them in NLP applications. In the case of Chodorow et al. (1985), that information was hypernymy relations. In the work of Hill et al. (2016), it was semantic composition. In works such as of Tissier et al. (2017), the focus was to convert the format of this information.

2.4 Definition Modeling and Reverse Dictionary

Dictionaries can also be construed as datasets for NLP tasks. In particular, researchers have been interested in generating definition glosses given the associated definiendum—known as the Definition Modeling task—, or finding which definiendum corresponds to a given input gloss—i.e., constructing softwares for reverse dictionaries.

2.4.1 Reverse Dictionary

A reverse dictionary, such as the Oxford reverse dictionary (Edmonds, 1999), is a type of lexical resource that matches concept descriptions to words that correspond to these descriptions. In that, a reverse dictionary is the mirror image of a dictionary: instead of mapping a word to a paraphrase that explains it, it maps a paraphrase to a word. One difficulty inherent to this type of lexical resource is that a given concept can be described using many different paraphrases—arguably, infinitely many descriptions can correspond to any given concept. The description that a user would come up with will certainly differ from the descriptions offered by a reverse dictionary. This entails that a physical reverse dictionary can never be an exhaustive resource. This also underscore the need for an automatic procedure that parses a user's input description and returns a matching concept. This can therefore be formulated as an NLP task, where the input is an user query, and the output is the target word to retrieve.

Siddique and Sufyan Beg (2019) propose a comprehensive review of the literature on the topic, on which we base our present discussion. Siddique and Sufyan Beg remark that there are few works addressing the task as such. They trace the research on this topic back to a patent held by H. V. Crawford and J. Crawford (1997). The works surveyed by Siddique and Sufyan Beg (2019) can be grouped in four classes: document-based models, graph-based models, vector space-based models and neural language model-based models.

The first class of models draws heavily from the field of Information Retrieval. The core idea is to match the input query words with the words contained in the definition gloss: the system will then return the definiendum for the definition

gloss that overlaps most with the input user query. To avoid problems of data sparsity, user queries are often expanded to include synonyms or other semantically related words, such as hypernyms or hyponyms. Works implementing this approach have been applied to multiple languages. The patent of H. V. Crawford and J. Crawford (1997), as well as the approach of R. Shaw et al. (2013) tackle English. Bila et al. (2004) address Japanese, and El Khalout and Oflazer (2004) study Turkish.

The second group of works listed by Siddique and Sufyan Beg (2019) is based on graph connectivity. Dutoit and Nugues (2002) use a resource called "The Integral Dictionary," and propose a two step algorithm that first delineates a subgraph to query, and then exhaustively looks up in the subset for the most similar entry to the query. Thorat and Choudhari (2016) use the graph structure of a dictionary (WordNet or Oxford English Dictionary). For each content word in a user query, they explore the sub-graph that surrounds it, and then rank all nodes they reached in their search to retrieve the most likely target word.

The third class of works attempt to transform the input query from the user into a vector. Méndez et al. (2013) propose to derive vectors from WordNet by selecting synsets that maximize a similarity measure, before performing a neighborhood search to extract the most relevant target word; Calvo et al. (2016) explore whether vectors obtained by other means, such as LDA, can be used instead of the WordNet-derived ones.

The last group corresponds to a work we have already mentioned above, that of Hill et al. (2016) (cf. Section 2.3). While Hill et al. (2016) suggested to use dictionaries as benchmarks for compositional semantics model, it is worth noting that the algorithmic approach they suggest was to use a LSTM to parse the

full definition gloss and use the hidden state at the last time-step to predict the definiendum. In effect, replacing the definition gloss with a user's query would lead to a reverse dictionary system. One precursor to the work of Hill et al. (2016) is that of Zanzotto et al. (2010), who used a shallow neural network to implement a compositional distributional semantics model and dictionaries as their training data.

Since the review of Siddique and Sufyan Beg (2019), a number of works have attempted to tackle the Reverse Dictionary task using a neural language model-based approach. The WantWords system (L. Zhang et al., 2020; Qi et al., 2020) is based on a BiLSTM architecture, and incorporates auxiliary tasks such as part-of-speech prediction to boost performances. Yan et al. (2020) seeks to replace the learned neural language models in Hill et al. (2016) or WantWords with a pre-trained model such as BERT (Devlin et al., 2019) and its multilingual variants, which allows them to use their system in a cross-lingual setting—querying in a language to obtain an answer in another. Most recently, Malekzadeh et al. (2021) used a neural-language model based approach to implement a Persian reverse dictionary.

2.4.2 Definition Modeling

The task of Definition Modeling, introduced by Noraset et al. (2017), generates a dictionary definition using a neural network. This network takes as input word embeddings, or neural vector representations of the word being defined. There are a number of applications of this task: for instance, Definition Modeling systems could provide definition drafts for under-documented languages. From a

linguist's point of view, there is theoretical significance in comparing two different types of semantic representations such as distributional word vectors and dictionary definitions. Where Definition Modeling systems find their core usefulness, however, is in their ability to explain the contents of word embeddings. They can translate computer-generated sequences of real numbers into dictionary definitions that humans will be able to comprehend without difficulty.

Definition Modeling was originally conceived as an evaluation tool by Noraset et al. (2017): the quality of the information captured in a word embedding should impact the quality of the definition generated. In fact, we should expect that results on the Definition Modeling reflect the contents of the input: if the embeddings we feed into our model lack semantic grounding, then we should expect the output definitions to be equally unanchored to the real world. This sort of confusion has been anecdotally attested in Definition Modeling systems. An example provided by Noraset et al. (2017) is listed in the definition 7:

feminine: of or pertaining to the human body (7)

This definition is obviously factually incorrect, as "*feminine*" should correspond pertain to females rather than "*the human body*".It is nonetheless interesting to see that this model got some elements right, even though the facts it states—how this symbol actually relates to the real world—are incorrect. This illustrative example should not be taken as the full extent of a Definition Modeling system's capability, but it is useful to stress how these models are reliant on the input word embeddings: we can see that the specific facts about a particular referent are difficult to retrieve from its distributional representation alone. It also stresses the breadth of work that remains to be done to obtain valid definitions.

Since the seminal work of Noraset et al. (2017), Definition Modeling has blos-

somed into a somewhat consequential body of work. Gadetsky et al. (2018) introduced the use of examples of usage as a secondary input; L. Yang et al. (2019) transposed the task into Chinese. A very noteworthy work here is that of Bear and Cook (2021), who introduce a cross-lingual English–Wolastoqey model. Further improvements and re-framing of the task have been suggested: for instance, H. Zhang et al. (2019) suggest training Definition Modeling architectures to also generate examples of usage, whereas Bevilacqua et al. (2020) stress how large, pre-trained models can be co-opted to perform impressively well on the task.

Nonetheless, authors in Definition Modeling cannot guarantee their artificial definitions to be factually correct. There are ways to mitigate this problem: in principle, we can give more information to the model, so that its guesses are more and more educated. For instance, authors have suggested providing the model with information on the definiendum, from the hypernymy relations it entertains (Noraset et al., 2017) to what sememes can be used to describe it (L. Yang et al., 2019).

Perhaps the piece of supplementary information that has been studied in most depth is contextual information. Gadetsky et al. (2018) first proposed to use examples of usage to deal with polysemy. H. Zhang et al. (2019) require their model to demonstrate the ability to use a definiendum coherently, by having it produce an example of usage as an auxiliary task. The model of Bevilacqua et al. (2020) functions by transforming a context with a highlighted word into a definition for this word.

Despite all this inventiveness, Definition Modeling systems are derived from text alone. In that, they provide a very good illustration of the sort of issues

highlighted in the thought experiments previously presented in Section 2.2. It may help to contrast what we expect from the dictionary to what we can glean from neural networks trained on Definition Modeling. Why do we trust what's written in a dictionary? At the most basic level, this has to do with the fact that lexicographers are humans. We can trust that the person who wrote a definition knows the world around them, that they are not completely clueless about the real-world object they are trying to define. When asked to define poppy, for instance, we can rely on our experience with poppies—we know what they look like, perhaps we know what they smell like, and that guides our definition-writing. A neural network, on the other hand, has no such experience: it has no eyes to see with, no nose to smell with, no experience to recall. It may be able to infer that a word such as flower should appear in the definition of poppy, but there's very little preventing it from producing a definition such as "*a blue flower*".[5]

In all, ensuring the factual correctness of these models remains an open question. Hence some have advocated side-stepping text generation altogether, like what is done in the related task of definition extraction from text (Navigli and Velardi, 2010). This task has seen recent interest, owing to the shared task of Spala et al. (2020). All this goes to showing that the generative aspect of definition modeling is perceived as a challenge in the NLP community.

[5]In fact, we may expect text-based neural networks to produce such errors, to a certain extent. As their input is solely text, they should be sensitive to whatever is written. Given the reporting bias of human speakers—viz., that we do not tend to state the obvious—we can expect texts to state the color of a poppy if and only if it is unusual, as for instance is the case with Himalayan blue poppies. This could in principle bias distributional models into associating *poppies* with *blue*.

2.5 Conclusions

Meaning is one of the more complex aspects of language. It is an immediate experience for any speaker, but properly explaining what it is, or how it comes to be, remains an arduous enterprise. It stands to reason that the same issue is to be found in NLP, the field of study that deals with the mechanization of language. As we discussed in Chapter 1, research in NLP has yet to establish firmly what counts and what doesn't count as a meaning representation, and this question is becoming all the more crucial as we witness neural networks growing ever more complex, and their productions ever more similar to what humans would produce.

Dictionaries, on the other hand, are curated inventories of definitions, as we saw in Section 2.1. They attempt to describe word meanings as objectively as possible. In that, they have proved to be an invaluable asset to NLP studies in semantics, as they provide dense descriptions in natural language that can be leveraged to inject semantic information in NLP models. In particular, we saw how dictionaries are a privileged vantage point to study semantic grounding in Section 2.2.

In this chapter, we have summarized how dictionaries have traditionally been used in NLP. We showed how they have been used to gather semantic information in Section 2.3. We more specifically looked into two domains of study: reverse dictionaries in Section 2.4.1 and the Definition Modeling task in Section 2.4.2.

This overview has underscored some existing gaps in the literature. Semantic grounding is far from being a solved question. Both reverse dictionaries and

the definition modeling tasks are understudied, compared to other NLP applications such as machine translation and or image captioning. As a consequence, there are legitimate concerns about how our models should be assessed, and what metrics are most fit to our purposes. These concerns will be explored at greater lengths in future chapters.

II

Distributional Semantics vs. Dictionaries

3

Comparing Dictionaries and DSMs using Topographic Similarity

The cracks weren't visible in the beginning
Now we're staring deep down into the guts of the earth
We've been watching these two tectonic plates slowly drifting apart
And when the first cracks appeared on the surfaces
We had surrendered our hopes to reality

— The Ocean Collective, *Ordovician: The Glaciation of Gondwana*

Previously in Chapter 1 and Chapter 2, we have detailed how distributional semantics and dictionaries fit within the larger framework of NLP. We have seen that they correspond to very different constructions: vectors on the one hand and sequences of text on the other.

These two different natures do not entail that we have to deal with incommensurable objects, and limit ourselves to manual annotations and qualitative observations. As we surveyed briefly in Section 2.3, a group of works has at-

This chapter is based on a previous publication (Mickus, Bernard, et al. 2020, "What Meaning-Form Correlation Has to Compose With: A Study of MFC on Artificial and Natural Language").

tempted to merge the two kinds of semantic descriptions. While the aims of all these works differ, from studying compositional semantics to building an explanatory tool for contextual embeddings, these models all attempt to vectorize definition glosses. Hill et al. (2016) propose to build a compositional semantics model. The dict2vec model of Tissier et al. (2017) constructs definienda embeddings from definition gloss contexts. Bosc and Vincent (2018) constructs an auto-encoder from definition glosses. Chang and Y.-N. Chen (2019) use sentence encoders to vectorize glosses, and map them to contextualized definiendum embeddings.

This common approach of converting glosses into vectors can be questioned. Here, we advocate a more direct approach. We can leverage the fact that texts and vector spaces are both metrizable: it is possible to mathematically define a distance between two sentences, much as we define the Euclidean distance between two vectors.

In this chapter, we will focus on measuring topographic similarity using Mantel tests, which we will review in Section 3.1. A crucial question for us to answer is whether topographic similarity computations yield linguistically coherent results, especially when it comes to *natural language*. We will then turn to assessing the validity of this methodology on the sort of issues we are interested in: comparing vectors and sequences of symbols, so as to delineate the confounding factors we are likely to encounter in Sections 3.2 and 3.3. We will finally perform our comparison of definitions and embeddings: we will measure topographic similarity on a dataset of comparable embeddings in Section 3.4, and then replicate and expand on this first study, this time using off-the-shelf pre-trained models in Section 3.5. We will close on a summary of our findings in Section 3.6.

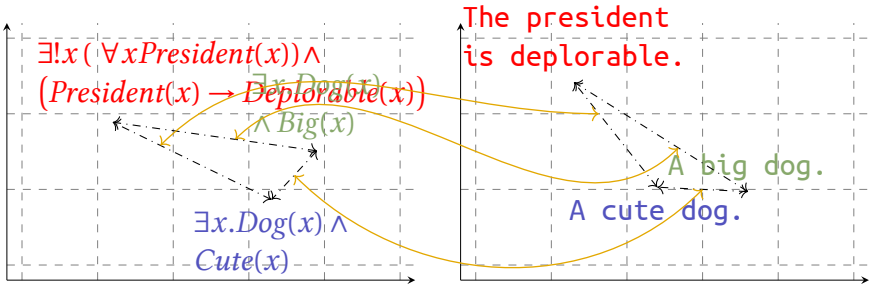


Figure 3.1: Overview of topographic similarity computations

3.1 Topographic Similarity

Let us consider two semantically similar words, such as "*dog*" and "*cat*". On the one hand, we may expect their distributional vector representations to be close to one another. On the other hand, the definitions that we give for these words are likely to be similar as well—both of them are likely to be noted as domesticated mammals. We can contrast these expectations to what we would observe if we compared "*dog*" and "*mammoth*". Both their distributional vectors and their definition glosses are likely to be further apart than that of "*dog*" and "*cat*". And the words "*dog*" and "*torque*" would be even further apart. Provided that we can measure the similarity between any two embeddings, and between any two definition glosses, this observation can be rephrased as follows: the distance between two definienda embeddings should correlate with the distance between their associated glosses. We refer to this characteristic as *topographic similarity*: the metric space for distributional representations should in principle display the same similarity structure as the metric space for definition glosses.

To evaluate whether two different spaces display the same structure, we will compute the correlation between distance measurements for two metric spaces.

A toy example of this process is described in Figure 3.1. In the figure are displayed two different metric spaces: one for logical formulas on the left, and one for graphical sentences on the right. We assume that some distance metric is defined within each space, represented in dashed lines. In the present chapter, our interest lies in comparing distributional semantics vectors with definition glosses: our experiment will therefore compare the textual distance between two definitions with the vector distance between two embeddings.

3.1.1 Measuring topographic similarity with Mantel tests

The methodology we employ to measure topographic similarity in this chapter is based on Mantel tests (Mantel, 1967). The initial requirements to run a Mantel test are that we possess:

(i) two distance metrics d_1 and d_2

(ii) a set of items S such that for any pair of items drawn from the set $\langle i_m \in S, i_n \in S \rangle$, the two distances d_1 and d_2 are defined and computable.

The two distances d_1 and d_2 can be seen as describing metric spaces M_1 M_2, which we will compare at the locations in M_1 and M_2 that correspond to the items $i_m, i_n \cdots \in S$ in our set S. Assuming these requirements are satisfied, we can define two distance matrices of shape $[\#S \times \#S]$:

$$\begin{aligned} (D_1)_{mn} &= d_1(i_m, i_n) && \text{with} \quad i_m, i_n \in S \\ (D_2)_{mn} &= d_2(i_m, i_n) && \text{with} \quad i_m, i_n \in S \end{aligned} \tag{3.1}$$

Since a distance metric should assign 0 to the distance between an item and itself,

the diagonal of D_1 and D_2 is equal to 0. Moreover, note that the two matrices are symmetric: for all indices m and n, we have $(D_1)_{mn} = (D_1)_{nm}$ and $(D_2)_{mn} = (D_2)_{nm}$. As such, the triangular matrices of D_1 and D_2 are sufficient to encode all distances $S \times S$.

These distance matrices can be considered as parallel sequences of observations O_1 and O_2:

$$\begin{aligned}(O_1) &= \left((D_1)_{0,1} \dots (D_1)_{0,\#S}, (D_1)_{1,2} \dots (D_1)_{\#S-1,\#S}\right) \\ (O_2) &= \left((D_2)_{0,1} \dots (D_2)_{0,\#S}, (D_2)_{1,2} \dots (D_2)_{\#S-1,\#S}\right)\end{aligned} \tag{3.2}$$

Such parallel sets of observations can be used to derive a measure of correlation that will quantify whether high values in O_1 will correspond to high values in O_2. This is generally done using a Pearson correlation coefficient, also known as Pearson's r. Given two series of observations $X = (x_1 \dots x_n)$ and $Y = \left(y_1 \dots y_n\right)$, Pearson's r is defined as:

$$r_{X,Y} = \frac{\sum_i^n (x_i - \mu_x)(y_i - \mu_y)}{\sqrt{\sum_i^n \left(x_i - \mu_x\right)^2} \cdot \sqrt{\sum_i^n \left(y_i - \mu_y\right)^2}} \tag{3.3}$$

where μ_x and μ_y are the mean values of X and Y respectively. The core idea behind Pearson's correlation coefficient is that we want to assess whether observations x_i and y_i have the same linear behavior: Pearson's r measures whether high x_i values entail high y_i values On a formal level, Pearson's correlation coefficient is highly related to the cosine function (cf. Equation (1.23)): if we consider X and Y as vectors of dimension n, then Pearson's r corresponds to the cosine

between the mean-centered vectors X and Y, $\cos\left(X - \mu_X \cdot \vec{1}_n, Y - \mu_Y \cdot \vec{1}_n\right)$, where $\vec{1}_n$ is the vector of dimension n with all components equal to 1. In other words, Pearson's r corresponds to removing the spurious co-directional components—as $\mu_X \cdot \vec{1}$ and $\mu_Y \cdot \vec{1}$ are biases inherited from the mean value in X and Y—before testing whether X and Y are co-directional. As a consequence, Pearson's r is bounded to real values within $[-1; 1]$; where 1 corresponds to two perfectly correlated sets of observations, -1 to two anti-correlated sets, and 0 to uncorrelated sets.

Another related measure of correlation is the Spearman correlation coefficient, also known as Spearman's ρ. It measures whether the two set of observations correspond to the same ordering of values. Hence it can be seen as a relaxation of Pearson's correlation coefficient, where no emphasis is put on ensuring that the two sets observations X and Y are in a linear scaling relationship. On a formal level, it is equivalent to computing Pearson's r, but replacing raw scalar value with rank information instead. Its definition relies on a ranking function that computes the number of items greater than i in some set S:

$$\text{rank}(i, S) = \#\{i' | i' \geq i \wedge i \in S\} \tag{3.4}$$

Using this function, we can define the Spearman correlation coefficient between two sets of observations $X = (x_1 \dots x_n)$ and $Y = \left(y_1 \dots y_n\right)$ as:

$$\begin{aligned} \hat{X} &= (\text{rank}(x_1, X), \dots \text{rank}(x_n, X)) \\ \hat{Y} &= \left(\text{rank}(y_1, Y), \dots \text{rank}(y_n, Y)\right) \\ \rho_{X,Y} &= r_{\hat{X},\hat{Y}} \end{aligned} \tag{3.5}$$

where $r_{A,B}$ is the Pearson correlation coefficient between A and B. Here, unless specifically noted, we will rely on Pearson's correlation coefficient.

Taken together, we call the topographic similarity τ of two metric spaces the correlation between the corresponding triangular distance matrices:

$$\tau_{M_1,M_2} = \text{correl}\,(O_1, O_2) \tag{3.6}$$

Where the correl function corresponds to Pearson's r, Spearman's ρ, or some other correlation coefficient.

Mantel tests include one additional step to derive a p-value, i.e., a quantification of the statistical significance of the topographic similarity τ. By randomly permuting one of the two sets of observations, O_1 or O_2, we can measure the topographic similarity τ' we obtain for a random pairing of distance measurements in the distance matrices D_1 and D_2. Repeating this random permutation measurement multiple times yields an estimate of what our baseline expectations for the topographic similarity of the metric spaces M_1 and M_2 evaluated on the set of items S ought to be. We can then compare our actual measurement τ to this baseline expectation to derive statistical significance: if it is among the 5% highest observed similarity scores τ', we will assign it a p-value of 0.05 or below, if we can establish a stricter cutoff. It is also frequent to use these baseline expectations τ' to produce a z-score, which we will refrain from doing here.

3.1.2 Relevant Applications

One important point to note is that this technique of topographic similarity is generally used for purposes other than the one we espouse here. Two relevant

domains of application correspond to research on semantic compositionality and on arbitrariness of the sign.

One suggestion, that can be traced back to Kirby (1999), and that is fully operationalized in Kirby (2001), Brighton and Kirby (2006) or Kirby, Cornish, et al. (2008), is that compositionality can be measured as a correlation between meaning and surface form (i.e., the sequence of tokens): as the components change, so should the composed meanings. This can be viewed as a topographic similarity between the meaning space and the form space. Topographic similarity has therefore been employed as a way to both detect and quantify compositionality in the field of emergent communication (Kirby, Cornish, et al., 2008; Kirby, Tamariz, et al., 2015; Spike, 2016; Ren et al., 2020, a.o.) which studies agents (artificial or human) who have to produce messages in order to express well defined meanings.

Other implementations of topographic similarity include studies centered on correlations between form and meaning at the word or sub-morphemic level—conflicting with the assumption of arbitrariness of the sign (Saussure, 1916). These studies generally use distributional representations to derive meaning distances. Gutiérrez et al. (2016) combine topographic similarity with kernel regression, in order to derive the most appropriate distance between word forms. This research trend has been applied to numerous languages: Kutuzov (2017) transfers this line of reasoning to Russian, whereas Dautriche et al. (2017) survey 100 different languages. We especially note the work of Pimentel et al. (2019), which use an information-theoretic formulation of the problem and rely on manual semantic resources to compute meaning distances.

3.2 Topographic similarity and Artificial Languages

Topographic similarity-based assessments implicitly assume that any change in one of the metric spaces should correspond to some change in the other. In the case of natural language processing, this can however be challenged: for instance, synonyms and paraphrases will introduce changes in form that should not entail change in meaning. Thus we expect topographic similarity to be sensitive to such phenomena, and this in turn suggests that factors such as synonymy could overpower the metric space similarity that we wish to detect using topographic similarity. To approach this question, we generate artificial languages containing varying degrees of potential confounding factors.

3.2.1 Methodology

Our experimental protocol consists in generating artificial languages with varying properties, and see what impact they have on topographic similarity measurements. All of our artificial languages are sets of paired representations. For simplicity, we refer to one of these representations as the underlying *meaning*, while the other will be the formal *message*. We represent meanings as binary vectors of five components, whereas messages are sequences of symbols. We refer to each of the five semantic dimensions as a *concept*. In most cases, the value of a concept will be denoted in a message by a specific symbol, which we call its *expression*. An example illustration for two meaning binary vectors $\vec{B}_1$ and $\vec{B}_2$ paired to their respective expressions F_1 and F_2 is shown in Figure 3.2.

As we are interested in whether topographic similarity accurately captures

$$\vec{B}_1 = \langle\ 0\ 0\ 0\ 1\ 1\ \rangle$$
$$\downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow$$
$$F_1 = \langle\ 5\ 6\ 1\ 20\ 12\ \rangle$$

$$\vec{B}_2 = \langle\ 0\ 0\ 1\ 1\ 1\ \rangle$$
$$\downarrow\ \downarrow\ \downarrow\ \downarrow\ \downarrow$$
$$F_2 = \langle\ 5\ 6\ 18\ 20\ 12\ \rangle$$

Figure 3.2: Artificial languages: basic setup

whether two different manners of encoding the same information are equivalent, we will design our languages so that some of the concepts are systematically expressed conjointly by unanalyzable holistic expressions—i.e., using symbols that cannot be attributed to any single concept, but rather correspond to a group of concepts at once. We generate languages where the values of the first h concepts are systematically expressed through a single expression, and the other $5-h$ are left untouched, with h varying from 1 to 5. When $h = 1$, the language is entirely compositional; when $h = 5$, the language is entirely holistic. Purely holistic messages should not display a structure similar to that of our meaning vectors, whereas purely compositional messages should display a one-to-one mapping between message symbols and meaning component values. We therefore expect the degree of *holisticity* displayed by a message to be inversely proportional to the topographic similarity scores. A visual depiction of the effects of $h = 1$ is presented in Figure 3.3.

$$\vec{B} = \langle\ 0\ 0\ 0\ 1\ 1\ \rangle$$
$$\searrow\swarrow\ \downarrow\ \downarrow\ \downarrow$$
$$F = \langle\ 0\ 1\ 20\ 12\ \rangle$$

Figure 3.3: Artificial languages: holisticity

The first confounding factor we consider is *synonymy*: as previously noted,

synonyms entail a variation in form that is not coupled with a variation in meaning. To model this phenomenon, we generate languages in which any single value of a concept can equally be expressed by s different expressions, with s ranging from 1 to 3. As a consequence, when $s = 1$, the language exhibits no synonymy, whereas if $s > 1$, the language will contain pairs such as those in Figure 3.4.

$$\begin{array}{lccccccc} \vec{B}_1 = \langle & 0 & 0 & 0 & 1 & 1 & \rangle \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\ F_1 = \langle & 5 & 6 & 1 & 20 & 12 & \rangle \end{array}$$

$$\begin{array}{lccccccc} \vec{B}_2 = \langle & 0 & 1 & 1 & 1 & 1 & \rangle \\ & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\ F_2 = \langle & 5 & 3 & 13 & 17 & 12 & \rangle \end{array}$$

Figure 3.4: Artificial languages: synonymy

Moreover, we expect that topographic similarity measurements might be influenced by the presence of *semantically ungrounded elements*—i.e., elements not associated with any concept or combination of concepts. We therefore generate languages where u specific ungrounded symbols appear once in every message at randomly chosen positions, with u varying between 0 and 3. In languages where $u = 0$, the language contains only semantically grounded expressions. In languages where $u > 0$, we have instead the behavior exemplified in Figure 3.5; more precisely this figure would correspond to a parameter $u = 1$.

$$\begin{array}{lccccccc} \vec{B} = \langle & 0 & 0 & 0 & 1 & 1 & \rangle \\ & \swarrow & \swarrow & \swarrow & & \searrow & \searrow & \\ F = \langle & 5 & 6 & 1 & 14 & 20 & 12 & \rangle \end{array}$$

Figure 3.5: Artificial languages: semantically ungrounded elements

Finally, we consider the case of *paraphrases*, sentences of different forms but

equivalent meanings. For a language to contain paraphrases, it must be able to express a single meaning with different messages. This is the case in our artificial languages that exhibit synonymy or contain semantically ungrounded elements; the variation they introduce allow distinct messages to have the same meanings. We thus generate languages for which p messages are produced for each meaning before dropping possible meaning-message pair duplicates. p ranges from 1 to 3. If $p = 1$, the language contains no paraphrase. This p parameter leads to languages containing pairs such as those in Figure 3.6.

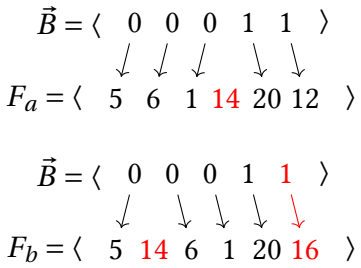


Figure 3.6: Artificial languages: paraphrases

We test all possible combinations of these four parameters. We also include *random baselines* where we assign meanings to random sequences of symbols, either of an arbitrarily fixed length of 5 symbols, or of a length chosen uniformly between 1 and 10. We generate 50 artificial languages for every combination of parameters to help us distinguish the stable effects of our parameters from spurious accidents due to our random generation process. We refer to each of the 50 generation processes as a separate *run*.

We compute Mantel tests using the Hamming distance between meaning

vectors—i.e., the number of differing components:

$$h(\vec{x}, \vec{y}) = \sum_{i}^{d} \mathbb{1}_{\{x_i \neq y_i\}} \tag{3.7}$$

As for message, we use the Levenshtein distance, also known as edit distance. The Levenshtein distance corresponds to the minimum number of editions (substitutions, deletions or additions) necessary to convert a string of characters A into a string of character B:

$$d_l(A,B) = \begin{cases} \max(\#A, \#B) & \text{if} \#A = 0 \vee \#B = 0 \\ d_l(A_{1\ldots m}, B_{1\ldots n}) & \text{if} A_0 = B_0 \\ 1 + \min \begin{pmatrix} d_l(A, B_{1\ldots n}), \\ d_l(A_{1\ldots m}, B), \\ d_l(A_{1\ldots m}, B_{1\ldots n}) \end{pmatrix} & \text{otherwise} \end{cases} \tag{3.8}$$

Here, A is of length $m+1$, B is of length $n+1$, and SA_i and B_i refer to the i^{th} (zero-indexed) character in the strings A and B respectively. In the present experiment, we normalize Levenshtein distance by the maximum length of the two messages.

$$\hat{d}_l(A,B) = \frac{d_l(A,B)}{\max(\#A, \#B)} \tag{3.9}$$

For each language, we study the corresponding p-value and the correlation score. For every combination of parameters, we study its average p-value and correlation score across all runs.

One limitation of this method is that our modeling may not comply fully with natural language—in particular, the existence of exact synonyms is debatable; likewise, natural function words do possess some semantic content, whereas our ungrounded symbols do not. Neither do we claim to conduct an exhaustive study of all relevant phenomena.

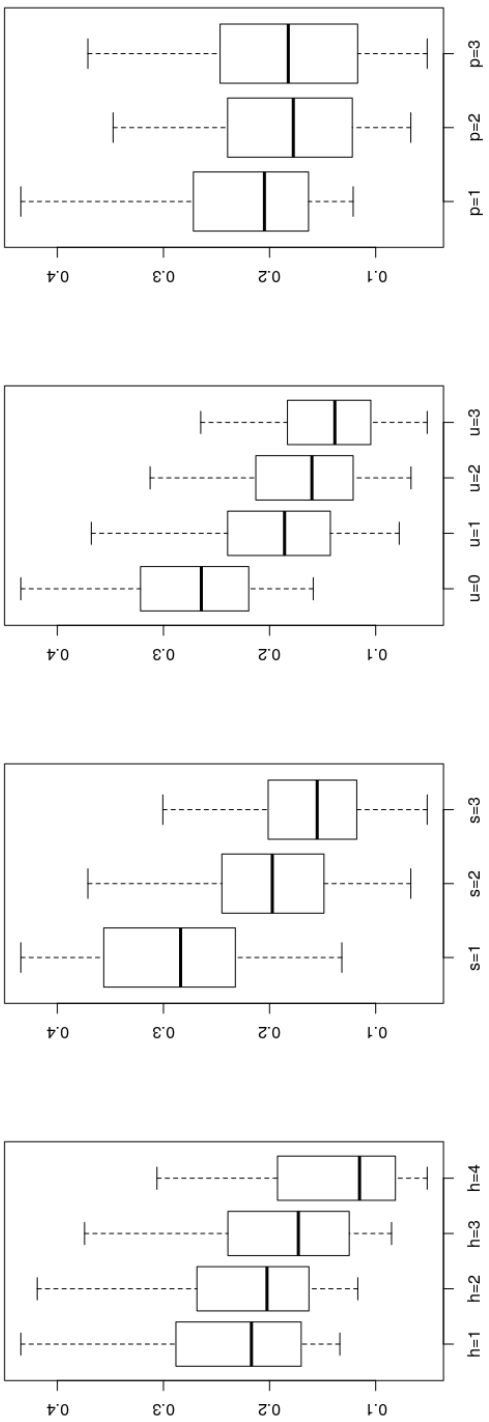
3.2.2 Results

A visualization of the results for the variation factors is shown in Figure 3.7. Each sub-figure corresponds to a different factor, and shows the distribution of topographic similarity scores according to the possible levels for that factor. As expected, random baselines were found to be insignificant (p-value ≥ 0.05).

If we focus on holisticity (Figure 3.7a), we do see that less compositional languages yield lower topographic similarity scores. When we consider the correlation values averaged over all 50 runs, we see that no holistic parameter configuration (where $h = 5$) is found to be significant, resulting in the missing boxplot in Figure 3.7a. 1^{st}, 2^{nd} and 3^{rd} quartiles are found to consistently decrease for higher values of h.[1] This tells us that topographic similarity is indeed sensitive to the structural similarity of meaning representations and message representations.

Synonymy and semantically ungrounded elements are found to be confounding factors (Figure 3.7b and Figure 3.7c). Higher values for the s and u parameters systematically entail that the distribution of topographic similarity scores

[1]Some languages with $h = 5$, which are fully holistic, were found to yield significant topographic similarity. Most of these also included multiple ungrounded symbols ($u \geq 1$). This can be explained by the effects of paraphrases: in holistic languages with multiple messages per meaning containing ungrounded symbols, paraphrastic messages for a given meaning differ only by their ungrounded symbols, whereas messages for different meanings will also differ by their grounded symbols—leading to nonzero correlation. However, on average over all 50 runs, the p-value for any of these settings is below our threshold.



(a) Holisticity (b) Synonymy (c) Ungrounded elements (d) Paraphrase

NB: No $h = 5$ is found significant.

Figure 3.7: Topographic similarity for artificial languages, grouped by parameter (significant items only, avg. of 50 runs)

(averaged over 50 runs) is globally lower, as 1st, 2nd and 3rd quartiles consistently decrease.

Lastly, for non fully-holistic settings ($h < 5$), we observe that some combinations of factors fail to produce significant topographic similarity scores. In 15.2 % of all possible factor combinations, this persists even when averaged over all 50 corresponding runs. Hence we conclude that this is an actual effect of the interaction of factors. All these languages are defined with at least one extreme factor: viz. either three synonyms per concept ($s = 3$), three ungrounded symbols ($u = 3$) or four concepts merged into a single expression ($h = 4$). Moreover all of them (except for two languages defined with $h = 4$, $s = 3$ and either $u = 2$ or $u = 3$) contained a single message per meaning ($p = 1$). Confirming this trend, we find that all non fully-holistic languages with up to three messages per meaning ($p = 3$) were found to have a significant topographic similarity on average. These shared characteristics can hint at the fact that confounding factors can significantly obfuscate the structural similarity of the two metric spaces. An alternative explanation could be that languages without paraphrases ($p = 1$) contain fewer messages and thus yield higher p-values, whereas paraphrases additionally entail that very low textual distances map to zero meaning distances.

3.2.3 Discussion & Conclusions

We observed that synonymy (Figure 3.7b) and ungrounded elements (Figure 3.7c) seemed detrimental to topographic similarity scores, whereas the effects of paraphrases were found to be more subtle (Figure 3.7d). We quantify this by computing a simple linear model in R (R Core Team, 2018) where the correlation

Coeffs.	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	0.402781	0.001936	208.07	$< 2 \cdot 10^{-16}$
$h = 2$	−0.016909	0.001577	−10.72	$< 2 \cdot 10^{-16}$
$h = 3$	−0.057501	0.001577	−36.46	$< 2 \cdot 10^{-16}$
$h = 4$	−0.113547	0.001577	−72.00	$< 2 \cdot 10^{-16}$
$h = 5$	−0.197191	0.001734	−113.75	$< 2 \cdot 10^{-16}$
$s = 2$	−0.096726	0.001748	−55.34	$< 2 \cdot 10^{-16}$
$s = 3$	−0.137808	0.001748	−78.85	$< 2 \cdot 10^{-16}$
$u = 1$	−0.099435	0.001524	−65.26	$< 2 \cdot 10^{-16}$
$u = 2$	−0.126905	0.001524	−83.29	$< 2 \cdot 10^{-16}$
$u = 3$	−0.145707	0.001524	−95.63	$< 2 \cdot 10^{-16}$
$p = 2$	0.031989	0.001354	23.63	$< 2 \cdot 10^{-16}$
$p = 3$	0.041219	0.001354	30.45	$< 2 \cdot 10^{-16}$

Table 3.1: Linear model of correlation with parameters as predictors. Intercept: $h = 1, s = 1, u = 0, p = 1$.

score is the dependent variable and the values of the four parameters are the predictors; data points correspond to specific runs. Results are reported in Table 3.1. For each coefficient (i.e., predictor value), we list its estimated coefficient in the regression, the standard residual error not captured by the model, the corresponding t-statistics (i.e., the number of standard deviations that separate the value assigned to a predictor from what we would observe in the case of the null hypothesis being true), and the probability of it being a significant factor.

While $h = 5$ was found to be the predictor with the strongest negative effect on topographic similarity scores, we found that factors $s = 3$, $u = 3$ and $u = 2$ had stronger effects than $h = 4$. In short, the model shows that factors such as synonymy impact topographic similarity measurements—sometimes to a greater extent than structural similarity as shown by t-value scores. It also stresses that paraphrases positively impact topographic similarity scores: in languages where $p > 1$, any single unreliable message is less likely to whittle down scores.

In all, our experiment suggests that taking topographic similarity applied to natural sentences comes with significant challenges. Factors that we expect from natural language, such as ungrounded symbols and synonyms, obfuscate the clear relationship between structural similarity and topographic similarity scores. At times, these factors can even annihilate the interpretability of topographic similarity scores for generated languages. Yet structural similarity does impact measurements: therefore, while topographic similarity scores in and of themselves may not be sufficient to establish or reject that two spaces encode similar information, they can serve as a diagnosis tool.

3.3 Topographic similarity and Sentence Encoders

We may expect another type of confounding factor arising from the models we use to compute semantic representations. As Wieting and Kiela (2019) point out, randomly initialized and untrained neural architectures can perform surprisingly well on some tasks, despite not being able to produce linguistically meaningful representations. If we are to use neural embeddings, it is important that we assess how topographic similarity copes with embeddings from untrained neural networks.

3.3.1 Methodology

In this experiment, our approach will be to compute topographic similarity for natural language data—more specifically, for sentences. We will consider two types of metrics: text-based metrics and vector-based metrics.

With respect to textual metrics, we will use the Levenshtein distance (defined

over words rather than characters), as well the Levenshtein distance normalized by sentence length. Moreover, as we saw in Section 3.2, extraneous factors such as synonymy or ungrounded tokens may impact our measurements. To check whether these factors also impact natural language examples, we perform simple modifications of our original process. To control for synonymy, we replace every word by the first lemma of its first synset in WordNet (Fellbaum, 1998), if any such lemma can be found. To control for ungrounded symbols, we remove stopwords from our sampled sentences.

The vector distances we use in this section rely on *sentence encoders*, computational models that convert sequences of tokens into vector representations. They can be trained on a variety of tasks, from predicting the entailment relation between a pair of sentences (Conneau et al., 2017) to reconstructing the context of a passage (Kiros et al., 2015). These tasks require capturing the meaning of the corresponding texts.

We first verify whether the distances over these vector spaces correspond to human judgments. If so, this would allow us to consider our embeddings as linguistically motivated. We expect sentence embeddings to anti-correlate to human similarity ratings: if sentence encoders capture sentence semantics, then words that humans judge to be highly similar in meaning should not be far from one another in the embedding spaces. We therefore compute the Spearman correlation between the human ratings present in the SICK benchmark (Marelli et al., 2014) and the cosine and Euclidean distances between the two corresponding sentence embeddings. SICK consists in a series of paired sentences, matched with a human rating of their semantic similarity. As such, we expect that distance between representations of sentence meaning should significantly anti-correlate

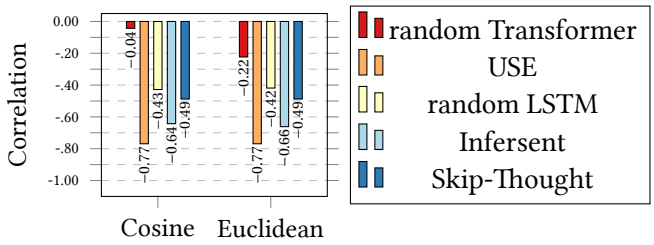


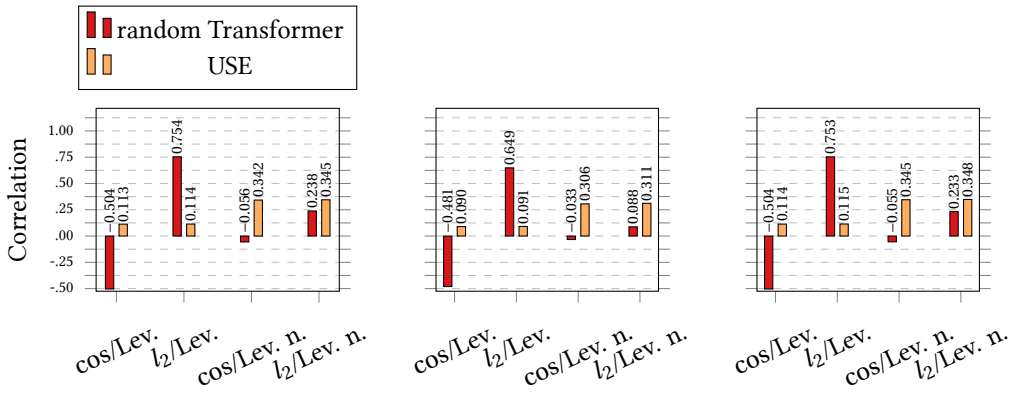
Figure 3.8: Meaning distance metrics evaluated on the SICK dataset. (Spearman correlation)

with human semantic similarity judgments.

Figure 3.8 summarizes correlation scores for a few sentence encoders. First is Skip-Thought (Kiros et al., 2015), a LSTM-based encoder trained to produce vector representatiosn that contain the information necessary to reconstruct the previous and next sentences. Next is Infersent (Conneau et al., 2017), a biLSTM model trained on the NLI task. The third and final model is the Universal Sentence Encoder (Cer, Y. Yang, et al., 2018, USE for short), trained on multiple tasks at once, which include sentence classification and an objective similar to that of Skip-Thought. Lastly, we include randomly initialized and untrained Transformer and LSTM models. We observe that USE yields the most consistent semantic representations and thus decide to focus in the following on this particular model. To contrast the effects of training, we will also include the random Transformer in our experiments.

To compute actual topography similarity scores, we randomly sample 4 123 sentences from the Toronto BookCorpus (Zhu et al., 2015)—a collection of English books of various genres—for computing Mantel tests using Levenshtein distance, both raw and normalized, as the textual distance.[2] We repeat the procedure 5 times before averaging results.

[2]This corpus size is defined in relation to a pilot study.



(a) No control (b) Controlling for stop-words (c) Controlling for synonymy

Figure 3.9: Topographic similarity scores for natural language sentences.

3.3.2 Results

Results are presented in Figure 3.9: Figure 3.9a corresponds to the control-less scenario; Figure 3.9b and Figure 3.9c present the effects of controlling for stop-words and synonyms respectively. Results are consistent across all five random samples of sentences: standard deviation of topographic similarity scores is systematically below 0.016, and often below 0.005.[3]

Most striking are the very high correlations and anti-correlations that are yielded by the random baseline: the anti-correlations and correlations derived from non-normalized Levenshtein distance have a greater magnitude than what we observe for USE (which is also based on the Transformer architecture). In the case of Euclidean distance, this magnitude can partly be explained by the architecture itself, which computes a vector that is not meaningful (as we saw in Figure 3.8) but that is still computed in a very compositional way (cf. Section 1.2.2).

[3]Only the randomly initialized Transformer when using Euclidean distance and non-normalized Levenshtein distance yields standard deviations above 0.01 (in all three scenarios). Standard deviations for USE embeddings are all below 0.003, with the exception of the two setups involving normalized Levenshtein distance and controlling for stop-words (around 0.007).

Put simply, random Transformers are compositional, but the corresponding notion of composition is not linguistically justified. The simple sum used to derive the sentence embeddings from the hidden states at each time-step[4] entails that the norm of every sentence representation grows proportionally to the number of words it contains. Moreover the residual connections used in Transformers entail that the hidden state for a given time-step bears some trace of the input word at this time-step: therefore, sentences with words in common will tend to be nearer in the Euclidean space.

Turning to the topographic similarity scores for USE, we observe that normalizing Levenshtein distance leads to higher scores, which would suggest that sentence length is not a semantically relevant factor. On the other hand, cosine-based setups overall are found to decrease scores by a low margin, of 0.005 at most. Removing stop-words lowers the correlation for USE embeddings: scores in Figure 3.9b are found to be lower than those without any form of control by a margin ranging from 0.02 to 0.04. Lastly, while we technically observe a higher topographic similarity when controlling for synonyms (Figure 3.9c), the effect is very subtle: correlation increases by no more than 0.005.

3.3.3 Discussion & Conclusions

This second experiment first and foremost cautions us that topographic similarity does not discriminate between linguistically motivated and randomly generated representations: in some instances, we observe high correlation scores for random model outputs. This entails that any comparison that we perform using this

[4]Like Cer, Y. Yang, et al. (2018), we divided the sentence embedding by the square root of the length of sentence.

tool must factor in the inherent noisiness of our distributional representations. Topographic similarity therefore assumes, and does not control, that the inputs we feed it are linguistically motivated. While this assumption is obviously not satisfied for a random model, it is defensible for USE given its training procedure and the high anti-correlation with respect to human judgments observed above in Figure 3.8.

On a purely practical level, this experiment also shows that topographic similarity can be applied on human language. We were able to measure somewhat high correlation scores ($\tau > 0.34$). This suggests that the methodology is applicable to linguistic data, and more specifically to definitions.

3.4 Topographic similarity and definitions

We now return to our inquiry of topographic similarity in definitions and embeddings. The results of our two preliminary experiments lead us to adopt a more cautious stance. We have established in Section 3.2 that we expect confounding factors to impact our measurements. From Section 3.3, we know that the quality of an embedding is also bound to affect how we can interpret topographic similarity measurements. Taking these caveats into account, we now turn to measuring the topographic similarity between the two metrizable spaces described by word embedding and definitions. To that end, we will first establish the data we will use in Section 3.4.1, before describing our exact methodology and experimental results in Sections 3.4.2 to 3.4.4.

3.4.1 Dataset

In order to properly contrast the topography of embedding spaces with that of dictionary definitions, we need to establish a dataset of embeddings paired with dictionary definitions. The main question we will address here is how to select hyperparameters to train our models and data so as to ensure that our comparisons are fair. We will first conduct a pilot study, on which we will build to produce our final dataset of embeddings and definitions broadly comparable across settings.

Pilot study on hyperparameter setting

The first trait to consider, which we pointed out earlier in Section 1.4, is that word embedding dimensions may impact distance, norm, and cosine similarity of embeddings. Euclidean distance and cosine similarity are especially relevant to our present study on the topographic similarity of word embeddings and definitions. As such, to rule out a potential confounding factor, we choose to use a constant value of vector size for all embedding models to consider, which we set to $d = 256$. Nonetheless, dimension is but one hyperparameter to us: there are still numerous other for us to consider,d depending on the exact distributional models under consideration.

One possibility that we may consider to select hyperparameters is Bayesian Optimization. This algorithm consists in iteratively training models, with initially random hyperparameters: at each iteration, we select the hyperparameter configuration most likely to yield an increase in performance on some objective, according to Bayes' rule. We provide a more thorough presentation in

Appendix A.

Given that this hyperparameter selection method is not consensual in the community, we start by evaluating the impact it would have on our models. We focus on finding hyperparameters suitable for word2vec architectures on all four comparable corpora. To compare their performances, we will compare them on manual translations of the BATS dataset of Gladkova et al. (2016): we provide an overview of these translations in Appendix B. This analogy benchmark is structured in two levels: individual sub-sections instantiating specific analogical relations (e.g., "*animal—young*" or "*infinitive—past participle*") are then grouped into four super-sections: Inflection, Derivation, Lexicography, Encyclopedia. The former two correspond to morphological relations whereas the two latter are more closely aligned to common-sense reasoning.

We consider four languages to start with: English, Spanish, French and Italian. All the models are trained on comparable corpora; data was collected from three different domains to ensure a broad linguistic coverage: Wikipedia, movie subtitles, and literature. Wikipedia data was retrieved from existing dumps.[5] Movie subtitles data comes from the OpenSubtitles corpus (Lison and Tiedemann, 2016). The literature corpora come from various sources: Project Gutenberg[6] for English and French, Wikisource[7] for Spanish, and Liber Liber[8] for Italian. For each language, we made sure to select data in standard modern language varieties only. For the literature corpora, this involved discarding books published prior to a date which corresponds to when each language started being

[5] `https://dumps.wikimedia.org/`
[6] `https://www.gutenberg.org`
[7] `https://es.wikisource.org/`
[8] `https://www.liberliber.it/online/`

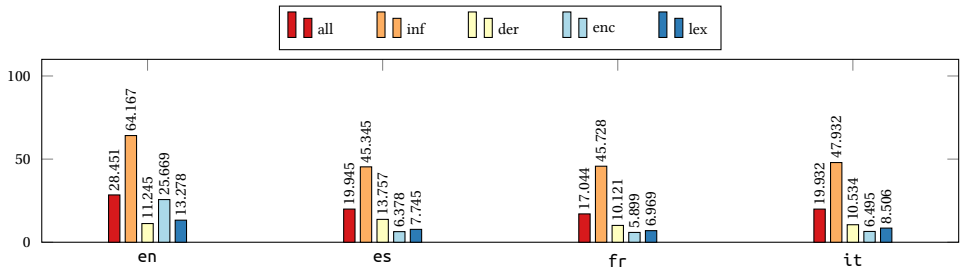
written in its modern form (around the 19th century depending on the language).

Pre-processing included normalization, cleaning, and tokenization. The following steps were performed with the aim of ensuring consistency across single languages. First, we normalized the data to the NFC form and re-encoded all texts to UTF-8. Then, we performed text normalization using regular expressions (quotes, diacritics, punctuation marks, trailing white spaces). After that, we removed metadata, headers and footers based on heuristics, such as identifying sentences at the beginning and end of files that contain metadata-related keywords. Lastly, we performed segmentation in sentences and tokenization. We used the `spaCy` library[9] to tokenize the data.

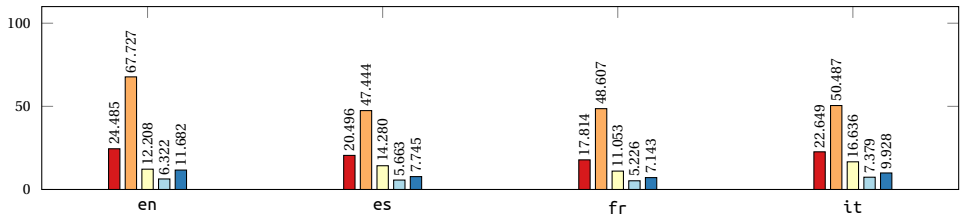
Word2vec models were trained on these corpora using the `gensim` library (Řehůřek and Sojka, 2010). We define the objective of this Bayes Optimization process as the performance over a random subset of our analogy datasets: for each sub-category, we select ten possible pairs of analogy instances, and attempt to maximize the accuracy of the models on these subsets. In total, we perform 50 iterations, the first 10 of which are purely random samples of hyperparameters to establish a prior distribution. This optimization process is implemented using the `scikit-optimize` library.[10]

Once we have determined the best model for each language using Bayesian Optimization, we compute its performances on the aforementioned analogy benchmarks (either the original BATS model of Gladkova et al. (2016) or our translations). The performances of these best models are displayed in Figure 3.10b. We also include the performances obtained by models with the default hyperparam-

[9] `https://spacy.io/`
[10] `https://scikit-optimize.github.io`



(a) Default hyperparameters (in %)



(b) Best hyperparameters (in %)

Figure 3.10: Hyperparameter selection: Results on BATS and translations

eters suggested in `gensim` in Figure 3.10a.

While tuning the hyperparameters does yield improvement, this has to be balanced with the intensive computations it requires. In fact, a more careful observation will show that this increase in performance is not statistically significant. For each of the four languages we consider in this preliminary experiment, we compute a Student's t-test for related samples between the tuned and untuned models, comparing their average accuracy on each subsection of our analogy benchmark.

A paired Student t-test such as the one we conduct here consists in evaluating whether two comparable sets of values are significantly different from one another. It assumes that we have two sets of measurements, A and B, of n measured values, and that we are able to link a measurement in A to a related measurement in B. Given that, we can compute a t-statistic based on the

mean-centered distributions of A and B:

$$\hat{A} = \left(A_1 - \mu_A \ \dots \ A_n - \mu_A\right)$$
$$\hat{B} = \left(B_1 - \mu_B \ \dots \ B_n - \mu_B\right)$$
$$t_{A,B} = \left(\sum_{i=1}^{n} \mu_A - \mu_B\right) \times \sqrt{\frac{n\,(n-1)}{\sum_{i=1}^{n}\left(\hat{A}_i - \hat{B}_i\right)^2}} \tag{3.10}$$

where μ_A and μ_B are the mean values of A and B respectively. In essence, the t-statistic corresponds to computing the difference in means $\mu_A - \mu_B$ to characterize how distinct A and B are, and then scaling this difference so that it is statistically interpretable. This scaling is related to the estimated standard deviation for the distribution of the mean-centered differences $(\hat{A}_1 - \hat{B}_1 \dots \hat{A}_n - \hat{B}_n)$, under the assumption that the average mean-centered difference is equal to 0. Simply put, the t-statistic compares the global difference $\mu_A - \mu_B$ to what one would expect if there was no difference between A and B.

We can then compare the t-statistic computed for A and B to what we would observe for completely unrelated observations, where differences could be imparted to simple random flukes in the measurements. In practice, this hypothetical baseline is defined as a t-distribution. Let $X = \mathcal{N}(0, \sigma)$ a normal distribution with mean O and standard deviation σ, and Y such that Y^2/σ^2 follows a χ^2 distribution with degree of freedom n—viz., $Y^2/\sigma^2 = \sum_{i=1}^{n} \mathcal{N}(0,1)$). From this, we can define the t-distribution with n degrees of freedom as:

$$t = \frac{X\sqrt{n}}{Y} \tag{3.11}$$

Using this t-distribution baseline, we can establish a p-value for the difference

between A and B. More precisely, we compute the probability of observing the t-statistic as defined in Equation (3.10) using a confidence interval over the t-distribution.

