From Text to Trust : A Priori Interpretability Versus Post Hoc Explainability in Natural Language Processing

Tom Bourgeade

Université Fédérale
Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE
Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue par

Tom BOURGEADE

Le 7 mars 2022

Interprétabilité A Priori et Explicabilité A Posteriori dans le Traitement Automatique des Langues

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Informatique et Télécommunications**

Unité de recherche :
IRIT : Institut de Recherche en Informatique de Toulouse

Thèse dirigée par
Philippe MULLER et Tim VAN DE CRUYS

Jury

Mme Aurélie NÉVÉOL, Rapporteure
M. Benoit FAVRE, Rapporteur
M. Pascal DENIS, Examinateur
M. Philippe MULLER, Directeur de thèse
M. Tim VAN DE CRUYS, Co-directeur de thèse
Mme Leila AMGOUD, Présidente

Abstract

With the advent of Transformer architectures in Natural Language Processing a few years ago, we have observed unprecedented progress in various text classification or generation tasks. However, the explosion in the number of parameters, and the complexity of these state-of-the-art blackbox models, is making ever more apparent the now urgent need for transparency in machine learning approaches. The ability to explain, interpret, and understand algorithmic decisions will become paramount as computer models start becoming more and more present in our everyday lives. Using eXplainable AI (XAI) methods, we can for example diagnose dataset biases, spurious correlations which can ultimately taint the training process of models, leading them to learn undesirable shortcuts, which could lead to unfair, incomprehensible, or even risky algorithmic decisions. These failure modes of AI, may ultimately erode the trust humans may have otherwise placed in beneficial applications. In this work, we more specifically explore two major aspects of XAI, in the context of Natural Language Processing tasks and models: in the first part, we approach the subject of intrinsic interpretability, which encompasses all methods which are inherently easy to produce explanations for. In particular, we focus on word embedding representations, which are an essential component of practically all NLP architectures, allowing these mathematical models to process human language in a more semantically-rich way. Unfortunately, many of the models which generate these representations, produce them in a way which is not interpretable by humans. To address this problem, we experiment with the construction and usage of Interpretable Word Embedding models, which attempt to correct this issue, by using constraints which enforce interpretability on these representations. We then make use of these, in a simple but effective novel setup, to attempt to detect lexical correlations, spurious or otherwise, in some popular NLP datasets. In the second part, we explore post-hoc explainability methods, which can target already trained