As for our pilot study on hyperparameter selections, the two sets we wish to compare correspond to the performances on the analogy dataset of Appendix B of a model using default parameters, and the best model obtained during our Bayesian Optimization hyperparameter selection. Our paired observations will correspond to the average performance on a subsection of the dataset, hence we have in total 40 different paired observations per language. We aim to assess whether the hyperparameter selection process yielded an increase in performance, i.e., whether the average accuracy on the analogy benchmark is greater after hyperparameter selection.

Language	p-value	t-stat
en	0.469	N/A
es	0.524	N/A
fr	0.213	N/A
it	0.003	3.224

Table 3.2: T-tests between tuned and untuned models

Results of these analyses are displayed in Table 3.2. Aside from Italian, the difference between tuned and untuned models were not found to be statistically significant, hence we do not report the t-statistic associated. To confirm these results, we also tried doubling the number of iterations of the Bayesian Optimization process for the French and English tuned models. While this did yield a slight increase in performance, the outcome still did not significantly differ from results of the two corresponding untuned models.

Language	with examples	without
en	0	806297
es	0	132583
fr	431793	573313
it	16127	86959
ru	122282	485208

Table 3.3: DBnary: number of items per language

There are two possible explanations for these puzzling results: either the Bayesian Optimization requires many more iterations before yielding any substantial improvement, or the performances we obtain with untuned word2vec models are already close to optimal. The latter case could be construed as evidence against the validity of the offset method to solve the word analogy task. In any event, taking these results in stride, we decide to train our embedding models using default hyperparameters. An important argument in favor of this decision is that our emphasis is on building comparable and controlled resources, rather than reaching state-of-the-art performances.

Selected embeddings & dictionary data

In the previous section, we have decided not to perform hyperparameter setting for our embedding algorithms. We have also started delineating comparable corpora in Section 3.4.1. The questions we now have to tackle are twofold: that of the embedding architectures and of the dictionary data we wish to include in our comparable datasets.

As a source of dictionary definitions, we will primarily be using the DBnary dataset (Sérasset, 2012).[11] It consists of an RDF-formatted version of some of

[11] http://kaiko.getalp.org/about-dbnary/

Language	N. Sents.	N. Tokens	N. Bytes
it	78761031	955474050	5001829910
es	78973969	975762257	5001999992
fr	82082118	1004767254	5001999368
en	97622760	1035154295	5001999755
ru	79526583	1035661601	10036395727

Table 3.4: Embeddings: corpus statistics

the existing Wiktionary projects.[12] Table 3.3 shows the DBnary dataset size for all four languages in our pilot study, plus Russian, which we introduce so as to introduce more typological and linguistic diversity to our observations.

To construct a comparable dataset for Russian, we select data from Wikisource, Wikipedia and OpenSubtitles. The corresponding corpora are displayed in Table 3.4. Note that we include twice the volume in bytes for Russian: this is due to idiosyncrasies of UTF-8 encoding, which generally uses 2 bytes for non-Latin characters. Comparing the number of sentences or tokens across languages nonetheless ensures us that the corpora are of similar size.

With that in mind, the last point to consider is which embedding architectures to include. Here, we focus on three architectures: word2vec models trained with gensim (Řehůřek and Sojka, 2010), the ELECTRA model of Clark, Luong, et al. (2020), and character-based embeddings. The word2vec and ELECTRA models were selected so as to provide some comparison between static and contextual embeddings; both are trained with default hyperparameters aside from output vector size, which we set to 256. As for the ELECTRA models, given that we need contexts to derive token representations, we train the models only in English, French and Russian. The French and Russian DBnary datasets both contain

[12] See https://www.wiktionary.org/

examples of usage that we can leverage to derive contexts for the embeddings of a word to be defined. While the English DBnary dataset does not contain examples of usage, they are nonetheless present in the original Wiktionary dumps. We therefore re-parse the English Wiktionary to extract examples of usage along with definition glosses and definienda. The same procedure did not yield convincing results for the Spanish and Italian Wiktionary projects, both of which contain too few examples of usage for any reliable downstream application.

The character-based embeddings are included to provide baseline expectations for non-semantic representations—as we can expect spelling to be more or less arbitrary with respect to word meaning. In practice, these embeddings are computed through a simple LSTM-based auto-encoder: the word is passed into an LSTM encoder as a sequence of characters, we sum all output hidden states, and use these summed hidden states to initialize an LSTM decoder, whose objective it is to reconstruct the input word. As a character-based representation, we can therefore use the summed output hidden states, as they are tailored to contain all the information necessary to reconstruct the spelling of the corresponding word. Given that we implement this module ourselves, we use a Bayesian Optimization algorithm to select hyperparameters for our five character auto-encoder. We use this process to decide learning rate, weight decay, dropout, β_1 and β_2 parameters of the AdamW optimizer, batch size, number of epochs over the full dataset, as well as whether to share a single weight matrix for encoder and decoder character embeddings. The datasets used to trained the models correspond to the set of all word types attested in our base corpora described in Table 3.4. All models achieve a 99% reconstruction accuracy.

3.4.2 Methodology

We now return to our primary concern: establishing whether word embeddings and definitions describe comparable metric spaces. As we explained above, we rely on the dataset we constructed in Section 3.4.1. We sample definitions for each of the five languages (English, Spanish, French, Italian, Russian). We construct two frequency registers for our samples: we either sample *common words*—viz., definitions whose definienda are among the 20% most common words in the embeddings' training corpus—or we sample *rare words*—viz., definitions whose definienda are among the 50% least common words in the embeddings' training corpus. To take into account effects due to sampling accidents, for each language, we construct five random samples of both kinds, and average results prior to any analysis.

As for confounding factors, we only check whether ungrounded symbols also impact natural language examples. We use the same control mechanism we introduced in Section 3.3. Namely, to control for ungrounded symbols, we remove stop-words from our definitions. We consider both Euclidean and cosine distances to measure the similarity between two embeddings. As for definition glosses, we consider the Levenshtein edit distance and the Levenshtein edit distance normalized by the maximum length of the two glosses. We also include he Jaccard index, viz. the "Intersection over Union," defined as:

$$d_J(A,B) = \frac{\#(A \cup B)}{\#(A \cap B)} \tag{3.12}$$

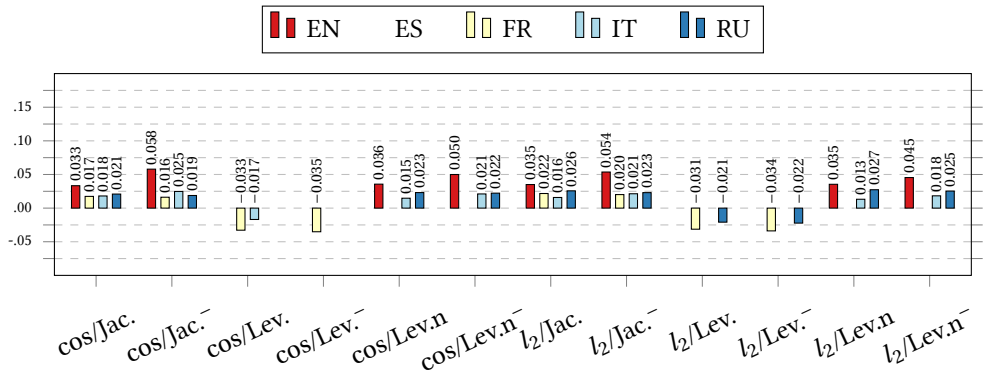
for two sets of tokens A and B.

3.4.3 Results

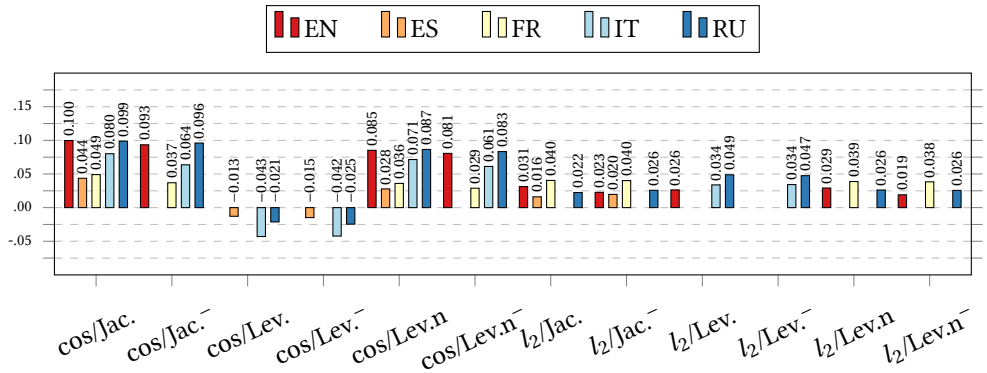
We summarize our findings in Figure 3.11 for common words, and in Figure 3.12 for rare words. Sub-figures correspond to different embedding architectures. Correlations measured on glosses from which stop-words were removed are indicated with a minus exponent $^{-}$. Statistically insignificant topographic similarity scores are not displayed.

Out of the total 312 correlations we compute, only 148 (47%) were found to yield a significant correlation, and 38 (12%) yielded significant anti-correlations. The remainder (126 setups, or about 40%) were found to be insignificant. Standard deviations of topographic similarity scores across all setups remain below 0.015 (with one exception, that English ELECTRA, using l_2/Lev.) and often below 0.008, with the exception of a few setups. In detail, out of the 3 setups yield a standard deviation above 0.01, all of which involve Euclidean distance. Furthermore, 9 more setups yield a standard deviation above 0.009, with no common trend between all the involved setups.

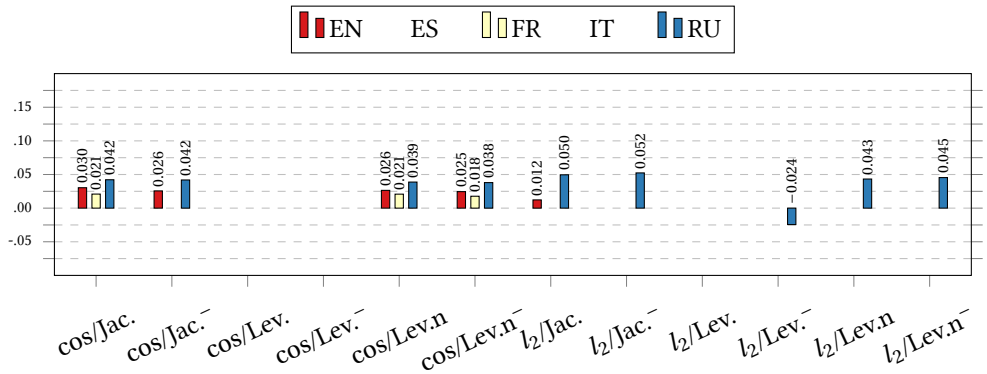
We now look at the results on common words displayed in Figure 3.11. Over all, the highest correlation seem to be those obtained with SGNS. Most setups that involve word2vec embeddings yield positive correlations, and magnitudes are generally higher than what is observed elsewhere. Character-based embeddings also seem to yield correlations, although the overall magnitude is lower than those observed for SGNS models. Turning to ELECTRA embeddings, we see that Russian seems like the only language which yields unequivocally higher correlation from ELECTRA than from character-based embeddings.



(a) Common words, char-based

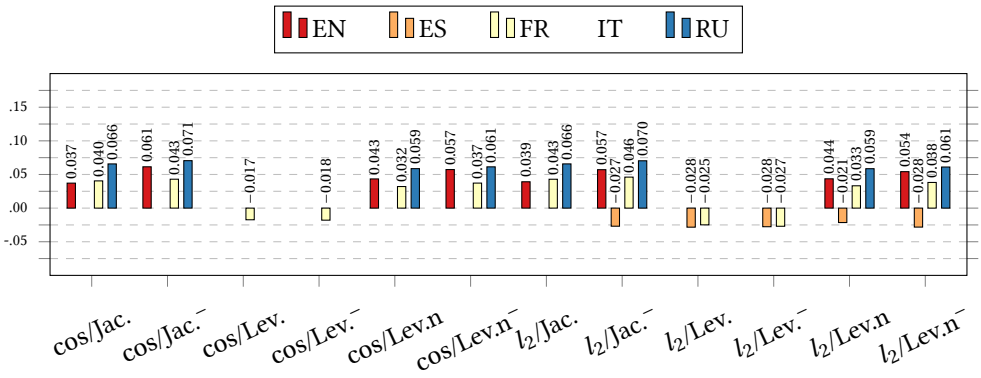


(b) Common words, word2vec

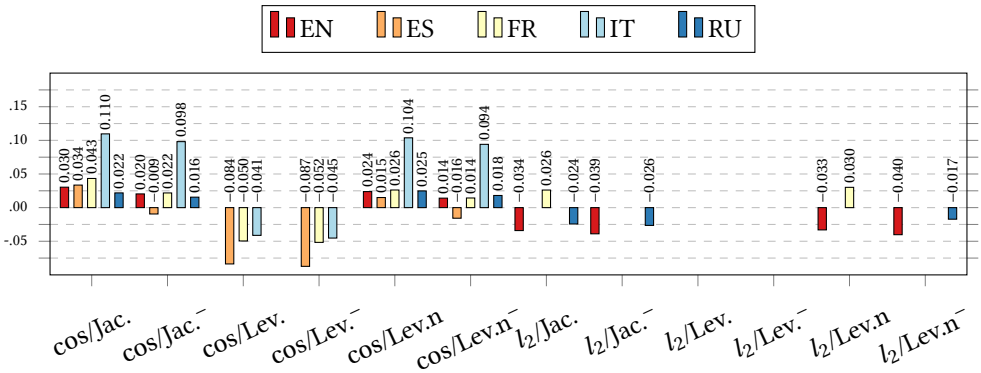


(c) Common words, ELECTRA

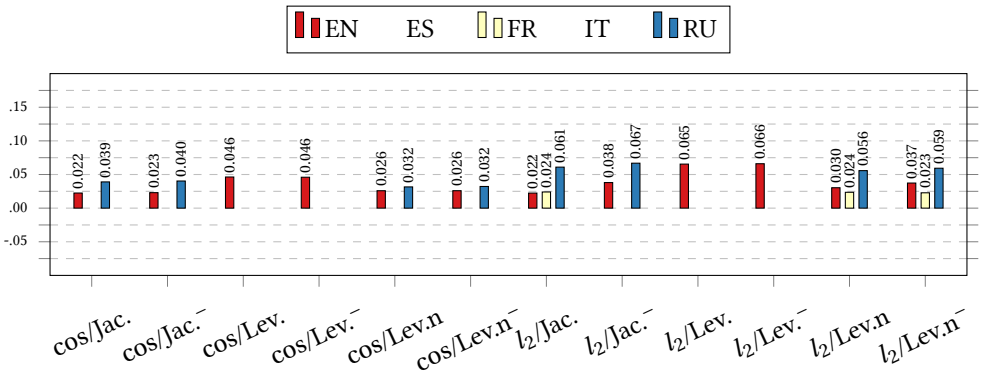
Figure 3.11: Topographic similarity scores for definitions (Common words).



(a) Rare words, char-based



(b) Rare words, word2vec



(c) Rare words, ELECTRA

Figure 3.12: Topographic similarity scores for definitions (Rare words).

If we focus on the results on rare words displayed in Figure 3.12, we see a different picture. SGNS-based setups are overall much poorer than what we observed in common words (Figure 3.11). Numerous setups are found insignificant or yield anti-correlations. Result from character embeddings sharply contrast across languages: in the English, French and Russian models, we observe higher results than what we have for SGNS, whereas Spanish and Italian yield insignificant setups or anti-correlations. Lastly, we can note that ELECTRA embeddings seem to fare better than with common words.

There are also some surprising observations to be made. The most straightforward setup (cos/Lev.) systematically fails to produce positive correlations. It also produces the highest anti-correlation scores (rare words, Spanish, with and without stop-words). Removing stop words is not always helpful: especially in SGNS-based setup, we can see an almost systematic decrease in correlation.

3.4.4 Discussion

The experiments suggest that common word definitions are more similar to SGNS vectors, whereas rare word definitions are more similar to character-based or contextual embeddings, as the magnitude of the correlation attests. Overall, the most regular relation is to be found between common word definitions and word2vec vectors.

Character-based embeddings are not a negligible predictor of definition distance. This makes sense: one might expect etymologically related words to have some part of their meaning in common; as a consequence, their definitions should highlight these similarities; hence, to some extent, orthographically

similar words should have similar definitions. On the other hand, our ELECTRA contextual embeddings often under-perform other types of embeddings. This is likely due to the amount of data they require in order to be properly trained.

Another point to take note of is that we find overall much lower correlations than what we observed for sentences in Section 3.3. The majority of the setups we test corresponds to either insignificant results or anti-correlation. The bewildering behavior observed on definitions might be due to particular artifacts from this dataset, in contrast to the more natural and varied dataset used in the previous experiment. Dictionaries are rather constrained in terms of style, and often make use of stylistic prefixes (e.g., "*of or pertaining to*"). This may drive the correlation scores down; and could in theory be factored in by more appropriate pre-processing or more elaborate textual metrics.

3.5 Replicating and expanding the study on definitions

The surprising results we observed in Section 3.4 warrants that we rule out any possible mistake on our part. To do so, we resort to a replication study, and proceed with greater caution so as to rule out any potential flaw in our methodology. We will rely on datasets and embeddings that we have not created ourselves, employ all control mechanisms at our disposal, and cover a greater number of textual metrics between definitions.

3.5.1 Methodology

We select definitions from the dataset distributed by Noraset et al. (2017). We restrict ourselves to definitions of nouns collected from the GCIDE, the GNU Collaborative International Dictionary of English (GCIDE),[13] where the definiendum is among the 100–10 000 most frequent words of the English Gutenberg corpus. This yields 4 123 distinct definienda and 20 109 definitions. For ambiguous definienda, we select one definition at random so as to ensure a strict one-to-one correspondence between embeddings and glosses. We repeat this process five times for all subsequent measurements before averaging results.

We consider four sets of pre-trained word embeddings: fastText trained on Common Crawl (Bojanowski et al., 2017), GloVe 6B, trained on Wikipedia and GigaWord, and GloVe 840B, trained on Common Crawl (Pennington et al., 2014), and word2vec trained on GoogleNews (Mikolov, K. Chen, et al., 2013).

We do not include contextual embeddings for three reasons. First, the GCIDE-based dataset of Noraset et al. (2017) does not provide contexts of usage for us to compute contextual representations. Second, we expect that the behavior of more complex models may be less easy to interpret overall. Third, there is no consensual methodology to compare the quality of static embeddings and contextual embeddings, making it difficult to know whether we can hold contextual embeddings to the same standard than static embeddings. Excluding contextual representations from this replication study is the simpler solution to these three problems.

We first verify that the semantic distances over these semantic spaces prop-

[13] `https://gcide.gnu.org.ua/`.

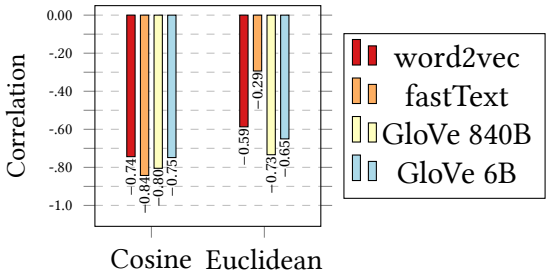


Figure 3.13: Meaning distance metrics evaluated on the MEN dataset. (Spearman correlation)

erly anti-correlate to human similarity ratings: words that humans judge to be highly similar in meaning should not be far from one another in the embedding spaces. We therefore assess the four sets of word embeddings on a word semantic similarity benchmark, the MEN dataset (Bruni et al., 2014), which maps pairs of words to human ratings of their semantic similarity. For all models, we test the Euclidean distance and the cosine distance as a distance over vectors; if they encode how different two meaning representations are, we expect them to anti-correlate with semantic similarity ratings. Results in Figure 3.13 highlight that while all semantic metrics properly anti-correlate, cosine distance yields higher correlations with human similarity judgments than Euclidean distance, for all embedding spaces.

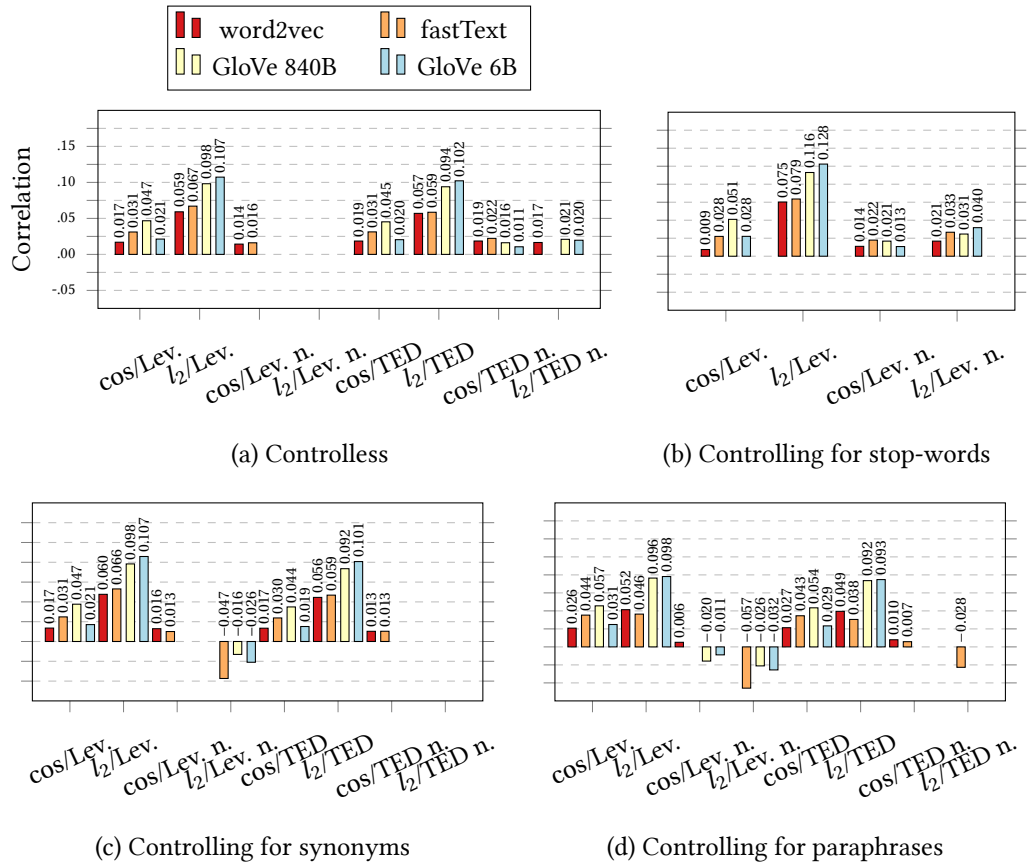
As in previous experiments, we consider cosine and Euclidean distance to measure the distance between definienda embedding. We also include Levenshtein distance and length-normalized Levenshtein distance, as well as the Jaccard index.

Moreover, natural language has a rich syntax: it therefore makes sense to assess the textual similarity of these sentences using syntactically-informed metrics. Thus, in addition to our previous definition gloss distances, we study the

Tree Edit Distance (TED) computed with the AP-TED algorithm (Pawlik and Augsten, 2015) over the corresponding parse trees obtained with the Reconciled Span Parser (V. Joshi et al., 2018). We also consider normalizing TED so that the maximum distance between any two definitions is 1. More formally, given the parse trees $\mathscr{F}$ and $\mathscr{G}$, we normalize their TED by $\#\mathscr{F} + \#\mathscr{G} - \min(h(\mathscr{F}), h(\mathscr{G}))$, where $\#T$ corresponds to the size of tree T and $h(T)$ to its height.

To check whether synonymy, ungrounded symbols and paraphrases also impact natural language examples as we saw in Section 3.2, we perform simple modifications of our original process. To control for synonymy, we use the same procedure as before: we replace every word by the first lemma of its first synset in WordNet (Fellbaum, 1998), if any such lemma can be found. To control for ungrounded symbols, we use the same mechanism as previously of removing stop-words. Given that this latter manipulation profoundly alters the sentence, we do not compute TED-based topographic similarity scores in this case.

As for paraphrases—i.e., multiple glosses associated to the same embedding—we redo the selection process, and this time randomly sample 4 123 items out of the total 20 109 definitions, without the constraint of having only one definition per definiendum: hence these samples only contain on average 2 507 distinct definienda (standard deviation: ±14.53). We duly note that different definitions correspond to distinct senses of the definiendum and thus are not strictly speaking paraphrases. However, we remark that static embeddings of polysemous words correspond to all possible senses for this word-type, thus a static word embedding ought to be matched to the entire set of definitions for the corresponding token. Moreover, the case of ambiguous definienda associated to multiple glosses correspond exactly to the confounding factor we described earlier



(a) Controlless

(b) Controlling for stop-words

(c) Controlling for synonyms

(d) Controlling for paraphrases

Figure 3.14: Topographic similarity scores for natural language definitions.

as paraphrases in Section 3.2: multiple sequences of tokens $M_1 \cdots M_n$ associated to the same vector representation.

3.5.2 Results

We summarize our findings in Figure 3.14. Figure 3.14a corresponds to results in the case where no supplementary control is applied. Figures 3.14b to 3.14d highlight the effects of controlling for stop-words, synonyms, and multiple definitions respectively. Statistically insignificant topographic similarity scores are not displayed. Standard deviations of topographic similarity scores across the

five runs remain below 0.008 and often around 0.005, with the exception of a few setups when controlling for paraphrases.[14]

While the control methods often produce inconsistent measures, they can be understood as suggesting that the identified topographic similarity confounding factors are noticeably at play in natural language.

As with artificial language experiments above, controlling for *stop-words* improved topographic similarity measurements in most setups (Figure 3.14b), however scores decrease for fastText and word2vec when using cosine and Levenshtein distances setups. Controlling for *synonymy* brought very small suggestive topographic similarity increases in the most standard setup (cosine and non-normalized Levenshtein) but observations across other setups are not consistent (Figure 3.14c). Controlling for *paraphrases* produces consistent and pronounced topographic similarity improvements on the most standard setup but quite diverse effects elsewhere (Figure 3.14d).

We also make two general unexpected observations about alternative setups. First, as seen e.g., from the controlless scenario in Figure 3.14a, normalizing textual metrics can be surprisingly detrimental. For instance, Euclidean distance between GloVe 6B vectors, when paired with Levenshtein distance, goes from the highest measured topographic similarity score to statistical insignificance—in fact, only 2 out of the 8 topographic similarity using normalized Levenshtein distance are significant. When controlling for confounding variables, normalization can even induce anti-correlations. Second, cosine distance also yields lower topographic similarity scores than Euclidean distance, despite it being more in

[14]In detail, 4 setups yield a standard deviation between 0.01 and 0.015; they all involve Euclidean distance and non-normalized form distances, using either fastText or GloVe 840B.

line with human ratings (Figure 3.13): Euclidean distance yields in many occasions correlation scores of 0.1 and higher, more than twice what we observe for cosine distance.

3.5.3 Discussion

Overall, this experiment on its own highlights that demonstrating the role of confounding factors expected to be detrimental to topographic similarity, such as ungrounded elements, is in principle possible for natural language; though we have employed blunt methods of control, effects could be perceived. On the other hand, our observations underscore how sensitive topographic similarity is to the choice of distance functions: considerations such as normalizing metrics between 0 and 1 or choosing Euclidean vs. cosine distance can impact results significantly.

These observations raise two questions:

(i) why does normalizing textual distances degrade topographic similarity scores?

(ii) do these results support that definitions and DSMs describe comparable metric spaces?

To answer both, we study more closely which items are detrimental to topographic similarity, as they may shed light on what topographic similarity measurements capture, and how normalization affects it. We consider items where measurements are mismatched: sentences with a relatively low meaning distance but a relatively high form distance—or vice versa—drive the topographic similarity score down. We convert distance measurements into rank values, and

consider the 100 pairs of sentences that yield maximal rank difference for a given setup. These pairs will be referred to as *problematic* below.

For the sake of clarity, we focus on GloVe 6B-based measurements, using Levenshtein distance (either raw or normalized) and compute all our measurements from the same random selection of 4 123 definitions. In the controlless scenario, the 100 most problematic items for non-normalized Levenshtein-based setups all involved two synonym-based definitions, such as Definition 8:

pilot: a steersman (8)

As synonym-based definitions can be identified to holistic messages, this evaluation suggests that our dataset contains a high number of non-compositional examples that this textual metric is not fit to handle.

All problematic pairs were found to have a high semantic distance (involving two unrelated definienda) but a low textual distance (both definitions are very short, usually composed of an article and a noun, which entails few edit operations). This suggests that normalizing the textual distance might be crucial to get reliable topographic similarity. However, while normalizing Levenshtein distance does reduce the number of pairs of synonymy-based definitions in the problematic pairs, they remain very frequent (98/100 for Euclidean distance, 88/100 for cosine distance), due to the fact that a common article usually means that half of the tokens in the two definitions are the same. Removing stop-words further reduces synonymy-based definitions in problematic pairs to a handful, but reveals other artifacts: many pairs share a common pattern such as Definitions 9 and 10:

sneer: the act of sneering (9)

wade: the act of wading (10)

Moreover, the topographic similarity scores obtained in these settings are lower than with raw Levenshtein distance, suggesting that such patterns are frequent and that length is (counter-intuitively) a relevant factor. We conjecture that short definitions are responsible for a large part of these artifacts, which answers our first question. A simple solution might therefore be to filter them out. Alternatively, a more complex textual distance could allow us to get rid of these artifacts. We leave this subject for future work.

The second question we asked above concerned whether this replication study presents a different pictures from what we saw in our first attempt in Section 3.4. To answer this, we compare all results of the present replication study to those we saw for English common words with word2vec (cf. Figure 3.11b).

First, we can remark that differences in setups do not seem to have quite the same effect: while cosine distance was found to yield higher correlations in the previous experiment, this time around, we generally get better results with Euclidean distance. In the previous study, removing stop-words did not seem helpful, whereas this time around this control mechanism appears to improve some, but not all setups.

In terms of raw correlation scores, if we restrict ourselves to the same architecture, we find better results in our previous experiment. If we adopt a more global perspective, we find a tentative improvement over the maximum correlation scores (0.100 previously, vs. 0.128 here). Nonetheless, this upgrade is far from satisfactory, as we are still quite below what we can expect for sentence encoders (cf. Section 3.3); moreover, it might not be a significant improvement

as we are using a different dataset of definition glosses.

Simply put, the results of this replication do not shed a greater light on our previous results. In both cases, we find counter-intuitive interactions between setups. In both cases, we find statistically insignificant correlations, as well as statistically significant correlations and anti-correlations. In both cases, the highest correlation scores we observe are far below what one cold expect from our previous experiments.

3.6 Conclusions

In all, what did we learn by measuring the topographic similarity of embeddings and definition glosses? On the one hand, some of the setups we surveyed did yield positive correlation scores. On the other hand, many also yielded no significant correlation, or even anti-correlation scores; furthermore the magnitude of the correlation in any setup that actually proved significant was fairly limited ($\tau < 0.15$). This contrasts with the sentence-based measurements we conducted in Section 3.3 ($\tau > 0.30$).

Another point that warrants caution is the lack of agreement from our setup. We do not observe a unanimous trend common to all setups, nor do the results straightforwardly point in the same direction. The choice of distances, embeddings, or control methods can annihilate or negate similarity scores. This tells us that while definitions can in principle be cast as a metric space, doing so is rife with caveats. This in turn obfuscates topographic similarity to a degree where it is hard to make any final conclusion.

This unclear set of results was the initial reason why we conducted the repli-

cation study in Section 3.5. Our conclusions in Section 3.4 hinted at a poor dataset: the one we had used might have been responsible for the surprising results we observed. However, replicating our study using GCIDE definitions (instead of DBnary definitions) and readily available embeddings (instead of the ones we trained) did not seem to improve our results convincingly—at least, the results we observed were still not up to the standards we could expect following our experiment with sentence encoders in Section 3.3. Moreover, we have been able to pinpoint through manual analysis that some characteristics of dictionaries hinder the application of the topographic similarity methodology: synonym-based definitions, in particular, can be equated to fully holistic messages in our earlier experiments with Artificial languages in Section 3.2, for which this topographic similarity metric is ill-suited.

This replication study has corroborated what emerged from our first approach in Section 3.4: the overall organizations of the space of definition glosses and the space of word embeddings differ. They are barely comparable to what we observe for sentences and sentence embeddings. We have ruled out that the data we provided is the root cause for this low similarity; hence, it is reasonable to conclude that this low correlation is to be imputed to the different nature of word embedding and definition spaces.

We could make some caveats and suggestions for improvement. For instance, a textual distance more complex than the ones we have focused on during this experiment could allow us to produce higher topographic similarity scores. However, if we consider this problem as some metric to improve on, we shift our methodology from measuring a property of two metrizable spaces to that of *modeling* the textual metric most topographically similar to some vector space, which

is another perspective altogether that we will investigate in future chapters.

4

Defining Inverse Functions Between Dictionaries and DSMs

Mara hörfar fráleitt fann
Feigðin lítið kríli
Fara draugar aldrei ann
Ógnin okkar býli

Býli okkar ógnin ann
Aldrei draugar fara
Kríli lítið feigðin fann
Fráleitt hörfar Mara

— Skálmöld, *Barnið*

Our comparison of definitions and embeddings as metric spaces in Chapter 3 was not conclusive. One potential explanation that emerges from our findings is that perhaps our problem is best casted as one of conversion, rather than com-

This chapter is based on previous publications (Mickus, Paperno, and Constant 2019, "Mark my Word: A Sequence-to-Sequence Approach to Definition Modeling"; Mickus, Constant, et al. 2020, "Génération automatique de définitions pour le français (Definition Modeling in French)") as well as the SemEval 2022 CoDWoE shared task: `https://competitions.codalab.org/competitions/34022`; article in press: (Mickus, Deemter, et al. 2022, *Semeval-2022 Task 1: CODWOE – Comparing Dictionaries and Word Embeddings*).

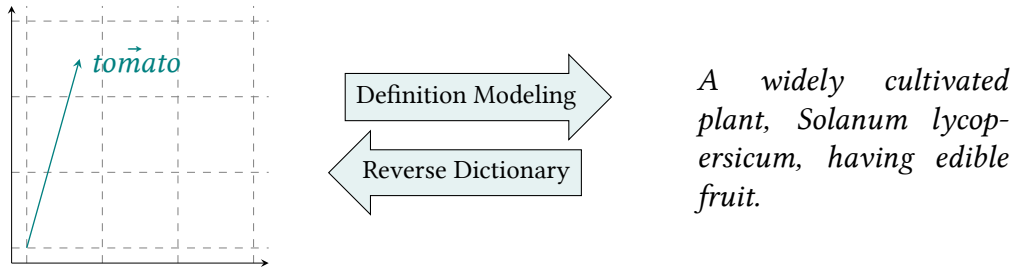


Figure 4.1: Inverse tasks of Definition Modeling and Reverse Dictionary

parison. Instead of measuring the properties of both types of semantic representations, we can focus on whether all the information necessary to reconstruct one representation is present in its counterpart. In other words: can a definition gloss be generated from the embedding of its definiendum, and can we conversely compute this vector from the gloss?

One appealing approach is to use neural networks, as they are in principle able to approximate any function. To address this problem, we would therefore need two types of neural models: those that convert definition glosses into embeddings, and those that convert embeddings into definition glosses. Such models are very reminiscent of existing NLP applications that we discussed previously: reverse dictionary applications (see Section 2.4.1) and definition modeling systems (see Section 2.4.2).

In all, we will investigate whether Definition Modeling and reverse dictionaries can be re-framed as each other's inverse, as illustrated in Figure 4.1. If this can be achieved, reverse dictionary models would allow us to convert glosses into embeddings, whereas definition modeling systems would allow us to convert embeddings into glosses.

This approach, though appealing, is not without issues. In this chapter, we will more narrowly focus on three foreseeable difficulties. First, aligning word

embeddings and word definitions is not a trivial problem, as we will describe in Section 4.1. Second, as we will see in Section 4.2, Definition Modeling as it is currently conceptualized in the NLP community is highly reliant on examples of usage, which are incompatible with our inverse functions. Third, it is not guaranteed that such systems are able to discriminate between embeddings based on their quality, as we will study in Section 4.3. Having covered these expected shortcomings, we will sketch some perspectives on this approach in Section 4.4.

4.1 Aligning word definitions and word embeddings

Earlier in Chapter 2, we reviewed a number of NLP systems that convert glosses into embeddings under the general denomination of "reverse dictionary systems."

Once can point out that the literature on the topic almost exclusively focuses on systems, rather than considering the problem of building a reverse dictionary as a task. One could nonetheless construct a dataset where definition glosses are associated with target embeddings to be reconstructed. Such a dataset could then be used to compare the performance of different approaches and architectures for reconstructing word embeddings from the associated definition glosses. This is in essence the approach adopted by works using dictionaries to perform compositional semantics, such as Zanzotto et al. (2010) and Hill et al. (2016).

As it stands, we do find some immediate issues with such a formulation of the *reverse dictionary task*—although it essentially corresponds to what we described in Section 3.4.1. One major problem to consider is that of word-sense alignment. While static embeddings correspond to a given word type, definition glosses instead describe a specific sense of a given word type. As such, typical

static embedding architectures such as word2vec, GloVe or FastText will map multiple inputs to the same output. While this may not appear too problematic for polysemous words, where the various definition glosses are somewhat related, we can ask ourselves whether this is justified for homographs. For instance, it is questionable that we should expect any model to produce the same output for two widely differing definitions such as those in 11 and 12:[1]

quail: Any of various small game birds of the genera Coturnix, Anurophasis or Perdicula in the Old World family Phasianidae or of the New World family Odontophoridae. (11)

quail: To lose heart or courage; to be daunted or fearful. (12)

The problem is all the more complex with contextual word embeddings such as BERT. Here, the first hurdle lies in aligning definition glosses with contexts of the word use in that sense. This can generally be done using examples of usage: these short texts should in principle contain the definiendum used with the meaning described by the definition gloss. However, not all dictionaries provide such examples of usage; and if a dictionary does include examples, it might not do so systematically.

Second, word senses do not necessarily correspond to specific contextualized token semantic representations. For instance, we might expect a gloss, as a word sense description, to only list essential properties of this concept (e.g., if we define the meaning of "*bird*" that corresponds to the *Aves* genus of animals,

[1] From `en.wiktionary.org`

we might include remarks such that a bird has wings, feathers, a beak, generally flies, etc.), whereas a token representation such as a contextual embedding might include contingent properties of the specific referent (e.g., in "*I saw a red bird*", the token representation for "*bird*" could contain enough information for us to rule out that the bird it refers to is a seagull, which is white). In other words, word sense representations could be less specific than token representations, and more specific than word type representations.

Yet another difficulty arises if we have multiple examples of usage associated to a single definition: here, we would have multiple contextual embedding targets for a single input sequence of definientia. Overall, these issues in alignment suggest that reverse dictionary systems can be seen as non-injective surjective functions; i.e., reverse dictionary systems correspond to a many-to-one set relation. This means that we cannot in principle establish a strict mathematical bijection between dictionary definitions and word embeddings.

4.2 Definition Modeling and Examples of Usage

The problem of word alignment we described in the previous is however not the sole issue we encounter when trying to establish inverse functions between definitions and embeddings. One key element to consider is that almost all Definition Modeling systems require examples of usage. This dependency on supplementary input aggravates the alignment problem we discussed above, as it entails we have asymmetrical tasks. We could convert glosses into embeddings, but to produce a gloss from an embedding one would need to supply an example of usage that is, for the most part, irrelevant to the reverse dictionary task.

The introduction of examples of usage in Definition Modeling can be traced back to Gadetsky et al. (2018). To our knowledge, the work of Gadetsky et al. (2018) is in fact the second paper to ever discuss Definition Modeling. In other words, Definition Modeling was re-framed so as to involve examples of usage almost immediately after the seminal work of Noraset et al. (2017).

The argument that Gadetsky et al. (2018) develop to justify including example of usages is that words are often ambiguous or polysemous, and thus generating a correct definition requires that we either use sense-level representations, or that we disambiguate the word embedding of the definiendum. It should be noted that the disambiguation that Gadetsky et al. (2018) proposed was based on a contextual cue—i.e. a short text fragment. As Chang, Chi, et al. (2018) notes, the cues in the dataset of Gadetsky et al. (2018) did not necessarily contain the definiendum or even an inflected variant thereof. For instance, one training example disambiguated the word "*fool*" using the cue "*enough horsing around—let's get back to work!*". As such, the cues used by Gadetsky et al. (2018) are not exactly examples of usage. Nonetheless, subsequent works such as Chang, Chi, et al. (2018) or Bevilacqua et al. (2020) fully embraced the idea of using examples of usage containing the definiendum.

One could also attempt to justify the inclusion of examples of usage in Definition Modeling systems using linguistic arguments. To take the more specific case of verb definitions, one can observe that context explicitly represents argument structure, which is obviously useful when defining the verb. There is no guarantee that a single embedding, even if it be contextualized, would preserve this wealth of information.

This contextual reformulation of definition modeling can appear contrary to

the original proposal by Noraset et al. (2017), which conceived definition modeling as a "word-to-sequence task." They argued for an approach related to, though distinct from sequence-to-sequence architectures. Nonetheless, the system of Noraset et al. (2017) applied a specific encoding procedure to the definiendum, based on spelling and hypernymy information. Hence even the non-contextual formulation of Noraset et al. (2017) involves supplementary input.

In all, despite some key differences, most Definition Modeling architectures can be casted as sequence-to-sequence models that rely on supplementary information on top of word embeddings. All approaches we are aware of devote distinct parameters or sub-modules to encode the definiendum representation. In the case of Noraset et al. (2017), the encoding was the concatenation of the embedding of the definiendum, a vector representation of its sequence of characters derived from a character-level CNN, and its "hypernym embedding." Gadetsky et al. (2018) used a sigmoid-based gating module to tweak the definiendum embedding. The architecture proposed by Chang, Chi, et al. (2018) is comprised of four modules, only one of which is used as a decoder: the remaining three are meant to convert the definiendum as a sparse embedding, select some of the sparse components of its meaning based on a provided context, and encode it into a representation adequate for the decoder. The model of Bevilacqua et al. (2020) is based on a pre-trained sequence-to-sequence architecture, whose encoder is tasked with modeling the context of occurrence.

On the other hand, the inclusion of examples of usage is here entirely contrary to our purpose. If we are to evaluate whether embeddings are equivalent to glosses, we cannot adopt a contextual formalization of Definition Modeling. A contextual formalization would in fact entail that we equate a given definition

to a vector paired with an example of usage; and we would have to re-frame reverse dictionaries accordingly as the task to generate a vector paired with an example of usage. The first question we will examine in this section is whether we can find a formalization of definition modeling that encompasses both contextual and non-contextual variants of this task (see Section 4.2.1). We will then shift our attention to the impact that a context-free formulation of Definition Modeling has on performance (see Section 4.2.2).

4.2.1 Formalization

The sequence-to-sequence formulation of definition modeling can formally be seen as a mapping between contexts of occurrence of definienda and their corresponding definitions. It moreover requires that the definiendum be formally distinguished from the remaining context: otherwise the definition could not be linked to any particular word of the contextual sequence, and thus would need to be equally valid for any word of the contextual sequence.

We formalize definition modeling as mapping to sequences of definientia from sequences of pairs $\langle w_1, i_1\rangle, \ldots, \langle w_n, i_n\rangle$, where w_k is the k^{th} word in the input and $i_k \in \{0, 1\}$ indicates whether the k^{th} token is to be defined. As only one element of the sequence should be highlighted, we expect the set of all indicators to contain only two elements: the one, $i_d = 1$, to mark the definiendum, the other, $i_c = 0$, to mark the context; this entails that we encode this marking using one bit only.[2]

[2]Multiple instances of the same definiendum within a single context should all share a single definition, and therefore could theoretically all be marked using the definiendum indicator $i_d = 1$. Likewise the words that make up a multi-word expression should all be marked with this i_d indicator. Here, for simplicity, we only mark a single item; in cases when multiple occurrences

To treat definition modeling as a sequence-to-sequence task, the information from each pair $\langle w_k, i_k \rangle$ has to be integrated into a single representation $\vec{marked}_k$:

$$\vec{marked}_k = mark(i_k, \vec{w}_k) \tag{4.1}$$

This marking function can theoretically take any form. Considering that definition modeling uses the embedding of the definiendum $\vec{w}_d = e(w_d)$, we focus on a multiplicative and an additive mechanism, as they are conceptually the simplest form this marking can take in a vector space. They are formally defined as:

$$\vec{marked}^{\times}_k = i_k \times \vec{w}_k \tag{4.2}$$

$$\vec{marked}^{+}_k = e(i_k) + \vec{w}_k \tag{4.3}$$

The last point to take into account is where to set the marking. Two natural choices are to set it either before or after encoded representations were obtained. We can formalize this using either of the following equation, with $\mathscr{E}$ the model's encoder:

$$\vec{marked}_k^{\text{after}} = mark(i_k, \mathscr{E}(\vec{w}_k)) \tag{4.4}$$

$$\vec{marked}_k^{\text{before}} = \mathscr{E}(mark(i_k, \vec{w}_k)) \tag{4.5}$$

of the same definiendum were attested, we simply marked the first occurrence.

Multiplicative marking

The first option we consider is to use scalar multiplication to distinguish the word to define. In such a scenario, the marked token encoding is

$$\vec{marked}_k^{\times} = i_k \times \vec{w}_k \tag{4.6}$$

As we use bit information as indicators, this form of marking entails that only the representation of the definiendum be preserved and that all other contextual representations are set to $\vec{0} = (0, \cdots, 0)$: thus multiplicative marking amounts to selecting just the definiendum embedding and discarding other token embeddings. The contextualized definiendum encoding bears the trace of its context, but detailed information is irreparably lost. Hence, we refer to such an integration mechanism as a SELECT marking of the definiendum.

When to apply marking, as introduced by Equation (4.4), is crucial when using the multiplicative marking scheme SELECT. Should we mark the definiendum before encoding, then only the definiendum embedding is passed into the encoder: the resulting system provides out-of-context definitions, like in Noraset et al. (2017) where the definition is not linked to the context of a word but to its definiendum only. For context to be taken into account under the multiplicative strategy, tokens w_k must be encoded and contextualized before integration with the indicator i_k.

In Figure 4.2 we present the contextual SELECT mechanism visually. It consists in coercing the decoder to attend only to the contextualized representation for the definiendum. To do so, we encode the full context and then select only

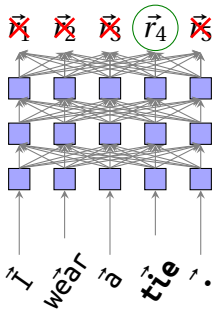


Figure 4.2: SELECT: Selecting from encoded items; items are contextualized and the definiendum is singled out from them

the encoded representation of the definiendum, dropping the rest of the context, before running the decoder. In the case of the Transformer architecture, this is equivalent to using a multiplicative marking on the encoded representations: vectors that have been zeroed out are ignored during attention and thus cannot influence the behavior of the decoder.

This SELECT approach may seem intuitive and naturally interpretable, as it directly controls what information is passed to the decoder—we carefully select only the contextualized definiendum, thus the only remaining zone of uncertainty would be how exactly contextualization is performed. It also seems to provide a strong and reasonable bias for training the definition generation system. Such an approach, however, is not guaranteed to excel: forcibly omitted context could contain important information that might not be easily incorporated in the definiendum embedding.

Being simple and natural, the SELECT approach resembles architectures like that of Gadetsky et al. (2018) and Chang, Chi, et al. (2018): the full encoder is dedicated to altering the embedding of the definiendum on the basis of its context; in that, the encoder may be seen as a dedicated contextualization sub-module.

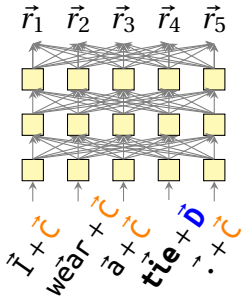


Figure 4.3: ADD: Additive marking in encoder; context items and definiendum are marked by adding dedicated embeddings

Additive marking

We also study an additive mechanism shown in Figure 4.3 (henceforth ADD). It concretely consists in embedding the word w_k and its indicator bit i_k in the same vector space and adding the corresponding vectors:

$$\vec{marked}^{+}_{k} = e(i_k) + \vec{w}_k \tag{4.7}$$

In other words, under ADD we distinguish the definiendum by adding a vector $\vec{\mathtt{D}}$ to the definiendum embedding, and another vector $\vec{\mathtt{C}}$ to the remaining context token embeddings; both markers $\vec{\mathtt{D}}$ and $\vec{\mathtt{C}}$ are learned during training. In our implementation, markers are added to the input of the encoder, so that the encoder has access to this information; we leave the question of whether to integrate indicators and words at other points of the encoding process, as suggested in Equation (4.4), to future work.

Additive marking of substantive features has its precedents. For example, BERT embeddings (Devlin et al., 2019) are trained using two sentences at once as input; sentences are distinguished with added markers called "segment encod-

ings." Tokens from the first sentence are all marked with an added vector $\vec{\texttt{seg}}_A$, whereas tokens from second sentences are all marked with an added vector $\vec{\texttt{seg}}_B$. The main difference here is that we only mark one item with the marker $\vec{D}$, while all others are marked with $\vec{C}$.

This ADD marking is more expressive than the SELECT architecture. Sequence-to-sequence decoders typically employ an attention to the input source (Bahdanau et al., 2015), which corresponds to a re-weighting of the encoded input sequence based on a similarity between the current state of the decoder (the 'query') and each member of the input sequence (the 'keys'). This re-weighting is normalized with a softmax function, producing a probability distribution over keys. However, both non-contextual definition modeling and the SELECT approach produce singleton encoded sequences: in such scenarios the attention mechanism assigns a single weight of 1 and thus devolves into a simple linear transformation of the value and makes the attention mechanism useless. Using an additive marker, rather than a selective mechanism, will prevent this behavior.

4.2.2 Experimental Protocol

We implement several sequence to sequence models with the Transformer architecture (Vaswani et al., 2017), building on the OpenNMT library (Klein et al., n.d.) with adaptations and modifications when necessary. Throughout this experiment, we use pre-trained GloVe vectors from Pennington et al. (2014) and freeze weights of all embeddings. Words not in GloVe but observed in train or validation data and missing definienda in our test sets were randomly initialized

with components drawn from a normal distribution $\mathcal{N}(0, 1)$.

We train a distinct model for each dataset. We batch examples by 8 192, using gradient accumulation to circumvent GPU limitations. We optimize the network using Adam with $\beta_1 = 0.99$, $\beta_2 = 0.998$, a learning rate of 2, label smoothing of 0.1, Noam exponential decay with 2000 warmup steps, and dropout rate of 0.4.

Model parameters are initialized using Glorot. Models were trained for up to 120,000 steps with checkpoints at each 1000 steps; we stopped training if perplexity on the validation dataset stopped improving. We report results from checkpoints performing best on validation.

Implementation of the Non-contextual Definition Modeling System

In non-contextual definition modeling, definienda are mapped directly to definitions. As the source corresponds only to the definiendum, we conjecture that few parameters are required for the encoder. We use 1 layer for the encoder, 6 for the decoder, 300 dimensions per hidden representations and 6 heads for multi-head attention. We do not share vocabularies between the encoder and the decoder: therefore output tokens can only correspond to words attested as definientia.[3] The dropout rate and warmup steps number were set using a hyperparameter search on the dataset from Noraset et al. (2017), during which encoder and decoder vocabulary were merged for computational simplicity and models stopped after 12,000 steps. We first fixed dropout to 0.1 and tested warmup step values between 1000 and 10,000 by increments of 1000, then focused on the most promising span (1000–4000 steps) and exhaustively tested dropout rates from 0.2

[3] In our case, not sharing vocabularies prevents the model from considering rare words only used as definienda, such as "*penumbra*" as potential outputs, and was found to improve performances.

to 0.8 by increments of 0.1.

Implementation of Contextualized Definition Modeling Systems

To compare the effects of the two integration strategies that we discussed in Section 4.2.1, we implement both the additive marking approach (ADD) and the alternative 'encode and select' approach (SELECT). To match with the complex input source, we define encoders with 6 layers; we reemploy the set of hyperparameters previously found for the non-contextual system. Other implementation details, initialization strategies and optimization algorithms are kept the same as described above for the non-contextual version of the model. We stress that the two approaches we compare for contextualizing the definiendum are applicable to almost any sequence-to-sequence neural architecture with an attention mechanism to the input source.[4] Here we chose to rely on a Transformer-based architecture (Vaswani et al., 2017), which has set the state of the art in a wide range of tasks, from language modeling (Dai et al., 2019) to machine translation (Ott et al., 2018). It is therefore expected that the Transformer architecture will also improve performances for definition modeling, if our arguments for treating it as a sequence to sequence task are on the right track.

Datasets

As our goal is to understand the effects of contextualization on definition modeling, we train our models on three distinct datasets, which are all borrowed or adapted from previous works on definition modeling. As a consequence, our

[4]For best results, the SELECT mechanism should require a bi-directional encoding mechanism.

experiments focus on the English language. The dataset of Noraset et al. (2017) (henceforth $D_{\mathbf{Nor}}$) maps definienda to their respective definientia, as well as additional information not used here. In the dataset of Gadetsky et al. (2018) (henceforth $D_{\mathbf{Gad}}$), each example consists of a definiendum, the definientia for one of its meanings and a contextual cue sentence. D_{Nor} contains on average shorter definitions than D_{Gad}. Definitions in D_{Nor} have a mean length of 6.6 and a standard deviation of 5.78, whereas those in D_{Gad} have a mean length of 11.01 and a standard deviation of 6.96.

Chang, Chi, et al. (2018) stress that the dataset D_{Gad} includes many examples where the definiendum is absent from the associated cue. About half of these cues doe not contain an exact match for the corresponding definiendum, but up to 80% contains either an exact match or an inflected form of the definiendum according to lemmatization by NLTK (Loper and Bird, 2002). To cope with this problematic characteristic, we converted the dataset into the word-in-context format assumed by our model by concatenating the definiendum with the cue. To illustrate this, consider the actual input from D_{Gad} comprised of the definiendum "*fool*" and its associated cue "*enough horsing around—let's get back to work!*": to convert this into a single sequence, we simply prepend the definiendum to the cue, which results in the sequence "*fool enough horsing around—let's get back to work!*" Hence the input sequences of D_{Gad} do not constitute linguistically coherent sequences, but it does guarantee that our sequence-to-sequence variants have access to the same input as previous models. The inclusion of this dataset in our experiments is intended mainly for comparison with previous architectures. We also note that this conversion procedure entails that our examples have a very regular structure: the word marked as a definiendum is always the first

word in the input sequence.

Our second strategy was to restrict the dataset by selecting only cues where the definiendum (or its inflected form) is present. The curated dataset (henceforth $D_{\mathbf{Ctx}}$) contains 78,717 training examples, 9,413 for validation and 9,812 for testing. In each example, the first occurrence of the definiendum is annotated as such. D_{Ctx} thus differs from D_{Gad} in two ways: some definitions have been removed, and the exact citation forms of the definienda are not given. Models trained on D_{Ctx} implicitly need to lemmatize the definiendum, since inflected variants of a given word are to be aligned to a common representation; thus they are not directly comparable with models trained with the citation form of the definiendum that solely use context as a cue—viz. Gadetsky et al. (2018) & Chang, Chi, et al. (2018). All this makes D_{Ctx} harder, but at the same time closer to a realistic application than the other two datasets, since each word appears inflected and in a specific *sentential context*.

4.2.3 Results

We use perplexity, a standard metric in definition modeling, to evaluate and compare our models. Informally, perplexity assesses the model's confidence in producing the ground-truth output when presented the source input. It is formally defined as the exponentiation of cross-entropy. We do not report BLEU or ROUGE scores due to the fact that an important number of ground-truth definitions are comprised of a single word, in particular in D_{Nor} ($\approx 25\%$). Single word outputs can either be assessed as entirely correct or entirely wrong using BLEU or ROUGE. However consider for instance the word "*elation*": that it be defined

	D_{Nor}	D_{Gad}	D_{Ctx}
Noraset et al.	48.168	45.620	–
Gadetsky et al.	–	43.540	–
Non-contextual	42.199	39.428	48.266
ADD	–	33.678	43.695
SELECT	–	33.998	62.039

Table 4.1: Results (perplexity)

either as "*mirth*" or "*joy*" should only influence our metric slightly, and not be discounted as a completely wrong prediction.

Table 4.1 describes our main results in terms of perplexity. Perplexity measures for Noraset et al. (2017) and Gadetsky et al. (2018) are taken from the authors' respective publications.

All our models perform better than previous proposals, by a margin of 4 to 10 points, for a relative improvement of 11–23%. Part of this improvement may be due to our use of Transformer-based architectures (Vaswani et al., 2017), which is known to perform well on semantic tasks (Radford, 2018; Cer, Y. Yang, et al., 2018; Devlin et al., 2019; Radford et al., 2019, eg.). Like Gadetsky et al. (2018), we conclude that disambiguating the definiendum, when done correctly, improves performances: our best performing contextual model outranks the non-contextual variant by 5 to 6 points. The marking of the definiendum out of its context (ADD vs. SELECT) also impacts results. Note also that we do not rely on task-specific external resources (unlike Noraset et al., 2017; L. Yang et al., 2019) or on pre-training (unlike Gadetsky et al., 2018).

Our contextual systems trained on the D_{Gad} dataset used the concatenation of the definiendum and the contextual cue as inputs. The definiendum was always

at the start of the training example. This regular structure has shown to be useful for the models' performance: all models perform significantly worse on the more realistic data of D_{Ctx} than on D_{Gad}. The D_{Ctx} dataset is intrinsically harder for other reasons as well: it requires some form of lemmatization in every three out of eight training examples, and contains less data than other datasets, only half as many examples as D_{Nor}, and 20% less than D_{Gad}.

The surprisingly poor results of SELECT on the D_{Ctx} dataset may be partially blamed on the absence of a regular structure in D_{Ctx}. Unlike D_{Gad}, where the model must only learn to contextualize the first element of the sequence, in D_{Ctx} the model has to single out the definiendum which may appear anywhere in the sentence. Any information stored only in representations of contextual tokens will be lost to the decoders. The SELECT model therefore suffers of a bottleneck, which is highly regular in D_{Gad} and that it may therefore learn to cope with; however predicting *where* in the input sequence the bottleneck will appear is far from trivial in the D_{Ctx} dataset. We also attempted to retrain this model with various settings of hyperparameters, modifying dropout rate, number of warmup steps, and number of layers in the encoder—but to no avail. An alternative explanation may be that in the case of the D_{Gad} dataset, the regular structure of the input entails that the first positional encoding is used as an additive marking device: only definienda are marked with the positional encoding $p(1)$, and thus the architecture does not purely embrace a selective approach but a mixed one.

In any event, even on the D_{Gad} dataset where the margin is very small, the perplexity of the additive marking approach ADD is better than that of the SELECT model. In short: supplying more contextual information to the decoder yields better performance. This fact can be construed as an indication of hard-

ships to come: if a strong argument can be made that the task ought to include contexts as inputs (as we discussed in the introduction of this Section 4.2) and that there is a clear impact in discarding the context, then perhaps definitions cannot be generated from embeddings alone—perhaps contextual information might be necessary. If so, this should bear consequences on our experiments down the line: viz., we may expect that contextual embeddings like ELECTRA yield better results than static embeddings like SGNS.

4.2.4 Qualitative Analysis

A manual analysis of definitions produced by our system reveals issues similar to those discussed by Noraset et al. (2017), namely self-reference,[5] POS mismatches, over- and under-specificity, antonymy, and incoherence. Annotating distinct productions from the validation set, for the non-contextual model trained on D_{Nor}, we counted 9.9% of self-references, 11.6% POS mismatches, and 1.3% of words defined as their antonyms. In contrast, Noraset et al. (2017) found 7.14% self-references but 8.57% antonyms. We counted POS mismatches whenever the definition seemed to fit another part-of-speech than that of the definiendum, regardless of both of their meanings.

filch: `to seize` (13)

grammar: `the science of language` (14)

implosion: `a sudden and violent collapse` (15)

[5]Self-referring definitions are those where a definiendum is used as a definiens for itself. Dictionaries can be expected to be exempt of such definitions, as readers are assumed not to know the meaning of the definiendum when looking it up.

Error type	**Context** (definiendum in bold)	**Production**
POS mismatch	her **major** is linguistics	`most important or important`
Self-reference	he wrote a letter of **apology** to the hostess	`a formal expression of apology`

Table 4.2: Examples of common errors (ADD model trained on D_{Nor})

sediment: `to percolate` (16)

deputation: `the act of inciting` (17)

ancestry: `lineage` (18)

We list a few examples of definitions generated by the non-contextual model trained on D_{Nor}. Definitions 13, 14 and 15 were manually selected by us to display the capabilities of the models: they are able to produce simplistic but conceivable definitions. Definitions 16, 17 and 18 were randomly selected from the validation set, so as to provide a clearer idea of the actual performances of our models.

For comparison, we annotated the first 1000 productions of the validation set from our ADD model trained on D_{Ctx}. We counted 18.4% POS mismatches and 4.4% of self-referring definitions; examples are shown in Table 4.2. The higher rate of POS mismatch may be due to the model's hardship in finding which word is to be defined since the model is not presented with the definiendum alone: access to the full context may confuse it. On the other hand, the lower number of self-referring definitions may also be linked to this richer, more varied input: this would allow the model not to fall back on simply reusing the definiendum as its own definiens. Self-referring definitions highlight that our models equate the

meaning of the definiendum to the composed meaning of its definientia. Simply masking the corresponding output embedding might suffice to prevent this specific problem.

As for POS mismatches, we do note that the work of Noraset et al. (2017) had a much lower rate of 4.29%: we suggest that this may be due to the fact that they employ a learned character-level convolutional network, which arguably would be able to capture orthography and rudiments of morphology. Adding such a sub-module to our proposed architecture might diminish the number of mistagged definienda. Another possibility would be to pre-train the model, as was done by Gadetsky et al. (2018): in our case in particular, the encoder could be trained for POS-tagging or lemmatization.

Lastly, one important kind of mistakes we observed is hallucinations. Consider for instance the production 19 by the ADD model trained on D_{Ctx}, for the word "*beta*":

beta: `The twentieth letter of the Greek alphabet (κ), transliterated as 'o'.` (19)

Nearly everything it contains is factually wrong, though the general semantics are close enough to deceive an unaware reader. We conjecture that filtering out hallucinatory productions will be a main challenge for future definition modeling architectures, for two main reasons. Firstly, the tools and metrics necessary to assess and handle such hallucinations have yet to be developed. Secondly, the input given to the system being word embeddings, research will be faced with the problem of grounding these distributional representations—how can we ensure that "*beta*" is correctly defined as "*the second letter of the Greek alphabet, transliterated as 'b'*", if we only have access to a representation derived from its

contexts of usage? Integration of word embeddings with structured knowledge bases might be needed for accurate treatment of such cases.

On a related note, other examples were found to contain unwanted social biases; consider the production 20 from the same model:

blackface: `relating to or characteristic of the theatre` (20)

Part of the social bias here may be blamed on the under-specific description that omits the offensive nature of the word; however contrast the definition of Merriam Webster in 21 for "*blackface*", which includes a note on the offensiveness of the term, with that of Wiktionary in 22, which does not.

blackface: dark makeup worn to mimic the appearance of a Black person and especially to mock or ridicule Black people
Note: *The wearing of blackface by white performers was, from the early 19th through the mid-20th centuries, a prominent feature of minstrel shows (see minstrel sense 3a) and similar forms of entertainment featuring exaggerated and inaccurate caricatures of Black people. The use of blackface is considered deeply offensive.* (21)

blackface: A style of makeup in which a non-black person blackens their face, usually in order to portray a black person. (22)

We refer the reader to Bolukbasi et al. (2016) or Swinger et al. (2018) for a

discussion on biases within embedding themselves, and to Russell (2021) for an overview from the perspective of lexicography.

4.2.5 Conclusions

These preliminary experiments suggest that re-framing Definition Modeling as a context-free task will come with its challenges. For one thing, we can expect lower scores overall: as we saw in Table 4.1, perplexity is at its highest for non-contextual models. In all, state-of-the-art results are probably outside of the reach of a non-contextual Definition Modeling system. We however stress that comparable contextual models, such as the ADD and SELECT models we proposed here, also display faulty productions.

Nonetheless, there are also some facts to consider that do encourage us in pursuing a non-contextual approach. It is important to underscore that our reformulation is coherent with the original formulation of Definition Modeling by Noraset et al. (2017). Lastly, this reformulation is also required by our object of inquiry: if we are to study whether distributional semantic representations and dictionary definitions encode the same semantic information, we must first carefully remove all confounding factors—including the use of context in the Definition Modeling task.

4.3 Can Definition Modeling discriminate DSMs?

In the previous sections, we saw that aligning word embeddings and word definitions is not a trivial enterprise, and that context is likely necessary to reach state-of-the-art performance on the Definition Modeling task.

We now turn to a third issue: are Definition Modeling systems able to discriminate between distributional semantics models? A model that is sufficiently complex could in principle overcome quality issues in some input DSM. This is not a desirable characteristic: if Definition Modeling systems are able to extrapolate and blur differences of quality between embeddings, what could we learn from them?

This echoes an argument we have previously stressed, with respect to the classifier probe architectures used to investigate contextual embeddings (see Section 1.3.2). A model that is too powerful ceases to be a good investigation tool. In the narrower scope of our present inquiry, having Definition Modeling systems that are able to overcome data limitations would entail that we are not able to draw any firm conclusion by setting up inverse functions between glosses and embeddings.

To answer this question, we will look at the performance of definition modeling on a variety of embeddings, and compare this behavior to what we observe on some other measure of word embedding quality, namely word analogy.

4.3.1 Dataset

The dataset we use in this next experiment is drawn from the GLAWI resource (Hathout and Sajous, 2016), a XML-formatted French Wiktionary dump. GLAWI associates each definiendum word type with a list of its attested POSs. Each POS, in turn, is associated with a list of word senses, complete with a definition gloss as well as an optional example of usage. We ignore definitions that either lack an example of usage, or where the definiendum was not found in the example of

# Items	# distinct definienda	Avg. example length	Avg. gloss length
232037	100288	25.36	12.01

Table 4.3: Items retrieved from GLAWI

usage. As previously, we include examples that contain an inflected variant of the definiendum. We case-fold the entire dataset. We do not remove multiword expressions, as long as they are realized as a contiguous span in the example of usage.

Table 4.3 presents some descriptive statistics pertaining to this dataset. Examples of usage are twice as long as definition glosses. We also note that 12% of the items correspond to multiword definienda; for these, average example length and average definition gloss length is relatively similar to what we observe for single-word definienda. The dataset is then split for train (80%, 185363 examples), validation (10%, 23178 examples) and test data (10%, 23496 examples), such that definienda are unique to the split where they are attested. We use the same splits for all models in this pilot experiment.

4.3.2 Embeddings and Definition Modeling system

We use a model similar to the ADD model from the previous Section 4.2. We mark all definienda tokens with a feature vector $+\vec{D}$, whereas context tokens marked with a feature vector $+\vec{C}$. We also prepend to this example of usage the definienda tokens between two special tokens `[DDUM]` and `[CTXT]`, to help the model delineate definienda from contexts. An overview of the architecture is shown in Figure 4.4. Encoder and decoder contains 12 Transformer layers each; as our interest lies in whether definition modeling does distinguish between em-

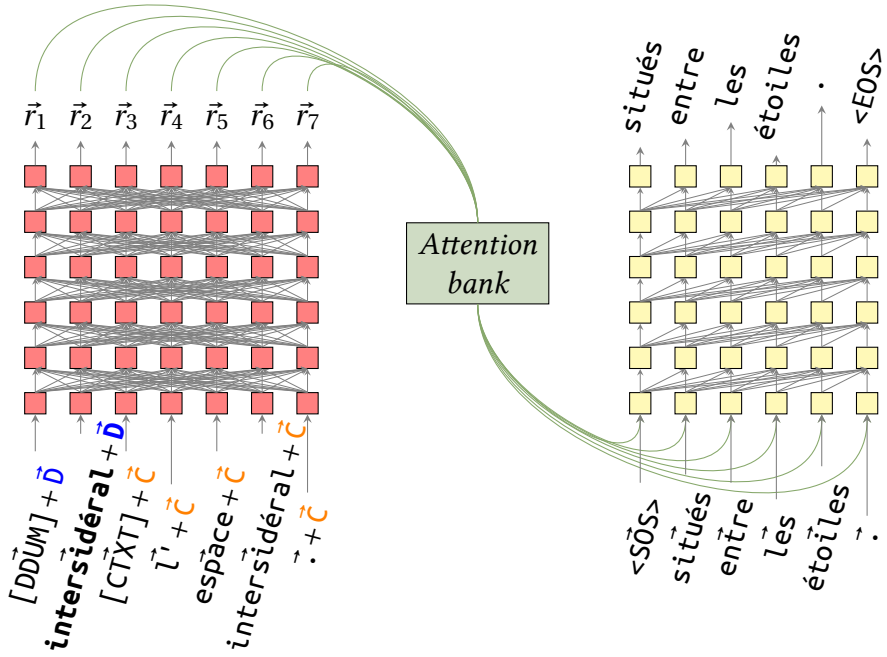


Figure 4.4: Overview of Definition Modeling for French data

bedding quality, we do not update the embeddings during training. hyperparameters are manually set, using a warmup of 10000 steps, a learning rate of 1 and a label smoothing of 0.15

As for word embeddings, we compare multiple architectures: CBOW word2vec (Mikolov, K. Chen, et al., 2013), GloVe (Pennington et al., 2014) and FastText (Bojanowski et al., 2017). Word2vec and GloVe models are trained on FRCOW and wikipedia, as lemmatized by Coavoux (2017);[6] all embeddings use 300 dimensions. Both corpora are case-folded; they correspond to a total of 7.25 billion tokens. For word2vec, we use 20 negative samples, a window of 10, and iterate 10 times over the full corpus. For GloVe, we train the model over 10 iterations using the default parameters suggested by the original demo script of Pennington et al. (2014). As for FastText, we consider two setups: one model that is trained

[6] Available at `http://www.llf.cnrs.fr/wikiparse/`

random	word2vec	GloVe	FastText SC	FastText FB
0.00	36.46	58.32	57.38	68.63

Table 4.4: Analogy results for embedding sets (in %)

on the same corpus for 5 iterations (referred to as "FastText SC"), as well as a model trained on roughly ten times the number of tokens, drawn from CommonCrawl and Wikipedia by Grave et al. (2018) ("FastText FB"). Comparing these two FastText embeddings sets should inform us on the sensitivity of Definition Modeling to hyperparameter choices and embedding training corpus size. Lastly, we include a randomly initialized matrix, i.e., standard Gaussian vectors, to define baseline expectations.

Table 4.4 summarizes the accuracy observed on the French analogy test set of Grave et al. (2018) for each of our sets of vector representations; that is to say, the proportion of correctly solved analogy questions. Note that we drop duplicate entries in the dataset of Grave et al. (2018), as well as examples containing the pair "*son–sa*" (third person singular possessive) as they correspond to a grammatical gender analogy, rather than a social gender analogy as do the other pairs in its group (e.g. "*oncle–tante*", 'uncle–aunt'). This corresponds to 646 examples. Finally, we case-fold the entire analogy dataset. Analogy results give us an overview of what our expectations are, in terms of embedding quality. These baseline expectations defined from analogy will provide us a basis of discussion when considering definition modeling results. If these two embedding quality measures appear very distinct from one another, we will need to either question the ability of definition modeling to discriminate embeddings based on their quality, or admit that analogy quality is orthogonal to definition modeling

Embeddings	Perplexity	BLEU
random	83.70	19.90
word2vec	52.13	30.60
GloVe	48.55	29.00
FastText SC	45.04	30.50
FastText FB	47.84	32.80

Table 4.5: General results on test

quality. The latter is not as implausible as it might first appear: nothing guarantees that linear regularity of semantic space has any practical usefulness for generating definitions. Nonetheless, both tasks should be sensitive to embedding quality; hence why we may expect that their results should overlap to some degree.

4.3.3 Results

We use two metrics to evaluate our models: perplexity and BLEU scores (Papineni et al., 2002). These metrics are fairly common in NLG. Perplexity aims to capture how uncertain a model is that it would have produced this target; it is closely related to cross-entropy. Models with lower perplexit therefore perform better than models with higher perplexity. BLEU computes the similarity of the vocabulary used in a target and the matching production: high BLEU should correspond to higher performances.

Performances are detailed in Table 4.5. For each set of embedding, we report perplexity and BLEU score on our test split. Two remarks can be made by looking at perplexity results: first, random embeddings yield much lower performances than non-random embeddings. This means that definition modeling

meets at least the bare minimum requirements as an evaluation tool. Second, differences in performances on the analogy task (Table 4.4) do not correspond to what we observe for perplexity. If the ordering of the three architectures is more or less stable, the difference between GloVe and FastText is much narrower than what we could expect from analogy results. Note however that FastText embeddings encode orthographic regularities as linear vector offsets. As a consequence, inflection-based analogies with more or less orthographically regular morphological exponents are likely to favor FastText over GloVe. Crucially, the dataset of Grave et al. (2018) contains examples matching these characteristics. Moreover, our model (FastText SC) yields results comparable to the one we retrieved from Grave et al. (2018), whereas analogy results clearly delineated the two. On the other hand, BLEU scores suggest a different picture. While random vectors are still below embeddings, GloVe vectors are this time seen as worse than word2vec. Lastly, the two FastText models seem to strike a balance between BLEU and perplexity, with our SC model now below what we observe for the original FB model.

Note however that hyperparameters were not set separately for each architecture; hence a more extensive research might have a significant impact on these results. Moreover, we can interpret these results as our models having access to sufficiently rich input to overcome the differences among embedding sets—which would cast doubt on the usefulness of definition modeling as an evaluation task for embeddings.

To provide a more thorough understanding of these results, we can also compare them to what we would observe by simply providing the definition for the most similar item in the train test. We consider two approaches for this baseline:

Embeddings	BLEU validation	Best overall	Best definiendum
random	19.80	17.10	16.20
word2vec	31.60	17.50	17.80
GloVe	28.40	17.60	18.00
FastText SC	30.30	17.80	18.70
FastText FB	32.30	17.30	18.80

Table 4.6: Supplementary baselines on validation data (BLEU scores)

either by selecting the most similar input overall—which we can approximate by computing the similarity between train and test examples using the cosine between their mean vector representations—or by considering that similar words should have similar definitions—hence we can compare definitions with the most similar definienda. Both of these approaches are computed on the validation set.

Corresponding results are presented in Table 4.6. We also include the BLEU scores obtained on the validation split for reference. First, we see that the difference between random and non-random embeddings is much less than what we observed earlier. This stems in part from the fact that none of these models were learned, hence random vectors are this time associated to stylistically perfect outputs.

Hence we can suppose that any improvement over these baselines is to be imputed to outputs that are more semantically appropriate rather than outputs that are more stylistically appropriate. If we do commit to this idea that the difference in BLEU scores on these baselines and those obtained through training models reflect semantic improvements, we must conclude two things. First, we ought to conclude that definition modeling is able to assess the semantic nature of the embeddings we tested: the high improvements of learned models compared to the low improvements of the random model suggest that there was indeed something

to learn. Second, we ought to conclude that Definition modeling, as a method of evaluation, is much less straightforward than analogy—since random and non-random baselines in Table 4.6 are fairly close to one another, and we must rely on trained models with numerous parameters to enlarge this distinction.

4.3.4 Manual analysis

Thus far, we have relied on automatic metrics to investigate our results on French definition modeling. Such metrics have their limitations. In particular, they are mostly fit to capture stylistic or formal coherence—they do not measure whether productions are factually correct, as we have discussed in Sections 2.2 and 2.4.2.

To take a concrete example, suppose the following definition gloss for the definiendum "*chemical*" in 23:[7]

chemical: of or relating to chemistry (23)

Automatic metrics such as BLEU are much likely to prefer an absurd production such as definition 24

chemical: `of or relating to Barbados` (24)

over the attested definition 25:[8]

chemical: of, relating to, used in, or produced by chemistry or the phenomena of chemistry (25)

This example is far from implausible. We selected a random sample S_G of 10000 definitions from GLAWI, and then computed for each definition gloss def_i in our sample S_G the minimal edit distance to any other definition in S_G, or more

[7] As found on `en.wiktionary.org`.
[8] As defined by the Merriam-Webster, see `www.merriam-webster.com`.

formally the set:

$$D = \left\{ \min_{\text{def}_j \in S_G - \{\text{def}_i\}} \hat{d}_l\left(\text{def}_i, \text{def}_j\right) \quad | \quad \text{def}_i \in S_G \right\} \tag{4.8}$$

where $\hat{d}_l$ corresponds to the Levenshtein edit distance defined on words, rather than characters. The set D of minimum edit distances was found to be composed at 77.19% of definitions with an edit distance of 1, i.e., 77.19% of our definitions had a counterpart in S_G that differed only by a word added, removed or swapped.

In all, if factual correctness is a problem that broadly applies to all NLG applications, it appears to be even more thorny in definition modeling due to the constrained style of definitions and the importance of factual correctness when determining what is a good or a bad production. Paradoxically, the prominence of factual correctness is also why this task is an interesting approach for the evaluation of word embeddings in the first place.

We sidestep these caveats to the study of definition modeling outputs by relinquishing automatic metrics and instead conducting a manual evaluation of 100 randomly sampled validation items. We first consider issues in the generation process itself. For each embedding architecture, we tabulate:

(i) the number of productions that would match another POS than the one attributed,

(ii) the number of productions where the definiendum is present in its own definition gloss,

(iii) the number of productions containing repetitions.

These criteria, as we pointed out above, are orthogonal to the productions being

Embeddings	Wrong POS	Self-reference	Repetitions
random	25	1	6
word2vec	19	7	4
GloVe	24	2	5
FastText SC	16	7	5
FastText FB	22	4	0

Table 4.7: Production errors typical of neural NLG systems

factually correct.

Results are displayed in Table 4.7. While random vectors and GloVe embeddings appear equally impacted at first glance, we note that the errors attested for random vector tend to have a much greater impact: e.g., phrases are repeated until the model hits the maximum production length. All embedding models seem to find it challenging to properly distinguish between parts of speech, which we can pin on the fact that we use a single encoder for both the definiendum and the example of usage. Non-random embeddings produce more self-referring definitions (where the definiendum is present in its own gloss), which suggests that the NLG models are aware of the structure of the embedding space, as the contextualized definiendum representation cues the decoder towards producing the most semantically similar word type, i.e., the definiendum itself.

We now focus on the semantic factors that can produce invalid definitions. It is hard to survey these factors systematically due to their entanglement with text generation issues: for instance, a wrong POS will necessarily entail that the definition is invalid, to some extent. As our models are poorly semantically grounded, some productions are hard to judge as well. We therefore focus on three criteria where we can hope to achieve some strictness:

Embeddings	Inappropriate semantic field		*Genus–differentia*		
	for target sense	**for any sense**	**# defs. impacted**	**# with wrong genera**	*(%)*
random	93	91	58	50	86.2 %
word2vec	63	56	62	35	56.5 %
GloVe	67	57	65	40	61.5 %
FastText SC	61	45	69	38	55.1 %
FastText FB	70	57	65	41	63.1 %

Table 4.8: Production errors of semantic nature

(i) whether the semantic field of the production has any link to the target meaning, given the example of usage

(ii) whether the semantic field of the production has any link to any meaning of the definiendum

(iii) the proportion of definitions using a genus–differentia pattern (cf. Section 2.3) where the genus is a hypernym of the definiendum.

Tabulated results are presented in Table 4.8. We include the number of definitions using a genus–differentia pattern for reference. We observe that random vectors often fall back to meta-linguistic definition pattern—e.g., "*synonyme de* ~" ('synonym of ~') "*variante orthographique de* ~" ('alternative spelling for ~')—which is likely a consequence of the corresponding model's inability to coherently link a definiendum with a possible hypernym. More generally, this manual evaluation suggests that semantic adequacy and factual correctness remain a major challenge to definition modeling. This time again, we do see that random vectors yield worse definitions than non-random embeddings: this suggests that definition modeling can serve as an evaluation tool for distributional represen-

tations, as random inputs yield clearly less acceptable outputs than non-random ones.

During this manual evaluation, we also made a number of surprising observations. Models corresponding to different embedding sets ended up producing the exact same definitions for certain entries, suggesting that the lexicographic material presented to the model plays a major role. Some models also tended at times to focus solely on one particular meaning, and did not exploit the example of usage. Lastly, many definitions corresponded to highly domain-specific knowledge, which could at time hinder our manual evaluation.

4.3.5 Conclusions

In all, the picture that emerges from this last experiment is not entirely clear. We do see that Definition Modeling systems delineate random inputs from pretrained word embedding systems. Committing to the idea that definition modeling can discriminate between embedding quality however entails that the aspects it captures are orthogonal to the aspects that are captured by word analogy. A manual evaluation of the productions clearly highlights that the models' outputs are far from usable, and that much remains to be done before definition modeling systems can map embeddings to definition glosses.

These limitations, along with the previous issues on word alignment that we mentioned in Section 4.1 and the reliance on examples of usage in Definition Modeling we explored in Section 4.2, question whether we can in fact define inverse functions between definition glosses and word embeddings. The requirements of the Definition Modeling and Reverse Dictionary tasks, as well as the

overall structures of definitions and word embeddings, appear to make the two lexical theories incommensurable.

4.4 The CoDWoE Shared task

The studies we have conducted thus far obviously require further validation by the community. To foster interest in this task and offer wider perspectives on what we suggest here, we proposed to re-frame our line of argument in the format of a shared task. Inviting contributions from the NLP community at large will allow us both to advertise the tasks as reformulated here as well as gather multiple replication studies. This in short will allow to strengthen any argument that might arise from our results, as they would be based on a scientific consensus, rather than a single study.

This shared task was hosted at the 2022 edition of the SemEval workshop[9] and titled CoDWoE—short for "*Comparing Dictionaries and Word Embeddings*".[10] Participants were given as material the datasets we described in Section 3.4.1 and directed to solve the two tasks of Definition Modeling and Reverse Dictionary, as two separate tracks. In the Definition Modeling track, participants had to generate definition gloss, using any or all of the embeddings we provided. In the Reverse Dictionary track, participants had to generate any (or all) embedding architectures, using the definition gloss we provided. We asked participants to refrain from using external resources, including static lexical resources like WordNet (Fellbaum, 1998) or pre-trained language models like BERT (Devlin

[9] See `https://semeval.github.io/SemEval2022/`.

[10] For further information on the shared task, visit the Codalab competition website at `https://competitions.codalab.org/competitions/34022`, as well as the dedicated code repository at `https://github.com/TimotheeMickus/codwoe`.

Figure 4.5: Logo for the CoDWoE Shared task

et al., 2019), so as to ensure the comparability and scientific worth of the shared task results.

We describe the metrics we used to rank submissions in Section 4.4.1. We then turn to a description of our baseline systems in Section 4.4.2, and finally discuss some of the findings from this shared task in Section 4.4.3.

4.4.1 Metrics

The first question we have to address is that of which metrics to use. Given that we frame our experiment as a shared task, there a three criteria we consider to select our metrics. First, their computation must only require a system's output and the corresponding targets. Second, we favor well-known, or easy to understand metrics over obscure ones. Third, we try to select metrics that are easy to compare across frameworks. Only the first of these three criteria is an absolute requirement.

Reverse Dictionary Metrics

The Reverse Dictionary task, as we have re-framed it here, consists in reconstructing embeddings. To that end, we consider three measures of vector similarity. First is MSE (mean squared error), which measures the difference between the components of the reconstructed and target embeddings. Mean-squared error is however not very easy to interpret on its own.

Second is cosine: the reconstructed and target embeddings should have a cosine of 1. It is hard to place specific expectations for what a random output would produce, as this essentially differs from architecture to architecture: for

instance, Transformer outputs are known to be anisotropic, so we shouldn't expect two random ELECTRA embeddings to be orthogonal.

As neither MSE nor cosine provides us with a clear diagnosis tool comparable across all targets, we also include a ranking based measure: we compare the cosine of the reconstructed embedding $\vec{p}_i$ and the target embedding $\vec{t}_i$ to the cosine of the reconstruction $\vec{p}_i$ and all other targets $\vec{t}_j$ in the test set, and evaluate the proportion of such targets that would yield a closer association—viz., the number of cosine values greater than $\cos(\vec{p}_i, \vec{t}_i)$. More formally, we can describe this ranking metric with Equation (4.9):

$$\text{Ranking}(\vec{p}_i) = \frac{\sum_{\vec{t}_j \in \text{Test set}} \mathbb{1}_{\cos(\vec{p}_i,\vec{t}_j) > \cos(\vec{p}_i,\vec{t}_i)}}{\#\text{Test set}} \tag{4.9}$$

Definition Modeling Metrics

A common trope in NLG is to stress the dearth of adequate automatic metrics. Most of the metrics currently existing focus on token overlap, rather than semantic equivalence. The very popular BLEU and ROUGE metric (Papineni et al., 2002; Lin, 2004) measures the overlap rate in n-grams of various lengths (usually 1-grams to 4-grams).

To alleviate this, researchers have suggested using external resources, such as lists of synonyms and stemmers (Banerjee and Lavie, 2005) or pre-trained language models (W. Zhao et al., 2019). The reliance of these augmented metrics on external resources is problematic. Different languages will use different resources with varying degrees of quality—and this will necessarily impact scores,

introducing a confounding factor for any analysis down the line. In the extreme case, if these resources are not available for a particular language, then the metric will have to be discarded. Even assuming the availability of the required external resources, none of these improved metrics is entirely satisfactory. In the case of synonymy-aware metrics such as METEOR (Banerjee and Lavie, 2005), we can stress that syntactically different sentences can express the same meaning, but would not be captured by such metrics. Embeddings-based metrics such as MoverScore (W. Zhao et al., 2019) are very recent, and therefore less well understood; moreover concerns can be raised about whether using a method derived from neural networks trained on text will prove of any help in studying the meaning of texts generated by other neural networks.

One alternative frequently used by the NLG community—and that we ourselves have used in Section 4.2.3—is perplexity, which weighs the probability that the model would generate the target. This last alternative is however not suited to a shared task setup, as it requires us to have access to the actual neural networks trained by participants so we can investigate the probability distributions they model—unlike the other metrics we mentioned thus far, which only require the outputs of these models.

In short, none of the currently available NLG metrics are fully satisfactory. Some are not applicable given the shared task format, some depend on external resources of varying quality, and some merely measure formal similarity, rather than semantic equivalence. Our approach is therefore twofold: on the one hand, we select multiple metrics with the expectation that each might shed light on one specific factor; on the other hand, we encourage participants to go beyond automatic scoring for the evaluation of their model.

As for which metrics we select, we narrow our choice to three. First is a basic BLEU score (Papineni et al., 2002) between a production p_i and the associated target t_i; our reasoning here is that as it is one of the most basic metrics, it is a consistent default choice. Second is the maximum BLEU score between a production p_i and any of the targets $t_i, t_j \dots t_n$ for which the definiendum is the same as that of p_i. This second metric is designed to not penalize models that rely solely on SGNS or char embeddings: as the input would always be the same, deterministic models would always produce the same definition $p_i = p_j = \dots = p_n$.[11] To distinguish between our two BLEU variants, we refer to the former as S-BLEU (or Sense-BLEU), and the latter as L-BLEU (or Lemma-BLEU).

Given that some definitions in our dataset can be very short, we also apply a smoothing to both BLEU-based metrics. In practice, BLEU computes an overlap of n-grams of size m and under; by default, $m = 4$. This overlap is a geometric mean across all n-gram sizes $1 \dots m$. If a definition d contains less than m tokens, then any associated production for which d is used as a target will contain 0 overlapping n-grams of size m. The use of a geometric then entails that the BLEU score for any production associated to d will be 0. To circumvent this limitation of BLEU, it is common to use some form of smoothing. Here, for any n-gram size $\hat{m}$ that would yield an overlap of 0 (i.e., $\hat{m}$ such that $\#d < \hat{m} \leq m$), we replace the overlap count with a pseudocount of $1/\log \#d$.

Lastly, we include MoverScore (W. Zhao et al., 2019), using a multilingual DistilBERT model as the external resource. The fact that this model is multilingual means that we can use it for all five languages of interest. Embedding-based

[11] One way of bypassing this problem would be to include a source of noise, as is done in GAN architectures (Goodfellow et al., 2014). This would still leave open the question of how to optimally align the outputs to the possible targets.

methods have the potential to overcome some of the limitations of purely token-based metrics, which is why we deem them worth including in our setup.

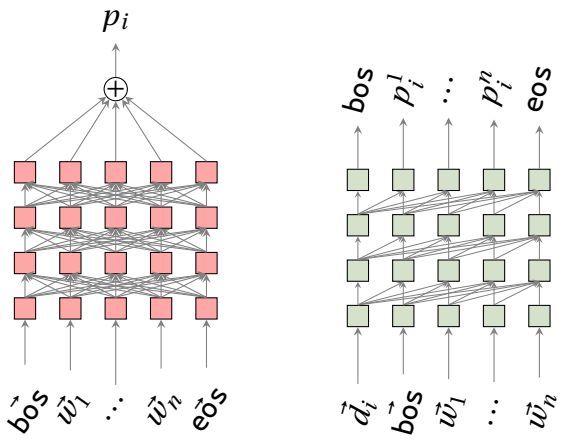
The second part of our approach for evaluating submissions consists in encouraging participants to not rely solely on the automatic scoring system of their outputs. Concretely, we provide participants with a richly annotated trial dataset, which contains frequency and hand-annotated semantic information, and strongly suggest participants to use it for a manual evaluation of their system. We include the presence of a manual evaluation as a criterion to evaluate the quality of a system description paper, and plan to formally recognize the most enlightening evaluations conducted by participants.

Neither our selection of metrics nor our insistence on manual evaluation solves the evaluation issues of NLG systems. We duly note the importance of this question, and plan to conduct a follow-up evaluation campaign on the CoD-WoE submissions.

4.4.2 Baseline Architectures

One remark that emerges from our selection of metrics is that it is difficult to see what baseline expectations should be for each of these models. We therefore implement simple neural network architectures to set a lower threshold for our expectations. We will be using the models shown in Figure 4.6. They are based on the Transformer architecture of Vaswani et al. (2017) and designed to be as simple as possible.

We illustrate our Reverse Dictionary baseline architecture in Figure 4.6a. It consists in feeding the input gloss $\langle \vec{\texttt{bos}}, \vec{w}_1, \ldots, \vec{w}_n, \vec{\texttt{eos}} \rangle$ into a simple Trans-



(a) Reverse dictionary (b) Definition modeling

Figure 4.6: Baseline architectures for the CoDWoE shared task

former encoder, and then summing all the hidden representations to produce the prediction p_i. In practice, the summed hidden states are passed into a small non-linear feed-forward module to derive the prediction:

$$p_i = W_p\left(\text{ReLU}\left(\sum_t \vec{h}_t\right)\right)$$

Our Definition Modeling baseline is presented in Figure 4.6b. It consists in a simple Transformer encoder, where earlier time-step representations are prevented from attending to later time-step representations. To provide information about the definiendum to the model, we use the definiendum embedding $\vec{d}_i$ as the input for the first time-step instead of a start-of-sequence token. We train the models with teacher-forcing: i.e., during training we ignore the definientia $p_i^1, \ldots, p_i^n$ that the model produces; instead we feed it the target $w_1, \ldots, w_m$ attested in the training set at each time-step. During inference, we feed the model with its own prediction. This creates a train-test mismatch, which we alleviate

by using a beam-search. We stop generation when all beams have produced an end-of-sequence token.

For both tracks, we train one model for each distinct pair of language and embedding architecture. We start by re-tokenize the datasets using sentence piece with a vocabulary size of 15000. This is done in order to mitigate the effects of different vocabulary sizes when training our Transformer baselines, and make the models overall easier to compare across different languages.

In the same vein, we set hyperparameters using a Bayesian Optimization procedure, with 100 hyperparameter configurations tested and 10 initial random samples. For the Reverse dictionary models, we tune the following hyperparameters: learning rate, weight decay penalty, the β_1 and β_2 hyperparameters of the Adam optimizing algorithm, dropout rate, length of warmup, batch size,[12] number of heads in the multi-head attention layers, and number of stack layers. For the Definition Modeling systems, we also include a label smoothing parameter to tune. Models are trained over up to 100 epochs; training is stopped early if no improvement of at least 0.1% is observed during 5 epochs. In all cases, we decay the learning rate after the warmup following a half cosine wave, such that the learning rate reaches 0 at the end of the 100 epochs.

4.4.3 Results

The CoDWoE shared task was scheduled to last until after the initial version of this dissertation was presented to the jury. We therefore include in this section early results on the development set as presented to the jury members in the next

[12] In practice, we first manually find the largest batch size that fits on our GPU, and then let the model select the number of batches it should accumulate gradient on.

Lang.	Embs.	MSE	cos	Ranking
en	SGNS	0.910 92	0.151 32	0.490 30
	char	0.147 76	0.790 06	0.502 18
	ELECTRA	1.412 87	0.842 83	0.498 49
es	SGNS	0.929 96	0.204 06	0.499 12
	char	0.569 52	0.806 34	0.497 78
fr	SGNS	1.140 50	0.197 74	0.490 52
	char	0.394 80	0.758 52	0.499 45
	ELECTRA	1.153 48	0.856 29	0.497 84
it	SGNS	1.125 36	0.204 30	0.476 92
	char	0.363 09	0.727 32	0.496 63
ru	SGNS	0.576 83	0.253 16	0.490 08
	char	0.134 98	0.826 24	0.494 51
	ELECTRA	0.873 58	0.720 86	0.491 20

Table 4.9: Reverse Dictionary track: results on development set (baseline models)

subsection, as well as the finalized competition results in the subsequent one.

Development set results

We begin by reviewing the performances of the baseline architectures over the reverse dictionary track, as displayed in Table 4.9[13] Overall, we can stress that the character-based embeddings yield by far the lowest MSE scores across all five languages. ELECTRA embeddings yield the highest cosine scores—which can certainly be attributed to their known anisotropy. Turning to ranking measurements, we find that no specific architecture manages to display convincing results: all models are equally disappointing, with on average half of the development set being preferred over the reconstruction. This could suggest that all

[13]We give results on the development set until the end of the evaluation phase of the shared task.

Lang.	Embs.	S-BLEU	L-BLEU	MoverScore
en	SGNS	0.030 48	0.040 62	0.083 07
	char	0.026 30	0.033 59	0.045 31
	ELECTRA	0.031 55	0.041 55	0.067 32
es	SGNS	0.035 28	0.052 73	0.066 85
	char	0.032 91	0.047 12	0.061 12
fr	SGNS	0.029 83	0.041 34	0.040 36
	char	0.029 13	0.039 85	0.019 35
	ELECTRA	0.030 61	0.039 54	0.038 55
it	SGNS	0.047 59	0.069 10	0.101 54
	char	0.025 32	0.035 22	0.040 68
ru	SGNS	0.038 05	0.051 21	0.115 59
	char	0.023 24	0.032 38	0.071 45
	ELECTRA	0.029 87	0.037 82	0.103 82

Table 4.10: Definition Modeling track: results on development set (baseline models)

our models tend to produce the median vector for the space, disregarding the output that we feed them.

Results for the definition modeling track are summarized in Table 4.10. Here, results are again rather disappointing: all models tend to produce very low scores across all metrics. In general, SGNS vector seem to yield the highest performances, followed by character-based embeddings and ELECTRA embeddings. We can observe that MoverScore is overall more lenient than the BLEU variants we use. Another point of interest is that the ELECTRA models also benefit from the L-BLEU, which suggests that these token representations do not necessarily coincide with word senses.

Test set results

In total, the shared task attracted 159 different submissions from more than 15 distinct users, and resulted in 11 system description papers. From these 15 teams, 9 tackled the Definition Modeling track, and 10 addressed the reverse dictionary track.

Leaderboards Scores attained by participants are shown in Tables 4.11 to 4.14. In Table 4.11, "Mv", "SB" and "LB" refer to Moverscore, Sense-BLEU and Lemma-BLEU respectively; in Tables 4.12 to 4.14, "rnk" refers to the cosine ranking metric (cf. Section 4.4.1). Across tables, highest participant scores per metric are displayed in bold font.

In total, we received 159 valid submissions from 15 different users; out of which 11 teams produced a submission paper. 9 of these teams tackled the Definition Modeling, and 10 addressed the reverse dictionary track. Competition rankings are established by ranking each submission received, selecting for each participant the best performance on all metrics, and finally taking the average best rank. Some participants' submissions were faulty and could not be processed by the evaluation website scoring program.

Among the system descriptions we received, two focused solely on definition modeling. Kong et al. (2022, BLCU-ICALL) use a multitasking framework for definition modeling, based on a generation and a reconstruction objectives. Mukans et al. (2022, RIGA) focus on what are the effects of model size and duration of training on GRUs and LSTMs for definition modeling, and whether MoverScore corroborates human judgment.

Team	en			es			fr			it			ru		
	Mv	SB	LB	Mv	SB	LB	Mv	SB	LB	Mv	SB	LB	Mv	SB	LB
Bl. SGNS	0.084	0.030	0.040	0.065	0.035	0.052	0.046	0.030	0.041	0.107	0.053	0.076	0.112	0.039	0.054
Bl. char	0.047	0.026	0.033	0.059	0.031	0.043	0.022	0.028	0.037	0.046	0.029	0.038	0.072	0.025	0.037
Bl. Electra	0.065	0.031	0.039				0.043	0.031	0.039				0.101	0.032	0.041
Locchi	0.049	0.022	0.027	0.038	0.020	0.026				0.071	0.008	0.012			
LingJing	−0.045	0.004	0.005	0.023	0.013	0.020	−0.113	0.003	0.005	−0.012	0.018	0.029	−0.010	0.011	0.014
BLCU-ICALL	**0.135**	0.031	0.040	**0.128**	0.039	0.056	0.042	0.027	0.037	**0.117**	**0.066**	**0.099**	**0.148**	0.048	0.065
IRB-NLP	0.094	**0.033**	0.042	0.093	**0.045**	**0.064**	0.056	0.028	0.033	0.077	0.010	0.015	0.080	0.027	0.036
RIGA	0.093	0.026	0.032	0.107	0.031	0.045	**0.075**	0.024	0.030	0.093	0.012	0.018	0.094	0.031	0.043
lukechan1231	0.071	0.022	0.027	0.068	0.025	0.036	0.054	0.021	0.026	0.101	0.037	0.054	0.109	0.029	0.040
Edinburgh	0.104	0.031	0.038	0.101	0.035	0.053	0.026	**0.029**	**0.038**	0.107	0.060	0.092	0.109	**0.049**	**0.072**
talent404	0.128	0.033	**0.043**												

Table 4.11: Definition Modeling track results

Team	en			es			fr			it			ru		
	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk
Baseline	0.911	0.151	0.490	0.930	0.204	0.499	1.141	0.198	0.491	1.125	0.204	0.477	0.577	0.253	0.490
Locchi	0.875	0.204	0.394							1.087	0.274	0.386			
BL.research	0.895	0.166	0.312	0.910	0.252	0.253	1.107	0.212	0.314	1.111	0.246	0.247	0.566	0.298	0.290
LingJing	0.862	0.243	0.329	**0.858**	0.353	0.251	1.030	0.328	0.282	1.039	0.360	0.230	**0.528**	**0.424**	0.187
MMG				0.911	**0.403**	**0.167**									
chlrbgus321	**0.854**	0.248	0.319												
IRB-NLP	0.964	**0.260**	**0.231**	0.883	0.367	0.197	1.068	**0.342**	**0.193**	1.076	**0.380**	**0.165**	0.568	0.421	**0.150**
Edinburgh	0.864	0.241	0.326	0.860	0.347	0.271	**1.026**	0.312	0.302	**1.031**	0.374	0.197	0.538	0.383	0.247
the0ne	0.900	0.185	0.500												
JSI	0.909	0.156	0.499	0.913	0.223	0.495	1.122	0.216	0.498	1.196	−0.004	0.499	0.615	0.006	0.499
1cadamy	0.915	0.194	0.374	0.906	0.262	0.375	1.100	0.228	0.439	1.097	0.260	0.384	0.578	0.335	0.291

Table 4.12: SGNS Reverse Dictionary track results

Team	en			es			fr			it			ru		
	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk
Baseline	0.148	0.790	0.502	0.570	0.806	0.498	0.395	0.759	0.499	0.363	0.727	0.497	0.135	0.826	0.495
Locchi	**0.141**	**0.798**	0.483							0.355	0.734	0.478			
BL.research	0.143	0.795	0.450	0.510	0.824	0.412	0.366	0.770	0.428	0.359	0.728	0.417	0.132	0.830	0.410
LingJing	0.176	0.782	0.486	0.583	0.824	0.500	0.411	0.752	0.502	0.438	0.681	0.496	0.184	0.791	0.472
IRB-NLP	0.162	0.770	**0.419**	0.526	0.819	**0.403**	0.390	0.756	0.421	0.366	0.724	**0.383**	0.140	0.824	**0.357**
Edinburgh	0.143	0.795	0.500	**0.467**	**0.839**	0.424	**0.335**	**0.789**	0.428	**0.334**	**0.747**	0.428	**0.116**	**0.852**	0.389
the0ne	0.143	0.796	0.500												
1cadamy	0.168	0.792	0.478	0.557	0.820	0.410	0.391	0.769	**0.416**	0.364	0.739	0.438	0.156	0.836	0.377

Table 4.13: Char Reverse Dictionary track results

Team	en			fr			ru		
	MSE	cos	rnk	MSE	cos	rnk	MSE	cos	rnk
Baseline	1.413	0.843	0.498	1.153	0.856	0.498	0.874	0.721	0.491
Locchi	**1.301**	0.843	0.478						
BL.research	1.326	0.844	0.434	1.112	0.858	0.442	0.864	0.721	0.399
LingJing	1.509	0.846	0.478	1.271	0.859	0.478	**0.828**	0.734	0.420
IRB-NLP	1.685	0.828	**0.432**	1.339	0.847	**0.429**	0.911	0.724	**0.345**
Edinburgh	1.310	**0.847**	0.490	**1.066**	**0.862**	0.476	0.828	**0.735**	0.417
the0ne	1.340	0.846	0.500						

Table 4.14: ELECTRA Reverse Dictionary track results

Five submissions specifically focus on the reverse dictionary task. Bendahman et al. (2022, BL.research) compare the performances of MLP-based to LSTM-based networks for reverse dictionary. B. Li et al. (2022, LingJing) study pretraining objectives for the reverse dictionary track. Ardoiz et al. (2022, MMG) pay specific attention to how the not-so-satisfactory quality of the Spanish dataset impacts results on Spanish reverse dictionary. Cerniavski and Stymne (2022, Uppsala) study whether foreign language entries can improve the performance of the English reverse dictionary baseline model. Z. Wang et al. (2022, 1cademy) introduce multiple technical tweaks for reverse dictionary, such as a dynamic weight averaging loss, language-specific tags and residual cutting.

The last four submissions addressed both tracks. P. Chen and Z. Zhao (2022, Edinburgh) propose to project embeddings and definitions on a shared representational space. Korenčić and Grubišić (2022, IRB-NLP) take inspiration from Noraset et al. (2017) to address definition modeling, and experiment with pooling strategies over Transformer embeddings for the reverse dictionary track. Tran et al. (2022, JSI) focus on comparing the effects of adding LSTM and BiLSTM layers on top of a Transformer model, as well as zero-shot cross-lingual generalization.

Srivastava and Harsha Vardhan (2022, TLDR) propose two Transformer-based architectures for the two tracks, leveraging contrastive learning and unsupervised pretraining.

Looking at Tables 4.11 to 4.14, we see that the metrics we chose in Section 4.4.1 are not always aligned. On the Definition Modeling track (Table 4.11), while the multitask framework of Kong et al. (2022, BLCU-ICALL) yields generally the most consistent performance, it is often outmatched in specific setups. For instance, BLEU-based metrics favor the shared projection technique of P. Chen and Z. Zhao (2022, Edinburgh) in Russian and French, while the pooling strategies of Korenčić and Grubišić (2022, IRB-NLP) appear especially effective on the Spanish dataset. As for the Reverse Dictionary track (Tables 4.12 to 4.14), the strongest contender is generally the Edinburgh team, although the IRB-NLP team almost systematically produces the highest cosine ranking score. Interestingly, BLCU-ICALL, IRB-NLP and Edinburgh all rely on multi-task learning. Note however that the SGNS targets seem to depict a rather different picture, where the pretraining objectives of B. Li et al. (2022, LingJing) bring about some of the best results.

Discussion & analyses When looking at the competition results, two trends emerge. First, the baseline architectures from Section 4.4.2 remain quite competitive with solutions proposed by participants. Second, scores are generally unsatisfactory, especially in the definition modeling track: we do not see a clear divide between char embeddings and distributional semantic representations. The NLG metrics are, in absolute terms, low compared to modern NLP standards and results reported elsewhere on other definition modeling benchmarks. As for the

reverse dictionary track, we see that across all submissions, at least a third of the test set is closer (in terms of cosine distance) to the production than the intended target.

Participants have suggested multiple reasons for these hardships. In particular, Ardoiz et al. (2022, MMG) highlight that the automated data compilation in DBnary (Sérasset, 2012) is of an unsatisfactory quality. Similar remarks can be made with respect to the embeddings, which are trained on rather small corpora. Other submissions such as Mukans et al. (2022, RIGA), P. Chen and Z. Zhao (2022, Edinburgh), Korenčić and Grubišić (2022, IRB-NLP) highlight the limited applicability of mainstream NLG metrics. One last remark is the limited size of our dataset, discussed by the Edinburgh and RIGA teams. All these remarks suggest avenues for future research: in particular, the release of the full dataset should alleviate some of the concerns with respect to dataset size. The MMG team also suggest some concrete preprocessing steps to handle some of the issues they identify in the proposed definitions.

In terms of solutions explored, we can stress that teams have adopted a variety of strategies and architectures: systems used Transformer, RNN and CNN components, often leveraging or exploring multilingualism (Tran et al. 2022, JSI; Cerniavski and Stymne 2022, Uppsala; Z. Wang et al. 2022, 1cademy; Bendahman et al. 2022, BL.research), multitasking, or multiple training objectives (Kong et al. 2022, BLCU-ICALL; 1cadamy; Korenčić and Grubišić 2022, IRB-NLP; Srivastava and Harsha Vardhan 2022, TLDR; P. Chen and Z. Zhao 2022, Edinburgh). Multi-task training tends to yield varied yet competitive results for our data. No preponderant architecture emerges from the system descriptions; we note that multiple submissions based their work on other contextualized embedding ar-

chitectures, trained from scratch on the CODWOE dataset (Z. Wang et al. 2022, 1cademy; B. Li et al. 2022, LingJing). The comprehensive review of architectures by team 1cadamy suggests nonetheless that Transformers might be less suited to this shared task than recurrent models.

As for manual evaluations, Kong et al. (2022, BLCU-ICALL) provide a thorough review of the errors produced by their model. Mukans et al. (2022, RIGA) provide some example outputs of their models, while Srivastava and Harsha Vardhan (2022, TLDR) and Z. Wang et al. (2022, 1cademy) include ablation studies. The most thorough analysis, however, is that of P. Chen and Z. Zhao (2022, Edinburgh), who provide both quantitative and qualitative (PCA-based) analyses across embedding architectures, languages, and trial dataset features. Korenčić and Grubišić (2022, IRB-NLP) provide an extremely well documented review of their systems performances, along multiple analyses of the embeddings proposed for the shared tasks, ranging from 2D down-projection visualizations to descriptive statistics of components. We refer the reader to the respective system papers for a more thorough review and focus here on a few promising approaches to summarize trends that emerge from these manual analyses:

(i) Current metrics are not satisfactory. The IRB-NLP team highlight that the BLEU scores reported on the shared task are dramatically lower than what is generally expected in the literature; the Edinburgh team even shows that the S-BLEU scores obtained by non-sensical glosses such as "`, or .`" can end up among the highest scores for some languages. The Reverse Dictionary metrics can also be sensitive to different aspects of the embeddings, as shown by the IRB-NLP team: this can lead to very different rankings of

model productions, especially when comparing the cosine-based ranking metric to the cosine and MSE metrics. BLEU-based scores are also often sensitive to the length of the production, the target, or both, as shown by both the Edinburgh and the Riga teams.

(ii) Erroneous productions abound. Related to the previous remark, many Definition Modeling systems produce irrelevant or under-specified glosses, for which the proposed metrics are not satisfactory. For instance, the BLCU-ICALL report 52% irrelevant glosses and 23.5% under-specified glosses, from a manual evaluation of 200 productions. Other participating teams, such as RIGA or IRB-NLP, also display generated glosses with varying degress of semantic accuracy.

(iii) Embeddings contain more than semantics. The Edinburgh team highlights how different linguistic features retrieved from the trial dataset can significantly impact the scores they observe. They also highlight that char embeddings are separable by length, and that the Electra embeddings are clustered according to their frequency.

(iv) Not all setups are created equal. The Uppsala team report that Russian seems to be the most effective data source in their multilingual transfer experiments. The IRB-NLP team stresses that vector component distributions across languages and architectures as well as gloss length across languages can take very different values, and they also include 2D visualization suggesting the Electra embeddings tend to form neat cluster not observed for SGNS embeddings. Scores also vary quite a lot across setups (cf. Tables 4.11 to 4.14).

4.5 Conclusions

In all, framing our investigation as finding inverse functions between the set of definition glosses and the set of word embeddings is an approach rife with caveats.

As we saw in Section 4.1, the structure of word embeddings and dictionary differ, making it impossible to define inverse functions in the mathematical sense. Further issues arise when we consider definition modeling. On the one hand, definition modeling systems require examples of usage in order to reach their maximum potential, as we surveyed in Section 4.2. On the other hand, as we saw in Section 4.3, the current state of definitions modeling productions is not up to the standard we could expect.

These problems add to other issues beyond what we discussed in the present chapter. Recall that in Section 2.2 we saw that semantic grounding was a major issue for text-only systems; and this more specifically applies to definition modeling systems, as we reviewed in Section 2.4.2. Likewise, in Chapter 3, our attempt to measure the correlation of distances between embeddings with distances between definition glosses was not very conclusive: our results could be construed as considering two related but distinct spaces.

Taking all these results in stride, we could argue that definitions and embeddings are not equivalent lexical theories. Of course, further research is required before we can believe beyond reasonable doubt the impossibility of perfectly translating definitions glosses into word embeddings and back, This is what we wished to foster through the organization of a shared task, which we described in Section 4.4. We expected that broadening the research landscape in definition

modeling and reverse dictionary systems would bring about more nuance, and help us refine our understanding of word embeddings and dictionary definitions as lexical theories. Yet, we found it is not trivial to tease apart the various factors that lead to the overall low results we observed. While the inadequacy of mainstream NLG metrics and the limitations of the dataset certainly play a role, they do not resolve the fundamental issue that we wished to investigate with CoDWoE. Whether word embeddings and dictionaries contain the same information is still not a solved research problem.

Nonetheless, the facts that we presently have at our disposal, as gathered through our experiments, suggest that what is encoded in an embedding differs from what is described by a definition gloss. This naturally leads us to asking what precisely word embeddings encode.

III

Word Embeddings are not Distributional Semantics Models

5

Limits of the distributional hypothesis

Spare me your riddles witch, can't you see?
They mean nothing to me

— Æther Realm, *Tarot*

In the two previous chapters, we have stressed how difficult it is to establish whether distributional semantics and dictionary definitions are equivalent semantic descriptions. We have however also stressed in Chapter 2 that we do not expect distributional information to suffice to encode all of meaning.

One criticism we should particularly take into account is that the DSMs we have been focused on are derived from gradient-based approaches. As such, they are approximate solutions, rather than exact ones, to the objective functions set up to train these models. Moreover, it is not certain that these objective functions describe the distributional hypothesis accurately. In short, we are in principle dealing with approximate solutions to improper simplifications of an unproven

This chapter is based on previously published work (Mickus, Constant, et al. 2021a, "A Game Interface to Study Semantic Grounding in Text-Based Models"), as well as work currently under review.

hypothesis.

It therefore makes sense for us to look into the quality of our embedding models, insofar they are implementations of the distributional hypothesis. This quality assessment should moreover not rely on some contingent property of these vector spaces, but rather sharply focus on whether they do implement the distributional hypothesis of Harris (1954). The first hurdle we have to overcome, however, is to determine whether or not distributional semantics models can be considered as a coherent group on a theoretical level: as we have seen in Chapter 1, the unity of the field seems more chronological in nature than theoretical, as the objects that have been dubbed DSMs are extremely varied. We will propose in Section 5.1 a unifying framework for implementations of distributional semantics based on distributional substitutability.

Armed with this framework, we will then question whether this evaluation procedure can be practically put to use through a pilot study in Section 5.2. We will then focus on what data we need to collect in Section 5.3, and analyze our results in Section 5.4. Finally, we discuss what conclusions can be drawn from this analysis in Section 5.5.

5.1 Distributional substitution

As we had reviewed in Chapter 1, DSMs are extremely varied in their architectures and target objectives. This variety strongly suggests that the exact architecture design of an embedding model bears on the analysis results we observe. It therefore makes sense to root any comparison we may conduct in the design choices of the model we compare. One consequence of this is that distributional

semantics may be a rather loosely defined label given to miscellaneous models, and any work purporting to discuss distributional semantics as a theoretical framework would have to juggle between different architectures.

There is worth in critically assessing this statement. While it is certain that existing models of distributional semantics do not adhere to a strictly defined framework, we can nonetheless examine and compare existing models to see whether a common characteristic is shared across implementations.

One characteristic that may be fit for this purpose is the distributional substitutability proposed in the seminal work of Harris (1954). He remarks that it is possible to establish classes of items for which we can make similar statements in terms of their occurrences. By manually examining a given corpus, we may find that the environments in which some item X occurs are essentially the same as the environments in which some item Y occurs. In essence, this entails that we are justified to think that we can *substitute* X for Y, and Y for X, based on their distribution.

Sahlgren (2008) reviews and builds on this idea of distributional substitutability. More precisely, Sahlgren stresses the theoretical connection between this distributional substitutability and the paradigmatic axis in the structuralist framework of linguistics (Saussure, 1916). A key remark he makes is that the distributional hypothesis can be re-framed as stating that differences of meaning entail differences of distribution. This differential view of meaning has its roots in the structuralist work of Saussure (1916) and Bloomfield (1933). Sahlgren more specifically draws on Saussure's concept of "*valeur*"—i.e., the purely linguistic difference in meaning attributed to a linguistic sign, owing to its unique combinatorial properties.

The Saussurian *valeur* of a sign, as a differential conceptualization of meaning, is characterized both by the allowed positions of the sign on the syntagmatic axis—simply put, the sorts of syntactic contexts where the corresponding sign may occur—as well as the relations this sign entertains within the paradigmatic axis—i.e., how it differs from other words that could fit in this slot. Or, directly quoting Sahlgren (his emphasis):

> Paradigmatic relations hold between linguistic entities that occur in the same context but not at the same time, like the words "*hungry*" and "*thirsty*" in the sentence "*the wolf is [hungry|thirsty]*". Paradigmatic relations are substitutional relations [...]. A *paradigm* is thus a set of such substitutable entities.

While Sahlgren (2008) does not explicitly equate the distributional substitutability of Harris (1954) with the paradigmatic axis of Saussure (1916), the connection immediately derives from his analysis.

This principle of substitutability has been used in other studies. One major contribution to highlight here is that of Ferret (2021), who proposes to substitute words in context to derive pairs of contextual embeddings for different word types in identical contexts. This allows Ferret (2021) to test contextual embeddings on typical word-type benchmarks and tasks, such as measuring the cosine similarity of contextual embeddings for co-hyponyms, antonyms, and other semantically related words.

We can however go beyond simply swapping items in a given context. One can continue the line of reasoning from Sahlgren (2008) by considering how statements of distributional substitutability should translate to DSMs. To do so, one

can propose a more formal definition of substitutability: two words w_1 and w_2 are substitutable if and only if they are equally likely to occur in any context c. This could be formally written as:

$$\text{Substitutable}(w_1, w_2) \iff \forall c \in C \ \Pr(w_1|c) = \Pr(w_2|c) \tag{5.1}$$

It should be noted that this definition is rather strict, and requires us to have a notion of the set of all contexts C. Let us also stress that the formal definition we propose in Equation (5.1) involves a probability distribution Pr, rather than observations of utterances as per the initial definition in Harris (1954). This modification is necessary if we want to translate Harris's definition to DSMs in general and neural embeddings in particular, as such models only output probability distributions, not utterances.

Let us re-frame Equation (5.1) to consider cases where words are not substitutable. For a given context c, we can consider which of two words is most likely. This naturally yields the following inequality:

$$\Pr(w_1|c) > \Pr(w_2|c) \tag{5.2}$$

In essence, we expect that DSMs are able to characterize the effect of substituting one word (w_2) for another one (w_1) within a given linguistic context (c). This corresponds to a slightly different approach than the original proposal of Harris (1954) we consider a fixed context, and investigate which words it is most appropriate for. As such, we can start from some given context c that we know will contain the first word w_1, and see what the effects of substituting in w_2 would

be.

Crucially, many—if not most—word embedding models are able to yield an expression such as the one in Equation (5.2).[1]

To begin with, we can highlight that some DSMs directly model $\Pr(t|c)$. This is in particular the case of sequence denoising objectives like BERT's Masked Language Model objective (Devlin et al., 2019), where a "masked" item has to be uncovered from its context. This is also the case of multinomial classification objectives like CBOW (Mikolov, K. Chen, et al., 2013), where the target item is to be predicted by the summed activation of the context embeddings. Lastly, bidirectional language models like ELMo (Peters et al., 2018) that predict a word from the past and future context are also subsumed in this category.

A category of DSMs where we need to put in a bit more effort to arrive at Equation (5.2) are models like Skip-gram (Mikolov, K. Chen, et al., 2013). Their loss function takes the form of $P(c|\cdot)$; meaning that we need to apply Bayes' rule to arrive at Equation (5.2); or formally:

$$\begin{aligned} P(t_1|c) > P(t_2|c) &= \frac{P(c|t_1)P(t_1)}{P(c)} > \frac{P(c|t_2)P(t_2)}{P(c)} \\ &= P(c|t_1)P(t_1) > P(c|t_2)P(t_2) \end{aligned} \tag{5.3}$$

It should be noted that this equation relies on the probability of a token $P(t_i)$. Crucially, said probability in word2vec architectures is truncated, by mean of a temperature sampling. Recall from our overview in Section 1.2.1 that in word2vec,

[1] A major category that we have to consider separately concerns document-based DSMs, as the sort of context c they rely on differs from word-based DSMs. Remark that Sahlgren (2008) argues that such models encode information on syntagmatic relations, rather than word paradigms.

a term w is dropped with probability:

$$\hat{P}(w) = 1 - \sqrt{\frac{t}{P(w)}}$$

where t is the temperature parameter, typically 10^{-5}, and $P(w)$ is the non-modified, frequency-based probability. The idea is to under-sample very frequent words, and over-sample rarer words. It is therefore more principled to use the re-sampled distribution $\hat{P}$ instead of the raw frequency P here, as it avoids a distributional shift between training and testing.

Moving on, we see that negative sampling approaches like FastText (Bojanowski et al., 2017) and ELECTRA (Clark, Luong, et al., 2020) measure whether a term is valid for a given context. Formally, this is written as $P(t \in c)$, and is reflected by the use of a sigmoid to compute a binomial distribution probability. For these objective functions, we can simply consider which term yields the highest probability of being a good fit for the current context, namely:

$$P(t_1 \in c) > P(t_2 \in c) \tag{5.4}$$

More precisely, to arrive exactly at the previous Equation (5.2), it is possible to renormalize the probability $P(t_i \in c)$ with respect to the entire vocabulary:

$$\Pr(t_i|c) = \frac{1}{\sum_{t_j \in V} P(t_j \in c)} \cdot P(t_i \in c)$$

The above does sum to 1 over the full probability space of the vocabulary V, is defined with respect to the context c, and simplifies to the aforementioned

Equation (5.4), as $\frac{1}{\sum_{t_j \in V} P(t_j \in c)}$ is constant for a given context c.

Generative language models—for instance, that of Bengio et al. (2003) or more recently, the GPT models of Radford (2018)—can also be folded into this same framework. Generative objectives seemingly ignore half of the context: typically, they yield the probability of t_i being the next token: $P(t_i|c_{1...i-1})$. While we could simply adopt this truncated form as the definition of a context for such models, it is in fact possible to coerce them into taking into account the full context. More precisely, we leverage an idea proposed in the literature on probing syntactic agreement in neural networks, as envisioned by Linzen et al. (2016) and Gulordava et al. (2018). We can therefore derive the comparison given the full context $c_{1...n}$ as follows:

$$P(c_n|c_{1...i-1},\ t_1,\ c_{i+1...n-1}) > P(c_n|c_{1...i-1},\ t_2,\ c_{i+1...n-1}) \tag{5.5}$$

Essentially, this consists in comparing the probability of generating the full context with the target to the probability of generating the full context with the distractor. We do note that this formulation may not be fully appropriate for generative models with a limited time window, such as that of Bengio et al. Lastly, a similar trick can also be used for models like BART (Lewis et al., 2020). While BART is technically trained using a denoising objective, its sequence-to-sequence generative architecture entails that we consider it along other generative language models.

Lastly, this characteristic further generalizes to non-neural models. If we adopt the "count" vs. "predict" dichotomy of Baroni et al. (2014), we can see that both "count" and "predict" models are based on estimate of the conditional prob-

ability of words given their linguistic contexts: the main difference being that "count" models derive this estimate from descriptive statistics, whereas "predict" models learn it using inferential models such as neural networks. To take a concrete example, recall that count-based matrices approaches, such as that of Erk and Padó (2010) or Reisinger and Mooney (2010), a cell M_{ij} denotes the strength of the association between a term t_i and a context c_j. It suffices to normalize this matrix to arrive at a probability distribution of the form required in Equation (5.2). Alternatively in the case of exemplars and prototype models, one can also apply the cluster assignment or similarity threshold filtering methods proposed to rank terms t given a specific context c.

This distributional substitution allows us to properly compare distributional models, using only the explicit objective functions of neural models or the counted observations of statistical models. The formulation moreover lends itself to an intuitive definition of DSMs: a distributional semantics model is a model capable of discriminating words from their distribution. This ease of statement and broadness of application make distributional substitution a very appealing theoretical ground to compare and contrast distributional models.

Moreover, it naturally lends itself to comparisons with human judgments on the same task: we can easily ask annotators which of two words they would prefer in some set of contexts. Comparing human preferences to model preferences therefore allows us to study whether a DSM matches with what we observe of human linguistic behavior.

In short, distributional substitution is a natural candidate when it comes to assessing the linguistic validity of embedding models. We can contrast it to what we would glean from, say, word analogy tasks (cf. Section 1.3.1). That seman-

tically regular processes are encoded by more or less regular vector offsets is certainly an interesting and highly useful characteristic of a vector space. However, it would be false to assume that not displaying this trait entails that our models are not distributional: at the end of the day, having regular semantic offsets is a contingent, nice-to-have feature of a vector space—not a prerequisite for a DSM.

On the other hand, distributional substitution stems from the very definition of distributional semantics: it is an essential trait that we expect a distributional semantic model to display. Moreover, testing this trait requires solely that we focus on the objective functions of our distributional semantic spaces, making it a purely intrinsic evaluation procedure. It therefore allows us to verify not only the validity of an objective function as an approximation of the distributional hypothesis, but also the degree of approximation that remains after trying to optimize this objective, i.e., the quality of the embeddings as an approximate solution to their objective function.

5.2 Pilot Study

If we wish to use distributional substitution to evaluate our models, we need to be able to characterize what our expectations are. For instance, we might expect that a pair of words such as "*potato*" and "*ecclesiastical*" are easy to distinguish, whereas "*two*" and "*three*" might not be. To evaluate this, collecting human judgments is a necessity. We conduct a pilot study to assess whether our hypothesis is coherent—i.e., whether humans indeed discriminate between easy and hard word pairs—and whether this can constitute a coherent data collection program.

In all, a collection of human judgments on distributional substitutability would correspond to a list of records, each of which containing the following:

1. two distinct candidate words, w_t and w_d,
2. a set of contexts for one of the candidate word w_t where said candidate has been blanked out, C_a,
3. human judgments of how easy it is to confuse the two candidates, h_{ab}.

Note that judgments are not necessarily symmetrical: a word could for instance be easily confused with one of its hypernyms, but not with its hyponyms—or conversely. Thus any pair w_t, w_d entails a second pair w_d, w_t.

Concretely, participants of this pilot study were invited to submit judgments through an online survey app developed specifically for the study.[2] The website is now retired.

5.2.1 Word type vs. word sense judgments

Given the scope of this dissertation, we could consider having annotators distinguish between senses, rather than between word types. In principle, this would allow us to propose a fine-grained comparison between definitions (as they aim to propose word sense descriptions) and word embeddings, both defined at the word type level—i.e., static embeddings, which should conglomerate all the possible senses for a word in a single representation—or at the word token level—i.e., contextual embeddings which should model the sense appropriate to a given context.

[2]Source code for the Django web application: `https://github.com/TimotheeMickus/the-pilot-is-a-dog`

A natural way of collecting judgments at the sense level would be to have our annotators select word definitions d_a, d_b (or any type of semantic gloss for a sense), rather than word types w_t, w_d. This however comes with its own challenges: it supposes that we are able to align senses to contexts. While we could rely on existing definition modeling datasets which link glosses to contexts, this would introduce a non-negligible train/test overlap between the human judgments dataset and models trained on our dataset. This is due in part to the collaborative nature of Wiktionary: as edits are open to any collaborator, it does appear that some definitions are copied or adapted from other online dictionaries—a well-known case being the inclusion of TLFi entries in the French Wiktionary project. A second issue stems from the bias in the sort of sentences used as examples of usage or citations: many dictionaries tend to favor literary examples, which correspond to a subset of all possible genres, styles and registers.

To avoid train/test overlap and the bias towards the literary genre, one would therefore need to learn to align contexts and senses, which is not a trivial problem. More precisely, it would require us to essentially produce a WSD system that would be highly reliable, or to perform a manual verification, which would necessarily limit the scope of our data collection. Collecting data at the word-sense level would therefore prove too costly.

An alternative solution we could consider would be to ask participants to select the definition that is the most appropriate from the pool of all possible definitions for the two words w_t, w_d. Again, this comes with major drawbacks: it is overall more demanding of the annotators—in essence, annotators would need to read all the possible dictionary entries before making a choice. Another issue with this approach lies in that definitions for a given word type can have

varying degrees of similarity between one another: two definitions can be fairly similar or highly distinct. This would require of any analysis down the line to include some metric of similarity between definition glosses; and we have already shown the sort of issues that this approach is met with in Chapter 3.

Lastly, it is worth pointing out that our stated goal being to evaluate word embeddings, it is not necessary to collect annotations at the sense level. In fact, if we want to keep in line with the theoretical approach developed in Section 5.1, it is in a certain respect more appropriate to collect annotations at the word-type level, as this corresponds more closely to what DSMs are confronted with.

In short, it seems both more practical and more theoretically appropriate to not collect human judgments at the word sense level, but rather focus on judgments at the word type level. To ensure that the annotations are at the word type level, we can provide annotators with multiple contexts. This will ensure that we collect annotations for the general trend for two word types—rather than provide one specific context where the two words happen to be ambiguous. To illustrate this last point, let us take a concrete example. We expect the words "*mouse*" and "*keyboard*" to be generally easy to distinguish; but some contexts, such as "*I just bought a new _____ for my computer*" happen to be ambiguous. If we were to provide a single context, we would end up collecting judgments about particular word tokens rather than word types.

5.2.2 Word Pairs Selection

We then need to establish the set of word pairs for which we would like to collect annotations. The first decision we make is to try to target word pairs that will be

difficult to distinguish. This is done for two reasons. First, we expect the majority of word pairs to not be significantly challenging: any two random words will have mostly orthogonal meanings, so to speak. Hence presenting these pairs should invariably result in annotators confidently distinguish the two words. Second, we are interested in finding the limitations of current DSMs. Focusing on a more challenging set of word pairs should make these limitations clearer.

An obvious source of candidate word pairs to start with is words with similar distributional representations, as these are the sort of items we wish to probe. This distributional semantics-based selection of annotated pairs may provide a useful starting point; however, one might want to find some other pair-selection mechanism to avoid any potential bias that distributional models may have implicitly carried. It therefore makes sense to use other lexical semantic resources, such as ontologies like WordNet (Fellbaum, 1998). We may also target word pairs that are known in the literature to be semantically similar. Another alternative would be to employ adversarial data collections, where annotators are divided into two groups: the first produces judgments about a word pair in a given context, the second suggests word pairs for the first group to judge.

In all, this entails we have four distinct word-pair collection strategies: distributional semantics-based, ontology-based, from previous studies, and adversarial examples. We now look at each strategy in more detail.

Distributional semantics. Using word2vec trained on Google-news, a random sample of 10K word pairs was selected. A raw random sample would include exceedingly rare word types, which might correspond to improperly pre-processed textual artifacts. Items were therefore filtered against a lexicon (all lemma names

Word pair		cos	Word pair		cos
aegis	auspices	0.851	fourteen	eighteen	0.851
allay	assuage	0.843	fourteen	nineteen	0.833
altercation	scuffle	0.843	gorgeous	beautiful	0.835
bluegill	crappie	0.837	immensely	tremendously	0.852
cello	viola	0.871	inscribed	engraved	0.851
chanting	chanted	0.889	kilo	kilogram	0.851
eight	six	0.945	male	female	0.841
featherweight	bantamweight	0.841	trombonist	trumpeter	0.843
featherweight	welterweight	0.843	viagra	cialis	0.912
fifth	sixth	0.968	welterweight	bantamweight	0.841

Table 5.1: Distributional pairs produced

from WordNet). The 20 pairs yielding the highest cosine similarity were selected.

Produced pairs are shown in Table 5.1.

Ontology. Using WordNet, items were selected based on how many hypernyms they had in common. More precisely, we selected pairs that maximized the intersection-over-union or Jaccard index $d_j(H_1, H_2)$, using the closure sets of hypernyms for the two words considered H_1 and H_2. Note that the Jaccard index between hypernym sets reaches 1 for synonymous lemmas and inflectional variants. In theory, synonyms should have exactly the same meaning and therefore the same distribution; as our interest lies in gradients of word pair substitutability, we should in principle disregard such word pairs. As for inflectionally related words, note that the setup would devolve in a grammar proficiency test, which is irrelevant to our present hypothesis. We therefore removed candidate word pairs with a Jaccard index equal to 1.

The corresponding word pairs are listed in Table 5.2.

Word pair		cos	Word pair		cos
pear	apple	0.958	haricot	frijole	0.958
fullback	quarterback	0.958	crabapple	apple	0.960
apple	quince	0.958	ingenue	heavy	0.962
loganberry	dewberry	0.958	buckskin	roan	0.962
tangelo	pomelo	0.958	vicuna	alpaca	0.964
tangelo	kumquat	0.958	chardonnay	riesling	0.966
tangelo	citrange	0.958	muscadet	riesling	0.966
tangelo	shaddock	0.958	verdicchio	riesling	0.966
tangelo	citron	0.958	manioc	cassava	0.968
tangelo	grapefruit	0.958	tokay	muscatel	0.969

Table 5.2: WordNet-based pairs

Previous studies of interest. Colors (Zaslavsky et al., 2018) and containers (White et al., 2017) have been suggested as semantically competing words that entertain complex semantic relationships, based on grounded factors. Likewise, embeddings of cities (Louwerse and Zwaan, 2009) are known to correlate to some extent with their geographical locations, i.e., some of their real-world grounded characteristics can be retrieved from their distributions.

We created a basic vocabulary for each of the three categories:

- **Colors**: red, orange, yellow, green, blue, purple, black, white, brown, pink, gray
- **Containers**: bottle, pot, tube, vial, drum, gourd, flask, vase, thermos, teapot, canister, jerrycan
- **Cities**: Birmingham, Leeds, Glasgow, Sheffield, Bradford, Manchester, Edinburgh, Liverpool, Bristol, Cardiff, Belfast, Leicester

The vocabulary for cities corresponds to the most populous UK cities, as listed in Wikipedia. The vocabulary for containers was adapted from (White et al., 2017).

Color		Container		Cities	
orange	purple	thermos	bottle	Belfast	Cardiff
brown	yellow	pot	vial	Birmingham	Bristol
brown	red	thermos	flask	Liverpool	Bristol
green	blue	vase	gourd	Glasgow	Sheffield
purple	yellow	vial	flask	Bradford	Leeds
red	white	teapot	drum	Sheffield	Manchester
orange	yellow	vial	gourd	Birmingham	Sheffield
red	purple	vase	tube	Leicester	Cardiff
green	purple	vial	thermos	Bristol	Belfast
orange	blue	vial	vase	Liverpool	Manchester
red	blue	pot	gourd	Glasgow	Bristol
green	orange	teapot	vase	Sheffield	Leeds
gray	yellow	pot	bottle	Leicester	Bradford
brown	purple	canister	flask	Edinburgh	Bristol
purple	gray	bottle	gourd	Liverpool	Leicester
purple	blue	pot	jerrycan	Sheffield	Bradford
brown	gray	vase	bottle	Liverpool	Edinburgh
brown	blue	vial	canister	Edinburgh	Belfast
green	white	drum	flask	Manchester	Leeds
orange	red	bottle	tube	Bristol	Leeds

Table 5.3: Hand-crafted pairs

We then computed all distinct pairs within categories, and randomly selected 20 pairs per category. The corresponding word pairs are listed in Table 5.3

Adversarial examples. We asked a small separate pool of participants to provide word pairs that they expected to be difficult to distinguish, based on distribution alone. All of them received the set of instructions transcribed in Appendix C.1. Participants tasked with producing adversarial examples were allowed to solicit others. Submitted pairs are shown in Table 5.4.

Submitter 1	
red	blue
fear	surprise
glass	mug
orange	apple
knowledge	belief
brother	friend
ashes	dirt
to give	to lend
cute	beautiful
correctly	accordingly
to run	to walk
democracy	dictatorship
Submitter 2	
annihilate	destroy
daddy	father
lie	fib
thing	entity
spit	sputter

Submitter 3	
box	package
parcel	package
fence	board
traverse	jump
shade	shadow
vampire	mosquito
tangerine	orange
dragon	dinosaur
concede	recognize
Submitter 4	
magic	illusion
witch	sage
sand	gravel
cardboard	paper
sponge	towel
dirt	powder

Submitter 5	
high	tall
blue	turquoise
interesting	fascinating
career	profession
glisten	gleam
Submitter 6	
envy	jealousy
sonnet	quatraine
gallon	litre
this	that
infer	imply

Table 5.4: Adversarially submitted word pairs

Word pair		Word pair	
lazer	female	drinking	siege
east	honeydew	N	chameleon
info	haddock	pocket	sonnet
combat	coop	sorcerer	antiseptic
liquor	possibility	rim	cherry
fluid	mess	hotspot	reparcelling
aeroplane	drag	plenipotentiary	medicine
tycoon	pillagings	dwarf	defens
krib	mite	compromise	countryside
troopship	osteo	plaza	cookie

Table 5.5: Control word pairs

Control items. Control pairs result from randomly selecting 20 word pairs from Universal Dependencies English tree banks (all treebanks excepted English ESL). We sample 40 tokens uniformly over the full vocabulary set derived from the UD datasets.[3] The resulting pairs are shown in Table 5.5; note that the sampling procedures leads to selecting rather infrequent tokens such as "*N*", "*krib*" or "*defens*", which are likely either spelling mistakes, non-standard orthographic variants or textual preprocessing artifacts.

5.2.3 Results

Each participant was presented with five uniformly randomly sampled items from each series: WordNet, word2vec, colors, cities, containers, adversarial examples, and control items. This totals to 35 items to annotate. Picking one word is required, participants may tick a checkbox if they believe the words to be synonyms in the provided contexts. Items (word pairs w_t, w_d and five contexts) are

[3] We first sample 20 items without replacement to be used as targets w_t, then do a second uniform sample without replacement to select distractors w_d.

Ontology	Distributional	Cities	Containers	Colors	Adversarial	Control
0.828	0.626	0.657	0.879	0.889	0.869	0.929

Table 5.6: Results of pilot experiment

presented in random order. For each item, the correct word and the distractor from the pair are presented in random order.

Two optional free-text feedback questions were made available to participants: one asking whether participants believe they can present more challenging word pairs than the one they just saw, one asking for any general comments.

For each word pair we collect four elements:

1. to which word pair this annotation corresponds;
2. whether the correct word was selected;
3. whether the participant considered the two words to be synonyms;
4. during which web session was this annotation produced

The fourth item, in essence, serves as an anonymous identifier for the annotator.

Most word pairs received at least one annotation, some up to 17. Accuracy results per series of word pair, based on the 99 first submitted annotations are listed in Table 5.6 All series are solved above chance (0.5).

Unsurprisingly, using cosine to retrieve distributionally similar words seems to be the hardest challenge, followed by distinguishing cities. There is almost a 20% difference with the next series, i.e., using WordNet hypernym set overlaps. Adversarially submitted pairs, followed by containers and then colors, yield accuracy scores of 0.04 to 0.06 points above what we observe for WordNet. Control

pairs are solved 92% of the time. Removing annotations by annotators having failed their control item does not modify the above ranking.

When asked whether they could produce more challenging word pairs, participants had varying responses. Out of the 13 responses we received, five participants had a high confidence that they could; out of which three went as far as suggest pairs on the spot. Another group of 4 responses expressed some degree of uncertainty as to the quality of the pairs they provided ("maybe," "I guess"). Only 2 participants replied negatively, suggesting that it would make sense to allow participants to provide word pairs.

We also note 2 participants who provided a more elaborate answer, stressing that the difficulty of the task depends on the exact implementation: e.g., which contexts are selected, and what cultural and social milieu the annotators were from. The general feedback question also echoed some of these remarks on the details of implementation: participants asked whether looking up information on google was allowed and suggested improvements to the interface. An important number of respondents also pointed out that the knowledge and skills required to solve questions varied from word pair to word pair.

5.2.4 Conclusions of the pilot study

In all, this pilot study was able to demonstrate three key elements.

The most crucial point that can be gathered from is that word pairs constructed from DSMs were among the most challenging. This signals that the task, as it is constructed, does indeed relate more to distributional semantics than to alternative means of construing and describing semantic contents, such as on-

tologies like WordNet. As such, the annotation task as it was framed in this pilot study appears to be a legitimate way to investigate distributional semantics.

Just as crucial is the fact that world-knowledge and the socio-cultural milieu of annotators impacts their overall success rate. This was especially apparent for the UK cities based questions: the vast majority of our participants were not born, raised, or connected to the United Kingdom, making this category especially difficult to solve.

The last element that was made clear by this pilot study concerns the construction of annotation items. On the one side, participants themselves were often disoriented by the variety of questions and possible strategies to answer them. On the other side, constructing annotation items for this pilot study also revealed some potential issues: some sentences contained inflected variants of the target or the distractor, making a simple hard-match strategy not viable for selecting contexts. Furthermore, we observed artifacts such as sentences in foreign languages (Spanish, Japanese, Chinese), as well as cases where all the selected contexts matched only with one of the possible senses of a polysemous target. As such, a more fault-tolerant strategy might be to dynamically select contexts for each annotator, rather than pre-compute a set of contexts that would then be presented to all annotators. This dynamic context selection would also simplify the interface and design requirements so as to allow participants to propose their own word pairs.

Overall, this pilot study suggests that the task is well suited to study the validity of the distributional hypothesis. We therefore now turn to implementing an interface for data collection at a larger scale.

5.3 Implementing an interface

The core focus of this chapter consists in testing the limits of distributional semantics as a theory of lexical semantics. There are two aspects in which human knowledge can be useful to our enterprise of investigating the value of the distributional hypothesis:

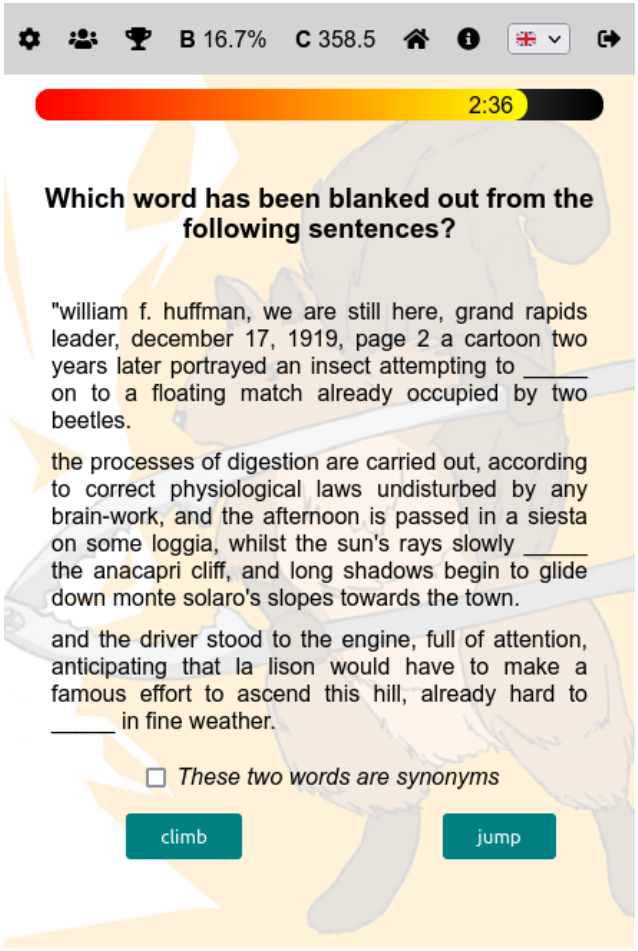
(a) How hard is it to distinguish terms from contexts alone?

(b) Are there terms indistinguishable from context alone?

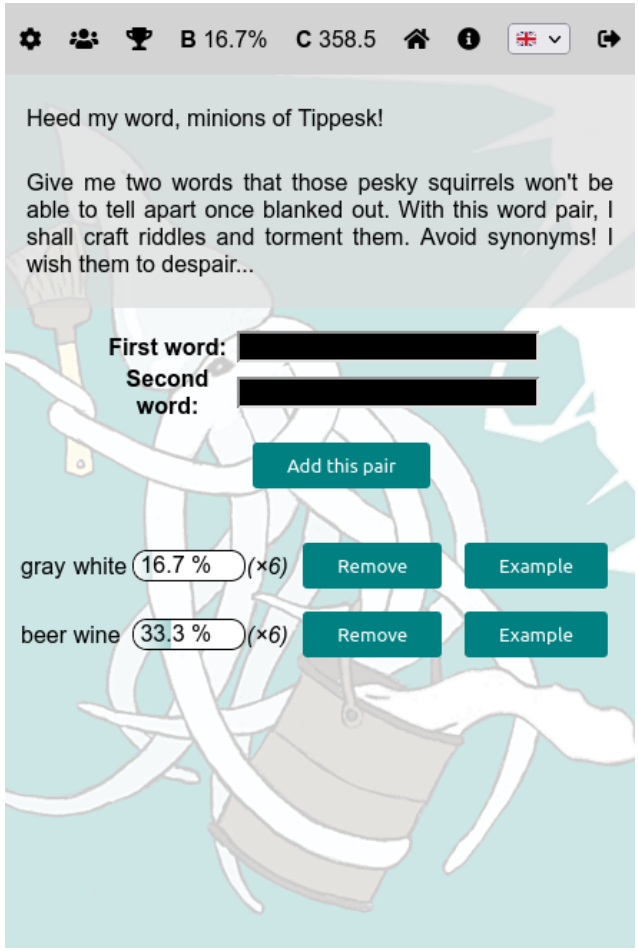
These two related questions lead to two distinct series of data to collect. To answer (a), we would need to collect judgments on the distributional substitution task. Later on, substituting distributional models with human annotators will allow us to compare human judgments to DSM performance. To answer (b), we would instead require human participants to suggest word pairs that they expect to be difficult to distinguish.

The pilot study we detailed in Section 5.2 demonstrated two key elements. First, it highlighted that it is possible to gather judgments and make non-trivial observations using an online platform, which would help us answer (a). Second, it suggested that participants, on the whole, were confident that they could propose challenging word pairs—i.e., exactly what we require to address (b).

As such, these two collection procedures lead us to an adversarial data collection project, where annotators can play either of two roles: proposing word pairs to answer (b) (henceforth (b)-annotators), or distinguishing word pairs proposed in (b) to answer (a) (likewise (a)-annotators).



(a) (a)-annotation interface



(b) (b)-annotation interface

Figure 5.1: Game Annotation Interface

More precisely, these two adverse roles naturally cast themselves into two antagonistic positions in a gamified setup, as the objectives of these two roles are opposite to one another. As such, we propose to collect such human judgments through a game interface, as displayed in Figure 5.1. Supplementary illustrations are available in Appendix D. This game is available online at `https://blankcrack.atilf.fr/`; code for the interface is to be made public at `https://github.com/TimotheeMickus/blankcrack`.

5.3.1 Dataset Construction

We start by some considerations regarding the data we annotate and the format of our collected annotations. In particular, we detail the sentence contexts we select, our initial set of word pairs, the data presented to the annotators, and the data we effectively collect.

As we are interested in establishing a widely applicable benchmark, we collect data for multiple languages: English, French, Italian, Spanish and Russian. These five languages were chosen on criteria of high data availability. We construct the interface so as to facilitate adding new languages to the interface in future releases.

The first element required for our game is a list of distributional contexts, or sentences. We further wish our data to be broadly comparable across languages: we therefore select a comparable number of sentences from comparable but varied corpora. The corpora of sentences furthermore need to be large enough to allow us to dynamically select sentences for any given word pair, as we have outlined in Section 5.2.4. We chose to select four million sentences

per language, equally drawn from four genres of corpora. One fourth of each corpus comes from Wikipedia dumps, one fourth from books corpora (Gutenberg Project, Wikisource, `LiberLiber.it`), one fourth from parliamentary debates (EuroParl (Koehn, 2005) or UN Corpus (Ziemski et al., 2016) for Russian) and the last fourth from OpenSubtitles (Lison and Tiedemann, 2016).

The second type of data we require is a set of word pairs to bootstrap our data collection process, so that both (a)-annotators and (b)-annotators can immediately start. To do so, we consider two strategies. The first "*a priori*" strategy consists in manually constructing pairs that one initially expects to be challenging, such as months, days of the week, numbers (cardinal and ordinal) and colors. Any pair of terms from one of these series can constitute a word pair to annotate.

The second strategy, which we call "distributional" or "w2v-based," consists in automatically discovering distributionally similar items given our corpus of sentence. We train distinct word2vec models for each of our five language-specific corpora. We select hyperparameters with Bayesian optimization, using performance on a formal analogy dataset as the objective to maximize. We randomly sample 1M word pairs, and narrow down to the 250 items whose vectors maximize cosine similarity.

From these word pairs and sentences, we can then automatically construct annotation items. We present each (a)-annotator with two words w_t and w_d from a given word pair $\langle w_t, w_d \rangle$, and k sentences randomly selected such that all sentences contain the target w_t and none contains a word with the same word stem as the distractor w_d. (a)-annotators can freely set $k \in \{1, 3, 5\}$; by default, $k = 5$.

We replace all occurrences of w_t by a blank token "_____" before present-

Target:	`pleura`	**Distractor**:	`diaphragm`
Correct:	No	**Time**:	35.84 s

Contexts:	`best way to dissect the aortic _____.`
	`the _____ and pericardium have both been recorded as points of outlet.`
	`if the _____ be implicated, greater expansion of the upper and outside portion of the left side of the chest in inspiration takes place.`		

Annotator ID:	`dYaGLiFsJz8`		
Creator ID:	N/A (distributional)		

Table 5.7: Example annotation item

ing them to the (a)-annotator. The annotator is then tasked with retrieving which of the target w_t or the distractor w_d corresponds to these blank tokens. Word pairs $\langle w_t, w_d \rangle$ can correspond either to our initial set of word pairs, or to items proposed by (b)-annotators.

To construct our dataset, we collect the following items: the target w_t, the distractor w_d, the k sentences provided to the annotator, whether the annotator correctly selected the target w_t, the time taken to provide an answer, as well as identifiers tracking the annotator and the creator of the word pair. We provide an example item in Table 5.7.

5.3.2 Player Engagement

At its core, our game is score-based, with two distinct scores per user corresponding to performances as (a)-annotators and (b)-annotators. The (b)-annotator score corresponds to the success rate (as a percentage) of the user's proposed word pairs, i.e., how often (a)-annotators failed to solve riddles constructed using the (b)-annotator's word pairs, and selected the distractor w_d instead of the tar-

get w_t. The (a)-annotator score is a running tally of points: (a)-annotator get between 0.1 and 3 points per correctly solved annotation item (where the (a)-annotator selected the target w_t, rather than the distractor w_d), depending on whether the item was solved under 3 minutes, was based on a known difficult pair, or whether the (a)-annotator had set a lower number k of example sentences.

The possibility to set the number k of sentences per riddle is presented in-game as a difficulty level setting. Aside from this setting, we further implement several mechanisms to attempt to keep players engaged. First, we include a competition mode, whereby users compete against one another; this competition mode is based on a "friends list." Second, we ensure that word pairs newly suggested by (b)-annotators get presented to (a)-annotators in priority, so that (b)-annotators receive feedback as early as possible. Third, we also include some materials to share on social media, e.g., when (a)-annotators successfully retrieve the blanked-out word in their annotations multiple times in a row, or at the end of a competition session. Thus far, sharing on social media and competitions have not been used much often by our users.

We also note that users tend to connect only once. One explanation may lie in that "manual" word pairs from our initial set (cf. Section 5.3.1) are felt to be very hard to solve. We are currently investigating mechanisms to combat this trend such as high-score leader-boards displaying username, language and score for top players; our intuition is that it may motivate players to return to the platform to ensure they still appear on the leader-board. Another possibility would be to provide users with a way to opt-out of these word pairs, which we leave for future investigation.

5.3.3 Implementation details

The game is implemented as a Django application (Django, 2020),[4] a python web framework. The web deployment of the application relies on a nginx proxy web-server (nginx, 2021). A continuous Integration workflow is implemented using a Docker container (Merkel, 2014) and triggered by a gitlab pipeline (Gitlab, 2021). The interface itself is coded using responsive web design principles, making the display appropriate to screen sizes ranging from smartphone to computer.

Translations of the web interface are handled using the Django functionalities to that effect; hence translators can handle a plain text file of the data that needs to be translated. Moreover, to facilitate the addition of new languages later on, the code is structured so as to separate all NLP services from the web interface implementations. They are implemented as a local python library, such that the domain where language-experts have to intervene is limited and well-delineated.

Data is handled through a PostgreSQL database (PostgreSQL Global Development Group, 2021). The SQL model is schematically presented in Figure 5.2 as a UML diagram. There are two groups of model classes: data description classes and game mechanics classes.

Data description classes serve to store all textual and annotation data. The `Word` class keeps track of all possible targets or distractors, according to our original sentence corpora; it includes information about the language (`lang`), the word type (`wtype`), and the corresponding stem (`wstem`). The `WordPair` class groups a target (`word1`) and a distractor (`word2`); we further keep track of the original `creator` and the relevant language (`lang`). Sentences are described using the

[4]Documentation, tutorials and information available here: `https://www.djangoproject.com/`

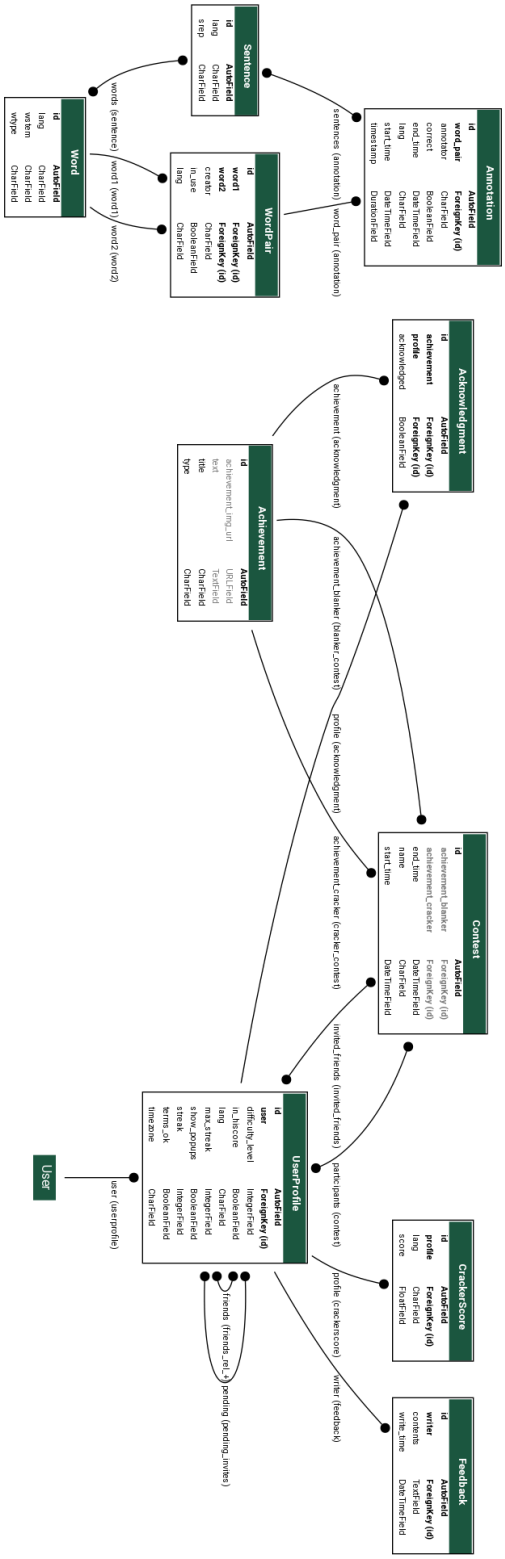


Figure 5.2: BlankCrack SQL model: UML diagram

`Sentence` class, which keeps track of all the `Word` objects it contains as a many-to-many relation, as well as the language (`lang`). Lastly, the `Annotation` class keeps track of the annotations we collect: it keeps track of all `Sentence` objects presented to the `annotator` using a many-to-many relations, and relates to a `WordPair` object using a many-to-one relation; further collected information include whether the user selected the target or the distractor (`correct`), the language (`lang`) and the time taken to produce an annotation (`timestamp`). We use this latter class to produce CSV dataset files, replacing usernames with random strings of characters.

The second group of model classes keep track of user statistics, as well as in-game information. We include two classes here. First is the `UserProfile` class which keeps track of a user's preferences (language, number of sentences per riddle, timezone, friends list, or highest number of back-to-back correct annotations, etc.) Second, we have the `CrackerScore` class, which keeps track of a user's score as a (a)-annotator in a given language. The `Feedback` class gathers suggestions from users. The `Contest` class models to player-versus-player time-bound sessions. The `Achievement` class keeps tracks of notable events that can be shared on social medias by the corresponding users, such as `Contest` results or noteworthy streaks of correct annotations. A newly generated `Achievement` object will produce a pop-up in the game interface, which will then be turned off using an `Acknowledgment` object.

5.4 Analyzing the collected data

Having detailed the online game that we used to annotate and collect data, we now turn to analyzing this data, in hope of answering the question we initially set to address in this chapter: what are the limits of the distributional hypothesis, and of distributional models? In this last section, we will first have a look at the overall contents of the data in Section 5.4.1. We will then try to compare our annotators' judgments to DSMs in Section 5.4.2 and Section 5.4.3. Lastly, we will attempt to manipulate the distributional hypothesis in Section 5.4.4.

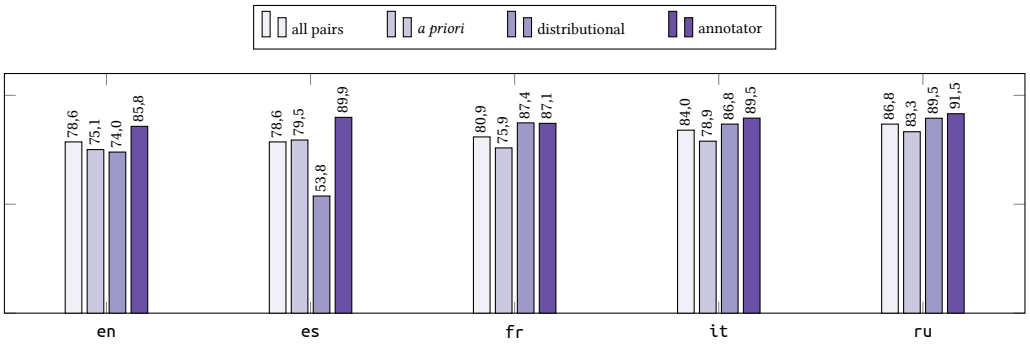
5.4.1 Contents Overview

	en	es	fr	it	ru
$k = 1$	329	110	540	161	113
$k = 3$	58	90	136	73	90
$k = 5$	2223	2044	3719	816	3991
Total	2610	2244	4395	1050	4194

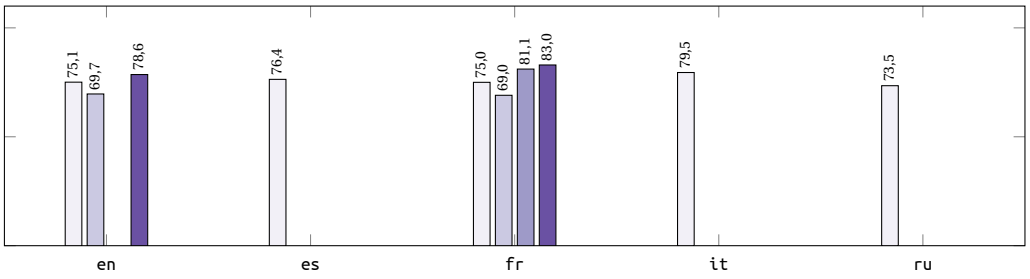
Table 5.8: Number of items collected

The analyses presented here are derived from a set of 14493 annotations. An overview of how these items are distributed across languages and numbers of contexts (k) is displayed in Table 5.8.

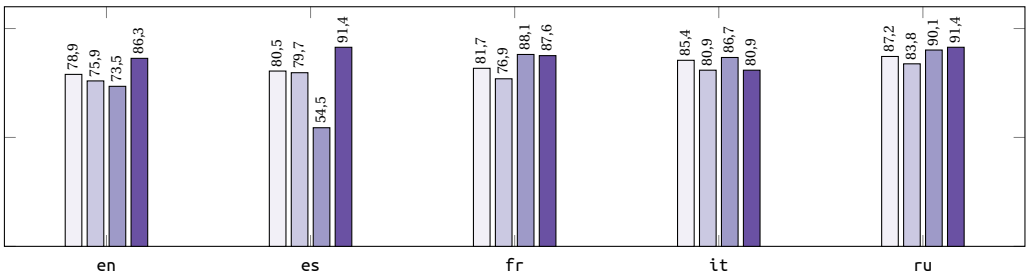
Figure 5.3 displays the overall success rate of annotators; i.e., the percentage of annotations where they were able to select the target word over the distractor. Each sub-figure presents a different condition: Figure 5.3a shows results over the full dataset, whereas Figure 5.3b and Figure 5.3c display results according to the number of contexts shown to the annotators. We do not include results for $k = 3$,



(a) Overall



(b) $k = 1$



(c) $k = 5$

Figure 5.3: Success rates (in %), groups with fewer than 100 items not included

as most groups contained less than 100 items.

We observe task is not as trivial as one might expect. If we look at the overall tally (Figure 5.3a), and average across all five languages of our study, we get a success rate of 82%. For all languages, at least 13% of the items considered here have received an incorrect response from human annotators. The overall difficulty can jump to more than 26% if we consider the most challenging setups, where annotators only have access to $k = 1$ sentences (Figure 5.3b). Even in the most informed setup with $k = 5$ (Figure 5.3c), we find that the best language remains below 90% accuracy overall. It is also instructive to compare the strategies used to define word pairs: those suggested by annotators tend to be the easiest of all; whereas *a priori* word pairs tend to be harder than the average case. Lastly, the surprising difficulty for Spanish distributional word pairs comes from the fact that our original Wikipedia sample contain a number of extremely similar sentences, focusing on botanical nomenclature.

Even in the best of cases, annotators select the distractor rather than the target almost one out of every ten items. This difficulty could be imparted, among other factors, to our approach in collecting this data: we preprocess the sentences we present to annotators automatically and rely on crowd-sourcing to retrieve human judgments on the distributional substitution task. Nonetheless, it suggests that meaning cannot be entirely retrieved from distribution alone: extra-linguistic context is necessary (cf. Section 2.2). Adding strength to this latter analysis, we can tentatively identify some word pairs that are not reliably distinguished by human annotators with access to linguistic contexts: for all languages, roughly 5% of all word pairs that have been seen by 5 or more annotators have a corresponding average success rate at or below chance level. Such

pairs often include co-hyponyms: in French we find aquarelle 'watercolor' vs. gouache 'gouache', in Spanish we have frambuesa 'raspberry' vs. fresa 'strawberry', Russian yields беркут 'golden eagle' vs. кречет 'gyrfalcon'), and in English we find baseball vs. basketball.

5.4.2 Success rates

If we wish to assess how well distributional models are able to assess Equation (5.2), we can look at how often models correctly retrieve the target.

Methodology

We start by considering a 1-gram baseline and a 2-gram baseline. Both are tabulated from corpora comparable to the ones used as basis for our dataset. We further ensure that there is no overlap between the corpora we use to compute our n-gram baselines and those used to construct our dataset.

We also include pre-trained models based on the BERT architecture of Devlin et al. (2019), or variants thereof. We select the following models: BERT (base, uncased) for English, BETO (Cañete et al., 2020) for Spanish, CamemBERT (Martin et al., 2020, base) for French, UmBERTo[5] for Italian and RuRoberta (large)[6] for Russian. Finally, we consider word2vec models (Mikolov, K. Chen, et al., 2013), trained on up to 500M sentences from the Oscar dataset (Ortiz Suárez et al., 2019), using the gensim library (Řehůřek and Sojka, 2010), with default hyper-parameters. We do so instead of relying on the DSMs we constructed in Section 3.4.1 so as to preemptively rule out any suspicion that data limitations or

[5] https://github.com/musixmatchresearch/umberto
[6] https://huggingface.co/sberbank-ai/ruRoberta-large

poor training are to influence our results.

As all our models are able to assess the probability of a word in a given context $p(w|c)$, we can extract a prediction by considering whether the probability associated to the target word $p(w_t|c)$ is greater than the probability associated to the distractor in the same context $p(w_t|c)$.

In practice, we found it more effective to consider the sum of log probabilities across all contexts $c_1, \ldots, c_k$ within an annotation item:

$$\sum_k \log p(w_t|c_k) - \sum_k \log p(w_d|c_k) > 0 \tag{5.6}$$

Whenever Equation (5.6) holds true, the associated model correctly assigns a higher probability to the target w_t than to the distractor w_d. It should be noted that Equation (5.2) and (5.6) are not strictly equivalent. However, using log-probabilities matches more closely the training objectives of the models we consider: both the MLM objective and the objective function of word2vec models are implemented as cross-entropy minimization objectives.

As BERT models rely on masking word pieces, rather than word tokens, we derive the scores in Equation (5.6) by masking all the word pieces of the target, and sum the associated log-probabilities, then substitute in the distractor and sum the log-probabilities associated with its word pieces. As for word2vec models, we derive the prediction by considering its explicit probability distribution, as it is computed during training using the weights associated to modeling the context.

	en	es	fr	it	ru
Size	2051	1686	3443	749	3926
Reduct. (%)	78.6	75.1	78.3	71.1	95.0

Table 5.9: Effects of filtering on dataset size

Results

We can now compare the success rate of models to that of humans. To tabulate these scores, we dropped annotations that took too long or too short: we dropped any annotations where the logarithm of the time taken by the annotator was more than one standard deviation apart from the mean, to ensure that we remove the least trustworthy annotations. To avoid likely train/test overlaps, we also remove any sentence originating from Wikipedia. The quantitative impact of this preprocessing is displayed in Table 5.9.

	en	es	fr	it	ru
human	83.1	86.9	83.8	89.1	87.8
1-gram	51.9	56.2	53.4	50.8	57.2
2-gram	60.4	71.2	66.0	70.7	60.1
BERTs	75.8	71.6	74.1	76.1	74.4
W2Vs	75.5	77.1	75.5	74.8	72.5

Table 5.10: Success rates (in %)

Results are described in Table 5.10. We include the success rates of human annotators on the items we retain for comparison. All models considered yield results above chance level (50%). The various BERT models attain a success rate between 71.6% and 76.1%; the macro-average across all languages reaches 74.4%. This is still below what we see for humans (83.1% to 89.1%, averaging to 86.1%), but systematically above n-gram baselines: the 1-gram average across languages

is at 53.9%, the 2-gram average is at 65.7%. The real surprise here is the performance of the word2vec models: despite being designed as purely static embeddings, they achieve a 75.1% average success rate on this contextual task, slightly above what we observe for the BERT models.

Discussion

This overview of models' success rates highlights that word2vec models can obtain performances comparable to what we observe for BERT-like models. This may be due in part to the size of our training corpora, ranging from 60G (EN) to 90G (RU) of data: this is often (but not always) above what some of the BERT models were trained with.

In all, it is surprising to see that these static embeddings can rival contextual embeddings on a contextual task. This lends depth to previous studies which have found static embeddings to be comparable to contextual embeddings on word-type benchmarks (Vulić et al., 2020; Lenci et al., 2021, a.o.). Nonetheless there is still a gap between these models and human performance.

5.4.3 Comparing human and model behaviors

Our previous experiment (Section 5.4.2) has given us a quantitative estimate of the performance of our distributional models. We now turn to assessing whether these models can be construed as models of the linguistic behavior of our annotators.

Binary classification approach

The first approach we consider is to re-frame this question as a binary classification problem. Let us assume our models are perfect linguistic models of human capabilities: if so, we would expect them to match human failure with failure. In other words, any incorrect annotation item should correspond to a negative score, as assessed by Equation (5.6).

Hence we can consider human behavior as the "gold standard" that a model of human linguistic capabilities would try to match. By assessing how our models perform on this binary classification task, we are able to surmise whether their behavior matches that of human—are they puzzled by sentences humans got wrong? Are they confident with sentences humans got right? To answer this question, we can use standard binary classification tools. More specifically, we turn to Matthews correlation coefficient (MCC) to see whether model predictions match with human behavior. This correlation coefficient is computed as:

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP}+\text{FP}) \cdot (\text{TP}+\text{FN}) \cdot (\text{TN}+\text{FP}) \cdot (\text{TN}+\text{FN})}} \tag{5.7}$$

With TP, TN, FP, FN being respectively the number of true positive, true negative, false positive and false negative. In other words, the MCC subsumes a confusion matrix under a single figure between 0 and 1, such that a higher value is assigned when the number of false positive or false negatives is minimized.

Results are shown in Table 5.11. The difference between n-gram baselines and distributional semantics models that clearly emerged from Table 5.10. For our three Romance languages, we find that the 2-gram baseline yields a higher correlation coefficient than both word2vec and BERT. In English, the word2vec

	en	es	fr	it	ru
1-gram	0.157	0.158	0.158	0.119	0.177
2-gram	0.156	0.211	0.200	0.193	0.143
BERTs	0.208	0.178	0.150	0.077	0.230
W2Vs	0.135	0.185	0.170	0.122	0.199

Table 5.11: Matthews' correlation coefficient

model is found to yield the lowest MCC; in French and Italian, the CamemBERT and UmBERTo models yield the lowest MCC.

It is hard to argue that the distributional models correlate more with human behavior than the n-gram baselines. On the other hand, we can stress that all the models we tested yielded a positive correlation. This suggests that the behavior of our DSMs is not unrelated to humans—although it is certainly not a close match either. In all, the mistakes and successes of our DSM models do not necessarily align with that of human annotators.

Ranking approach

There are two obvious caveats that one can think of in the methodology we adopted in Section 5.4.3. First, it pits model efficiency against linguistic validity: a model can't be both always correct and match human failures with failures of its own. Second, it relies entirely on treating human annotations as a gold standard—even when annotators have selected the wrong answer.

The simplest way to address both of these concerns is to depart from the binary approach, and see instead whether human uncertainty is matched with lower scores from the models. In principle, a model could always choose the right answer, but lower its score for difficult items—i.e., those annotators struggle with.

Norm.	en	es	fr	it	ru
none	–	–	–	0.417	0.390
sents.	–	0.458	0.449	0.373	0.376
words	0.447	0.385	0.454	0.421	0.452
chars.	0.462	0.395	0.455	0.417	0.459

Table 5.12: Common language effect size from Mann–Whitney U tests for log time taken when answering correctly vs. incorrectly

Considering the uncertainty of our annotators also entails that we factor in how confident they are in their judgments.

This approach requires some sort of measurement of annotator uncertainty, beyond the binary annotations we have exploited thus far. To that end, we focus on the time it takes an annotator to answer a question. We can expect that an annotation item that is easy to judge should take less time than an item requiring careful consideration. Furthermore, as annotators should have no difficulty to correctly guess easier items, we expect that the time taken to answer correctly should be less than the time taken to answer incorrectly. We also consider normalizing the time taken by the number of sentences (i.e., k), the number of words across all sentences, or the number of characters across all sentences. Our reasoning is that the time taken by an annotator also depends on how much text they have to read.

In Table 5.12, we consider various time indicators: either the raw log seconds taken,[7] or variants normalized by some measure of the length of the annotation item. Measurements are done using a Mann–Whitney U test. For two sets A and

[7]Our dataset contains both annotations completed in a few seconds, as well as annotations completed in more than ten minutes. A logarithmic transformation shifts the distribution from a power law to an almost normal distribution.

B, this test defines a U statistic as:

$$U_{A,B} = \sum_{A_i \in A} \sum_{B_j \in B} \begin{cases} 1 & \text{if } A_i > B_j \\ 1/2 & \text{if } A_i = B_j \\ 0 & \text{if } A_i < B_j \end{cases} \tag{5.8}$$

In other words, higher values are assigned when items in A are greater than items in B Here, we use this test to see whether the distributions of time indicators differ between correctly annotated items and incorrectly annotated items: we then compute the common-language effect size, i.e., the U statistic divided by the maximum value it could assume, $\#A \times \#B$, so as to normalize it between 0 and 1. Here, a lesser value of ρ entails a greater certainty that the incorrect annotations have greater associated values—that is to say, they require more time than the correct annotations. Statistically insignificant effect sizes are not reported.

By studying the results of Table 5.12, we see that the raw time measurement is not always significant. However, when factoring in the length of an annotation item, we do detect that annotators take longer when they answer incorrectly than correctly: this is consistent with time being an indicator of uncertainty. Interestingly, we note that the best length normalization differs across languages, but explaining what typological factors drive this difference is beyond the scope of the present study.

Having found a way to quantify uncertainty, we can now include it in our original annotations. We re-weight human annotations to factor in time, such that highly confident correct answers lie at one end of the spectrum, and highly confident wrong answers lie at the other end of the spectrum. This also ensures

that we match as closely as possible how we derive scores from our models. Technically, we re-weight human judgments as follows:

$$\left(\max s^* - s\right) \times \begin{cases} +1 & \textit{if correct} \\ -1 & \textit{otherwise} \end{cases} \tag{5.9}$$

where s is the length-normalized time indicator $\log t/N$, with N either the number of sentences (for FR, IT, RU) or words (for EN and ES), and $\max s^*$ is the maximum value observed for s across all annotations for that language.

	en	es	fr	it	ru
1-gram	0.149	0.115	0.132	0.163	0.147
2-gram	0.119	0.150	0.228	0.267	0.146
BERTs	0.225	0.152	0.204	0.218	0.258
W2Vs	0.145	0.196	0.244	0.165	0.248

Table 5.13: Spearman correlations of model scores and time-weighted human judgments

As we have two related series of continuous measurements, we can apply a simple correlation metric, such as Spearman's ρ, between time-weighted annotator responses and model scores. This is shown in Table 5.13. In English, French and Italian, either or both DSMs yield a lower correlation than what we observe for n-grams, while in Spanish the margin between BETO model and the 2-gram baseline is less than 0.002. Only in Russian do we find a sharp distinction between DSMs and n-grams. Overall, although correlation scores are always positive, they remain fairly low ($\rho < 0.27$).

Discussion

In all, while the models do display some degree of performance (as shown in Section 5.4.2), neither the sort of mistakes they do (Section 5.4.3) nor the confidence in their answer (Section 5.4.3) matches closely human behavior. In many cases, distinguishing DSMs from n-gram baselines can prove very arduous. In other words, models that perform relatively well on the distributional substitution task are not necessarily linguistically accurate.

5.4.4 Manipulating the distributional hypothesis

Our experiments thus far have focused on seeing whether DSMs model human behavior. We could instead reverse the setup, and see whether a low score from a DSM entails a greater hesitation from the human annotator. In effect, this mirror approach would imply that we do not focus on the entire distribution of scores, but rather put more emphasis on the most extreme values—where we would expect the greatest impact on human behavior.

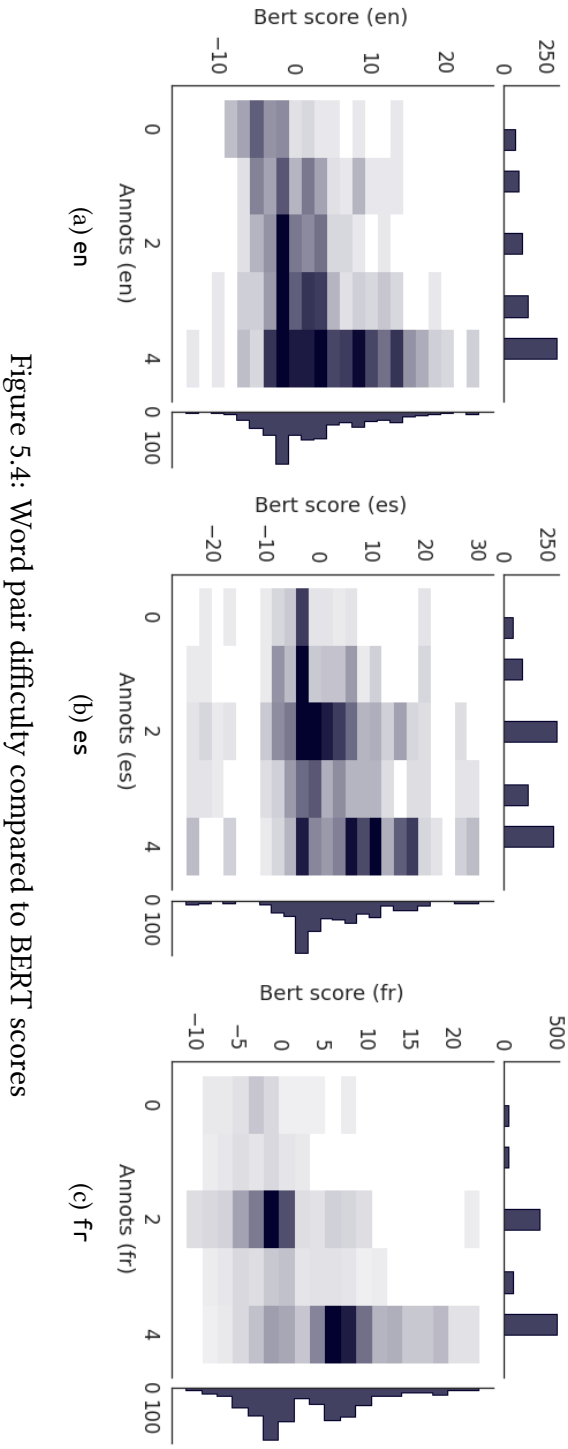
Methodology

Our approach this time around will be to select sentences that either maximize or minimize Equation (5.6), and see how human annotators fare on these contexts, and how confident they are in their answers. We start by selecting the most extreme word pairs, in terms of average success rate. For each word pair, we select a random sample of up to 10000 sentences from the original sentence corpora detailed in Section 5.3, and rank them according to the score a BERT-like model would give them, following Equation (5.6). We then restrict our random

sample to the five sentences with the lowest scores and the five sentences with the highest scores, while making sure that sentences are uniquely associated to word pairs. In other words, if some sentence c_p is among the ten items chosen for a pair $\langle w_t^n, w_d^n \rangle$, then it will not be chosen for any other pair $\langle w_t^m, w_d^m \rangle$. This last restriction is required so as to ensure that participants never have access to the target word in its actual context: otherwise, we would present twice the same context c_p—the first time with the word w_t^n blanked out, the second time with the word w_t^m blanked out—and annotators may be able to recall the target they saw previously.

We then recruit annotators to review this data. Unlike the main dataset, we only present contexts one at a time: annotators only see one sentence with the target word replaced by a blank token. Our reasoning is that we are interested in the ability of a DSM to rank sentential contexts, and presenting multiple sentences at once would prevent us from retrieving which specific context clued in the annotator. Another difference is that we ask annotators to express themselves using a five-point Likert scale, ranging from high confidence in the target to high confidence in the distractor.

Annotators are asked to review up to 500 items; for each language, two annotators work on the full dataset. All annotators are native speakers of the language they worked with, and have been raised in a country where this language is commonly spoken. Due to these limitations, we were only able to gather data for English (200 items), Spanish (430 items) and French (500 items).



(a) en (b) es (c) fr

Figure 5.4: Word pair difficulty compared to BERT scores

Results

Figure 5.4 pits the scores derived from BERT on the y-axis against the corresponding Likert scale annotations, for each language; the heatmap in the middle of each picture displays how the two distributions coincide. These illustrations clearly show that both annotators and the BERT models behave differently across languages. However there are similarities: in all three languages, annotators match high BERT scores with a strong preference for the target. In French (Figure 5.4c), annotators and BERT seem to closely match in their behavior: a neutral response is elicited when the score is low, whereas a confident preference for the target corresponds to a high score. In the other two languages, low scores are spread out across the scale. In Spanish (Figure 5.4b), scores around zero elicit a neutral response, but scores below zero do not seem associated to a specific response. In English (Figure 5.4a), we see a linear trend: the very lowest BERT scores tend to elicit a strong preference for the distractor.

To provide a more quantitative outlook, we turn to a dominance analysis of which factor is most closely related to our annotators' behavior. Dominance analysis consists in learning a simple linear regression, computing the associated r^2 to measure its fitness, and computing what proportion of this r^2 can be imparted on each predictor. We refer the reader to the original description by Budescu (1993) for the full derivation and only provide here a basic exposition.

A r^2 score, also known as a coefficient of determination, corresponds to the proportion of the variation of the dependent variable is explained by a linear

model's predictors. It is computed as:

$$r^2 = 1 - \frac{\sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2}{\sum_{i=1}^{n} \left(y_i - \mu_Y\right)^2} \tag{5.10}$$

Where $Y = (y_1, \ldots, y_n)$ corresponds to our observations of the dependent variable, $\hat{Y} = (\hat{y}_1, \ldots, \hat{y}_n)$ corresponds to the predictions of the model at hand, and μ_Y is the mean value for Y. The r^2 score involves a sum of squared error $\left(y_i - \hat{y}_i\right)^2$, normalized by the observed variance $\sum_{i=1}^{n} \left(y_i - \mu_Y\right)^2$. Note that a theoretically perfect model that produces no error should receive a score of 1. Any degree of error will decrease this score; negative values correspond to models worse than systematically producing the mean observed value of the dependent variable

Dominance analysis is based on the fact that the fitness of linear model, as measured by its r^2 score, can be rewritten as the sum of contributions C_{p_i} from specific model predictors $P_M = (p_1 \ \ldots \ p_n)$.

$$r^2_{P_M} = \cdot \sum_{i=i}^{n} C_{p_i} \tag{5.11}$$

where $r^2_{P_X}$ is the fit associated with the set of predictors P_X. Individual contributions C_{p_i} are here defined based on the effect of including or excluding the related predictor p_i. For any predictor p_i, consider all possible subsets $P_{M'} \subseteq P_M \setminus \{p_i\}$ of predictors other than p_i. For each subset $P_{M'}$, we can compare the fit we observe using $P_{M'}$ on its own, to what we would gain by adding p_i:

$$C_{p_i}^{(P_{M'})} = r^2_{P_{M'} \cup \{p_i\}} - r^2_{P_{M'}} \tag{5.12}$$

By summing over all possible subsets $P_{M'}$, we can quantify the importance of the predictor p_i based on the improvements it brings, taking into account the information captured by all other predictors. In practice, to ensure that individual contributions C_{p_i} sum at the r^2 of the initial set of predictors P_M, we take a two-step average. We first compute $C^k_{p_i}$, the average importance across subsets with the same number k of predictors to consider, and then take the average across all subset sizes k:

$$C^k_{p_i} = \frac{1}{\binom{n-1}{k}} \cdot \left(\sum_{\{P_{M'} | \#P_{M'} = k\}} C^{(P_{M'})}_{p_i} \right)$$
$$C_{p_i} = \frac{1}{n} \cdot \sum_{k=0}^{n-1} C^k_{p_i} \tag{5.13}$$

Crucially, individual contributions $C^{(P_{M'})}_{p_i}$ only depend on r^2 measurements of sub-models. These can be obtained directly by performing linear regressions for the corresponding sub-models and computing the associated r^2 scores.

In our case, we will use dominance analysis to look at a model predicting the average Likert score by annotation item. As predictors, we will use the original average success rate for that word pair, as well as the original BERT score. This will allow us to compare these two metrics as competing explanations for the Likert annotations we collected. We also introduce other predictors that we expect to play a role: the original source of the sentence shown to the Likert annotators (as a four-dimensional one-hot vector), the average length-normalized log time taken for the word pair in the original dataset, and the log frequency of the target and the distractor.

Results are presented in Table 5.14. The r^2 of each linear regression is given

	BERT	succ.	time	freq.	src.	r^2
en	76.86	8.97	9.01	3.80	1.36	0.28
es	59.18	4.22	4.20	21.24	11.16	0.21
fr	81.38	7.15	7.30	1.94	2.23	0.44

Table 5.14: Proportion of r^2 explained by type of predictor (in %)

in the last column; columns 1 through 5 detail the proportion of this r^2 imparted on each predictor (in %). The fitness of the regression, as measured by r^2 scores, suggests that more than half of the variance in annotations is not explained by a simple linear relation between predictors. This is especially striking in Spanish, where the r^2 score is at 0.21. Yet all models consistently rank the BERT score as the most important predictor. The French and English both impart more than 75% of the explained variance on BERT scores and 15% to 20% to average success rates and time taken on the previous dataset. The Spanish model emphasizes more the frequency of the target and distractor (21.24%) and the corpora from which the presented context originate (11.16%).

Discussion

In short, this last experiment stresses that in specific conditions BERT models can prove to be useful tools to manipulate the distributional hypothesis.

This is especially visible on the case of the French data, which yields the most obvious bimodal distribution (Figure 5.4c), the highest r^2, and the largest proportion of variance explained by the BERT model scores. These elements suggest that the CamemBERT model was able to select sentences that strongly cued the target.

On the other hand, we are not able to reliably find French contexts that elicit

a strong preference for the distractor. This is something we only tentatively observe for English, where paradoxically the dominance analysis suggests that our current predictors are less well-suited to explain the phenomena we recorded ($r^2 = 0.28$). This is in line with previous experiments: while high BERT scores translate into a confident preference for the target, much remains to be done in order to accurately depict the full breadth of human behaviors, ranging from strong preferences in the distractor and accurately depicting less confident human judgments.

Opposite to this is Spanish: the lowest scores from BETO do not bias the annotators towards neutral or negative responses. The main reason of this difference is unclear: the quality of the sentences presented to annotators might play a role, but so might the quality of BETO. We also find a much lower inter-annotator agreement for this language: the Pearson r correlation coefficient for our two Spanish annotators is of only 0.11, compared to the 0.59 we observe for English or the 0.74 for French.

Improvements could be made on our analyses: one could use as predictors the average success rate and the average time taken restricted to items with $k = 1$ contexts, as these would be more representative. We leave this to future investigations, as we haven't collected enough data to establish such baselines (cf. Table 5.8). Another point we leave for future study is the number of datapoints in our original datasets: some predictors are derived from them (average time and success rate) and it is unclear how size discrepancy impacts them.

5.5 Conclusions

In the present chapter, we have approached the question of how to quantify our expectations with respect to distributional semantics and distributional semantics models. The short, incomplete answer that we presented here is that distributional information would allow humans to retrieve about 82% of pairwise meaning distinctions. In contrast, embedding models like BERT or word2vec would only reach 75% accuracy, and the way they achieve these performances begs the question of whether we should consider them as models of distributional semantics. We however need to take these numbers in context, as we derived them from an experiment designed to focus on pairs of words difficult to distinguish from distributional information alone. Rather than the exact figure, the crucial point to keep in mind is that embeddings cannot be thought of as perfect implementations of the distributional semantics theory.

To answer our questions, we have provided a dataset of human judgments on the distributional substitution task (Section 5.3), spanning five languages and over 14000 items. We showed how DSMs in general—both modern BERT-like models and earlier word2vec embeddings—had still some margin for improvement, especially if we consider not just their overall efficiency (as in Section 5.4.2), but rather how well they match human behavior (Section 5.4.3). Nonetheless, despite this gap, we find that in limited circumstances, BERT-like models can be used as tools to manipulate the distributional hypothesis (Section 5.4.4), which we deem encouraging for the prospects in the field.

This discrepancy between word embeddings and human behaviors is not without practical consequences. In particular, contextual embeddings-based met-

rics such as the MoverScore metric of W. Zhao et al. (2019) implicitly rely on the assumption that embedding models describe—to some extent—the meaning of words. Such an assumption is perhaps not warranted by the current state of the art; in any event a case-by-case assessment is certainly advisable. Broadly speaking, a similar remark can be addressed to linguistic works using distributional representations as a proxy for meaning, be it when studying compositional semantics (Hill et al., 2016), arbitrariness of the sign (Gutiérrez et al., 2016) or morphological regularity (Bonami and Paperno, 2018). Whether an embedding model behaves in a linguistically coherent fashion on any given dataset should not be taken for granted and must be counted among the assumptions any such study relies on.

Taking a more linguistic-oriented point of view, this chapter suggest two key elements that will be relevant to future research. On the one hand, a unifying framework to study DSMs despite their high degree of variation can be put forward, as we have shown with distributional substitution in Section 5.1. On the other hand, our analyses in Section 5.4 reveal that much remains to be done before we can confidently say that modern NLP models can be construed as linguistically valuable models of distributional semantics. This leaves open a number of perspectives for future research, besides simply augmenting the size of our datasets: how would this translate to other languages, especially non-European ones? What is required of DSMs for them to accurately describe the human judgments we collected? Which factors are the most adequate to model human behavior on the distributional substitution task?

After these analyses, we can now return to the question we initially started this chapter with: how much faith can we put in our distributional semantics

models as theories of lexical semantics? We have established that the currently available models are imperfect, both in that they under-perform humans on the distributional substitution task, and in that they are limited in their capacity to reflect human behavior. To an extent, dictionaries are imperfect as well: lexicographers are fallible and systemic social biases will creep into any cultural production—among which dictionaries are to be counted. This is perhaps best captured by the key fact that year after year, dictionaries see new editions based on earlier versions. In short, dictionaries and embeddings alike have their flaws: the present dissertation might be best construed as investigating whether word embeddings should be taken as seriously as dictionaries when discussing meaning inventories.

Relevant to this discussion, the way in which a dictionary is imperfect is very much distinct from the way a distributional semantics model is imperfect. While we may criticize dictionaries for containing social biases, these biases will also be present in DSMs. More central to our argument is that distributional models are imperfect implementations of the distributional hypothesis: there is an inherent degree of noise in distributional representations that we cannot currently abstract away. And even if we did overcome these implementation issues, we would still be faced with the greater problem: the distributional hypothesis itself does not seem sufficient to properly encode all meaning distinctions. The 82% success rate that we mentioned earlier can also be read as 18% of word pairs that linguistic context on its own is not sufficient to distinguish.

Again, what matters here is not the exact figure we arrive at—be it 18% or any other—but rather that we can arrive at a quantified estimate at all. The experiments we have devised here are incomplete, and call for more research. In

reaching our conclusion that distributional semantics and its implementations have their flaws, we should be mindful of the caveats that come with our experiment protocol. We should keep in mind that we have used an adversarial setup to collect word pairs. We should keep in mind that the number of contexts displayed to annotators was limited to a few, whereas the methodology proposed by Harris (1954) was conceived with large corpora of manually collected linguistic evidence in mind. We should keep in mind that a number without a scale is never straightforward to interpret: we cannot formulate what our expectations are for success on this substitutability task. All of these aspects can and should be discussed: there is a need to confirm and replicate these claims. Nonetheless, to arrive at a quantified observations is a step forward and away from putative judgments of a model's intrinsic limitations; it provides objective facts on which to base our work. It provides a concrete basis for us to discuss the intrinsic limitations of the distributional enterprise.

6

The Structure of Transformer Embedding Spaces

I've done the math enough to know
The dangers of our second guessing
Doomed to crumble unless we grow
And strengthen our communication

— Tool, *Schism*

The previous Chapter 5 has underscored that distributional semantics models did not appear to correspond to human behavior on the distributional substitution task, casting doubt on their validity as models of lexical semantics. That is however not the sole issue we have encountered thus far. As we saw in Section 1.3, methodologies applied to analyze word embedding models are diverse. They distinctly delineate two or three family groups: word-type representations or *static* representations like word2vec, word-token or *contextual* rep-

This chapter is based on previously published work (Mickus, Paperno, Constant, and Deemter 2020, "What do you mean, BERT?") as well as work currently in press (Mickus, Paperno, and Constant 2022, *How to Dissect a Muppet: The Structure of Transformer Embedding Spaces*).

resentations, and attention-based representations (which are invariably contextual). These different methodologies seem to suggest that word-type representations like word2vec are inherently distinct from word-token representations like BERT. Whether we can apply the methodological approach developed for word-type models to word-token models is in fact not trivial.

Whereas traditional DSMs match word types with numeric vectors, contextualized embeddings produce distinct vectors per token. Ideally, the contextualized nature of these embeddings should reflect the semantic nuances that context induces in the meaning of a word—with varying degrees of subtlety, ranging from broad word-sense disambiguation (e.g. 'bank' as a river embankment or as a financial institution) to narrower sub-types of word usage ('bank' as a corporation or as a physical building) and to more context-specific nuances. Regardless of how apt contextual embeddings such as BERT are at capturing increasingly finer semantic distinctions, we expect the contextual variation to preserve the basic DSM vector-space properties. Namely, we expect that the space structure encodes meaning similarity and that variation within the embedding space is semantic in nature. Similar words should be represented with similar vectors, and only semantically pertinent distinctions should affect these representations. But that it *should be* so is no guarantee that it *is* actually so.

Do the different architectures of DSMs entail that the vector spaces they describe are qualitatively different? To answer this question, we will perform a series of experiment on the BERT model of (Devlin et al., 2019). Our first approach in Section 6.1 will be to see whether basic properties of distributional semantics models also hold for BERT contextual embeddings. This will lead us in Section 6.2 to see whether we can provide a mathematical reformulation that

links our observations to characteristics of the Transformer architecture. Using this better informed framework, we will conduct intrinsic and extrinsic evaluations of how the Transformer architecture shapes the BERT embedding space in Sections 6.3 and 6.4. To wrap up this chapter, we will discuss our findings in Section 6.5.

6.1 Is BERT a vector-space model of meaning?

In the previous chapter, we have seen that the probability distribution underlying BERT did not line up with our expectations for DSMs. There is another aspect that we have left aside, namely whether the embeddings describe a coherent semantic space.

6.1.1 Word type cohesion

Our starting point will be that similar words should lie in similar regions of the semantic space. This should hold all the more so for identical words, which ought to be maximally similar. By design, contextualized embeddings like BERT exhibit variation within vectors corresponding to identical word types. Thus we expect that word token representations corresponding to the same word type form natural, distinctive clusters in the embedding space. Here, we assess the coherence of word type clusters by means of their *silhouette scores*, as proposed by Rousseeuw (1987).

Experimental protocol

In this first experiment, we used the Gutenberg corpus as provided by the NLTK platform, out of which we removed older texts (King John's Bible and Shakespeare), whose style and vocabulary was distinct from the remainder of the corpus. Sentences are enumerated two by two; each pair of sentences is then used as a distinct input source for BERT. As we treat the BERT algorithm as a black box, we retrieve only the embeddings from the last layer, discarding all intermediary representations and attention weights. To derive embeddings, we used the `bert-large-uncased` model.

To study the basic coherence of BERT's semantic space, we can consider types as clusters of tokens—i.e. specific instances of contextualized embeddings—and thus leverage the tools of cluster analysis. In particular, silhouette score is generally used to assess whether a specific observation $\vec{v}$ is well assigned to a given cluster C_i drawn from a set of possible clusters C. The silhouette score is defined in Equation (6.1):

$$\begin{aligned}
\text{separation}(\vec{v}, C_i) &= \min\{\underset{\vec{v'} \in C_j}{\text{mean}}\ d(\vec{v}, \vec{v'}) \forall\ C_j \in C - \{C_i\}\} \\
\text{cohesion}(\vec{v}, C_i) &= \underset{\vec{v'} \in C_i - \{\vec{v}\}}{\text{mean}}\ d(\vec{v}, \vec{v'}) \\
\text{silhouette}(\vec{v}, C_i) &= \frac{\text{separation}(\vec{v}, C_i) - \text{cohesion}(\vec{v}, C_i)}{\max\{\text{separation}(\vec{v}, C_i),\ \text{cohesion}(\vec{v}, C_i)\}}
\end{aligned} \tag{6.1}$$

We used Euclidean distance for d.

Silhouette scores consist in computing for each vector observation $\vec{v}$ a cohesion score (viz. the average distance to other observations in the cluster C_i) and a separation score (viz. the minimal average distance to other observations, i.e. the

minimal 'cost' of assigning $\vec{v}$ to any other cluster than C_i). Optimally, cohesion is to be minimized and separation is to be maximized, and this is reflected in the silhouette score itself: scores are defined between -1 and 1; -1 denotes that the observation $\vec{v}$ should be assigned to another cluster than C_i, whereas 1 denotes that the observation $\vec{v}$ is entirely consistent with the cluster C_i.

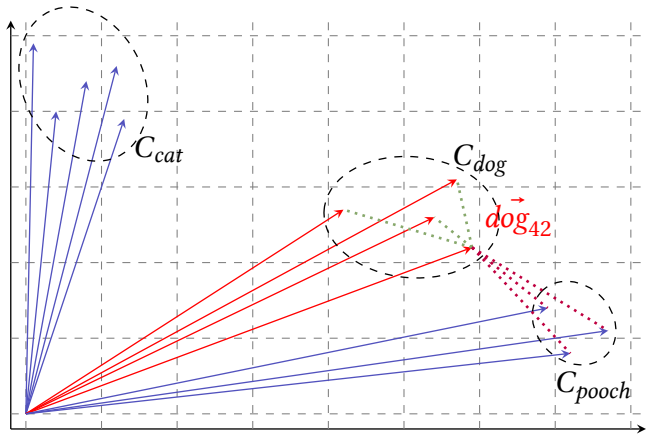


Figure 6.1: Visualization of silhouette score

A visualization of what silhouette scores assess is given in Figure 6.1. In our case, observations $\vec{v}$ therefore correspond to tokens (that is, *word-piece* tokens), and clusters C_i to types. In the illustration, the 42$^{\text{nd}}$ "*dog*" token vector is highlighted as belonging to the cluster C_{dog} for the corresponding word type. We assess whether the distance of this $\vec{dog}_{42}$ vector to other vectors in the C_{dog} is on average smaller than to the distance of vectors belonging to the nearest cluster C_{pooch}.

Keeping track of silhouette scores for a large number of vectors quickly becomes computationally intractable, hence we use a slightly modified version of the above definition, and compute separation and cohesion using the distance to the average vector for a cluster rather than the average distance to other vectors

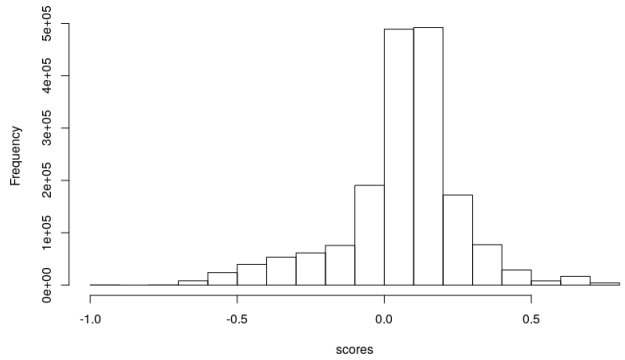


Figure 6.2: Distribution of token silhouette scores

in a cluster, as suggested by Vendramin et al. (2013). Though results are not entirely equivalent as they ignore the inner structure of clusters, they still present a gross view of the consistency of the vector space under study.

We do note two caveats with our proposed methodology. Firstly, BERT uses subword representations, and thus BERT tokens do not necessarily correspond to words. However we may conjecture that some subwords exhibit coherent meanings, based on whether they tightly correspond to morphemes—e.g. "*##s*", "*##ing*" or "*##ness*". Secondly, we group word types based on character strings; yet only monosemous words should describe perfectly coherent clusters—whereas we expect some degree of variation for polysemous words and homonyms according to how widely their meanings may vary.

Results & discussion

We compared cohesion to separation scores using a paired Student's t-test, and found a significant effect (p-value $< 2 \cdot 2^{-16}$). This highlights that cohesion scores are lower than separation scores. To provide a quantitative estimate of the mpor-

tance of this effect, we turn to Cohen's d:

$$d_{A,B} = \frac{\mu_A - \mu_B}{\hat{\sigma}_{A,B}} \tag{6.2}$$

Where $\hat{\sigma}_{A,B}$ is the pooled standard deviation derived from comparing the estimated variances $\hat{v}(A)$ and $\hat{v}(B)$ of A and B respectively:

$$\begin{aligned} \hat{v}(X) &= \frac{\sum_{x \in X} x - \mu_X}{\#X - 1} \\ \hat{\sigma}_{A,B} &= \sqrt{\frac{(\#A - 1) \cdot v(A) + (\#B - 1) \cdot v(B)}{\#A + \#B - 2}} \end{aligned} \tag{6.3}$$

Simply put, Cohen's d normalizes the difference between the mean value in A and the mean value in B by the standard deviation we expect for $A \cup B$.

The effect size as measured by Cohen's d is however rather small, reaching only -0.121, suggesting that cohesion scores are only 12% lower than separation scores. More problematically, we can see in Figure 6.2 that 25.9% of the tokens have a negative silhouette score: one out of four tokens would be better assigned to some other type than the one they belong to. When aggregating scores by types, we found that 10% of types contained only tokens with negative silhouette score.

The standards we expect of DSMs are not always upheld strictly; the median and mean score are respectively at 0.08 and 0.06, indicating a general trend of low scores, even when they are positive. We previously noted that both the use of sub-word representations in BERT as well as polysemy and homonymy might impact these results. The amount of meaning variation induced by polysemy and homonymy can be estimated by using a dictionary as a sense inventory:

we can in principle distinguish monosemous word types from ambiguous word types by verifying if they have more than one definition. First, we observed that monosemous words, as identified by their number of dictionary definitions, yielded higher silhouette scores than ambiguous words ($p < 2 \cdot 2^{-16}$, Cohen's $d = 0.236$), though they still include a substantial number of tokens with negative silhouette scores.

As for ambiguous words, note that the number of distinct entries for a word type can serve as a proxy measure of how much meaning varies in use. We thus used a linear model to predict silhouette scores with log-scaled frequency and log-scaled definition counts, as listed in the Wiktionary, as predictors. We selected tokens for which we found at least one entry in the Wiktionary, out of which we then randomly sampled 10 000 observations. Both definition counts and frequency were found to be significant predictors, leading the silhouette score to decrease. This suggests that polysemy degrades the cohesion score of the type cluster, which is compatible with what one would expect from a DSM.

Similarity also includes related words, and not only tokens of the same type. Other studies (Vial et al., 2019; Coenen et al., 2019, e.g.) already stressed that BERT embeddings perform well on word-level semantic tasks. To directly assess whether BERT captures this broader notion of similarity, we used the MEN word similarity dataset (Bruni et al., 2014), which lists pairs of English words with human annotated similarity ratings. We removed pairs containing words for which we had no representation, leaving us with 2290 pairs. We then computed the Spearman correlation between similarity ratings and the cosine of the average BERT embeddings of the two paired word types, and found a correlation of 0.705, showing that cosine similarity of average BERT embeddings encodes

semantic similarity. For comparison, a word2vec DSM (Mikolov, K. Chen, et al., 2013) trained on BookCorpus (Zhu et al., 2015) using the same tokenization as BERT achieved a correlation of 0.669.

6.1.2 Cross-sentence coherence

As observed in the previous section, overall the word type coherence in BERT tends to match our basic expectations. In this section, we do further tests, leveraging our knowledge of the design of BERT. We look at the interaction of two features of the BERT model: on the one hand, *segment encodings* to distinguish between paired input sentences, on the other hand, *residual connections*.

Formal approach

We begin by examining the architectural design of BERT. We refer the reader to Section 1.2.2 as well as Vaswani et al. (2017) and Devlin et al. (2019) for a thorough overview of the architecture. In this section we focus on the interaction of two design choices in the BERT architecture: the use of residual connections throughout the model, as well as segment encodings used for the NSP objective.

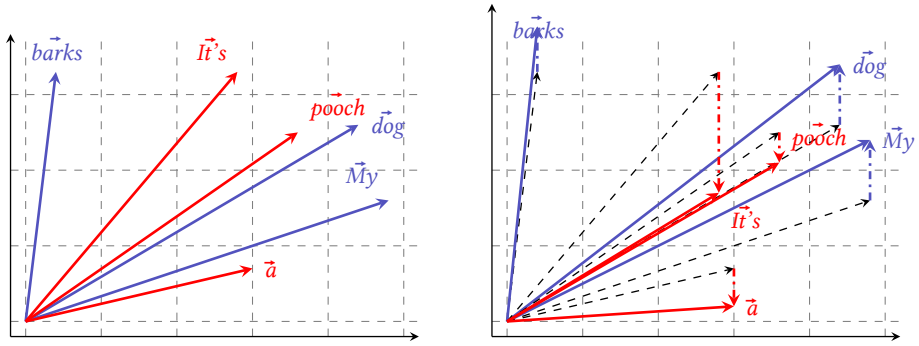
As brief reminder, on a formal level, BERT is a deep neural network composed of superposed layers of computations. Each layer is composed of two sub-layers: the first performing multi-head attention, the second being a simple feed-forward network. After each sub-layer, residual connections and layer normalization are applied; thus the intermediary output $\vec{y}_\lambda$ after sub-layer λ can be written as a function of the input $\vec{x}_\lambda$, as $\vec{y}_\lambda = LayerNorm(S_\lambda(\vec{x}_\lambda) + \vec{x}_\lambda)$, with S_λ the sublayer function.

Moreover, BERT includes positional encodings to explicitly model word order, as well as segment encodings to provide the model with some leverage point in its next sentence prediction (NSP) objective. Using the same example that we provided in Section 1.2.2, if the initial training example was "*My dog barks. It is a pooch.*," the actual input would correspond to the following sequence of vectors:

$$\begin{aligned}
[\vec{\texttt{CLS}}] + \vec{p}(0) + \vec{\texttt{seg}}_A,\ \vec{My} + \vec{p}(1) + \vec{\texttt{seg}}_A,\\
\vec{dog} + \vec{p}(2) + \vec{\texttt{seg}}_A,\ \vec{barks} + \vec{p}(3) + \vec{\texttt{seg}}_A,\\
\vec{.} + \vec{p}(4) + \vec{\texttt{seg}}_A,\ [\vec{\texttt{SEP}}] + \vec{p}(5) + \vec{\texttt{seg}}_A,\\
\vec{It} + \vec{p}(6) + \vec{\texttt{seg}}_B,\ \vec{is} + \vec{p}(7) + \vec{\texttt{seg}}_B,\\
\vec{a} + \vec{p}(8) + \vec{\texttt{seg}}_B,\ \vec{pooch} + \vec{p}(9) + \vec{\texttt{seg}}_B,\\
\vec{.} + \vec{p}(10) + \vec{\texttt{seg}}_B,\ [\vec{\texttt{SEP}}] + \vec{p}(11) + \vec{\texttt{seg}}_B
\end{aligned}$$

Due to the general use of residual connections, marking the sentences using the segment encodings $\vec{\texttt{seg}}_A$ and $\vec{\texttt{seg}}_B$ can introduce a systematic offset within sentences. Consider that the first layer uses as input vectors corresponding to word, position, and sentence information: $\vec{w}_i + \vec{p}(i) + \vec{\texttt{seg}}_i$; for simplicity, let $\vec{i}_t = \vec{w}_i + \vec{p}(i)$; we also ignore the rest of the input as it does not impact this reformulation. The output from the first sub-layer $\vec{y}_{1,i}$ can be written:

$$\begin{aligned}
\vec{y}_{1,i} &= LayerNorm(S_1(\vec{i}_i + \vec{\texttt{seg}}_i) + \vec{i}_i + \vec{\texttt{seg}}_i)\\
&= \vec{\beta}_1 + \gamma_1 \odot \frac{1}{\sigma_{1,i}} S_1(\vec{i}_i + \vec{\texttt{seg}}_i) + \gamma_1 \odot \frac{1}{\sigma_{1,i}} \vec{i}_i\\
&\quad - \gamma_1 \odot \frac{1}{\sigma_{i,1}} \mu(S_1(\vec{i}_i + \vec{\texttt{seg}}_i) + \vec{i}_i + \vec{\texttt{seg}}_i)\\
&\quad + \gamma_1 \odot \frac{1}{\sigma_{i,1}} \vec{\texttt{seg}}_i
\end{aligned}$$



(a) without segment encoding bias (b) with segment encoding bias

Figure 6.3: Toy example for segment encoding bias

$$= \vec{\tilde{y}}_{1,i} + \gamma_1 \odot \frac{1}{\sigma_{i,1}} \vec{\mathsf{seg}}_i \tag{6.4}$$

This equation is obtained by simply injecting the definition for layer-normalization (cf. Equation (1.17)).

Therefore, by recurrence, the final output $\vec{y}_{\Lambda,i}$ after all the Λ stacked Transformer sublayers for a given input $\vec{w}_i + \vec{p(i)} + \vec{\mathsf{seg}}_i$ can be written as:

$$\vec{y}_{\Lambda,i} = \vec{\tilde{y}}_{\Lambda,i} + \left(\bigodot_{\lambda=1}^{\Lambda} \gamma^{\lambda} \right) \odot \left(\prod_{\lambda=1}^{\Lambda} \frac{1}{\sigma_i^{\lambda}} \right) \times \vec{\mathsf{seg}}_i \tag{6.5}$$

This rewriting trick shows that segment encodings are partially preserved in the output. All embeddings within a sentence contain a shift in a specific direction, determined only by the initial segment encoding and the learned gain parameters for layer normalization. In Figure 6.3, we illustrate what this systematic shift might entail. Prior to the application of the segment encoding bias, the semantic space is structured by similarity ('*pooch*' is near '*dog*'); with the bias, we find a different set of characteristics: in our toy example, tokens are linearly

separable by sentences.

The presence of a trace imputable to segment encodings questions the nature of the vector space described by BERT. If BERT properly describes a purely semantic vector space, we should overall observe no significant difference in token encoding imputable to the segment the token belongs to.

Experimental protocol

This is a testable proposition: we can verify whether the BERT vector space is shaped by non-semantic factors. For a given word type w, we may constitute two groups: $w_{\texttt{seg}_A}$, the set of tokens for this type w belonging to first sentences in the inputs, and $w_{\texttt{seg}_B}$, the set of tokens of w belonging to second sentences. If BERT counterbalances the segment encodings, random differences should cancel out, and therefore the mean of all tokens $w_{\texttt{seg}_A}$ should be equivalent to the mean of all tokens $w_{\texttt{seg}_B}$.

We used the same dataset and BERT model as in Section 6.1.1. This setting (where all paired input sentences are drawn from running text) allows us to focus on the effects of the segment encodings. We retrieved the output embeddings of the last BERT layer and grouped them per word type.

To assess the consistency of a group of embeddings with respect to a purported reference, we used a mean of squared error (MSE): given a group of embeddings E and a reference vector $\vec{r}$, we computed how much each vector in E strayed from the reference $\vec{r}$. It is formally defined as:

$$\mathrm{MSE}(E, \vec{r}) = \frac{1}{\#E} \sum_{\vec{v} \in E} \sum_{d} (\vec{v}_d - \vec{r}_d)^2 \tag{6.6}$$

This MSE can also be understood as the average squared distance to the reference $\vec{r}$. When $\vec{r} = \mu_E$, i.e. $\vec{r}$ is set to be the average vector in E, the MSE measures variance of E via Euclidean distance. We then used the MSE function to construct pairs of observations: for each word type w, and for each segment encoding $\texttt{seg}_i$, we computed two scores. The first, MSE($w_{\texttt{seg}_i}, \mu_{\texttt{seg}_i}$) gives us an assessment of how coherent the set of embeddings $w_{\texttt{seg}_i}$ is with respect to the mean vector in that set, $\mu_{\texttt{seg}_j}$. The second, MSE($w_{\texttt{seg}_i}, \mu_{\texttt{seg}_j}$) assesses how coherent the same group of embeddings is with respect to $\mu_{\texttt{seg}_j}$, the mean vector for the embeddings of the same type, but from the other segment $\texttt{seg}_j$. If no significant contrast between these two scores can be observed, then BERT counterbalances the segment encodings and is coherent across sentences.

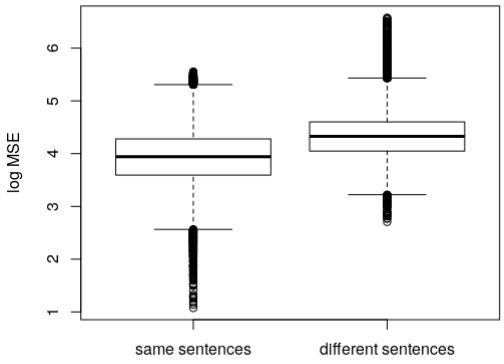


Figure 6.4: Log-scaled MSE per reference

Results & discussion

We compared results using a paired Student's t-test, which highlighted a significant difference based on which segment types belonged to (p-value $< 2 \cdot 2^{-16}$);

the effect size (Cohen's $d = -0.527$) was found to be stronger than what we computed when assessing whether tokens cluster according to their types (cf. Section 6.1.1). A visual representation of these results, log-scaled, is shown in Figure 6.4. For all sets $w_{\texttt{seg}_i}$, the average embedding from the set itself was systematically a better fit than the average embedding from the paired set $w_{\texttt{seg}_j}$. We also noted that a small number of items yielded a disproportionate difference in MSE scores and that frequent word types had smaller differences in MSE scores: roughly speaking, very frequent items—punctuation signs, stop-words, frequent word suffixes—received embeddings that are *almost* coherent across sentences.

Although the observed positional effect of embeddings' inconsistency might be entirely due to segment encodings, additional factors might be at play. In particular, BERT uses absolute positional encoding vectors to order words within a sequence: the first word w_1 is marked with the positional encoding $p(1)$, the second word w_2 with $p(2)$, and so on until the last word, w_n, marked with $p(n)$. As these positional encodings are added to the word embeddings, the same remark made earlier on the impact of residual connections may apply to these positional encodings as well. Lastly, we also note that many downstream applications use a single segment encoding per input, and thus sidestep the caveat stressed here.

6.1.3 Sentence-level structure

We have seen in Section 6.1.2 that BERT assigns to the same word type somewhat different representations for token occurrences in even and odd sentences. However, comparing tokens of the same type in consecutive sentences is not necessarily the main application of BERT and related models.

Does the segment-based representational variance affect the structure of the semantic space, instantiated in similarities between tokens of different types? Here we investigate how segment encodings impact the relation between any two tokens in a given sentence: our expectation for a semantic space is that segment encodings should not impact semantic similarity metrics.

Experimental protocol

Consistent with previous experiments, we used the same BERT model and dataset (cf. Section 6.1.1); in this experiment also mitigating the impact of the NSP objective was crucial. Sentences were thus passed two by two as input to the BERT model. As cosine has been traditionally used to quantify semantic similarity between words (Mikolov, Yih, et al., 2013; Levy and Goldberg, 2014a, e.g.), we then computed pairwise cosine of the tokens in each sentence. This allows us to reframe our assessment of whether lexical contrasts are coherent across sentences as a comparison of semantic dissimilarity across sentences. More formally, we compute the following set of cosine scores C_S for each sentence S:

$$C_S = \{\cos(\vec{v}, \vec{u}) \mid \vec{v} \neq \vec{u} \ \wedge \ \vec{v}, \vec{u} \in E_S\} \tag{6.7}$$

with E_S the set of embeddings for the sentence S. In this analysis, we compare the union of all sets of cosine scores for first sentences against the union of all sets of cosine scores for second sentences. To avoid asymmetry, we remove the `[CLS]` token (only present in first sentences), and as with previous experiments we neutralize the effects of the NSP objective by using only consecutive sentences as input.

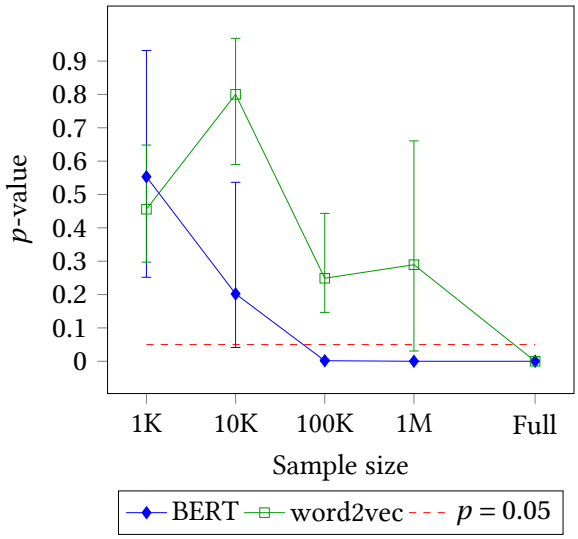


Figure 6.5: Mann–Whitney U tests, 1st vs. 2nd sentences

Results & discussion

We compared cosine scores for first and second sentences using a Mann–Whitney U test. We observed a significant effect, however small (Cohen's $d = 0.011$). This may perhaps be due to data idiosyncrasies, and indeed when comparing with a word2vec (Mikolov, K. Chen, et al., 2013) trained on BookCorpus (Zhu et al., 2015) using the same tokenization as BERT, we do observe a significant effect ($p < 0.05$). However the effect size is six times smaller ($d = 0.002$) than what we found for BERT representations; moreover, when varying the sample size (cf. Figure 6.5), p-values for BERT representations drop much faster to statistical significance.

A possible reason for the larger discrepancy observed in BERT representations might be that BERT uses absolute positional encodings, i.e. the k$^{\text{th}}$ word of the input is encoded with $p(k)$. Therefore, although all first sentences of a given length l will be indexed with the same set of positional encodings $\{p(1), \ldots, p(l)\}$,

only second sentences of a given length l preceded by first sentences of a given length j share the exact same set of positional encodings $\{p(j+1), \ldots, p(j+l)\}$. As highlighted previously, the residual connections ensure that the segment encodings were partially preserved in the output embedding: the same argument can be made for positional encodings. In any event, the fact is that we do observe on BERT representations an effect of segment on sentence-level structure. This effect is greater than one can blame on data idiosyncrasies, as verified by the comparison with a traditional DSM such as word2vec. If we are to consider BERT as a DSM, we must do so at the cost of cross-sentence coherence.

The analysis above suggests that embeddings for tokens drawn from first sentences live in a different semantic space than tokens drawn from second sentences, i.e. that BERT contains two DSMs rather than one. In the hypothetical case that BERT is a "two in one" DSM, the comparison between the two sentence representations that we can derive from a single input would be meaningless, or at least less coherent than the comparison of two sentence representations drawn from the same sentence position. To test this conjecture, we use two compositional semantics benchmarks: STS (Cer, Diab, et al., 2017) and SICK (Marelli et al., 2014). These datasets are structured as triplets, grouping a pair of sentences with a human-annotated relatedness score. The original presentation of BERT (Devlin et al., 2019) did include a downstream application to these datasets, but employed a learned classifier, which obfuscates results (Wieting and Kiela, 2019; Cover, 1965; Hewitt and Liang, 2019). Hence we simply reduce the sequence of tokens within each sentence into a single vector by summing them, a simplistic yet robust semantic composition method. We then compute the Spearman correlation between the cosines of the two sum vectors and the sentence pair's

Model	**STS cor.**	**SICK-R cor.**
Skip-Thought	0.25560	0.48762
USE	0.66686	0.68997
InferSent	0.67646	0.70903
BERT, *2 sent. ipt.*	0.35913	0.36992
BERT, *1 sent. ipt.*	0.48241	0.58695
word2vec	0.37017	0.53356

Table 6.1: Correlation (Spearman ρ) of cosine similarity and relatedness ratings on the STS and SICK-R benchmarks

relatedness score. We compare two setups: a "two sentences input" scheme (or *2 sent. ipt.* for short)—where we use the sequences of vectors obtained by passing the two sentences as a single input—and a "one sentence input" scheme (*1 sent. ipt.*)—using two distinct inputs of a single sentence each.

Results are reported in Table 6.1; we also provide comparisons with three different sentence encoders and the aforementioned word2vec model. As we had suspected, using sum vectors drawn from a two sentence input scheme degrades performances below the word2vec baseline. On the other hand, a one sentence input scheme seems to produce coherent sentence representations: in that scenario, BERT performs better than word2vec and the older sentence encoder Skip-Thought (Kiros et al., 2015), but worse than the modern USE (Cer, Y. Yang, et al., 2018) and Infersent (Conneau et al., 2017). The comparison with word2vec also shows that BERT representations over a coherent input are more likely to include some form of compositional knowledge than traditional DSMs; however it is difficult to decide whether some true form of compositionality is achieved by BERT or whether these performances are entirely a by-product of the positional encodings. In favor of the former, other research has suggested that Transformer-based

architectures perform syntactic operations (Raganato and Tiedemann, 2018; Hewitt and Manning, 2019; Clark, Khandelwal, et al., 2019; Jawahar et al., 2019; Voita et al., 2019; Michel et al., 2019). In all, these results suggest that the semantic space of token representations from second sentences differ from that of embeddings from first sentences.

6.2 The Linear Structure of Transformers

Our experiments in Sections 6.1.2 and 6.1.3 have demonstrated that the vector space of BERT is shaped, among other things, by the segment encodings used to solve its NSP objective. Crucially, the reasoning that we have applied to first formulate our intuition in Section 6.1.2 is based on features of the Transformer architecture—namely, in that BERT, as a Transformer, contains residual connections and LayerNorms at each sublayer, thus preserving inputs and intermediary representations in its outputs. We now focus on whether the intuition we had for segment encodings can be generalized to more broadly characterize Transformer embedding.

6.2.1 Mathematical re-framing

If we follow the logic we developed in Section 6.1.2 of tracking how inputs propagate throughout the network, we can show that the Transformer embedding e_t for a token t is as a sum of four *terms*:

$$\vec{e}_t = \vec{I}_t + \vec{F}_t + \vec{H}_t + \vec{C}_t \tag{6.8}$$

where $\vec{I}_t$ is proportional to the input $\vec{\text{ipt}}_{0,t}$, $\vec{F}_t$ and $\vec{H}_t$ are the cumulative contributions of the feed-forward sub-modules and the attention heads respectively, and $\vec{C}_t$ is a bias term applying to all positions t. We will review how to derive these terms from the definition of a Transformer architecture (cf. Section 1.2.2) in Section 6.2.2, and focus for now on characterizing each of the four terms in more details.

$\vec{I}_t$ corresponds to the input embedding (i.e., the positional encoding, the input word-type embedding, and the segment encoding in BERT-like models), after having gone through all the LayerNorm gains γ_λ for each of the Λ stacked sub-layers, and re-scaling by the inverse of the standard deviations σ_λ:

$$\vec{I}_t = \frac{\bigodot_{\lambda}^{\Lambda} \gamma_\lambda}{\prod_{\lambda}^{\Lambda} \sigma_\lambda} \odot \vec{ipt}_{0,t} \tag{6.9}$$

This first term is very reminiscent of the segment encoding bias we noted earlier in Equation (6.5): the main difference is that we consider all the linear biases inherited from the input vectors rather than specifically focus on the segment encoding bias.

The term $\vec{F}_t$ is the sum of the outputs of the feed-forward sub-modules for all L layers. sub-modules in lower layers have to pass through the LayerNorm of all the layers above, which gives:

$$\vec{F}_t = \sum_{l}^{L} \frac{\bigodot_{\lambda=2l}^{\Lambda} \gamma_\lambda}{\prod_{\lambda=2l}^{\Lambda} \sigma_\lambda} \odot \vec{\mathscr{F}}_{l,t} \tag{6.10}$$

where $\vec{\mathscr{F}}_{l,t} = W_O^{\mathscr{F}_l} \cdot \left(\Phi\left(W_I^{\mathscr{F}_l} \cdot \vec{ipt}_{\mathscr{F}_l,t} + b_I^{\mathscr{F}_l}\right)\right)$ is the unbiased output at the position t of the feed-forward sub-module for this layer l.

$\vec{H}_t$ corresponds to the sum across layers of each multi-head attention sub-module, having passed through the relevant LayerNorms. As multi-head attentions are entirely linear, we can further describe each output as a sum over all h heads of a weighted bag-of-words of the input representations to that sub-module. Or formally:

$$\vec{H}_t = \sum_{l}^{L} \left(\frac{\bigodot_{\lambda=2l-1}^{\Lambda} \gamma_\lambda}{\prod_{\lambda=2l-1}^{\Lambda} \sigma_\lambda} \odot \sum_{h_l} \sum_{t'} a_{t'}^{h_l} Z_{h_l}(\vec{ipt}_{l,t'}) \right) \tag{6.11}$$

where $a_{t'}^{h_l}$ corresponds to the attention weights assigned to position t' by the attention head h_l. The transformation $Z_{h_l} = W_O^{H_l} \cdot M^h \cdot W_V^{h,l}$ corresponds to passing an input embedding through the unbiased values projection $W_V^{h,l}$ of the head h, then projecting it from a d_v-dimensional subspace onto a $d_v \times h$-dimensional space using a zero-padded identity matrix:

$$M^h = \begin{bmatrix} 0_{d_v,(h-1)\times d_v} & I_{d_v} & 0_{d_v,(H-h)\times d_v} \end{bmatrix}$$

and finally passing it through the unbiased outer projection $W_O^{H_l}$ of the relevant MHA sub-module.

In the last term $\vec{C}_t$, we collect all the biases:

$$\vec{C}_t = \sum_{\lambda}^{\Lambda} \left(\frac{\bigodot_{\lambda'=\lambda+1}^{\Lambda} \gamma_{\lambda'}}{\prod_{\lambda'=\lambda+1}^{\Lambda} \sigma_{\lambda'}} \odot \beta_\lambda - \frac{\bigodot_{\lambda'=\lambda}^{\Lambda} \gamma_{\lambda'}}{\prod_{\lambda'=\lambda}^{\Lambda} \sigma_{\lambda'}} \odot \vec{\mu}_\lambda \right)$$

$$+\sum_{l}^{L}\frac{\bigodot_{\lambda=2l-1}^{\Lambda}\gamma_\lambda}{\prod_{\lambda=2l-1}^{\Lambda}\sigma_\lambda}\odot\left(b_O^{H_l}+W_O^{H_l}\cdot\bigoplus_{h_l}b_V^{h,l}\right)$$
$$+\sum_{l}^{L}\frac{\bigodot_{\lambda=2l}^{\Lambda}\gamma_\lambda}{\prod_{\lambda=2l}^{\Lambda}\sigma_\lambda}\odot\left(b_O^{\mathscr{F}_l}\right) \tag{6.12}$$

This term $\vec{C}_t$ includes the biases β_λ and mean-shifts $\vec{\mu}_\lambda$ of the LayerNorms, the output biases of the feed-forward sub-modules $b_l^{\mathscr{F}}$, the outer projection bias in each multi-head attention sub-module $b_{W_l^O}$, as well as those from the value projections in each head, after having passed through the outer projection $W_l^O \cdot \bigoplus_{h_l} b_{h,l}^{W_V}$.[1]

This derivation can be extended to account for multiple Transformer variants. For instance, in the case of relative positional embeddings applied to value projections (P. Shaw et al., 2018), it is rather straightforward to follow the same logic so as to include relative positional offset in the most appropriate term.

Remark that Equation (6.8) does not entail that the terms are independent from one another. For instance, the scaling factor $1/\prod\sigma_\lambda$ systematically depends on the magnitude of earlier hidden representations. Neither do we analyze feed-forward outputs, which are derived from previous representations—in fact, the non-linear function Φ they contain prevents us from using basic linear algebra to study them. Another noteworthy case concerns the $\vec{H}$ term, where we rely on the input of the previous layer. While we could decompose the input according to individual modules, in practice we did not find a computationally tractable

[1] In general, concatenation is equivalent to a sum of zero-padded identity matrices, hence this last term is equivalent to $\sum_h M^h b_{h,l}^{W_V}$, with M^h as we described for Equation (6.11).

way of doing so. We leave this question for future study. In all, Equation (6.8) only stresses that a Transformer embedding can be decomposed as a sum of the outputs of its sub-modules: it does not fully disentangle computations done in separate sub-modules.

Nonetheless, we can draw connections between the derivation we provide in Equation (6.8) and a wide range of previous studies setting out to explain how Transformers behave (Rogers, Kovaleva, et al., 2020). For instance, the $\vec{I}_t$ term also provides a principled way of comparing static and contextual embeddings (Lenci et al., 2021). Likewise, works that employ probes to unearth a linear structure corresponding to the syntactic structure of the input sentence (Raganato and Tiedemann, 2018; Hewitt and Manning, 2019, a.o.) can be construed as relying on the explicit linear dependence across tokens through the $\vec{H}_t$ term. Also relevant is the study on sparsifying Transformer representations of Yun et al. (2021): the linearly dependent nature of Transformer embeddings has some implications when it comes to dictionary coding. Likewise, our approach can provide some quantitative argument for the validity of attention-based studies (Serrano and Smith, 2019; Jain and Wallace, 2019; Wiegreffe and Pinter, 2019; Pruthi et al., 2020), and naturally expands on earlier work attempt to go beyond attention weight overviews (Kobayashi et al., 2020).

6.2.2 Step-by-step derivation of Equation (6.8)

Given that a Transformer layer consists of a stack of L layers, with each layer being comprised of two sublayers, we can treat a Transformer as a stack of $\Lambda = 2L$ sublayers. For simplicity of notation, we link the sublayer index λ to the layer

index l, by noting that the first sublayer of layer l is the $2l-1$-th sublayer, and the second is the $2l$th sublayer.[2]

Consider that each sublayer includes a residual connection before the final LayerNorm:

$$\vec{y} = \gamma_\lambda \odot \frac{(S(\vec{x}) + \vec{x}) - \vec{\mu}_\lambda}{\sigma_\lambda} + \beta_\lambda$$

For simplicity, let:

$$T_\lambda(\vec{v}) = \gamma_\lambda \odot \left(\frac{1}{\sigma_\lambda} \times \vec{v}\right)$$

Note that this normalization transformation T is distributive over vector addition. We can rewrite a sublayer as:

$$\begin{aligned}\vec{y} &= T_\lambda\left(S_\lambda(\vec{x}) + \vec{x} - \vec{\mu}_\lambda\right) + \beta_\lambda \\ &= T_\lambda(S_\lambda(\vec{x})) + T_\lambda(\vec{x}) - T_\lambda\left(\vec{\mu}_\lambda\right) + \beta_\lambda\end{aligned}$$

We can then consider what happens to this additive structure in the next sublayer. Likewise, let $T_{\lambda+1}(\vec{v}) = \gamma_{\lambda+1} \odot \left(\frac{1}{\sigma_{\lambda+1}} \times \vec{v}\right)$ be the normalization transformation associated to the next sublayer. If we consider the effects of combining multiple normalizations, we remark that:

$$\begin{aligned}T_\lambda \circ T_{\lambda+1}(\vec{v}) &= \gamma_{\lambda+1} \odot \gamma_\lambda \odot \left(\frac{1}{\sigma_\lambda} \times \frac{1}{\sigma_{\lambda+1}} \times \vec{v}\right) \\ &= \frac{\bigodot_{\lambda'=\lambda}^{\lambda+1} \gamma_{\lambda'}}{\prod_{\lambda'=\lambda}^{\lambda+1} \sigma_{\lambda'}} \odot (\vec{v})\end{aligned}$$

[2] In the case of BERT, we also need to include a LayerNorm before the first layer, which is straightforward if we index it as $\lambda = 0$.

The latter expression generalizes to any sequence of normalization transformations $T_{\lambda\ldots\lambda+n} = T_\lambda \circ \cdots \circ T_{\lambda+n}$. Assume for simplicity that the normalization transformation that covers no sublayer $T_{\lambda+1\ldots\lambda}$ is a simple identity function, i.e., we take that $T_{\lambda+2\ldots\lambda+1}(w) = w$.

Let us now consider passing the input $\vec{x}$ through a complete layer, i.e., through sublayers λ and $\lambda+1$:

$$\begin{aligned}\vec{y} = & T_{\lambda+1}\left(S_{\lambda+1}\left[T_\lambda\left(S_\lambda\left(\vec{x}\right)\right) + T_\lambda\left(\vec{x}\right) - T_\lambda\left(\vec{\mu}_\lambda\right) + \beta_\lambda\right]\right) \\ & + T_{\lambda+1}\left(T_\lambda\left(S_\lambda\left(\vec{x}\right)\right) + T_\lambda\left(\vec{x}\right) - T_\lambda\left(\vec{\mu}_\lambda\right) + \beta_\lambda\right) \\ & - T_{\lambda+1}(\vec{\mu}_{\lambda+1}) + \beta_{\lambda+1}\end{aligned}$$

As we are interested in the combined effects of a layer, S_λ is a multi-head attention mechanism and $S_{\lambda+1}$ a feed-forward.

For ease of consultation, we include here our earlier definition of the feed-forward outputs as:

$$\begin{aligned}\vec{y} &= W_O^{\mathscr{F}} \cdot \left(\Phi\left(W_I^{\mathscr{F}} \cdot \vec{x} + b_I^{\mathscr{F}}\right)\right) + b_O^{\mathscr{F}} \\ &= \vec{\mathscr{F}}_t + b_O^{\mathscr{F}}\end{aligned}$$

By substituting the actual sublayer functions in our previous equation:

$$\begin{aligned}\vec{y} = & T_{\lambda+1}\left(\vec{\mathscr{F}}_l + b_O^{\mathscr{F}_l}\right) \\ & + T_{\lambda+1}\left(T_\lambda\left(\vec{\mathscr{H}}_l + b_{\mathscr{H}_l}\right) + T_\lambda\left(\vec{x}\right) - T_\lambda\left(\vec{\mu}_\lambda\right) + \beta_\lambda\right) \\ & - T_{\lambda+1}(\vec{\mu}_{\lambda+1}) + \beta_{\lambda+1}\end{aligned}$$

This can be rewritten to match more closely Equation (6.8) by delineating input, multi-head attention, feed-forward and normalization terms:

$$
\begin{aligned}
\vec{y} &= \vec{i} + \vec{h} + \vec{f} + \vec{c} \\
\vec{i} &= T_{\lambda \ldots \lambda+1}(\vec{x}) \\
\vec{h} &= T_{\lambda \ldots \lambda+1}(\vec{\mathscr{H}}_l) \\
\vec{f} &= T_{\lambda+1}\left(\vec{\mathscr{F}}_l\right) \\
\vec{c} &= T_{\lambda \ldots \lambda+1}(b_{\mathscr{H}_l}) + T_{\lambda+1}\left(b_O^{\mathscr{F}_l}\right) \\
&\quad + \sum_{\lambda'=\lambda}^{\lambda+1} \left(T_{\lambda'+1 \ldots \lambda+1}(\beta_{\lambda'}) - T_{\lambda' \ldots \lambda+1}(\vec{\mu}_{\lambda'})\right)
\end{aligned}
$$

To express the output of an entire Transformer model, we then need to pass the input across multiple layers. We substitute $\vec{x}$ in our previous equation with the output of the previous layer:

$$
\begin{aligned}
\vec{y}_{l+1} &= \vec{i}_{l+1} + \vec{h}_{l+1} + \vec{f}_{l+1} + \vec{c}_{l+1} \\
\vec{i}_{l+1} &= T_{\lambda+2 \ldots \lambda+3}\left(\vec{i}_l + \vec{h}_l + \vec{f}_l + \vec{c}_l\right) \\
\vec{h}_{l+1} &= T_{\lambda+2 \ldots \lambda+3}\left(\vec{\mathscr{H}}_{l+1}\right) \\
\vec{f}_{l+1} &= T_{\lambda+3}\left(\vec{\mathscr{F}}_{l+1}\right) \\
\vec{c}_{l+1} &= T_{\lambda+2 \ldots \lambda+3}(b_{\mathscr{H}_{l+1}}) + T_{\lambda+3}\left(b_O^{\mathscr{F}_{l+1}}\right) \\
&\quad + \sum_{\lambda'=\lambda+2}^{\lambda+3} \left(T_{\lambda'+1 \ldots \lambda+3}(\beta_{\lambda'}) - T_{\lambda' \ldots \lambda+3}(\vec{\mu}_{\lambda'})\right)
\end{aligned}
$$

Which we can rearrange to match Equation (6.8):

$$
\vec{y}_{l+1} = \vec{I} + \vec{H} + \vec{F} + \vec{C}
$$

$$
\begin{aligned}
\vec{I} &= T_{\lambda \dots \lambda+3}(\vec{x}) \\
\vec{H} &= T_{\lambda+2 \dots \lambda+3}\left(\vec{\mathscr{H}}_{l+1}\right) + T_{\lambda \dots \lambda+3}\left(\vec{\mathscr{H}}_{l}\right) \\
\vec{F} &= T_{\lambda+3}\left(\vec{\mathscr{F}}_{l+1}\right) + T_{\lambda+1 \dots \lambda+3}\left(\vec{\mathscr{F}}_{l}\right) \\
\vec{C} &= T_{\lambda+2 \dots \lambda+3}(b_{\mathscr{H}_{l+1}}) + T_{\lambda \dots \lambda+3}(b_{\mathscr{H}_{l}}) \\
&\quad + T_{\lambda+3}\left(b_O^{\mathscr{F}_{l+1}}\right) + T_{\lambda+1 \dots \lambda+3}\left(b_O^{\mathscr{F}_{l}}\right) \\
&\quad + \sum_{\lambda'=\lambda}^{\lambda+3} \left(T_{\lambda'+1 \dots \lambda+3}(\beta_{\lambda'}) - T_{\lambda' \dots \lambda+3}(\vec{\mu}_{\lambda'})\right)
\end{aligned}
$$

As such, we can simplify the $\vec{H}$, $\vec{F}$ and $\vec{C}$ terms as:

$$
\begin{aligned}
\vec{H} &= \sum_{l'=l}^{l+1} T_{2l'-1 \dots 2(l+1)}\left(\vec{\mathscr{H}}_{l}\right) \\
\vec{F} &= \sum_{l'=l}^{l+1} T_{2l' \dots 2(l+1)}\left(\vec{\mathscr{F}}_{l}\right) \\
\vec{C} &= \sum_{l'=l}^{l+1} T_{2l'-1 \dots 2(l+1)}\left(b_{\mathscr{H}_{l}}\right) + \sum_{l'=l}^{l+1} T_{2l' \dots 2(l+1)}\left(b_O^{\mathscr{F}_{l}}\right) \\
&\quad + \sum_{\lambda'=\lambda}^{\lambda+3} \left(T_{\lambda'+1 \dots \lambda+3}(\beta_{\lambda'}) - T_{\lambda' \dots \lambda+3}(\vec{\mu}_{\lambda'})\right)
\end{aligned}
$$

This logic carries on across layers: adding one new layer corresponds to (i) mapping the existing term through the two new sublayers normalization transformations, (ii) adding a new term for the multi-head attention, (iii) adding a new term for the feed-forward, (iv) tallying up the biases and mean centering introduced in the current layer. Hence, by recurrence over all layers and providing the initial input $\vec{ipt}_{0,t}$, we obtain Equations (6.8) to (6.12) exactly.

6.3 Intrinsic analyses

We now focus on a quantitative description of the decomposition we described in Equation (6.8). We begin with an overview of the interplay between the four terms in Section 6.3.1 and then turn to quantifying the degree of non-linearity present in the BERT model Section 6.3.2.

6.3.1 Visualizing the contents of embeddings

We have shown that Transformer embeddings have a highly linear structure. Given that Equations (6.9) to (6.12) are all defined as sums across layers or sub-layers, it is straightforward to adapt them to derive the decomposition at each intermediate representation. One question we might be interested in asking is that of the relative importance of the four terms $\vec{I}_t$, $\vec{F}_t$, $\vec{H}_t$ and $\vec{C}_t$.

Experimental protocol

Typically, we are looking for some metric able to compare one of the terms $\vec{T}_t$ to the total $\vec{e}_t$. Ideally, we would want this metric to capture whether $\vec{T}_t$ and $\vec{e}_t$ have roughly the same orientation, as well as whether $\vec{T}_t$ is a major component of $\vec{e}_t$. In other words, we need a metric sensitive to co-directionality and relative magnitude.

A normalized dot-product of the form:

$$m(\vec{e}_t, \vec{T}_t) = \frac{\langle \vec{e}_t \cdot \vec{T}_t \rangle}{\|\vec{e}_t\|_2^2} \tag{6.13}$$

satisfies both of these requirements. Moreover, given that dot products dis-

tributes over addition (i.e., $\langle a \cdot \sum_i b_i \rangle = \sum_i \langle a \cdot b_i \rangle$), and that the dot-product of a vector with itself is its magnitude squared (i.e., $\langle a \cdot a \rangle = \|a\|_2^2$), we get that summing over all four terms equals one:

$$m(\vec{e}_t, \vec{I}_t) + m(\vec{e}_t, \vec{F}_t) + m(\vec{e}_t, \vec{H}_t) + m(\vec{e}_t, \vec{C}_t) = 1$$

Hence this metric intuitively measures the importance of a term relative to the total sum.

Following what we observed in Sections 6.1.2 and 6.1.3, we provide only a single sentence per input to avoid any obvious confounding factors—both in this experiment and any subsequent one. We contrast embeddings from three related models: the BERT base uncased model of Devlin et al. (2019) and fine-tuned variants on CONLL 2003 NER (Tjong Kim Sang and De Meulder, 2003)[3] and SQuAD v2 (Rajpurkar et al., 2018).[4] Our reason to depart from the model used in previous experiments lies in that the code used to compute the derivation in Equation (6.8) has yet to be optimized, and using a smaller model has the practical benefit of speeding up our computations significantly. Unlike the previous experiments, we randomly sample 10 000 sentences from the EuroParl English section (Koehn, 2005) which corresponds to almost 900 000 word-piece tokens. This different dataset was selected to ensure that all three models of interest would be tested on comparable conditions—viz., on an out-of-domain setting.

[3] `https://huggingface.co/dslim/bert-base-NER-uncased`
[4] `https://huggingface.co/twmkn9/bert-base-uncased-squad2`

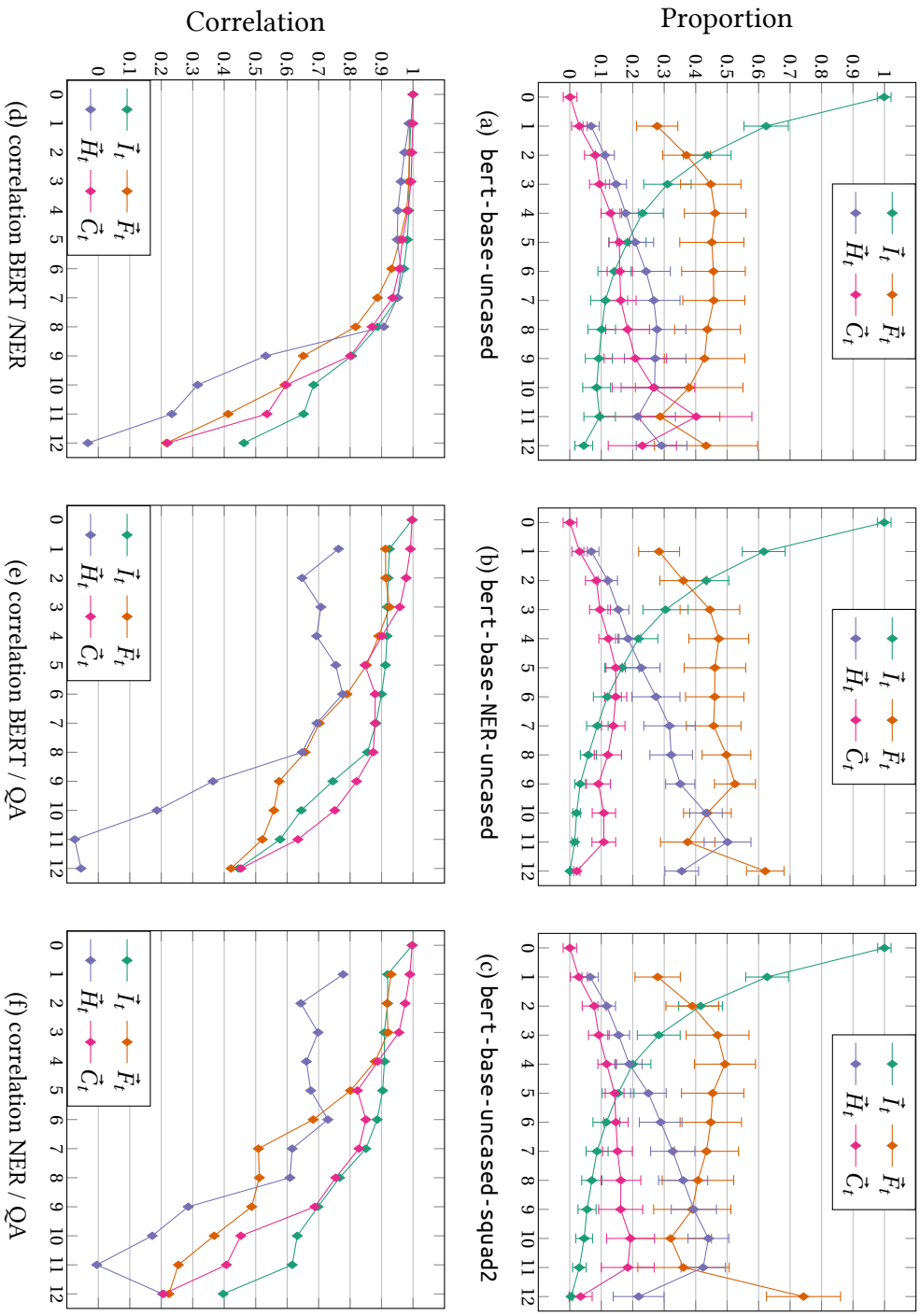


Figure 6.6: Relative importance of main terms

Results

Figure 6.6 summarizes the relative importance of the four terms of Equation (6.8), as measured by the normalized dot-product defined in Equation (6.13); ticks on the x-axis correspond to different layers. Figures 6.6a to 6.6c display the evolution of our proportion metric across layers for all three BERT models, whereas Figures 6.6d to 6.6f display how our normalized dot-product measurements correlate across pairs of models, using Spearman's ρ. [5]

Looking at Figure 6.6a, we can make a few important observations. The input term $\vec{I}_t$, which corresponds to a static embedding, initially dominates the full output, but quickly decreases in prominence, until it reaches 0.045 at the last layer. This should explain why lower layers of Transformers generally give better performances on static word-type tasks (Jawahar et al., 2019; Vulić et al., 2020, a.o.).

Interestingly, the constant term $\vec{C}_t$ is far from negligible: at layer 11, it is actually the most prominent term of the four and defines 23% of the output embedding. Note that $\vec{C}_t$ defines a set of offsets embedded in a 2Λ-dimensional hyperplane. In fact, we can re-write Equation (6.12) to highlight that is comprised only of scalar multiplications applied to constant vectors. Let:

$$b^S_\lambda = \begin{cases} \left(b^{H_l}_O + W^{H_l}_O \cdot \bigoplus_{h_l} b^{h,l}_V \right) & \text{if } \lambda = 2l-1 \\ b^{\mathscr{F}_l}_O & \text{if } \lambda = 2l \end{cases}$$

$$\vec{m}_\lambda = \bigodot_{\lambda'=\lambda+1}^{\Lambda} \gamma_{\lambda'} \odot (\beta_\lambda + b^S_\lambda)$$

[5] Layer 0 is the layer normalization conducted before the first sublayer, hence $\vec{F}_t$ and $\vec{H}_t$ are undefined here.

$$\vec{n}_\lambda = \bigodot_{\lambda'=\lambda+1}^{\Lambda} \gamma_{\lambda'} \odot \vec{1}$$

If there is a LayerNorm before the first layer (e.g. BERT), let $b_0^S = \vec{0}$. Then Equation (6.12) is equivalent to:

$$\vec{C}_t = \sum_{\lambda}^{\Lambda} \left(\frac{1}{\prod_{\lambda'=\lambda+1}^{\Lambda} \sigma_{\lambda'}} \cdot \vec{m}_\lambda \right) + \sum_{\lambda}^{\Lambda} \left(\frac{-\mu_\lambda}{\prod_{\lambda'=\lambda+1}^{\Lambda} \sigma_{\lambda'}} \cdot \vec{n}_\lambda \right)$$

Note that the vectors $\vec{m}_\lambda$ and $\vec{n}_\lambda$ are all constant for any input and position. Assuming they are all independent puts an upper bound of 2Λ independent vectors necessary to express any $\vec{C}_t$ vector. Moving back to our concrete example of BERT base, 23% of the output can be expressed using a $2\Lambda = 50$ dimensional vector,[6] or 6.5% of the 768 explicit dimensions of the model. As such, it is very likely that this term induces part of the anisotropy often attributed to Transformer embeddings (Ethayarajh, 2019; Timkey and Schijndel, 2021, e.g.).

The $\vec{H}_t$ term, which corresponds to the multi-head attention sub-modules, is not as prominent as one could expect from the vast literature that focuses on it. Its normalized dot-product is barely above what we observe for the constant term $\vec{C}_t$, and never averages above 0.3 across any layer. This can be partly pinned down on the prominence of the feed-forward term $\vec{F}_t$, which yields a normalized dot-product of 0.4 or above across most layers. Given that the feed-forward sub-modules are always the last component added to each hidden state, we can see that the sub-terms of $\vec{F}_t$ went through fewer LayerNorms, as compared to the

[6]Recall that BERT base contains 12 layers containing 2 sublayers each plus an initial LayerNorm applied before the first layer, for a totoal of $\Lambda = 25$ LayerNorms to consider.

sub-terms of $\vec{H}_t$. As a consequence, the sub-terms of $\vec{F}_t$ also underwent fewer scalar multiplications which likely affects their magnitude.

If we now turn to the finetuned models (Figures 6.6b and 6.6c), we find that they impart a much lower proportion of the contextual embeddings to the $\vec{I}_t$ and $\vec{C}_t$ terms. While the $\vec{F}_t$ term seems to dominate in the final embedding, looking at the correlations in Figures 6.6d and 6.6e suggest that the $\vec{H}_t$ terms are those that undergo the most modifications. Proportions assigned to the terms correlate with those assigned in the non-finetuned model more in the case of lower layers than higher layers (Figures 6.6d and 6.6e). The required adaptations seem task-specific as the two fine-tuned models do not correlate highly with each other (Figure 6.6f). Lastly, updates in the NER model impact mostly layer 8 and upwards (Figure 6.6d), whereas the QA model (Figure 6.6e) sees important modifications to the $\vec{H}_t$ term at the first layer, suggesting that SQuAD requires more drastic adaptations than CONLL 2003.

6.3.2 Quantifying non-linearity

Figure 6.6a show the quantitative importance of $\vec{F}_t$ to the final embedding. The intuitive reasoning for adding the feed-forward terms is that the model would otherwise devolve into a sum of bag-of-words and static embeddings. While both approaches have had their successes in the NLP literature (Mikolov, Yih, et al., 2013; Mitchell and Lapata, 2010), adding a non-linearity would in principle make the model more expressive. Non-linearity moreover enables the model to capture interactions between static input and context, which we have thus far quantified indirectly.

As the non-linear functions used in Transformers are generally either ReLU or GELU, which both behave almost linearly for a high enough input value, it is in principle possible that the feed-forward sub-modules can be approximated by a purely linear transformation, depending on the exact set of parameters they converged onto.

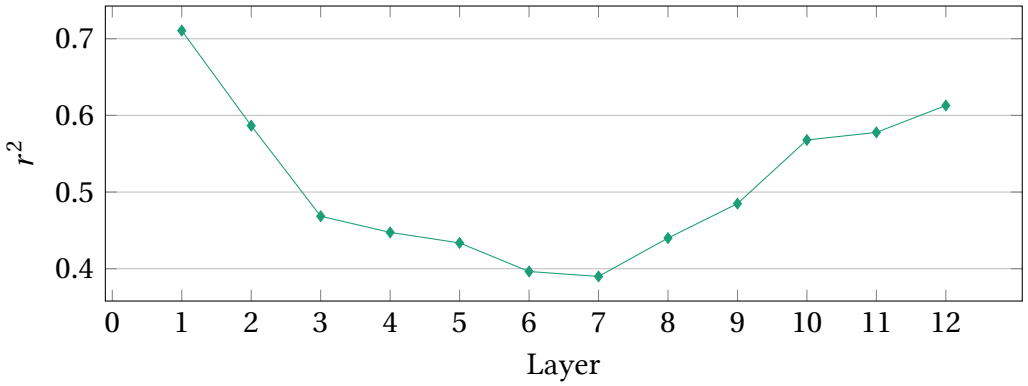


Figure 6.7: Fitting the $\vec{F}t$ term: r^2 across layers

To assess this possibility, we learn a simple least-square linear regression mapping the z-scaled inputs of every feed-forward sub-module to its respective z-scaled output. We use the `bert-base-uncased` model and the random sample from EuroParl from Section 6.3.1, and fit the regressions using all the 900 000 embeddings at our disposal. Figure 6.7 displays the quality of these linear approximations, as measured by a r^2 score. While we do see some variation across layers, we do not observe a perfect fit, suggesting the non-linearity is actively exploited by BERT.

6.4 Extrinsic analyses

We have provided quantitative description of the importance of the four terms delineated in Section 6.2. This previous analysis of our suggested decomposition was essentially intrinsic in nature, in that it only considered the embeddings themselves. It is however useful to look into what this decomposition of four terms may entail for downstream applications, which is why we now pivot to an extrinsic analysis.

6.4.1 The MLM objective

In Section 6.3.2, we saw that the $\vec{F}_t$ term could not be simply explained as a linear combinations of the correspond sub-module inputs. An interesting follow-up question concerns how that non-linearity is exploited by the model: does this non-linear $\vec{F}_t$ make the model more effective? More generally, it makes sense to see how the four terms allows us to retrieve the target word-piece.

Experimental protocol

We consider two approaches: either using the actual projection learned by the non-finetuned BERT model, or by learning a simple categorical regression for a specific term. To test the model, we use our EuroParl sample, and select 15% of the word-pieces at random; as in the original work of Devlin et al. (2019), 80% of the sampled word-pieces are replaced by a mask token, 10% are replaced by a random word-piece, and 10% are left as is; we split embeddings in three groups (80% for train, 10% for validation, 10% for test). When learning categorical regressions

from scratch, we use an AdamW optimizer (Loshchilov and Hutter, 2019) and iterate 20 times over the train set; hyperparameters (learning rate, weight decay, dropout, and the β_1 and β_2 AdamW hyperparameters) are set using Bayesian Optimization, with 50 hyperparameter samples and accuracy as objective.

Results

Results are displayed in Table 6.2. In the first row ("def.") we derive predictions using the default output projection from the original matrix, whereas the second ("lrn.") corresponds to our learned projections. Columns display the results of using the sum of 1, 2, 3 or 4 of the terms $\vec{I}_t$, $\vec{H}_t$, $\vec{F}_t$ and $\vec{C}_t$ to derive vector representations. On the one hand, the default projection benefits from a more extensive training: when using all four terms, the default projection is almost 2% more accurate than learning one from scratch. On the other hand learning a regression allows us to consider more specifically what can be retrieved from individual terms, as is apparent when comparing the behavior of the $\vec{F}_t$ in the two setups: when using the default output projection, we get 1.36% accuracy, whereas learning a specific categorical regression yields 53.92%.

The default projection matrix is also highly dependent on the normalization offsets $\vec{C}_t$ and the feed-forward terms $\vec{F}_t$ being added together: removing this $\vec{C}_t$ term from any experiment using $\vec{F}_t$ is highly detrimental to the accuracy. On the other hand, combining the two produces the highest accuracy scores. Our logistic regressions show that most of this performance can be imputed to the $\vec{F}_t$ term. Learning a projection from the $\vec{F}_t$ term already yields an accuracy of almost 54%. On the other hand, a regression learned from $\vec{C}_t$ only has a limited accuracy has a limited performance of 9.24%. Interestingly, this is still above what one would

	$\vec{I}_l$	$\vec{H}_l$	$\vec{F}_l$	$\vec{C}_l$	$\vec{I}_l + \vec{H}_l$	$\vec{I}_l + \vec{F}_l$	$\vec{I}_l + \vec{C}_l$	$\vec{H}_l + \vec{F}_l$	$\vec{H}_l + \vec{C}_l$	$\vec{F}_l + \vec{C}_l$	$\vec{I}_l + \vec{H}_l + \vec{F}_l$	$\vec{I}_l + \vec{H}_l + \vec{C}_l$	$\vec{I}_l + \vec{F}_l + \vec{C}_l$	$\vec{H}_l + \vec{F}_l + \vec{C}_l$	$\vec{I}_l + \vec{H}_l + \vec{F}_l + \vec{C}_l$
def.	3.08	39.64	1.36	4.44	40.55	2.99	4.40	22.20	35.69	48.78	32.45	35.23	49.41	57.17	57.33
lrn.	14.51	48.33	53.92	9.24	49.72	53.66	13.18	55.45	48.10	53.97	55.58	49.08	54.05	55.79	55.53

Table 6.2: Masked language model accuracy, in %

observe if the model always predicted the most frequent word-piece (viz. `the`, corresponding to 6% of the test targets), suggesting that even these very semantically bare items can be exploited by a categorical regression. As $\vec{C}_t$ variations are tied to the z-scaling performed in the LayerNorms, this would suggest that the magnitude of Transformer representations are not entirely meaningless, we leave a more in-depth investigation to future research.

In all, do feed-forward modules make the model more effective? The $\vec{F}_t$ term is necessary to achieve the highest accuracy on the training objective of BERT. On its own, it doesn't achieve the highest performances: for that we also need to add the multi-head attention outputs $\vec{H}_t$. However, the performances we can associate to $\vec{F}_t$ on its own are higher than what we observe for $\vec{H}_t$, suggesting that the feed-forward sublayers do help in making the Transformer architecture more effective on the MLM objective.

6.4.2 Lexical contents & WSD

The effectiveness on the training objective is however not necessarily linked to the utility of the term on downstream applications. We first look at how the vector spaces are organized, and which term describes the most linguistically appropriate vector space. Here, we turn to WSD, as it is fair to expect that distinct senses should correspond to different vector representations.

Experimental protocol

We consider an intrinsic KNN-based setup and an extrinsic probe-based setup. The former is inspired from (Wiedemann et al., 2019): we assign to a target the

most common label in its neighborhood. We restrict neighborhoods to words with the same annotated lemma and use the $k = 5$ nearest neighbors (using cosine distance). The latter is a 2-layer MLP similar to Du et al. (2019), where the first layer is shared for all items and the second layer is lemma-specific.

We use the NLTK SemCor dataset (Landes et al., 1998; Bird et al., 2009), with an 80%–10%–10% split between train, development and test. We ignore targets with monosemous or OOV lemmas. We sum over word-pieces to convert them into single word vector representations. Learning rate, dropout, weight decay, β_1 and β_2, learning rate scheduling are selected with Bayesian Optimization, using 100 random configuration samples and accuracy as objective.

Results

Results are shown in Table 6.3, we report results using accuracy. For reference, selecting the most frequent sense would yield an accuracy of 57%, whereas picking a sense at random would yield an accuracy of 24%. The input terms $\vec{I}_t$ and the offsets $\vec{C}_t$ struggle to outperform the most frequent sense baseline: the relevant KNN accuracy scores are lower, whereas the corresponding classifier accuracy scores are barely above.

Overall the same picture emerges from the intrinsic and extrinsic setups. The $\vec{F}_t$ term does not yield the highest performances in our experiment, instead, the $\vec{H}_t$ term systematically dominates. When considering a single term, $\vec{H}_t$ is ranked first and $\vec{F}_t$ second. When considering sums of two terms, the setups ranked 1^{st}, 2^{nd}, and 3^{rd} are those that include $\vec{H}_t$; setups ranked 3^{rd} to 5^{th}, those that include $\vec{F}_t$. Even more surprisingly, when summing three of the terms, the highest ranked setup is the one where we exclude $\vec{F}_t$, and the lowest ranked setup is

	$\vec{I}_t$	$\vec{H}_t$	$\vec{F}_t$	$\vec{C}_t$	$\vec{I}_t + \vec{H}_t$	$\vec{I}_t + \vec{F}_t$	$\vec{I}_t + \vec{C}_t$	$\vec{H}_t + \vec{F}_t$	$\vec{H}_t + \vec{C}_t$	$\vec{F}_t + \vec{C}_t$	$\vec{I}_t + \vec{H}_t + \vec{F}_t$	$\vec{I}_t + \vec{H}_t + \vec{C}_t$	$\vec{I}_t + \vec{F}_t + \vec{C}_t$	$\vec{H}_t + \vec{F}_t + \vec{C}_t$	$\vec{I}_t + \vec{H}_t + \vec{F}_t + \vec{C}_t$
KNN	54.36	64.07	62.45	55.40	64.22	62.22	56.34	63.37	64.40	62.43	63.56	64.44	62.18	64.10	63.94
Cls.	58.42	66.84	64.46	57.32	66.65	63.88	57.83	65.71	66.95	64.46	65.76	66.77	64.45	65.88	65.99

Table 6.3: Accuracy on SemCor WSD (in %)

the one where we exclude $\vec{H}_t$. This suggests that the $\vec{F}_t$ term is not necessarily helpful to the final representation for WSD.

One argument that could be made here would be to posit that the predictions derived from the different sums of terms are intrinsically different, hence a purely quantitative ranking might not capture this important distinction. To verify whether this holds, we can look at the proportion of predictions that agree for any two models. This is summarized in Figure 6.8: an individual cell will detail the proportion of the assigned labels shared by the models for that row and that column. In short, we see that model predictions tend to a high degree of overlap. For both KNN and classifier setups, the three models which appear to make the most distinct predictions turn out to be computed from the $\vec{I}_t$ term, the $\vec{C}_t$ term or their sum: i.e., the models that struggle to perform better than the MFS baseline and are derived from static representations. In other words, we find no strong evidence that the $\vec{H}_t$ and $\vec{F}_t$ produce very different results.

6.4.3 Effects of finetuning & NER

Downstream application can also be achieved through fine-tuning, i.e., restarting a model's training to derive better predictions on a narrower task. As we saw from Figures 6.6b and 6.6c, the modifications brought upon this second round of training are task specific, meaning that an exhaustive experimental survey is out of our reach.

We consider the task of Named Entity Recognition, using the WNUT 2016 shared task dataset (Strauss et al., 2016). We contrast the performances of terms extracted from the non-finetuned BERT model to that of the aforementioned vari-

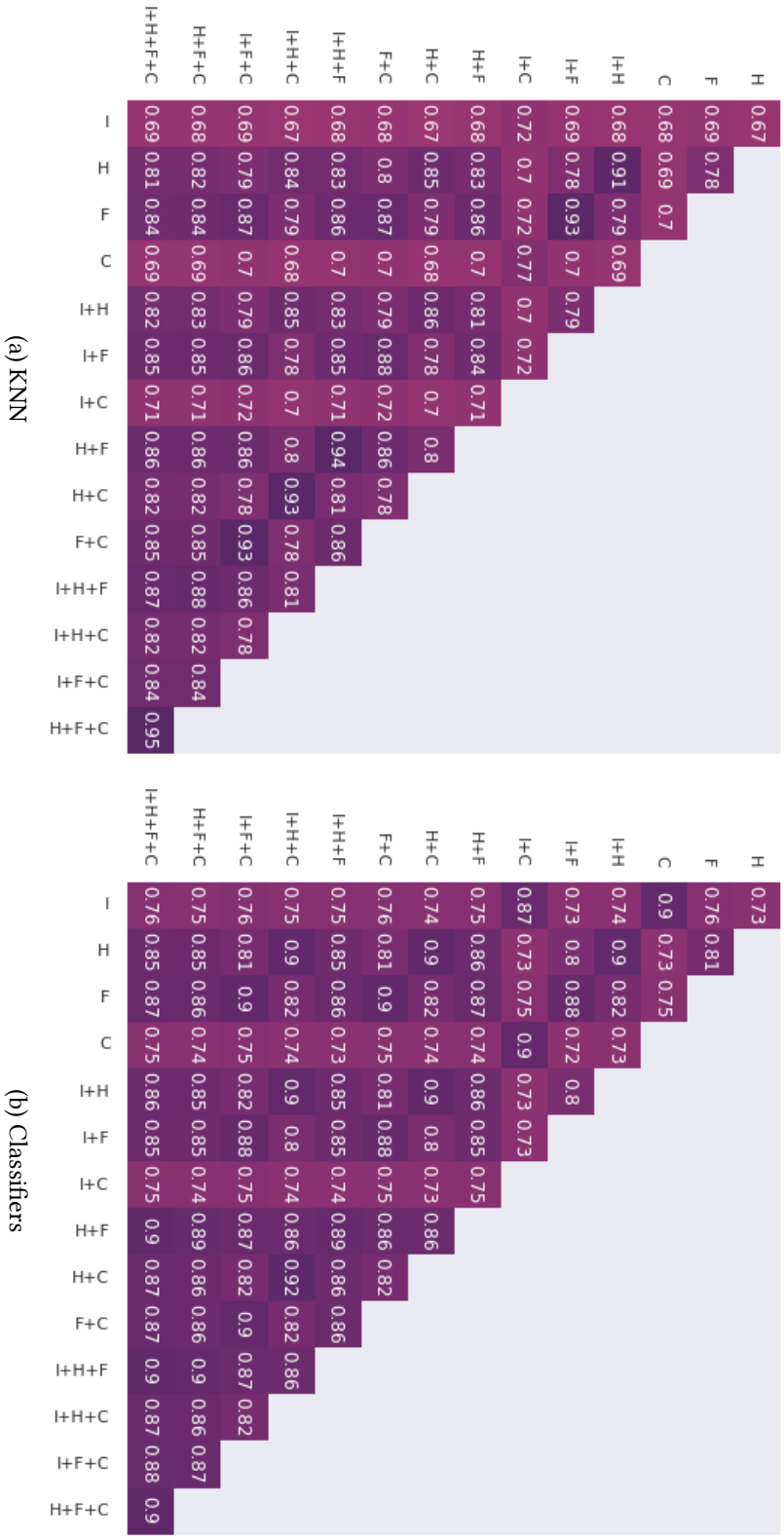


Figure 6.8: Prediction agreement for WSD models

ant finetuned on the CONLL 2003 NER dataset. We learn shallow logistic regressions, setting hyperparameters with Bayesian Optimization, using 100 samples and macro-f_1 as the objective.

Results are presented in Table 6.4. Finetuning BERT on another NER dataset unsurprisingly has a systematic positive impact. More interesting is the impact this finetuning has on the $\vec{F}_t$ term: when used as sole input, we observe an increase in performance of 10%, and similar improvements are observed consistently across all setups involving $\vec{F}_t$. The highest performance is reached by using $\vec{I}_t + \vec{H}_t$ as input (46.96%), and in the base setting the highest performance is reached by using $\vec{I}_t + \vec{H}_t + \vec{C}_t$—suggesting that even in this setup, $\vec{F}_t$ might be superfluous.

We can also look at whether the various classifiers produce different outputs. Given the high class imbalance of the dataset at hand, we macro-average the prediction overlaps by label. The result is shown in Figure 6.9; Figure 6.9a details the behavior of the untuned model, whereas Figure 6.9b details that of the NER-finetuned model. In this round of experiments, we see much more distinctly that the $\vec{I}_t$ model, the $\vec{C}_t$ model and the $\vec{I}_t + \vec{C}_t$ model behave markedly different from the rest. Rather surprising here is that $\vec{I}_t + \vec{H}_t$ non-finetuned model also behaves quite differently from the remaining models; for this we have no convincing explanation. Looking at the NER-finetuned model (Figure 6.9b), we find that aside from the aforementioned static representations, most predictions display a degree of overlap much higher than what we observe for the non-finetuned model: both feed-forwards and multi-head attention are skewed towards producing outputs more adapted to NER tasks.

	$\vec{I}_t$	$\vec{H}_t$	$\vec{F}_t$	$\vec{C}_t$	$\vec{I}_t + \vec{H}_t$	$\vec{I}_t + \vec{F}_t$	$\vec{I}_t + \vec{C}_t$	$\vec{H}_t + \vec{F}_t$	$\vec{H}_t + \vec{C}_t$	$\vec{F}_t + \vec{C}_t$	$\vec{I}_t + \vec{H}_t + \vec{F}_t$	$\vec{I}_t + \vec{H}_t + \vec{C}_t$	$\vec{I}_t + \vec{F}_t + \vec{C}_t$	$\vec{H}_t + \vec{F}_t + \vec{C}_t$	$\vec{I}_t + \vec{H}_t + \vec{F}_t + \vec{C}_t$
base	17.85	41.09	34.20	5.48	21.13	35.55	16.13	36.75	40.84	36.89	38.19	39.70	37.22	39.01	37.69
tune	18.08	42.25	44.90	12.36	46.96	44.47	16.80	43.53	46.43	45.08	42.95	46.49	45.60	44.38	43.54

Table 6.4: Macro-f_1 on WNUT 2016 (in %)

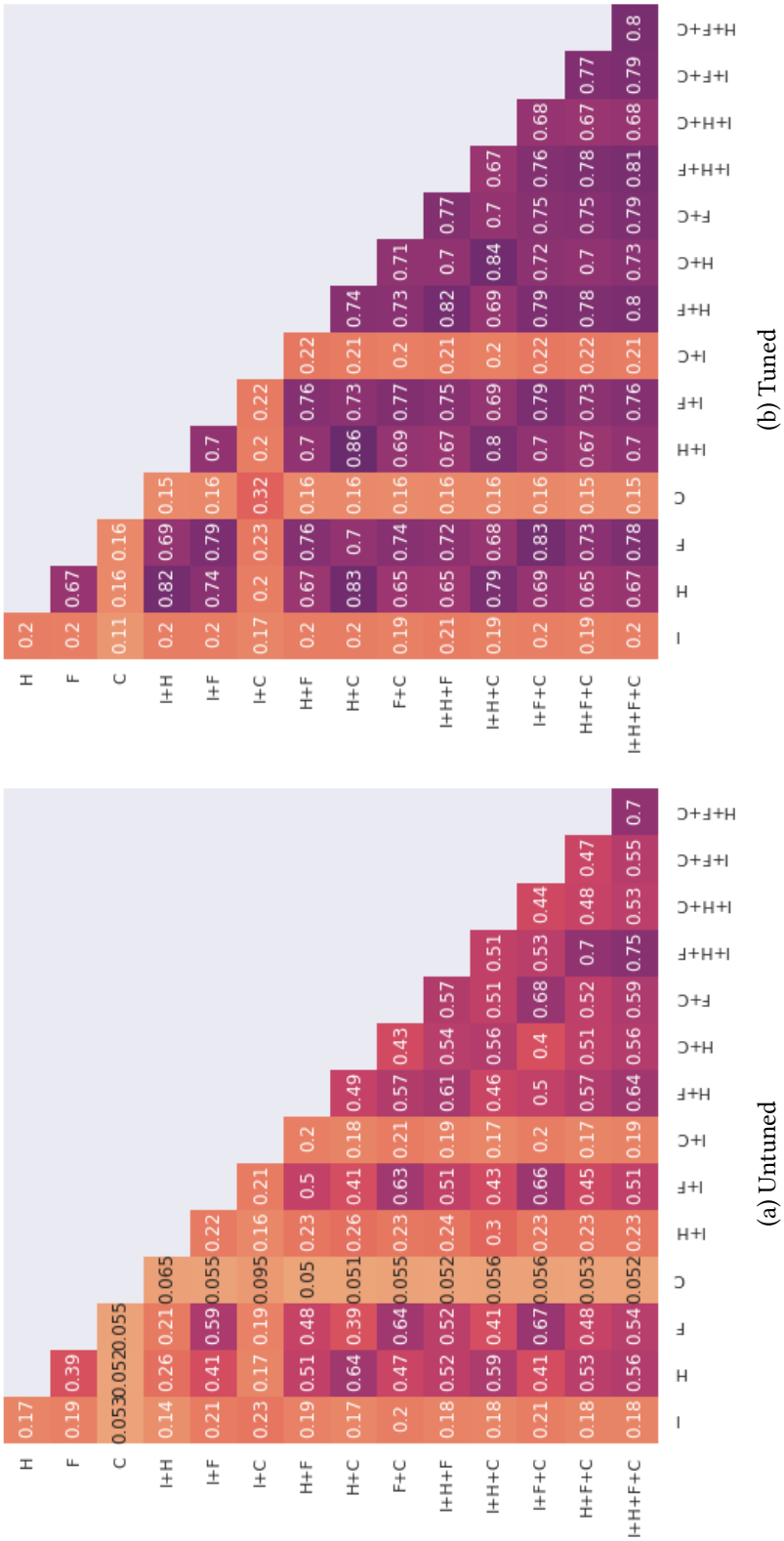


Figure 6.9: NER prediction agreement (macro-average)

6.5 Conclusions

In this chapter, we have investigated whether the vector space described by Transformer embeddings in general, and BERT in particular, encodes the properties we can expect from a vector-space model of meaning. We have used a diverse array of tools for doing so, from clustering techniques to mathematical analysis and from quantitative descriptions to classifier probes.

In Section 6.1, we saw that type-level semantics seem to match our general expectations about DSMs—and yet that focusing on details leaves us with a much foggier picture. The main issue we identify stems from BERT's next sentence prediction objective, which requires tokens to be marked according to which sentence they belong. This introduces a distinction between *first* and *second sentence of the input* that runs contrary to our expectations in terms of cross-sentence coherence. The validity of such a distinction for lexical semantics may be questioned. Similarly, other works (Lample and Conneau, 2019; Z. Yang et al., 2019; M. Joshi et al., 2020; Liu et al., 2019) stress that the usefulness and pertinence of the NSP task were not obvious. While the primary assessment we conducted in Section 6.1.1 showed that token representations did tend to cluster naturally according to their types, a finer study detailed in section Section 6.1.2 highlighted that tokens from distinct sentence positions (even vs. odd) tend to have different representations.

This artifact can be seen as a direct consequence of BERT's architecture: residual connections, along with the use of specific vectors to encode sentence position, entail that tokens for a given sentence position are 'shifted' with respect to tokens for the other position. This issue is however only the tip of the iceberg,

so to speak. As we have shown in Section 6.2, the Transformer architecture itself shapes and biases the vector space described by Transformer embeddings. Transformer embeddings can be decomposed linearly to describe the impact of each network sub-modules.

As we discussed in Sections 6.2.1 and 6.3.1, we can use this linear structure to draw connections to a wide array of studies on Transformer behavior, from anisotropy to their syntactic abilities, and from attention-based studies to variations of performances across layers. In other words, the derivation we presented in Section 6.2.1 provides a general explanation of the structure of Transformer embeddings that is directly relevant to a wide a range of known characteristics of these embeddings. Simply put, this derivation can be seen as an effective means of explaining how to exploit Transformer embeddings. This explanation is purely mechanical in nature: it relates to specific network sub-modules and does not intuitively mesh with our expectations of a description of the meaning of a word.

One of the conclusions that we had drawn from our attempt to testing the distributional hypothesis in Chapter 5 was that DSMs in general, and BERT-like models in particular, did not seem to properly represent the behavior of human speakers: in particular, in Section 5.4.3, we saw that DSMs, both static and contextual, did not coincide with human behavior any better than n-gram baselines on a task derived from what we expect of implementations of distributional semantics (Section 5.1). This has casted doubt on their status of implementations of a lexical semantic theory. The results of the experiments conducted in the present chapter strengthen this claim even more: vector spaces derived from Transformer-based models such as BERT bear the very recognizable marks of

the architecture that engenders them. Overall, there is a discernible gap between what we expect of a proper implementation of a lexical semantic theory and what we observe from the vectorization algorithms produced by NLP engineers. The characterization that emerges from an inspection of the Transformer architecture (Section 6.2) appears to be much more successful at explaining the behavior of Transformer embeddings than what a theory-driven approach does yield (Section 5.1). Or, to put it in a provocative manner: *word embeddings are not distributional semantics models.*

Conclusions

The keeper stokes the fires of the beacon at the top
Of a tower that drives us all onto the rocks
I thought the water would be quiet I was right
They warned me of the lighthouse but it shines so bright

— Dyscarnate, *Nothing Seems Right*

Word embeddings, insofar they are practical implementations of distributional semantics, leave much to be desired.

Distributional semantics is, as we saw in Chapter 1, a very diverse field of study, which connects to semantics, psycholinguistics, information retrieval and neural NLP. Even when restricting ourselves to the major works of the last decade, we find distinct architectures tied to distinct experimental protocols. Even when focusing very narrowly on a single model, we see that hyperparameters such as the associated vector space dimension impact the interpretation of the metrics we have at our disposal. In short, implementations of distributional semantics are necessarily subject to variation.

Yet, we were able to propose a tentative formalization for this field. In Section 5.1, we developed the idea that distributional subtitutablity was a promising

framework for unifying and formalizing distributional semantics as a consistent semantic theory. If we adopt a practical point of view, we see that many distributional semantics models can be construed as inference models for the distributional substitution task. From a theoretical stance, there is a rich connection to the seminal work of Harris (1954) as well as subsequent theoretical reviews and development: distributional substitutability is baked in the founding assumptions of distributional semantics. Adopting this formalization yields a very intuitive definition of distributional semantics models: a DSM is a model that can discriminate words based on context.

By explicitly stating the criteria we expect DSMs to conform with, we can quantify how far off this mark current word embedding models are. On the one hand, we see that word embeddings achieve some degree of performance on the distributional substitution task (Section 5.4.2) and, in some circumstances, they can serve as tools to manipulate the distributional hypothesis (Section 5.4.4). On the other hand, how they achieve these results does not appear to be in line with human behavior (Section 5.4.3). Furthermore, we can identify artifacts in the vector spaces of some of the most popular word embedding models, as we saw in Chapter 6. These artifacts range from word types not being entirely consistently described in word token models (Section 6.1.1), to biases imputable to the objective function (Sections 6.1.2 and 6.1.3), and to issues stemming from the very architecture these neural models employ (Sections 6.2 and 6.3.1).

While we can re-frame the objective functions of word embeddings so as to coincide with our expectations for distributional semantics models (Section 5.1), the models they converge to fall short of what we expect of an implementation of the lexical theory of distributional semantics (Section 5.4.3). This is in essence

a problem of calibration: the probabilities estimated by word embedding models do not match with what we observe of human behavior. This reflects a certain disconnect between the tools at our disposal and the requirements we as linguist may have: neural networks are built so as to *maximize* the likelihood of producing the optimal token with respect to their objective function. The expectations we have for distributional semantics models are more subtle, in that we want them to account for *degrees of uncertainty*. Word embeddings, as they currently are, cannot be construed as complete, coherent and consistent implementations of distributional semantics, as they are instead geared towards solving their objective functions—as appeared obvious when we looked at the effects of finetuning in Section 6.4.3.

This contrasts with what we can observe for dictionary definitions. While there is quite a variation in the lexical resources we dub "dictionaries," one can easily establish a consensual working concept of a dictionary (Section 2.1). This has led the NLP community to use these resources as ground truth meaning inventories in their applications (Sections 2.3 and 2.4). One question we have not addressed is whether this trust is in fact warranted: dictionaries do not provide a clear manner of relating words to real-world objects (Section 2.2), nor do they explicitly discuss how they establish their inventories of meanings, nor are they especially handy to study how linguistic context impacts meaning. Lastly, establishing quantifiable and principled expectations for dictionaries—as we did with word embeddings—also seems currently outside of our grasp.

These limitations of dictionaries and embeddings explain in part the overall low results we observe when we do compare vectors and glosses. Be it by comparing metric spaces directly, as we did in Chapter 3, or through neural net-

works as in Chapter 4, our results were overall inconclusive. We also identified a number of supplementary confounding factors that impact our comparisons across lexical semantic theories. How to align word types or word tokens to word senses (Sections 4.1 and 4.2), the quality of embedding representations (Sections 3.3 and 4.3) and the presence of confounding factors in natural language (Section 3.2) also weigh on such comparisons.

*

* *

Our initial line of inquiry concerned itself with the nature of distributional semantics as a lexical semantic theory—in the same sense that lexicography can be understood as an endeavor towards providing the meaning of every item in the lexicon. The experiments we conducted have given us numerous elements to develop and answer it. What transpires is that distributional semantics and lexicography are likely not equivalent lexical semantic theories. Setting aside whether we can reach any conclusions pertaining to distributional semantics based solely on observations from word embeddings, we find that confounding factors weigh heavily on the less-than-conclusive comparisons we can make. This strongly hints at a very distinct underlying concept of meaning: word sense as is described by distributional vectors appears to be quite different from word sense as we commonly find it depicted in dictionaries. Perhaps we could find a more suitable lexical theory to compare word embeddings to—perhaps a true distributional semantics model would be able to solve the distributional substi-

tution task as humans would—perhaps the issues we pointed out are merely a problem of calibrating the model—perhaps solving these issues would then yield distributional representations rich enough to compare with meaning representations, be it definitions or other. For now, we lack solid evidence that would point to these limitations being eventually overcome.

In this dissertation, we have attempted to demonstrate three key facts about distributional semantics. First, the diversity of studies in distributional semantics do not entail that no formal statements regarding this theory can be made. Here, we have proposed to use distributional substitutability, but we could more generally stress that the probability distribution that is explicitly modeled by a DSM is a very convenient handle for the linguist to grasp. Second, an inherent difficulty of this framework is that it cannot easily be equated or related to lexicography, making its value as an explanatory framework less secure. Our definition of the goal of a lexical semantic theory was to provide a description of the meaning of every item in the lexicon. However, that we cannot easily relate distributional semantics to another lexical semantic theory questions whether the distributional hypothesis actually provides an alternative account, or whether it deals with a very distinct set of facts altogether. Third, the gap in quality between practical implementations of distributional semantics and our expectations necessarily adds on to the confusion. Nonetheless, that we can make quantitative statements about this gap should be taken as a very encouraging sign for future research.

The hurdles and limitations we have identified throughout this dissertation are very much interlinked. The high focus on engineering practical applications in the field of NLP entails that word embeddings models are discussed and studied

first and foremost owing to their effectiveness on some set of tasks. Theoretical considerations that might interest the (computational) linguist come second—if they at all factor in. Hence the sort of limitations we underscored here should not surprise us. In short, it seems that *what a model can do* often matters more than *what it does model*. In this dissertation, we have been more concerned with the latter, and said very little of the former. That is not to say that we do not believe that nothing of value is to be gained by a more performance-driven approach; but rather that the discrepancy between NLP and linguistics appears to us a more salient and pressing question to answer. In the last decades, the NLP community has shifted from shallow statistical perceptrons and rule-based models to first fully espousing neural networks and now to committing to a handful of models pre-trained by a very small number of research groups. In such a context where datasets shelf-life dwindles and explainable NLP becomes a necessity, it is important that we in the NLP community do not renege our links to the study of language.

*

* *

An unstated goal of this dissertation was to explore the variety of tools available to the NLP researcher. We have established baselines using functional algebra (Sections 1.4 and 6.2) and collected human judgments with an online game (Section 5.3). We have studied metrics of statistical correlation (Chapter 3) and probing classifiers (Sections 6.4.1 to 6.4.3), neural networks (Sections 4.2 and 4.3)

and thought experiments (Section 2.2). This dissertation does not claim by any mean to be an exhaustive review of all the existing means at our disposal to investigate NLP systems, but we do hope to have illustrated the boons that come from adopting an approach partly rooted in linguistic concerns.

There are many aspects that we have left untouched. We can name the more obvious limitations of any NLP research agenda: we have studied a limited number of languages, have reviewed a limited number of architectures, have used a limited number of metrics, and so on. An important point to mention here is that we have adopted definitions as a gold standard against which to compare word embeddings. This position is at best debatable—few, if any, among the lexicography abd cognitive science communities would argue that meaning as it exists in a speaker's mind has much to do with dictionary definitions. Other frameworks such as word association maps or semantic networks might be more immediately comparable to distributional semantic representations. In all, the argument that was of this dissertation is perhaps best put as follow: when looking for models of word meaning, we should take distributional semantics at least as seriously as dictionary definitions.

Another limitation that is perhaps more central to our argument is that we have not tackled the question of semantic grounding, but rather introduced it as a perspective for future research. As we saw in Section 2.2, how to link a word with the real world object it refers to is a salient problem both in dictionaries and in word embeddings. Here, word embeddings provide an interesting vantage point. Firstly, we can simply focus on the difference between multimodal representations and text-only embeddings. Another angle of approach consists in delineating the capacities of text-only models, as we saw in Chapter 5. Con-

trasting their behavior to human judgments gives us a general idea of how far one can push a text-only model—from which we can quantify the gap that still needs to be covered by introducing semantic grounding information. A more complete overview would require an in-depth study of the performances of multimodal representations on the dataset we constructed.

Another aspect we have addressed but briefly and yet merits a lengthier investigation is that of textual metrics. As we saw in Chapters 3 and 4, defining metrics for textual similarity (between two attested definitions or between a target and a production) is far from a trivial task. Here we have looked into potential confounding factors and known limitations. One family of metrics which our results impact is that of embeddings-based metrics, such as the MoverScore we used in our shared task (Section 4.4). The limited linguistic interpretability of word embeddings likely entails that such metrics should also induce caution in the NLP practitioner that seeks explainability and transparency from their system. This also we leave for future research.

To summarize what we have argued for in a single word: word embeddings cannot be viewed as concrete implementations of a well-formalized semantic framework derived from word distribution. The distributional semantics community often echoes the famous quote of Firth (1957): "*You shall know a word by the company it keeps*". The train of thoughts we outlined in the present dissertation leads us to question whether the company a word keeps is actually enough for us to know it.

Bibliography

Sweet amnesia here to free you
As the pages burn
All your trials solved by fire
As the pages burn

— Arch Enemy, *As the Pages Burn*

Abell, Martha L et al. (1999). *Statistics with mathematica*. Vol. 1. Academic Press.

Ardoiz, Alfonso et al. (2022). *MMG at SemEval-2022 Task 1: A Reverse Dictionary approach based on a review of the dataset from a lexicographic perspective*.

Arora, Sanjeev et al. (2016). "Linear Algebraic Structure of Word Senses, with Applications to Polysemy". In: *CoRR* abs/1601.03764. arXiv: 1601.03764. URL: http://arxiv.org/abs/1601.03764.

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton (2016). *Layer Normalization*. arXiv: 1607.06450 [stat.ML].

Bahdanau, Dzmitry, Kyung Hyun Cho, and Yoshua Bengio (Jan. 2015). *Neural machine translation by jointly learning to align and translate*. English (US). 3rd International Conference on Learning Representations, ICLR 2015 ; Conference date: 07-05-2015 Through 09-05-2015.

Banerjee, Satanjeev and Alon Lavie (June 2005). "METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments". In: *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*. Ann Arbor, Michigan: Association for Computational Linguistics, pp. 65–72. URL: https://www.aclweb.org/anthology/W05-0909.

Baroni, Marco, Georgiana Dinu, and Germán Kruszewski (June 2014). "Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Baltimore, Maryland: Association for Computational Linguistics, pp. 238–247. DOI: 10.3115/v1/P14-1023. URL: https://aclanthology.org/P14-1023.

Bear, Diego and Paul Cook (Sept. 2021). "Cross-Lingual Wolastoqey-English Definition Modelling". In: *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*. Held Online: INCOMA Ltd., pp. 138–146. URL: https://aclanthology.org/2021.ranlp-main.17.

Béjoint, Henri (2016). "Dictionaries for General Users". In: *The Oxford Handbook of Lexicography*. Ed. by Philip Durkin. Oxford: Oxford University Press. Chap. 2, pp. 7–24.

Bendahman, Nihed et al. (2022). *BL.Research at SemEval-2022 Task 1: Deep networks for Reverse Dictionary using embeddings and LSTM autoencoders*.

Bender, Emily M. and Alexander Koller (July 2020). "Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online:

Association for Computational Linguistics, pp. 5185–5198. DOI: 10.18653/v1/2020.acl-main.463. URL: https://aclanthology.org/2020.acl-main.463.

Bengio, Yoshua et al. (Mar. 2003). "A Neural Probabilistic Language Model". In: *J. Mach. Learn. Res.* 3.null, pp. 1137–1155. ISSN: 1532-4435.

Bergenholtz, Henning (Nov. 2012). "What is a Dictionary?" In: *Lexikos* 22, pp. 20–30. DOI: 10.5788/22-1-995.

Bevilacqua, Michele, Marco Maru, and Roberto Navigli (Nov. 2020). "Generationary or "How We Went beyond Word Sense Inventories and Learned to Gloss"". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 7207–7221. DOI: 10.18653/v1/2020.emnlp-main.585. URL: https://aclanthology.org/2020.emnlp-main.585.

Bila, Slaven et al. (2004). "Dictionary search based on the target word description". In: *Proceedings of the 10th Annual Meeting of the Association for Natural Language Processing (ANLP 2004)*.

Bird, Steven, Ewan Klein, and Edward Loper (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*." O'Reilly Media, Inc.".

Bloomfield, L. (1933). *Language*. U.S.A: Holt, Rinehart and Winston, Inc.

Bojanowski, Piotr et al. (2017). "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146. DOI: 10.1162/tacl_a_00051. URL: https://aclanthology.org/Q17-1010.

Boleda, Gemma (2020). "Distributional Semantics and Linguistic Theory". In: *Annual Review of Linguistics* 6.1, pp. 213–234. DOI: 10.1146/annurev-linguistics-011619-030303. eprint: https://doi.org/10.1146/annurev-linguistics-011619-030303. URL: https://doi.org/10.1146/annurev-linguistics-011619-030303.

Bolukbasi, Tolga et al. (2016). "Man is to Computer Programmer as Woman is to Homemaker? Debiasing Word Embeddings". In: *Advances in Neural Information Processing Systems 29*. Ed. by D. D. Lee et al. Curran Associates, Inc., pp. 4349–4357. URL: http://papers.nips.cc/paper/6228-man-is-to-computer-programmer-as-woman-is-to-homemaker-debiasing-word-embeddings.pdf.

Bonami, Olivier and Denis Paperno (2018). "A characterisation of the inflection-derivation opposition in a distributional vector space". In: *Lingua e Langaggio*.

Bosc, Tom and Pascal Vincent (Oct. 2018). "Auto-Encoding Dictionary Definitions into Consistent Word Embeddings". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, pp. 1522–1532. DOI: 10.18653/v1/D18-1181. URL: https://aclanthology.org/D18-1181.

Brighton, Henry and Simon Kirby (Mar. 2006). "Understanding Linguistic Evolution by Visualizing the Emergence of Topographic Mappings". In: *Artif. Life* 12.2, pp. 229–242. ISSN: 1064-5462. DOI: 10.1162/106454606776073323. URL: https://doi.org/10.1162/106454606776073323.

Bruni, Elia, Nam Khanh Tran, and Marco Baroni (Jan. 2014). "Multimodal Distributional Semantics". In: *J. Artif. Int. Res.* 49.1, pp. 1–47. ISSN: 1076-9757.

Brunner, Gino et al. (Aug. 2019). "On the Validity of Self-Attention as Explanation in Transformer Models". In: *arXiv e-prints*, arXiv:1908.04211, arXiv:1908.04211. arXiv: 1908.04211 [cs.CL].

Budescu, David V. (1993). "Dominance analysis: A new approach to the problem of relative importance of predictors in multiple regression." In: *Psychological Bulletin* 114.3, pp. 542–551. DOI: 10.1037/0033-2909.114.3.542. URL: https://doi.org/10.1037/0033-2909.114.3.542.

Cai, Xingyu et al. (2021). "Isotropy in the Contextual Embedding Space: Clusters and Manifolds". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=xYGNO86OWDH.

Calvo, Hiram, Oscar Méndez, and Marco A. Moreno-Armendáriz (Nov. 2016). "Integrated Concept Blending with Vector Space Models". In: *Comput. Speech Lang.* 40.C, pp. 79–96. ISSN: 0885-2308. DOI: 10.1016/j.csl.2016.01.004. URL: https://doi.org/10.1016/j.csl.2016.01.004.

Cañete, José et al. (2020). "Spanish Pre-Trained BERT Model and Evaluation Data". In: *PML4DC at ICLR 2020*.

Cer, Daniel, Mona Diab, et al. (Aug. 2017). "SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation". In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Vancouver, Canada: Association for Computational Linguistics, pp. 1–14. DOI: 10.18653/v1/S17-2001. URL: https://aclanthology.org/S17-2001.

Cer, Daniel, Yinfei Yang, et al. (Nov. 2018). "Universal Sentence Encoder for English". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for

Computational Linguistics, pp. 169–174. DOI: 10.18653/v1/D18-2029. URL: https://www.aclweb.org/anthology/D18-2029.

Cerniavski, Rafal and Sara Stymne (2022). *Uppsala University at SemEval-2022 Task 1: Multilingualism in Reverse Dictionaries: Can Foreign Entries Enhance an English Reverse Dictionary?*

Chang, Ting-Yun and Yun-Nung Chen (Nov. 2019). "What Does This Word Mean? Explaining Contextualized Embeddings with Natural Language Definition". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 6064–6070. DOI: 10.18653/v1/D19-1627. URL: https://www.aclweb.org/anthology/D19-1627.

Chang, Ting-Yun, Ta-Chung Chi, et al. (2018). *xSense: Learning Sense-Separated Sparse Representations and Textual Definitions for Explainable Word Sense Networks*. arXiv : 1809.03348. arXiv: 1809.03348. URL: http://arxiv.org/abs/1809.03348.

Chen, Pinzhen and Zheng Zhao (2022). *Edinburgh at SemEval-2022 Task 1: Jointly Fishing for Word Embeddings and Definitions.*

Chodorow, Martin S., Roy J. Byrd, and George E. Heidorn (July 1985). "Extracting Semantic Hierarchies From a Large On-Line Dictionary". In: *23rd Annual Meeting of the Association for Computational Linguistics*. Chicago, Illinois, USA: Association for Computational Linguistics, pp. 299–304. DOI: 10.3115/981210.981247. URL: https://www.aclweb.org/anthology/P85-1037.

Clark, Kevin, Urvashi Khandelwal, et al. (Aug. 2019). "What Does BERT Look at? An Analysis of BERT's Attention". In: *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Florence, Italy: Association for Computational Linguistics, pp. 276–286. DOI: 10.18653/v1/W19-4828. URL: https://aclanthology.org/W19-4828.

Clark, Kevin, Minh-Thang Luong, et al. (Nov. 2020). "Pre-Training Transformers as Energy-Based Cloze Models". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 285–294. DOI: 10.18653/v1/2020.emnlp-main.20. URL: https://www.aclweb.org/anthology/2020.emnlp-main.20.

Coavoux, Maximin (2017). "Discontinuous Constituency Parsing of Morphologically Rich Languages". PhD thesis. Univ Paris Diderot, Sorbonne Paris Cité.

Coenen, Andy et al. (June 2019). "Visualizing and Measuring the Geometry of BERT". In: *arXiv e-prints*, arXiv:1906.02715, arXiv:1906.02715. arXiv: 1906.02715 [cs.LG].

Collobert, Ronan and Jason Weston (2008). "A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning". In: *Proceedings of the 25th International Conference on Machine Learning*. ICML '08. Helsinki, Finland: Association for Computing Machinery, pp. 160–167. ISBN: 9781605582054. DOI: 10.1145/1390156.1390177. URL: https://doi.org/10.1145/1390156.1390177.

Conneau, Alexis et al. (Sept. 2017). "Supervised Learning of Universal Sentence Representations from Natural Language Inference Data". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*.

Copenhagen, Denmark: Association for Computational Linguistics, pp. 670–680. DOI: 10.18653/v1/D17-1070. URL: https://aclanthology.org/D17-1070.

Cover, Thomas M. (1965). "Geometrical and Statistical Properties of Systems of Linear Inequalities with Applications in Pattern Recognition". In: *IEEE Trans. Electronic Computers* 14.3, pp. 326–334. DOI: 10.1109/PGEC.1965.264137. URL: https://doi.org/10.1109/PGEC.1965.264137.

Crawford, H. Vance and Joel Crawford (1997). *Reverse electronic dictionary using synonyms to expand search capabilities*. U.S. patent number 5,649,221.

Dai, Zihang et al. (2019). "Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context". In: *CoRR* abs/1901.02860. arXiv: 1901.02860. URL: http://arxiv.org/abs/1901.02860.

Dautriche, Isabelle et al. (2017). "Wordform Similarity Increases With Semantic Similarity: An Analysis of 100 Languages". In: *Cognitive Science* 41.8, pp. 2149–2169. DOI: 10.1111/cogs.12453. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cogs.12453. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/cogs.12453.

Devlin, Jacob et al. (June 2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

Django (2020). *Django Project*. Version 3.1. URL: https://www.djangoproject.com/.

Dong, Yan (2015). "The prosody and morphology of elastic words in Chinese: annotations and analyses". PhD thesis. University of Michigan.

Du, Jiaju, Fanchao Qi, and Maosong Sun (2019). "Using BERT for Word Sense Disambiguation". In: *CoRR* abs/1909.08358. arXiv: 1909.08358. URL: http://arxiv.org/abs/1909.08358.

Duanmu, San (2007). *The phonology of standard Chinese*. OUP Oxford.

Durkin, Philip (2016). "Introduction". In: *The Oxford Handbook of Lexicography*. Ed. by Philip Durkin. Oxford: Oxford University Press. Chap. 1, pp. 1–4.

Dutoit, Dominique and Pierre Nugues (2002). "A Lexical Database and an Algorithm to Find Words from Definitions". In: *Proceedings of the 15th European Conference on Artificial Intelligence*. ECAI'02. Lyon, France: IOS Press, pp. 450–454. ISBN: 9781586032579.

Edmonds, David (1999). *The Oxford reverse dictionary*. New York: Oxford University Press.

El Khalout, Ilknur Durgar and Kemal Oflazer (2004). "Use of Wordnet for Retrieving Words from Their Meanings". In: *Proceedings of the Second Global Wordnet Conference (GWC 2004)*, pp. 118–123.

Erk, Katrin and Sebastian Padó (July 2010). "Exemplar-Based Models for Word Meaning in Context". In: *Proceedings of the ACL 2010 Conference Short Papers*. Uppsala, Sweden: Association for Computational Linguistics, pp. 92–97. URL: https://www.aclweb.org/anthology/P10-2017.

Ethayarajh, Kawin (Nov. 2019). "How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embed-

dings". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 55–65. DOI: 10.18653/v1/D19-1006. URL: https://aclanthology.org/D19-1006.

Fellbaum, Christiane (1998). *WordNet: An Electronic Lexical Database*. Bradford Books.

Ferret, Olivier (2021). *Using Distributional Principles for the Semantic Study of Contextual Language Models*. arXiv: 2111.12174 [cs.CL].

Firth, John Rupert (1952). "Linguistic Analysis as a Study of Meaning (1952)". In: *Selected papers of J.R. Firth, 1952–59*. Ed. by F. R. Palmer.

— (1957). "A synopsis of linguistic theory 1930-55." In: *Studies in Linguistic Analysis (special volume of the Philological Society)* 1952-59, pp. 1–32.

Frazier, Peter I. (2018). *A Tutorial on Bayesian Optimization*. arXiv: 1807.02811 [stat.ML].

Gadetsky, Artyom, Ilya Yakubovskiy, and Dmitry Vetrov (2018). "Conditional Generators of Words Definitions". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 266–271. URL: http://aclweb.org/anthology/P18-2043.

Gaume, Bruno, Nabil Hathout, and Philippe Muller (2004). "Word Sense Disambiguation Using a Dictionary for Sense Similarity Measure". In: *Proceedings of the 20th International Conference on Computational Linguistics*. COLING '04. Geneva, Switzerland: Association for Computational Linguistics. DOI:

10.3115/1220355.1220528. URL: https://doi.org/10.3115/1220355.1220528.

Geeraerts, Dirk (2016). "Lexicography and Theories of Lexical Semantic". In: *The Oxford Handbook of Lexicography*. Ed. by Philip Durkin. Oxford: Oxford University Press. Chap. 26, pp. 62–75.

Gitlab (2021). *GitLab Community Edition*. Version 14.5.2. URL: https://docs.gitlab.com/.

Gladkova, Anna, Aleksandr Drozd, and Satoshi Matsuoka (June 2016). "Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't." In: *Proceedings of the NAACL Student Research Workshop*. San Diego, California: Association for Computational Linguistics, pp. 8–15. DOI: 10.18653/v1/N16-2002. URL: https://aclanthology.org/N16-2002.

Goldberg, Yoav and Omer Levy (2014). "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method". In: *CoRR* abs/1402.3722. arXiv: 1402.3722. URL: http://arxiv.org/abs/1402.3722.

Goodfellow, Ian et al. (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems*. Ed. by Z. Ghahramani et al. Vol. 27. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Grave, Edouard et al. (May 2018). "Learning Word Vectors for 157 Languages". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. Miyazaki, Japan: European Language Resources Association (ELRA). URL: https://aclanthology.org/L18-1550.

Griffiths, Thomas L., Mark Steyvers, and Joshua B. Tenenbaum (2007). "Topics in semantic representation". In: *Psychological Review*, pp. 211–244.

Gulordava, Kristina et al. (June 2018). "Colorless Green Recurrent Networks Dream Hierarchically". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 1195–1205. DOI: 10.18653/v1/N18-1108. URL: https://www.aclweb.org/anthology/N18-1108.

Gulrajani, Ishaan et al. (2017). "Improved Training of Wasserstein GANs". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2017/file/892c3b1c6dccd52936e27cbd0ff683d6-Paper.pdf.

Guo, Shaoyu (1938). "中国语词之弹性作用 (The function of elastic word length in Chinese)". In: *Yen Ching Hsueh Pao* 24, pp. 1–34.

Gutiérrez, E. Dario, Roger Levy, and Benjamin Bergen (Aug. 2016). "Finding Non-Arbitrary Form-Meaning Systematicity Using String-Metric Learning for Kernel Regression". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 2379–2388. DOI: 10.18653/v1/P16-1225. URL: https://www.aclweb.org/anthology/P16-1225.

Hanks, Patrick (2016). "Definition". In: *The Oxford Handbook of Lexicography*. Ed. by Philip Durkin. Oxford: Oxford University Press. Chap. 7, pp. 94–122.

Harnad, Stevan (June 1990). "The Symbol Grounding Problem". In: *Phys. D* 42.1–3, pp. 335–346. ISSN: 0167-2789. DOI: 10.1016/0167-2789(90)90087-6. URL: https://doi.org/10.1016/0167-2789(90)90087-6.

Harris, Zellig (1954). "Distributional structure". In: *Word* 10.23, pp. 146–162.

Hathout, Nabil and Franck Sajous (May 2016). "Wiktionnaire's Wikicode GLAWIfied: a Workable French Machine-Readable Dictionary". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Ed. by Nicoletta Calzolari (Conference Chair) et al. Portorož, Slovenia: European Language Resources Association (ELRA). ISBN: 978-2-9517408-9-1.

Hendrycks, Dan and Kevin Gimpel (2016). "Gaussian Error Linear Units (GELUs)". In: *arXiv preprint arXiv:1606.08415*.

Hewitt, John and Percy Liang (Nov. 2019). "Designing and Interpreting Probes with Control Tasks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 2733–2743. DOI: 10.18653/v1/D19-1275. URL: https://aclanthology.org/D19-1275.

Hewitt, John and Christopher D. Manning (June 2019). "A Structural Probe for Finding Syntax in Word Representations". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4129–4138. DOI: 10.18653/v1/N19-1419. URL: https://aclanthology.org/N19-1419.

Hill, Felix et al. (2016). "Learning to Understand Phrases by Embedding the Dictionary". In: *Transactions of the Association for Computational Linguistics* 4, pp. 17–30. DOI: 10.1162/tacl_a_00080. URL: https://aclanthology.org/Q16-1002.

Huang, Cheng-Zhi Anna et al. (2018). "An Improved Relative Self-Attention Mechanism for Transformer with Application to Music Generation". In: *CoRR* abs/1809.04281. arXiv: 1809.04281. URL: http://arxiv.org/abs/1809.04281.

Huang, Luyao et al. (Nov. 2019). "GlossBERT: BERT for Word Sense Disambiguation with Gloss Knowledge". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 3509–3514. DOI: 10.18653/v1/D19-1355. URL: https://aclanthology.org/D19-1355.

Jackson, Frank (1982). "Epiphenomenal Qualia". In: *Philosophical Quarterly* 32.April, pp. 127–136. DOI: 10.2307/2960077.

Jain, Sarthak and Byron C. Wallace (June 2019). "Attention is not Explanation". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 3543–3556. DOI: 10.18653/v1/N19-1357. URL: https://aclanthology.org/N19-1357.

Jawahar, Ganesh, Benoît Sagot, and Djamé Seddah (July 2019). "What Does BERT Learn about the Structure of Language?" In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 3651–3657. DOI: 10.18653/v1/P19-1356. URL: https://aclanthology.org/P19-1356.

Joshi, Mandar et al. (2020). "SpanBERT: Improving Pre-training by Representing and Predicting Spans". In: *Transactions of the Association for Computa-*

tional Linguistics 8, pp. 64–77. DOI: 10.1162/tacl_a_00300. URL: https://aclanthology.org/2020.tacl-1.5.

Joshi, Vidur, Matthew Peters, and Mark Hopkins (July 2018). "Extending a Parser to Distant Domains Using a Few Dozen Partially Annotated Examples". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 1190–1199. DOI: 10.18653/v1/P18-1110. URL: https://www.aclweb.org/anthology/P18-1110.

Kay, Martin (Nov. 2011). "Zipf's Law and l'Arbitraire du Signe". In: *Linguistic Issues in Language Technology* 6.

Kilgarriff, Adam (Mar. 1997). ""I Don't Believe in Word Senses"". In: *Computers and the Humanities* 31.2, pp. 91–113. ISSN: 1572-8412. DOI: 10.1023/A:1000583911091. URL: https://doi.org/10.1023/A:1000583911091.

— (Nov. 2000). "Business models for Dictionaries and NLP". In: *International Journal of Lexicography* 13, pp. 107–118. DOI: 10.1093/ijl/13.2.107.

Kirby, Simon (1999). "Syntax out of Learning: The Cultural Evolution of Structured Communication in a Population of Induction Algorithms". In: *Proceedings of the 5th European Conference on Advances in Artificial Life*. ECAL '99. Berlin, Heidelberg: Springer-Verlag, pp. 694–703. ISBN: 3540664521.

— (Apr. 2001). "Spontaneous Evolution of Linguistic Structure-an Iterated Learning Model of the Emergence of Regularity and Irregularity". In: *Trans. Evol. Comp* 5.2, pp. 102–110. ISSN: 1089-778X. DOI: 10.1109/4235.918430. URL: https://doi.org/10.1109/4235.918430.

Kirby, Simon, Hannah Cornish, and Kenny Smith (2008). "Cumulative cultural evolution in the laboratory: An experimental approach to the origins of struc-

ture in human language". In: *Proceedings of the National Academy of Sciences* 105.31, pp. 10681–10686. ISSN: 0027-8424. DOI: 10.1073/pnas.0707835105. eprint: https://www.pnas.org/content/105/31/10681.full.pdf. URL: https://www.pnas.org/content/105/31/10681.

Kirby, Simon, Monica Tamariz, et al. (2015). "Compression and communication in the cultural evolution of linguistic structure". In: *Cognition* 141, pp. 87–102.

Kiros, Ryan et al. (2015). "Skip-Thought Vectors". In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes et al. Curran Associates, Inc., pp. 3294–3302. URL: http://papers.nips.cc/paper/5950-skip-thought-vectors.pdf.

Klein, G. et al. (n.d.). "OpenNMT: Open-Source Toolkit for Neural Machine Translation". In: *ArXiv e-prints* (). eprint: 1701.02810.

Kobayashi, Goro et al. (Nov. 2020). "Attention is Not Only a Weight: Analyzing Transformers with Vector Norms". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 7057–7075. DOI: 10.18653/v1/2020.emnlp-main.574. URL: https://aclanthology.org/2020.emnlp-main.574.

Koehn, Philipp (Sept. 2005). "Europarl: A Parallel Corpus for Statistical Machine Translation". In: *Proceedings of Machine Translation Summit X: Papers*. Phuket, Thailand, pp. 79–86. URL: https://aclanthology.org/2005.mtsummit-papers.11.

Kong, Cunliang et al. (2022). *BLCU-ICALL at SemEval-2022 Task 1: Cross-Attention Multitasking Framework for Definition Modeling*.

Korenčić, Damir and Ivan Grubišić (2022). *IRB-NLP at SemEval-2022 Task 1: Exploring the Relationship Between Words and Their Semantic Representations.*

Kosem, Iztok (2016). "Interrogating a Corpus". In: *The Oxford Handbook of Lexicography*. Ed. by Philip Durkin. Oxford: Oxford University Press. Chap. 6, pp. 76–93.

Kupietz, Marc (2016). "Constructing a Corpus". In: *The Oxford Handbook of Lexicography*. Ed. by Philip Durkin. Oxford: Oxford University Press. Chap. 5, pp. 62–75.

Kutuzov, Andrei (2017). "Arbitrariness of Linguistic Sign Questioned: Correlation between Word Form and Meaning in Russian". In:

Lample, Guillaume and Alexis Conneau (2019). "Cross-lingual Language Model Pretraining". In: *CoRR* abs/1901.07291. arXiv: `1901.07291`. URL: `http://arxiv.org/abs/1901.07291`.

Landauer, Thomas K and Susan T. Dumais (1997). "A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge". In: *Psychological Review 1997, Vol. 104.*

Landes, Shari, Claudia Leacock, and Randee I. Tengi (1998). "Building Semantic Concordances". In: *WordNet: An Electronic Lexical Database*. Bradford Books. Chap. 8, pp. 199–216.

Lenci, Alessandro (2018). "Distributional models of word meaning". In: *Annual review of Linguistics* 4, pp. 151–171.

Lenci, Alessandro et al. (May 2021). *A comprehensive comparative evaluation and analysis of Distributional Semantic Models.*

Lesk, Michael (1986). "Automatic Sense Disambiguation Using Machine Readable Dictionaries: How to Tell a Pine Cone from an Ice Cream Cone". In: *Proceed-*

ings of the 5th Annual International Conference on Systems Documentation. SIGDOC '86. Toronto, Ontario, Canada: Association for Computing Machinery, pp. 24–26. ISBN: 0897912241. DOI: 10.1145/318723.318728. URL: https://doi.org/10.1145/318723.318728.

Levy, Omer and Yoav Goldberg (2014a). "Linguistic Regularities in Sparse and Explicit Word Representations". In: *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*. Ann Arbor, Michigan: Association for Computational Linguistics, pp. 171–180. DOI: 10.3115/v1/W14-1618. URL: http://www.aclweb.org/anthology/W14-1618.

— (2014b). "Neural Word Embedding as Implicit Matrix Factorization". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., pp. 2177–2185. URL: http://papers.nips.cc/paper/5477-neural-word-embedding-as-implicit-matrix-factorization.pdf.

Lewis, Mike et al. (July 2020). "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703. URL: https://www.aclweb.org/anthology/2020.acl-main.703.

Li, Bin et al. (2022). *LingJing at SemEval-2022 Task 1: Multi-task Self-supervised Pre-training for Multilingual Reverse Dictionary*.

Li, Shen et al. (July 2018). "Analogical Reasoning on Chinese Morphological and Semantic Relations". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne,

Australia: Association for Computational Linguistics, pp. 138–143. DOI: 10.18653/v1/P18-2023. URL: https://aclanthology.org/P18-2023.

Lin, Chin-Yew (July 2004). "ROUGE: A Package for Automatic Evaluation of Summaries". In: *Text Summarization Branches Out*. Barcelona, Spain: Association for Computational Linguistics, pp. 74–81. URL: https://www.aclweb.org/anthology/W04-1013.

Linzen, Tal (Aug. 2016). "Issues in evaluating semantic spaces using word analogies". In: *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*. Berlin, Germany: Association for Computational Linguistics, pp. 13–18. DOI: 10.18653/v1/W16-2503. URL: https://www.aclweb.org/anthology/W16-2503.

Linzen, Tal, Emmanuel Dupoux, and Yoav Goldberg (2016). "Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies". In: *Transactions of the Association for Computational Linguistics* 4, pp. 521–535. DOI: 10.1162/tacl_a_00115. URL: https://aclanthology.org/Q16-1037.

Lison, Pierre and Jörg Tiedemann (May 2016). "OpenSubtitles2016: Extracting Large Parallel Corpora from Movie and TV Subtitles". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), pp. 923–929. URL: https://aclanthology.org/L16-1147.

Liu, Yinhan et al. (2019). "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *CoRR* abs/1907.11692. arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692.

Loper, Edward and Steven Bird (2002). "NLTK: The Natural Language Toolkit". In: *In Proceedings of the ACL Workshop on Effective Tools and Methodolo-*

gies for Teaching Natural Language Processing and Computational Linguistics. Philadelphia: Association for Computational Linguistics.

Loshchilov, Ilya and Frank Hutter (2019). "Decoupled Weight Decay Regularization". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

Louwerse, Max M. and Rolf A. Zwaan (2009). "Language Encodes Geographical Information". In: *Cognitive Science* 33.1, pp. 51–73. DOI: 10.1111/j.1551-6709.2008.01003.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1551-6709.2008.01003.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1551-6709.2008.01003.x.

Lund, Kevin and Curt Burgess (1996). "Producing high-dimensional semantic spaces from lexical co-occurrence". en. In: *Behavior Research Methods, Instruments, & Computers* 28.2, pp. 203–208. ISSN: 0743-3808, 1532-5970. DOI: 10.3758/BF03204766. URL: http://link.springer.com/article/10.3758/BF03204766 (visited on 09/09/2015).

Malekzadeh, Arman, Amin Gheibi, and Ali Mohades (2021). "PREDICT: Persian Reverse Dictionary". In: *CoRR* abs/2105.00309. arXiv: 2105.00309. URL: https://arxiv.org/abs/2105.00309.

Mantel, Nathan (1967). "The detection of disease clustering and a generalized regression approach". In: *Cancer research* 27.2 Part 1, pp. 209–220.

Marelli, Marco et al. (May 2014). "A SICK cure for the evaluation of compositional distributional semantic models". In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland: European Language Resources Association (ELRA), pp. 216–223. URL:

http://www.lrec-conf.org/proceedings/lrec2014/pdf/363_Paper.pdf.

Martin, Louis et al. (July 2020). "CamemBERT: a Tasty French Language Model". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 7203–7219. URL: https://www.aclweb.org/anthology/2020.acl-main.645.

Méndez, Oscar, Hiram Calvo, and Marco A. Moreno-Armendáriz (2013). "A Reverse Dictionary Based on Semantic Analysis Using WordNet". In: *Advances in Artificial Intelligence and Its Applications*. Ed. by Félix Castro, Alexander Gelbukh, and Miguel González. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 275–285. ISBN: 978-3-642-45114-0.

Merkel, Dirk (2014). "Docker: lightweight linux containers for consistent development and deployment". In: *Linux journal* 2014.239, p. 2.

Michel, Paul, Omer Levy, and Graham Neubig (May 2019). "Are Sixteen Heads Really Better than One?" In: *arXiv e-prints*, arXiv:1905.10650, arXiv:1905.10650. arXiv: 1905.10650 [cs.CL].

Mickus, Timothee, Timothée Bernard, and Denis Paperno (Dec. 2020). "What Meaning-Form Correlation Has to Compose With: A Study of MFC on Artificial and Natural Language". In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, pp. 3737–3749. DOI: 10.18653/v1/2020.coling-main.333. URL: https://aclanthology.org/2020.coling-main.333.

Mickus, Timothee, Mathieu Constant, and Denis Paperno (June 2020). "Génération automatique de définitions pour le français (Definition Modeling in French)".

French. In: *Actes de la 6e conférence conjointe Journées d'Études sur la Parole (JEP, 33e édition), Traitement Automatique des Langues Naturelles (TALN, 27e édition), Rencontre des Étudiants Chercheurs en Informatique pour le Traitement Automatique des Langues (RÉCITAL, 22e édition). Volume 2 : Traitement Automatique des Langues Naturelles*. Nancy, France: ATALA et AFCP, pp. 66–80. URL: https://aclanthology.org/2020.jeptalnrecital-taln.6.

Mickus, Timothee, Mathieu Constant, and Denis Paperno (2021a). "A Game Interface to Study Semantic Grounding in Text-Based Models". In: *Proceedings of the 2021 IEEE Conference on Games*. URL: https://ieee-cog.org/2021/assets/papers/paper_268.pdf.

— (2021b). "About Neural Networks and Writing Definitions". In: *Dictionaries* 42.2.

Mickus, Timothee, Kees van Deemter, et al. (2022). *Semeval-2022 Task 1: CODWOE – Comparing Dictionaries and Word Embeddings*. DOI: 10.48550/ARXIV.2205.13858. URL: https://arxiv.org/abs/2205.13858.

Mickus, Timothee, Denis Paperno, and Mathieu Constant (Sept. 2019). "Mark my Word: A Sequence-to-Sequence Approach to Definition Modeling". In: *Proceedings of the First NLPL Workshop on Deep Learning for Natural Language Processing*. Turku, Finland: Linköping University Electronic Press, pp. 1–11. URL: https://aclanthology.org/W19-6201.

— (2022). *How to Dissect a Muppet: The Structure of Transformer Embedding Spaces*. DOI: 10.48550/ARXIV.2206.03529. URL: https://arxiv.org/abs/2206.03529.

Mickus, Timothee, Denis Paperno, Mathieu Constant, and Kees van Deemter (Jan. 2020). "What do you mean, BERT?" In: *Proceedings of the Society for Com-*

putation in Linguistics 2020. New York, New York: Association for Computational Linguistics, pp. 279–290. URL: https://www.aclweb.org/anthology/2020.scil-1.35.

Mikolov, Tomas, Kai Chen, et al. (Jan. 2013). "Efficient Estimation of Word Representations in Vector Space". In: *Proceedings of Workshop at ICLR* 2013.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig (June 2013). "Linguistic Regularities in Continuous Space Word Representations". In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational Linguistics, pp. 746–751. URL: https://www.aclweb.org/anthology/N13-1090.

Miller, George (1967). "Empirical methods in the study of semantics". In: *Journeys in Science: Small Steps – Great Strides*, pp. 51–73.

Mitchell, Jeff and Mirella Lapata (2010). "Composition in Distributional Models of Semantics". In: *Cognitive Science* 34.8, pp. 1388–1429. DOI: https://doi.org/10.1111/j.1551-6709.2010.01106.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1551-6709.2010.01106.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1551-6709.2010.01106.x.

Mukans, Eduards, Gus Strazds, and Guntis Barzdins (2022). *RIGA at SemEval-2022 Task 1: Scaling Recurrent Neural Networks for CODWOE Dictionary Modeling*.

Navigli, Roberto and Paola Velardi (July 2010). "Learning Word-Class Lattices for Definition and Hypernym Extraction". In: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Uppsala, Sweden:

Association for Computational Linguistics, pp. 1318–1327. URL: https://www.aclweb.org/anthology/P10-1134.

nginx (2021). *nginx*. URL: http://nginx.org/en/.

Nie, Weili, Nina Narodytska, and Ankit Patel (2019). "RelGAN: Relational Generative Adversarial Networks for Text Generation". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=rJedV3R5tm.

Noraset, Thanapon et al. (2017). "Definition Modeling: Learning to define word embeddings in natural language". In: *AAAI*.

Ortiz Suárez, Pedro Javier, Benoît Sagot, and Laurent Romary (2019). "Asynchronous pipelines for processing huge corpora on medium to low resource infrastructures". en. In: ed. by Piotr Bański et al. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-7) 2019. Cardiff, 22nd July 2019. Mannheim: Leibniz-Institut für Deutsche Sprache, pp. 9–16. DOI: 10.14618/ids-pub-9021. URL: http://nbn-resolving.de/urn:nbn:de:bsz:mh39-90215.

Ott, Myle et al. (Oct. 2018). "Scaling Neural Machine Translation". In: *Proceedings of the Third Conference on Machine Translation: Research Papers*. Belgium, Brussels: Association for Computational Linguistics, pp. 1–9. DOI: 10.18653/v1/W18-6301. URL: https://www.aclweb.org/anthology/W18-6301.

Packard, Jerome L (2000). *The morphology of Chinese: A linguistic and cognitive approach*. Cambridge University Press.

Papineni, Kishore et al. (July 2002). "Bleu: a Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Philadelphia, Pennsylvania, USA: Asso-

ciation for Computational Linguistics, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://www.aclweb.org/anthology/P02-1040.

Pawlik, Mateusz and Nikolaus Augsten (Mar. 2015). "Efficient Computation of the Tree Edit Distance". In: *ACM Trans. Database Syst.* 40.1. ISSN: 0362-5915. DOI: 10.1145/2699485. URL: https://doi.org/10.1145/2699485.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (Oct. 2014). "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://aclanthology.org/D14-1162.

Peters, Matthew et al. (June 2018). "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: 10.18653/v1/N18-1202. URL: https://www.aclweb.org/anthology/N18-1202.

Pimentel, Tiago et al. (July 2019). "Meaning to Form: Measuring Systematicity as Information". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 1751–1764. DOI: 10.18653/v1/P19-1171. URL: https://www.aclweb.org/anthology/P19-1171.

PostgreSQL Global Development Group (2021). *PostgreSQL*. Version 14.1. URL: https://www.postgresql.org/.

Pruthi, Danish et al. (July 2020). "Learning to Deceive with Attention-Based Explanations". In: *Proceedings of the 58th Annual Meeting of the Association for*

Computational Linguistics. Online: Association for Computational Linguistics, pp. 4782–4793. DOI: 10.18653/v1/2020.acl-main.432. URL: https://aclanthology.org/2020.acl-main.432.

Qi, Fanchao et al. (2020). "WantWords: An Open-source Online Reverse Dictionary System". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 175–181.

R Core Team (2018). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. URL: http://www.R-project.org/.

Radford, Alec (2018). *Improving Language Understanding by Generative Pre-Training*.

Radford, Alec et al. (2019). *Language Models are Unsupervised Multitask Learners*.

Raganato, Alessandro and Jörg Tiedemann (Nov. 2018). "An Analysis of Encoder Representations in Transformer-Based Machine Translation". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, pp. 287–297. DOI: 10.18653/v1/W18-5431. URL: https://aclanthology.org/W18-5431.

Rajpurkar, Pranav, Robin Jia, and Percy Liang (July 2018). "Know What You Don't Know: Unanswerable Questions for SQuAD". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 784–789. DOI: 10.18653/v1/P18-2124. URL: https://aclanthology.org/P18-2124.

Rasmussen, Carl Edward and Christopher K. I. Williams (2006). *Gaussian Processes for Machine Learning*. The MIT Press.

Řehůřek, Radim and Petr Sojka (May 2010). "Software Framework for Topic Modelling with Large Corpora". English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. http://is.muni.cz/publication/884893/en. Valletta, Malta: ELRA, pp. 45–50.

Reisinger, Joseph and Raymond Mooney (Jan. 2010). "Multi-Prototype Vector-Space Models of Word Meaning." In: pp. 109–117.

Ren, Yi et al. (2020). "Compositional languages emerge in a neural iterated learning model". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HkePNpVKPB.

Rogers, Anna, Aleksandr Drozd, and Bofang Li (Aug. 2017). "The (too Many) Problems of Analogical Reasoning with Word Vectors". In: *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*. Vancouver, Canada: Association for Computational Linguistics, pp. 135–148. DOI: 10.18653/v1/S17-1017. URL: https://www.aclweb.org/anthology/S17-1017.

Rogers, Anna, Olga Kovaleva, and Anna Rumshisky (2020). "A Primer in BERTology: What We Know About How BERT Works". In: *Transactions of the Association for Computational Linguistics* 8, pp. 842–866. DOI: 10.1162/tacl_a_00349. URL: https://aclanthology.org/2020.tacl-1.54.

Rousseeuw, Peter (Nov. 1987). "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis". In: *J. Comput. Appl. Math.* 20.1, pp. 53–65. ISSN: 0377-0427. DOI: 10.1016/0377-0427(87)90125-7. URL: http://dx.doi.org/10.1016/0377-0427(87)90125-7.

Rubenstein, Herbert and John Goodenough (Oct. 1965). "Contextual correlates of synonymy". In: *Commun. ACM* 8, pp. 627–633. DOI: 10.1145/365628.365657.

Russell, Lindsay Rose (2021). "Dictionary Boycotts and the Power of Popular (Re)Definition". In: *Dictionaries: Journal of the Dictionary Society of North America* 42.1, pp. 235–247. DOI: 10.1353/dic.2021.0013.

Sahlgren, Magnus (2008). "The distributional hypothesis". In: *Italian Journal of Linguistics* 20.1, pp. 33–54.

Salton, G., A. Wong, and C. S. Yang (Nov. 1975). "A Vector Space Model for Automatic Indexing". In: *Commun. ACM* 18.11, pp. 613–620. ISSN: 0001-0782. DOI: 10.1145/361219.361220. URL: http://doi.acm.org/10.1145/361219.361220.

Saussure, Ferdinand de (1916). *Cours de linguistique générale*. 1995th ed. Paris: Payot & Rivage.

Schluter, Natalie (June 2018). "The Word Analogy Testing Caveat". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 242–246. DOI: 10.18653/v1/N18-2039. URL: https://www.aclweb.org/anthology/N18-2039.

Searle, John R. (1980). "Minds, brains, and programs". In: *Behavioral and Brain Sciences* 3, pp. 417–424.

Sérasset, Gilles (May 2012). "Dbnary: Wiktionary as a LMF based Multilingual RDF network". In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey: European Lan-

guage Resources Association (ELRA), pp. 2466–2472. URL: http://www.lrec-conf.org/proceedings/lrec2012/pdf/387_Paper.pdf.

Serrano, Sofia and Noah A. Smith (July 2019). "Is Attention Interpretable?" In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 2931–2951. DOI: 10.18653/v1/P19-1282. URL: https://aclanthology.org/P19-1282.

Shaw, Peter, Jakob Uszkoreit, and Ashish Vaswani (2018). "Self-Attention with Relative Position Representations". In: *CoRR* abs/1803.02155. arXiv: 1803.02155. URL: http://arxiv.org/abs/1803.02155.

Shaw, Ryan et al. (2013). "Building a scalable database-driven reverse dictionary". In: *IEEE Transactions on Knowledge and Data Engineering* 25.3, pp. 528–540.

Siddique, Bushra and Mirza Mohd Sufyan Beg (2019). "A Review of Reverse Dictionary: Finding Words from Concept Description". In: *Next Generation Computing Technologies on Computational Intelligence*. Ed. by Manish Prateek et al. Singapore: Springer Singapore, pp. 128–139. ISBN: 978-981-15-1718-1.

Snoek, Jasper, Hugo Larochelle, and Ryan P Adams (2012). "Practical Bayesian Optimization of Machine Learning Algorithms". In: *Advances in Neural Information Processing Systems*. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf.

Spala, Sasha et al. (Dec. 2020). "SemEval-2020 Task 6: Definition Extraction from Free Text with the DEFT Corpus". In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for

Computational Linguistics, pp. 336–345. URL: https://aclanthology.org/2020.semeval-1.41.

Spike, Matthew John (2016). "Minimal requirements for the cultural evolution of language". en. Ph.D. Thesis. Edinburgh, Scotland, UK.: University of Edinburgh. URL: https://era.ed.ac.uk/handle/1842/25930.

Srivastava, Aditya and Vemulapati Harsha Vardhan (2022). *TLDR at SemEval-2022 Task 1: Using Transformers to Learn Dictionaries and Representations.*

Sterkenburg, Piet van (2003). "'The' dictionary: Definition and history". In: *A Practical Guide to Lexicography*. Ed. by Piet van Sterkenburg. Amsterdam/Philadelphia: John Benjamins Publishing Company. Chap. 1, pp. 3–17.

Strauss, Benjamin et al. (Dec. 2016). "Results of the WNUT16 Named Entity Recognition Shared Task". In: *Proceedings of the 2nd Workshop on Noisy User-generated Text (WNUT)*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 138–144. URL: https://aclanthology.org/W16-3919.

Stump, Gregory (1998). "The handbook of morphology". In: ed. by A. Spencer & A. M. Zwicky. Oxford: Blackwell. Chap. Inflection. Pp. 13–43.

Swinger, Nathaniel et al. (2018). "What are the biases in my word embedding?" In: *CoRR* abs/1812.08769. arXiv: 1812.08769. URL: http://arxiv.org/abs/1812.08769.

Tarp, Sven (2017). "The Concept of Dictionary". In: *The Routledge Handbook of Lexicography*. Ed. by Pedro A. Fuertes-Olivera. Abingdon/New York: Routledge. Chap. 15, pp. 237–249.

Taylor, Wilson (1953). "Cloze Procedure: A New Tool for Measuring Readability". In: *Journalism Quarterly* 30, pp. 415–433.

The GIMP Development Team (June 12, 2019). *GIMP*. Version 2.10.12. URL: https://www.gimp.org.

Thorat, Sushrut and Varad Choudhari (Dec. 2016). "Implementing a Reverse Dictionary, based on word definitions, using a Node-Graph Architecture". In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 2797–2806. URL: https://aclanthology.org/C16-1263.

Timkey, William and Marten van Schijndel (Nov. 2021). "All Bark and No Bite: Rogue Dimensions in Transformer Language Models Obscure Representational Quality". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, pp. 4527–4546. URL: https://aclanthology.org/2021.emnlp-main.372.

Tissier, Julien, Christophe Gravier, and Amaury Habrard (Sept. 2017). "Dict2vec : Learning Word Embeddings using Lexical Dictionaries". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, pp. 254–263. DOI: 10.18653/v1/D17-1024. URL: https://aclanthology.org/D17-1024.

Tjong Kim Sang, Erik F. and Fien De Meulder (2003). "Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition". In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pp. 142–147. URL: https://aclanthology.org/W03-0419.

Tran, Thi Hong Hanh et al. (2022). *JSI at SemEval-2022 Task 1: CODWOE - Reverse Dictionary: Monolingual and cross-lingual approaches*.

Vaswani, Ashish et al. (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon et al. Curran Associates, Inc., pp. 5998–6008. URL: http://papers.nips.cc/paper/7181-attention-is-all-you-need.

Vendramin, Lucas, Pablo A. Jaskowiak, and Ricardo J. G. B. Campello (2013). "On the Combination of Relative Clustering Validity Criteria". In: *Proceedings of the 25th International Conference on Scientific and Statistical Database Management*. SSDBM. Baltimore, Maryland, USA: ACM, 4:1–4:12. ISBN: 978-1-4503-1921-8. DOI: 10.1145/2484838.2484844. URL: http://doi.acm.org/10.1145/2484838.2484844.

Vial, Loïc, Benjamin Lecouteux, and Didier Schwab (2019). "Sense Vocabulary Compression through the Semantic Knowledge of WordNet for Neural Word Sense Disambiguation". In: *Global Wordnet Conference*. Wroclaw, Poland. URL: https://hal.archives-ouvertes.fr/hal-02131872.

Vincent-Lamarre, Philippe et al. (2016). "The Latent Structure of Dictionaries". In: *Topics in Cognitive Science* 8.3, pp. 625–659. DOI: https://doi.org/10.1111/tops.12211. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/tops.12211. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/tops.12211.

Voita, Elena et al. (July 2019). "Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 5797–5808. DOI:

10.18653/v1/P19-1580. URL: https://www.aclweb.org/anthology/P19-1580.

Vulić, Ivan et al. (Nov. 2020). "Probing Pretrained Language Models for Lexical Semantics". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 7222–7240. DOI: 10.18653/v1/2020.emnlp-main.586. URL: https://aclanthology.org/2020.emnlp-main.586.

Wang, Alex et al. (2019). "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding". In: In the Proceedings of ICLR.

Wang, Zhiyong, Ge Zhang, and Nineli Lashkarashvili (2022). *1cademy at SemEval-2022 Task 1: Investigating the Effectiveness of Multilingual, Multitask, and Language-Agnostic Tricks for the Reverse Dictionary Task*.

Westera, Matthijs and Gemma Boleda (23–27 5 2019). "Don't Blame Distributional Semantics if it can't do Entailment". In: *Proceedings of the 13th International Conference on Computational Semantics - Long Papers*. Gothenburg, Sweden: Association for Computational Linguistics, pp. 120–133. URL: https://www.aclweb.org/anthology/W19-0410.

White, Anne, Barbara C. Malt, and Gert Storms (2017). "Convergence in the Bilingual Lexicon: A Pre-registered Replication of Previous Studies". In: *Frontiers in Psychology* 7, p. 2081. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2016.02081. URL: https://www.frontiersin.org/article/10.3389/fpsyg.2016.02081.

Wiedemann, Gregor et al. (2019). "Does BERT Make Any Sense? Interpretable Word Sense Disambiguation with Contextualized Embeddings". In: *ArXiv* abs/1909.10430.

Wiegreffe, Sarah and Yuval Pinter (Nov. 2019). "Attention is not not Explanation". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 11–20. DOI: 10.18653/v1/D19-1002. URL: https://aclanthology.org/D19-1002.

Wieting, John and Douwe Kiela (2019). "No Training Required: Exploring Random Encoders for Sentence Classification". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=BkgPajAcY7.

Wittgenstein, Ludwig (1953). *Philosophical Investigations*. Oxford: Basil Blackwell. ISBN: 0631119000.

Yan, Hang et al. (Nov. 2020). "BERT for Monolingual and Cross-Lingual Reverse Dictionary". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, pp. 4329–4338. DOI: 10.18653/v1/2020.findings-emnlp.388. URL: https://aclanthology.org/2020.findings-emnlp.388.

Yang, Liner et al. (2019). *Incorporating Sememes into Chinese Definition Modeling*. arXiv preprint : 1905.06512. arXiv: 1905.06512. URL: http://arxiv.org/abs/1905.06512.

Yang, Zhilin et al. (2019). "XLNet: Generalized Autoregressive Pretraining for Language Understanding". In: *CoRR* abs/1906.08237. arXiv: 1906.08237. URL: http://arxiv.org/abs/1906.08237.

Yun, Zeyu et al. (June 2021). "Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors". In: *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on*

Knowledge Extraction and Integration for Deep Learning Architectures. Online: Association for Computational Linguistics, pp. 1–10. DOI: 10.18653/v1/2021.deelio-1.1. URL: https://aclanthology.org/2021.deelio-1.1.

Zanzotto, Fabio Massimo et al. (Aug. 2010). "Estimating Linear Models for Compositional Distributional Semantics". In: *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Beijing, China: Coling 2010 Organizing Committee, pp. 1263–1271. URL: https://aclanthology.org/C10-1142.

Zaslavsky, Noga et al. (2018). "Efficient compression in color naming and its evolution". In: *Proc. Natl. Acad. Sci. USA* 115.31, pp. 7937–7942. DOI: 10.1073/pnas.1800521115. URL: https://doi.org/10.1073/pnas.1800521115.

Zhang, Haitong et al. (Dec. 2019). *Improving Interpretability of Word Embeddings by Generating Definition and Usage*. arXiv preprint : 1912.05898. arXiv: 1912.05898 [cs.CL].

Zhang, Lei et al. (2020). "Multi-channel reverse dictionary model". In: *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 312–319.

Zhao, Wei et al. (Nov. 2019). "MoverScore: Text Generation Evaluating with Contextualized Embeddings and Earth Mover Distance". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, pp. 563–578. DOI: 10.18653/v1/D19-1053. URL: https://aclanthology.org/D19-1053.

Zhu, Yukun et al. (2015). "Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books". In: *Proceedings of the*

2015 IEEE International Conference on Computer Vision (ICCV). ICCV '15. USA: IEEE Computer Society, pp. 19–27. ISBN: 9781467383912. DOI: 10.1109/ICCV.2015.11. URL: https://doi.org/10.1109/ICCV.2015.11.

Ziemski, Michał, Marcin Junczys-Dowmunt, and Bruno Pouliquen (May 2016). "The United Nations Parallel Corpus v1.0". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), pp. 3530–3534. URL: https://aclanthology.org/L16-1561.

IV

Appendices

A

Bayesian Optimization

Please, teacher teacher! Leave us alone
As we accept life lessons from no one

— Jinjer, *Teacher, Teacher!*

Bayesian Optimization is an optimization algorithm—i.e., an algorithm that seeks the optimal parameters for some given input function—tailored towards objective functions that are expensive to compute. It has been applied to select hyperparameters for neural networks (Snoek et al., 2012), and is therefore an alternative approach to random search and exhaustive grid search. The presentation below is based on the tutorial by Frazier (2018), to which we refer the reader for a more in-depth presentations as well as additional material.

In essence, when selecting hyperparameters with Bayesian Optimization, we repeat the following sequence of instructions:

1. compute what our prior beliefs are, given the previous hyperparameter configurations we have sampled and the performances they yielded;

2. select the next hyperparameter configuration according to our prior beliefs;

3. observe the actual performance of this latest configuration, and update our beliefs accordingly.

More formally, Bayesian Optimization consists in iteratively selecting *parameters* that we can expect to improve the current optimal value, using Bayes' rule. When applied to hyperparameter selection for a neural architecture $\mathscr{A}$, the function that we will optimize consists in taking some configuration of hyperparameters H as input, training a model of architecture $\mathscr{A}$ with the hyperparameters H, and outputing a scalar value representing the performance reached by said model. This scalar output value can be the loss of the model on a validation split or any other metric of its performance. This function to optimize is more formally called the *objective function.* Crucially, Bayesian Optimization does not require the objective function to be differentiable, hence we can use to select the optimal hyperparameters for any scalar measure of model performance.

Successful applications require having few parameters to set, therefore Bayesian Optimization is not applicable for a direct estimation of neural network parameters. It also assumes that all parameters to set take real values and that the objective function isn't noisy—although it is possible to relax both of these assumptions. On the other hand, Bayesian Optimization focuses on finding a global optimum under a minimum budget of objective function evaluations, which is why it is appropriate for hyperparameter selection, as training a neural network can be costly.

An overview of Bayesian Optimization is shown in Algorithm 1. Given the

Algorithm 1: Bayesian optimization

Data: f, objective function to optimize;
N, total number of samples of f to perform, $N > 0$;
n, number of samples of f to compute prior, $N > \mathrm{n} > 0$;
I, valid input space for f, $I \subset \mathbb{R}^h$;
Acq, an acquisition function.

Let Obs $\leftarrow$ [ ];
while $n \neq 0$ **do**
  Randomly sample $x \sim I$;
  Append $f(x)$ to Obs;
  Decrement n and N;
end
while $N \neq 0$ **do**
  Construct the posterior distribution $\mathscr{D}$ on f using Obs;
  $x^* \leftarrow \underset{x}{\operatorname{argmax}}\, \mathrm{Acq}(x, \mathscr{D}, \mathrm{Obs})$;
  Append $f(x^*)$ to Obs;
  Decrement N;
end
return optimum value in Obs;

assumption that all the parameters to set are real scalars, we can consider f as a function that takes a vector in input and outputs a scalar, that is to say $f : \mathbb{R}^p \to \mathbb{R}$, with p the number of parameters to set.

The two key concepts that Bayesian Optimization relies on are the construction of a posterior distribution $\mathscr{D}$ and the acquisition function Acq. The former is generally done using Gaussian processes: that is to say, we construct a multivariate Gaussian distribution that matches with the observations Obs = $\big(f(x_1),\ \ldots,\ f(x_i)\big)$. This multivariate Gaussian distribution is defined using a *mean function* Φ_μ and a *covariance function* Φ_σ (also known as a "kernel"), both of which are computed using the previously sampled input values $(x_1,\ \ldots,\ x_i)$. In short, we construct:

$$\begin{pmatrix} f(x_1) \\ \vdots \\ f(x_i) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \Phi_\mu(x_1) \\ \vdots \\ \Phi_\mu(x_i) \end{pmatrix}, \begin{bmatrix} \Phi_\sigma(x_1, x_1) & \ldots & \Phi_\sigma(x_1, x_i) \\ \vdots & \ddots & \vdots \\ \Phi_\sigma(x_i, x_1) & \ldots & \Phi_\sigma(x_i, x_i) \end{bmatrix}\right) \tag{A.1}$$

Covariance functions are generally chosen so that they have the property that points closer in the input space yield a higher value. As such, a simple and commonly used covariance function is the *power exponential*:

$$\Phi_\sigma(x_n, x_m) = \alpha \exp\left(-\|x_n, x_m\|_2^2\right) \tag{A.2}$$

Where α is a vector parameter that is adjusted globally so as to match with our previous observations Obs. One can remark that the maximum value of $\exp\left(-\|x_n, x_m\|_2^2\right)$ is attained when $x_n = x_m$. More complex covariance functions

exist, such as the Màtern kernel, which we fit to our observations using the parameters α and ν:

$$\Phi_\sigma(x_n, x_m) = \alpha \frac{2^{1-\nu}}{\Gamma(\nu)} \cdot \left(\sqrt{2\nu}\|x_n - x_m\|_2\right)^\nu \cdot K_\nu(\sqrt{2\nu}\|x_n - x_m\|_2) \tag{A.3}$$

where K_ν is the modified Bessel function of the second kind:

$$K_\nu(z) = \frac{\Gamma\left(\nu + \frac{1}{2}\right)(2z)^\nu}{\sqrt{\pi}} \cdot \int_0^\infty \frac{\cos t}{\left(t^2 + z^2\right)^{\nu+1/2}} dt \tag{A.4}$$

While this second covariance function appears much more complex, it plays essentially the same role of quantifying the similarity between two inputs x_m and x_n.

Mean functions tend to be simpler overall. A frequent common choice is to simply use a constant value that we adjust to match with our observations Obs. The usefulness of this Gaussian process is that we can use it to define what we expect to observe for any future input we might test. As the mean and covariance functions are defined with respect to the previous inputs $(x_1, \ldots, x_i)$, we can apply sample a new input $x_{i+1} \sim I$ and expand the multivariate Gaussian distribution to predict $f(x_{i+1})$, the value our objective function f will take at this next point x_{i+1}. More precisely, we can apply Bayes' rule, as it is defined for multivariate Gaussian distribution:

$$f(x_{i+1}) \mid f(x_1), \ldots, f(x_i) \sim \mathcal{N}(\hat{\Phi}_\mu(x), \hat{\Phi}_\sigma(x))$$

$$
\begin{aligned}
A &= \begin{pmatrix} \Phi_\sigma(x_{i+1}, x_1) \\ \vdots \\ \Phi_\sigma(x_{i+1}, x_i) \end{pmatrix} \cdot \begin{bmatrix} \Phi_\sigma(x_1, x_1) & \dots & \Phi_\sigma(x_1, x_i) \\ \vdots & \ddots & \vdots \\ \Phi_\sigma(x_i, x_1) & \dots & \Phi_\sigma(x_i, x_i) \end{bmatrix}^{-1} \\
\hat{\Phi}_\mu(x) &= A \cdot \begin{pmatrix} f(x_1) - \Phi_\mu(x_1) \\ \vdots \\ f(x_i) - \Phi_\mu(x_i) \end{pmatrix} + \Phi_\mu(x_{i+1}) \\
\hat{\Phi}_\sigma(x) &= \Phi_\sigma(x_{i+1}, x_{i+1}) - A \cdot \begin{pmatrix} \Phi_\sigma(x_1, x_{i+1}) \\ \vdots \\ \Phi_\sigma(x_i, x_{i+1}) \end{pmatrix}
\end{aligned} \tag{A.5}
$$

We refer the reader to textbooks on Gaussian processes, such as Rasmussen and Williams (2006).

This is also where the second key element comes into play, namely the acquisition function Acq. This function determines the input point we sample next: as shown on line 9 of Algorithm 1, the next observation $f(x^*)$ corresponds to the point x^* which maximizes this Acq function, given the distribution $\mathscr{D}$ (as modeled through a Gaussian process).

The simplest and most commonly discussed acquisition function is the *Expected Improvement* function:

$$
\mathrm{Acq}(x, \mathscr{D}, \mathrm{Obs}) = \mathbb{E}\left(\max\left(f(x) - \max \mathrm{Obs}\right),\ 0 | \mathscr{D}\right) \tag{A.6}
$$

Here, the expectation with respect to $\mathscr{D}$ is given by Equation (A.5). Simply put, the Expected Improvement function computes the expectation at point x of exceeding the previous optimum value $\max \mathrm{Obs}$. Again, other acquisition functions

can be defined. In particular, it might be useful to explore the input space, that is to say, to sample points where we are less certain of what the objective function will yield, which should correspond to a high variance in our constructed distribution.

B

Analogy Dataset translated from BATS

Stranger fruit
Got holes in flesh
But it ain't gonna spoil
'Cause it never was fresh

— Zeal & Ardor, *Stranger Fruit*

One of the contributions of this work consists of the introduction of translations of the Balanced Analogy Test Set (Gladkova et al., 2016, BATS) for Dutch, French, German, Italian, Mandarin, and Spanish. This analogy benchmark is structured in two levels: individual sub-sections instantiating specific analogical relations (e.g., "*animal—young*" or "*infinitive—past participle*") are then grouped into four super-sections: Inflection, Derivation, Lexicography, Encyclopedia. The former two correspond to morphological relations, such as the relation between two inflected forms of a word or the relation between a verb and the corresponding agent noun, whereas the two latter are more closely aligned to common-sense reasoning, and include relations such as synonymy or the relation between the

We are highly indebted to Eduardo Caló & Léo Jacqmin in the production of these resources.

name of a country and that of its capital city. The original resource by Gladkova et al. (2016) emphasizes balance by ensuring that each of the four super-sections contains exactly 10 sub-sections, and that each of the 10 sub-sections contains exactly 50 instances of the same analogical relation. Models are tested exhaustively on all possible pairs of instances within each sub-category. We direct the reader to the original paper by Gladkova et al. (2016) for a more thorough overview. To create these resources, direct translations from the original English version were taken as starting point before performing language-specific adaptations.

Dutch. The encyclopedic semantic section E03 was localized using Dutch *provincies* and their capital cities.

French. The inflectional morphology section I03 was replaced with gender inflection of adjectives since comparatives are realized using periphrastic constructions (e.g., *jolie* 'cute', *plus jolie* 'cuter'). The derivational morphology section D01 was replaced with denominal adjectives using the suffix *-el*, as the formation of privatives using suffixes is not a productive morphological operation. The encyclopedic semantic section E03 was localized using a random selection of 50 French *départements* and their capital cities, barring those that would be tokenized as MWE.

German. The encyclopedic semantic section E03 was localized using German *Länder* and their capital cities.

Italian. The inflectional morphology section I03 was replaced with gender inflection of adjectives since in Italian too comparatives are realized using pe-

riphrastic constructions (e.g., *bella* 'cute', *più bella* 'cuter'). The derivational morphology section D01 was replaced with noun diminutives using the suffixes *-ino*, *-ina*, for the same reason as in French. The encyclopedic semantic section E03 was localized using Italian *regioni* and their capital cities.

Mandarin. Given the typological differences with English, we removed the whole section concerning inflectional morphology and completely reshaped the one on derivational morphology. In particular, given that derivation by means of affixes is a very productive process (Packard, 2000), we selected eight affixes, namely -度 *-dù* '-ness/-ity', -化 *-huà* '-ize', -性 *-xìng* '-ness/-ity', -学 *-xué* '-ology', -主义 *-zhǔyì* '-ism', -儿 *-r* 'prosodic suffix', -机 *-jī* 'instrument', 小- *xiǎo-* 'diminutive prefix/small/young', and created corresponding categories. We set the focus of D09 on agent formation from verbs, much as D08 in all other languages, whereas for D10 we took inspiration from S. Li et al. (2018) focusing on reduplication of monosyllabic verbs having "a bit" as semantic nuance. In the lexicographic semantic section, we exploited the so-called "elastic words" (Guo, 1938; Duanmu, 2007) to build L08. We filled it using the list of elastic words in the Appendix of Dong (2015), focusing only on free monomorphemic adjectives and their corresponding long forms. The encyclopedic semantic section E03 was localized using Chinese 省 *shěng* and their capital cities. We incorporated the original E06 in D08 and replaced it with a category on nouns and their respective classifiers, disregarding the general classifier 个 *gè* that is not semantically informative.

Spanish. The inflectional morphology section I03 was replaced with gender inflection of adjectives since also in Spanish comparatives are realized using periphrastic constructions (e.g., *linda* 'cute', *más linda* 'cuter'). The derivational morphology section D01 was replaced with noun diminutives using the suffixes *-ito*, *-ita*, for the same reasons as in French and Italian. The encyclopedic semantic section E03 was localized using Spanish *comunidades autónomas* and their capital cities.

Cross-linguistically, when direct translations sounded unnatural or were not compliant with the patterns of the categories, we deleted the pair, and where applicable, more than one correct answer was supplied. Eventually, we removed multi-word expressions and duplicates when appearing in translations. Finally, all the categories, except E03, were padded to 50 entries following the morphological or semantic pattern of each category. After completing the translation process, we obtained 11,328 pairs overall. Resources will be made publicly available upon acceptation of the paper. An overview of our BATS translations with examples and figures can be found in Tables B.1 and B.2.

Rel	de	es	fr	it	nl	zh
I01	*Tag : Tage*	*día : dias*	*jour : jours*	*dio : dèi*	*rol : rollen*	✗
I02	*Rat : Räte*	*voz : voces*	*bail : baux*	*base : basi*	*vlo : vlooien*	✗
I03	*süß : süßer*	*barato : barata*	*chanceux : chanceuse*	*colto : colta*	*oud : ouder*	✗
I04	*rein : reinste*	*feo : feísimo*	*drôle : drôlissime*	*duro : durissimo*	*rijk : rijkst*	✗
I05	*hören : hört*	*crear : crea*	*dire : dit*	*godere : gode*	*vraagen : vraagt*	✗
I06	*teilnehmen : teilnehmend*	*creer : creyendo*	*gérer : gérant*	*gestire : gestendo*	*leren : lerend*	✗
I07	*sehen : gesehen*	*decir : dicho*	*croire : cru*	*perdere : perso*	*hoor : gehoord*	✗
I08	*glaubend : glaubt*	*girando : gira*	*lisant : lit*	*succedendo : succede*	*gaand : gaat*	✗
I09	*fragend : gefragt*	*uniendo : unido*	*ratant : raté*	*capendo : capito*	*vragend : gevraagd*	✗
I10	*wird : geworden*	*ejecuta : ejecutado*	*suit : suivi*	*sente : sentito*	*volgt : gevolgd*	✗
D01	*Arm : armlos*	*cabeza : cabecita*	*culture : culturel*	*stella : stellina*	*ego : egoloos*	强: 强度
D02	*fähig : unfähig*	*edito : inédito*	*pair : impair*	*certo : incerto*	*zeker : onzeker*	国际: 国际化
D03	*Kind : kindlich*	*real : realmente*	*fort : fortement*	*ampio : ampiamente*	*feest : feestelijk*	重要: 重要性
D04	*mäßig : übermäßig*	*poblado : sobrepoblado*	*aigu : suraigu*	*umano : sovrumano*	*vol : overvol*	语言: 语言学
D05	*fest : Festigkeit*	*fijo : fijeza*	*fou : folie*	*raro : rarità*	*vast : vastheid*	自由: 自由主义
D06	*geben : wiedergeben*	*mandar : remandar*	*lire : relire*	*spedire : rispedire*	*bouwen : herbouwen*	虫: 虫儿
D07	*haften : haftbar*	*evitar : evitable*	*jeter : jetable*	*vivere : vivibile*	*eeten : eetbaar*	打火: 打火机
D08	*tun : Täter*	*diseñar : diseñador*	*tuer : tueur*	*gestire : gestore*	*boksen : bokser*	孩子: 小孩子
D09	*reduzieren : Reduktion*	*acusar : acusación*	*priver : privation*	*mutare : mutazione*	*inspireren : inspiratie*	开发: 开发员
D10	*erklären : Erklärung*	*elevar : elevamiento*	*licencier : licenciement*	*pagare : pagamento*	*verklaren : verklaring*	想: 想想

Table B.1: BATS translations: examples and numbers per category

Rel	de	es	fr	it	nl	zh
L01	*Kuh : Wirbeltier/...*	*ganso : pájaro/...*	*caille : vertébré/...*	*ape : insetto/...*	*coyote : carnivoor/...*	猫头鹰: 鸟/...
L02	*Foto : Bild/...*	*sofá : mueble/...*	*bureau : objet/...*	*pompelmo : frutto/...*	*jas : eenheid/...*	架: 家具/...
L03	*Boot : Post/...*	*color : blanco/...*	*mois : décembre/...*	*canzone : inno/...*	*tasse : gral/...*	-甜点: 蛋糕/...
L04	*Bart : Haar*	*agua : oxígeno/...*	*océan : eau*	*neve : acqua/...*	*staal : ijzer/...*	旗: 纸/...
L05	*Kalb : Vieh/...*	*cantante : coro/...*	*juré : jury*	*pecora : gregge*	*kal : vee/...*	鹅: 群
L06	*Byte : Bit*	*guitarra : cuerda/...*	*film : épisode/...*	*corpo : petto/...*	*euro : cent*	门: 铰链/...
L07	*ängstlich : entsetzt/...*	*amar : adorar/...*	*poney : cheval*	*triste : depresso/...*	*aap : gorilla*	湿: 浸泡/...
L08	*Fahrrad : Rad*	*madre : mamá*	*marché : bazar*	*roccia : sasso*	*vader : papa*	勇: 勇敢
L09	*heiß : frostig/...*	*claro : oscuro*	*sec : humide/...*	*sano : pazzo/...*	*jong : gaga/...*	甜: 酸/...
L10	*tot : lebendig*	*sucio : limpio*	*chute : montée*	*dopo : prima*	*west : oost*	内: 外
E01	*Lima : Peru*	*Bagdad : Irak*	*Damas : Syrie*	*Kiev : Ucraina*	*Zagreb : Kroatië*	安曼: 约旦
E02	*Iran : Persisch*	*Camboya : jemer*	*Égypte : arabe*	*Marocco : berbero/...*	*Cuba : Spaans*	伯利兹: 英语
E03	*München : Bayern* (13)	*Barcelona : Cataluña* (11)	*Nîmes : Gard* (50)	*Roma : Lazio* (17)	*Maastricht : Limburg* (10)	西安: 陕西 (27)
E04	*Marx : Deutsch*	*Homero : griego*	*Tolstoi : russe*	*Pascal : francese*	*Hegel : Duits*	孟子: 中国
E05	*Dante : Dichter*	*Depp : actor/...*	*Lincoln : président*	*Hawking : fisico/...*	*Locke : filosoof*	孔子: 哲学家
E06	*Ente : Küken*	*cigüeña : cigoñino*	*daim : faon*	*ape : larva*	*eend : eendje/...*	筷子: 双/...
E07	*Kuh : muhen*	*lobo : aúlla*	*hyène : rire*	*cane : abbaiare*	*ezel : balken/...*	猫: 喵/...
E08	*Wal : Meer/...*	*castor : río*	*bovin : étable*	*corvo : nido/...*	*beer : kooi/...*	狐狸: 洞穴
E09	*Kirsch : rot/...*	*peonía : roja/...*	*sel : blanc*	*tè : nero/...*	*bloed : rood*	蚂蚁: 黑色/...
E10	*Stier : Kuh*	*niño : niña*	*roi : reine*	*leone : leonessa*	*opa : oma*	老公: 老婆
Tot	1,963	1,961	2,000	1,967	1,960	1,477

Table B.2: BATS translations: examples and numbers per category (continued)

C

Instructions Provided to Annotators

Now you do what they told ya

— Rage Against The Machine, *Killing in the Name*

At multiple stages during the conception and analysis of the Blankcrack project, we relied on annotators. We collect here the annotation guidelines, instructions, etc., for ease of consultation.

C.1 BlankCrack Pilot Study, Adversarial Word Pair Submissions

I'm looking for English speakers that would have a bit of time to spare.

I'm looking into collecting some data for my thesis. It's probably going to turn into a web-based application broadly open to the public and set up as a game, but before I actually set things in motion and ask my supervisors to fund this, I need to conduct a pilot study.

Typically, I'm interested in how feasible it is to guess the meaning of a word based on its context. Annotators/players will be presented with a set of sentences, all of which containing some word X that will be blanked out, and will then be tasked with selecting which of two definitions is most likely to apply to the blanked out word. Here's an example:

Given the contexts:

- At one point they told me the _____ had been fixed, the next day it hadn't.
- There is already a little _____ house that I would turn into a little rabbit hutch being kept back there and I was thinking about keeping the rabbit in the run with the chickens, but keeping the chickens in the coop and the rabbit in the hutch at night.
- This type of harness goes around the _____s chest and legs and has a clip for the leash up on the back of the _____ in middle of shoulder blades.
- You say you work a lot, and that you have a young _____; so I have little doubt that your _____ is just filled with energy to burn; and it is good of you to look for a place to take him.
- went in there and got my _____ groomed came home to an uneven _____ then took him back to get evened up what a mistake!

which definition is the most likely for the blanked out word?

1. A mammal, that has been domesticated for thousands of years, of highly variable appearance due to human breeding.
2. A domesticated subspecies of feline animal, commonly kept as a house pet.

I'm currently working on how to gather word pairs. One possibility would be to have annotators/players submit word pairs that other annotators/players would then have to disambiguate. I would like to include this as part of the pilot study, and for this I need people to come up with some word pairs they think will be hard to guess. If you have the time, can you come up with 5-ish word pairs that I'd be able to test in this pilot study?

C.2 BlankCrack Pilot Study, On-Screen Instructions

This is a pilot survey!

Below and on the next page are 30 questions. Each question is based on a word pair; we randomly selected 5 sentences where one of the two words occurred and the other didn't.

Can you guess which word we blanked out?

You are required to give a guess for each question, i.e., *you may not leave a blank choice.* If you think the two words always mean the same thing, make your guess nonetheless and check the tickbox at the end of each question.

Upon completing the survey, you should be redirected to a page confirming your answers have been processed. If nothing happens upon clicking the "Submit" button, make sure you've made a guess for each word pairs, even those you judge to be synonyms.

Lastly, you will find 2 optional questions for feedback at the very end. Don't hesitate!

Data collected through this survey will serve academic and/or educational purposes only. No personal data is being collected.

C.3 BlankCrack Online Game, On-Screen Instructions

C.3.1 (a)-annotation instructions

The fiendish Tippesk has issued forth a challenge! The riddle is made of 2 words and *k* sentences. One of these words has been blanked out from all the sentences; the other actually never occurs in any of them.

Let's show them we can crack this in no time!

C.3.2 (b)-annotation instructions

Heed my word, minions of Tippesk!

Give me two words that those pesky squirrels won't be able to tell apart once blanked out. With this word pair, I shall craft riddles and torment them. Avoid synonyms! I wish them to despair…

C.4 BlankCrack Online Game, Instructions for Re-Annotation of BERT-Selected Contexts

I need people to go through a closed-format "fill-in-the-gaps" questionnaire.

You'll see one sentence at a time, where one word has been manually blanked out.

Here's an example in French:

```
bien entendu, une hirondelle ne fait pas le _____, mais le belarus
est un grand état important, un voisin de l'union européenne et nous
```

`sommes évidemment ravis de toute évolution positive.`

The interface will propose two possible words: in this French example, these two words are automne and printemps.

We ask you to indicate which of the two words you believe was in the original version of this sentence, before it was replaced with a blank space.

Rather than a hard binary choice, we want you to state how confident you are in your answer. Continuing with our French example, you can select any of the five following answers:

- You think it's very likely printemps
- You think it's perhaps printemps
- You don't know
- You think it's perhaps automne
- You think its very likely automne

Behind the scenes, the website will convert your responses into annotations on a 5 point Likert scale, ranging from "high confidence in the attested word" to "high confidence in the unattested word."

The sentences & word pairs weren't selected at random: we used some Natural Language Processing software to do that. Your responses will provide us with information to test the value of that software as a linguistic model.

D

Illustrations, pictures, visual supports

Legends spoke of the ancient monster
From a time beyond the dawn of time
In a pit of primordial ooze
Many years we have been kept waiting
But tonight that squid will surely die
Revenge is a dish best served fried
Deep fried

— Alestorm, *Death Throes of the Terrorsquid*

The following images were not included in the main text, and are collected here instead to reward the happy few who made it through to this appendix. Illustrations listed here, as well as those presented in Chapter 2 and the CoDWoE logo (Figure 4.5) were realized by the author of this dissertation. In all cases, rough drafts were first realized by hand, and finalized and colorized using GIMP (The GIMP Development Team, 2019). The leaflets in Figure D.11 and Figure D.12 were used as advertisement material at the *Forum des Sciences Cognitives et du TAL à NANCY 2021* and the *Salon 360 Grand Est 2021*.

Figure D.1: BlankCrack website banner picture

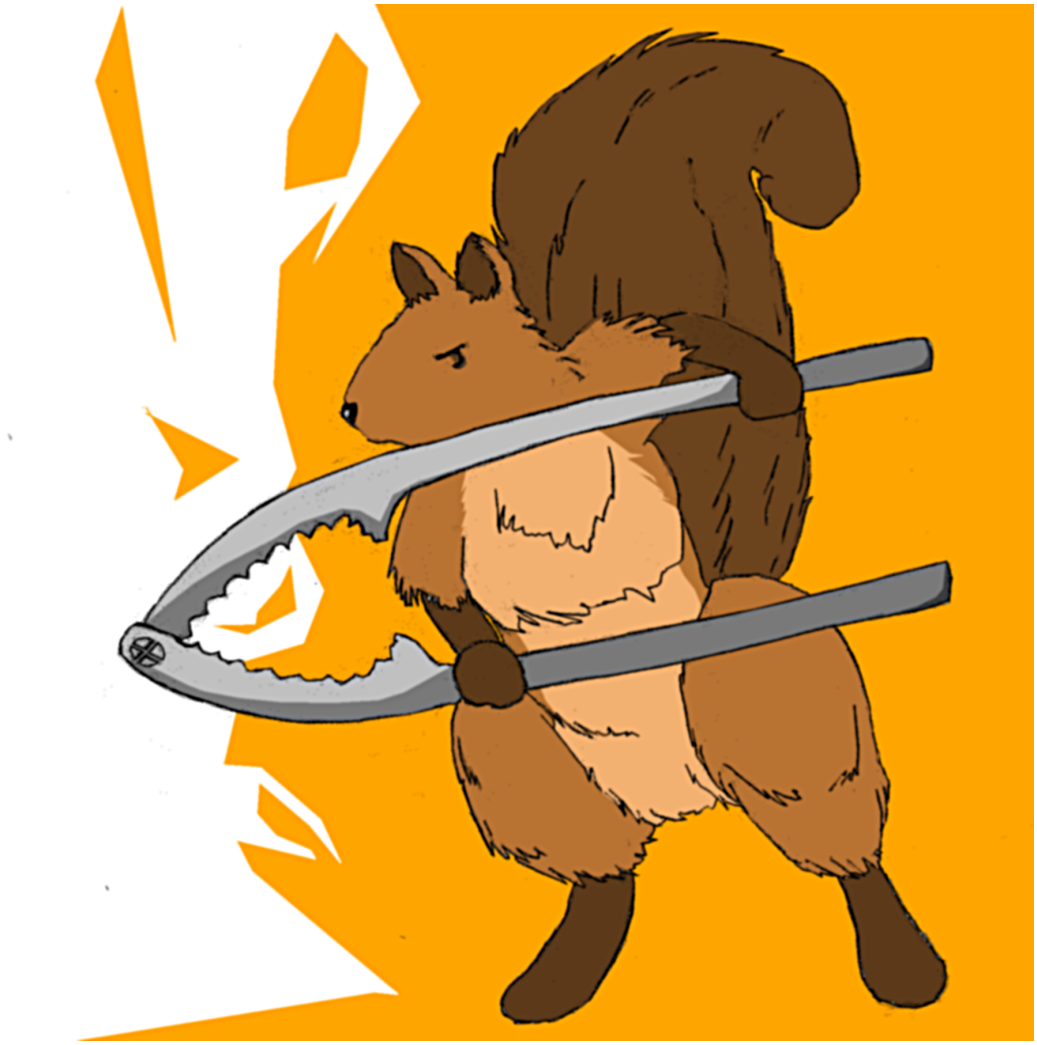


Figure D.2: Member of the United Riddle Solving Squirrels, used to represent cracker playstyle ((a)-annotators)

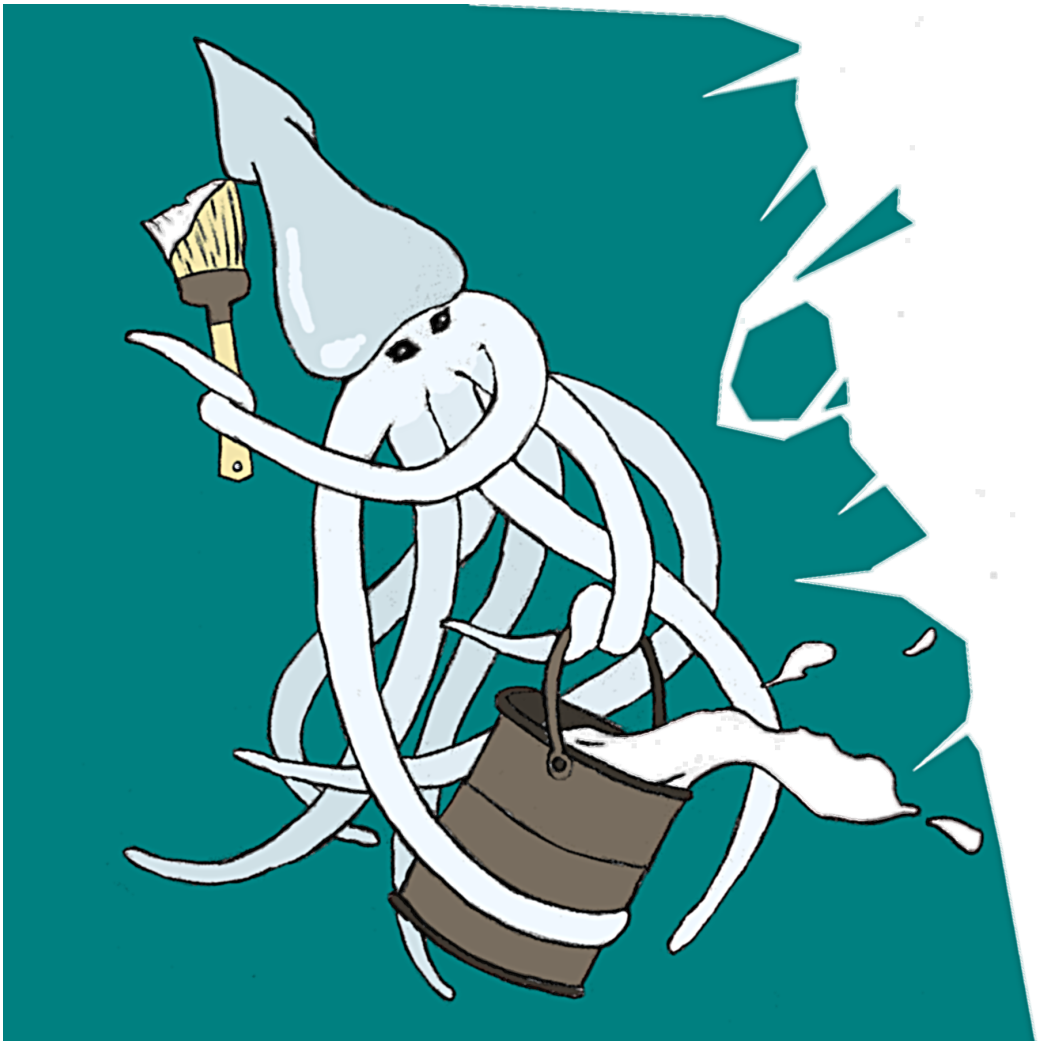


Figure D.3: Minion squid of Tippesk, used to represent blanker playstyle ((b)-annotators)

Figure D.4: Displayed for correct (a)-annotations

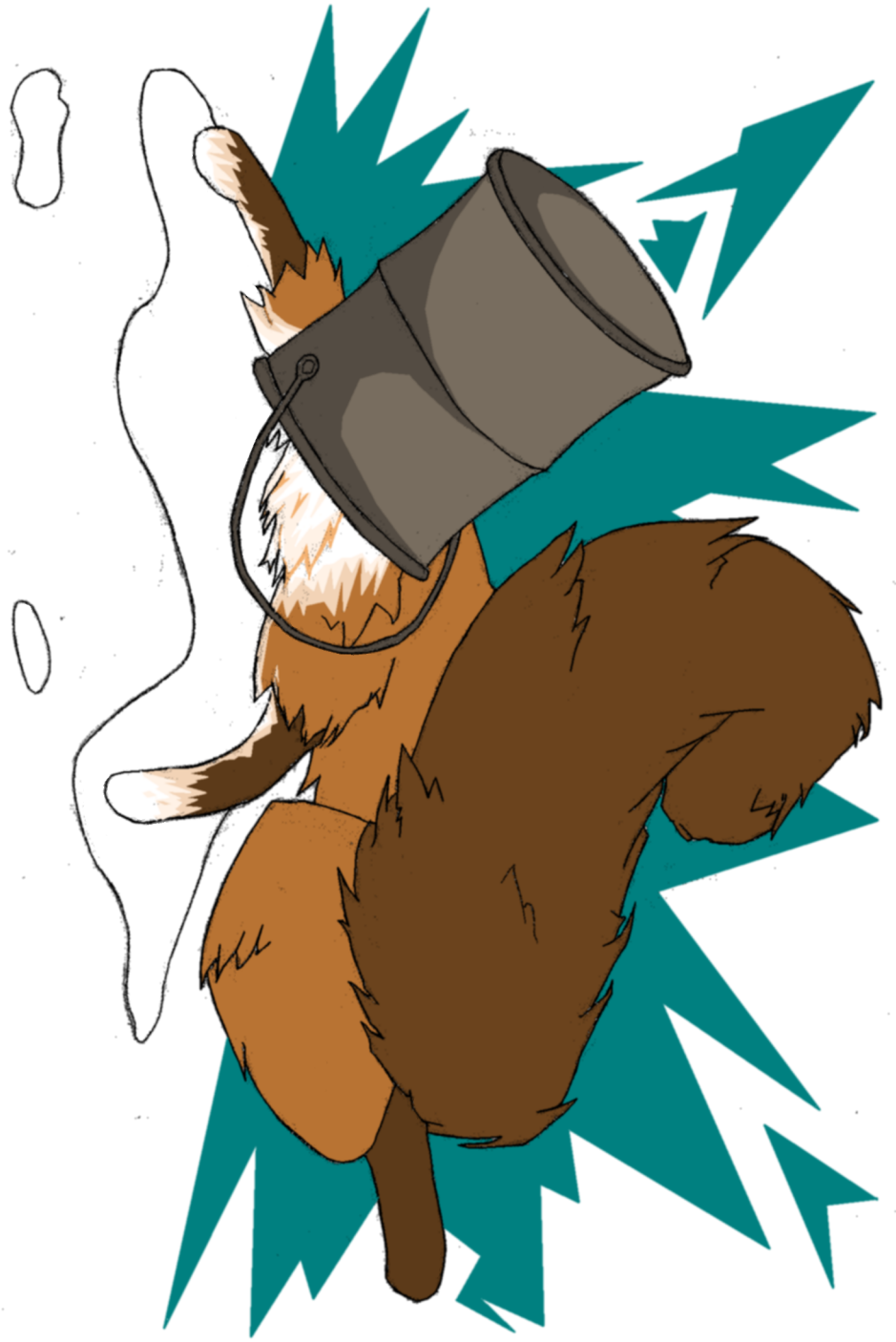


Figure D.5: Displayed for incorrect (a)-annotations

Figure D.6: Displayed for highly efficient (b)-annotations

Figure D.7: Displayed for mildly efficient (b)-annotations

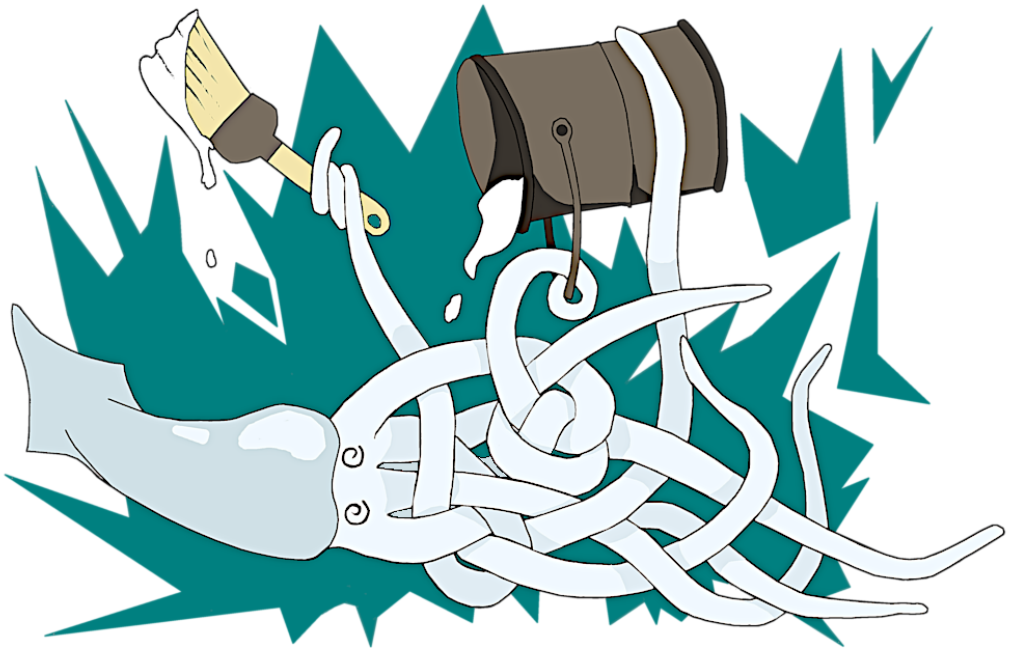


Figure D.8: Displayed for poor (b)-annotations

Figure D.9: Displayed during contests ((a)-annotators ranking)

Figure D.10: Displayed during contests ((b)-annotators ranking)

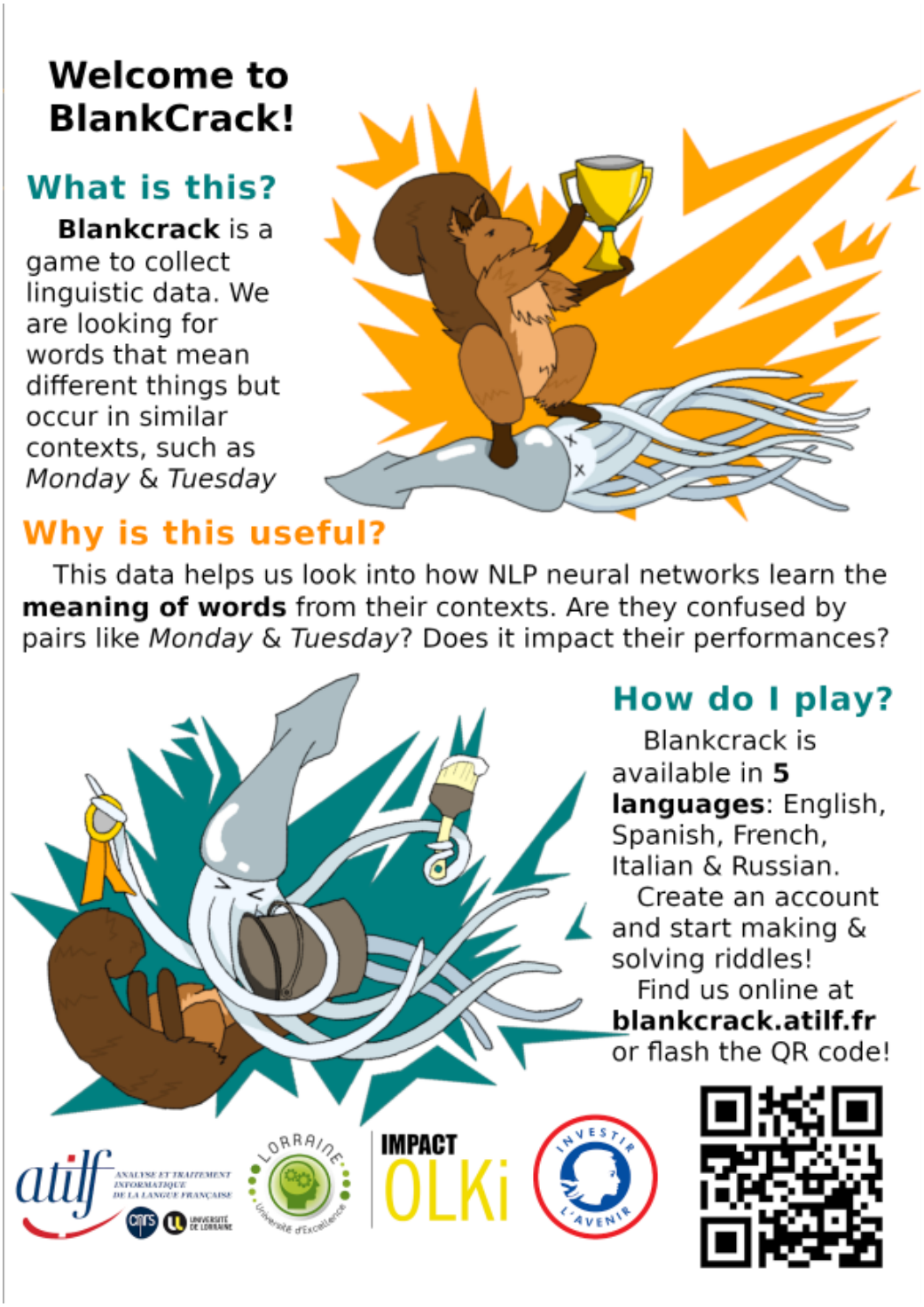


Figure D.11: Advertisement leaflet (English)

Bienvenue à BlankCrack !

C'est quoi ?

Blankcrack est un jeu pour collecter des informations linguistiques. Nous cherchons des mots de sens différents et qui apparaissent dans des contextes similaires, comme *mardi* et *lundi*.

Ça sert à quoi ?

Ces données nous permettent d'étudier comment les réseaux de neurones en TAL apprennent le **sens des mots** à partir du contexte. Sont-ils perturbés par des paires comme *mardi* et *lundi* ? Est-ce que cela joue sur leurs performances ?

Comment jouer ?

Blankcrack existe en **5 langues** : Français, Anglais, Italien, Russe et Espagnol.

Inscris-toi et viens résoudre et construire des énigmes !

Rejoins-nous **blankcrack.atilf.fr** ou flash le QR code !

atilf — Analyse et Traitement Informatique de la Langue Française — cnrs — Université de Lorraine

Lorraine Université d'Excellence

IMPACT OLKi

Investir l'avenir

Figure D.12: Advertisement leaflet (French)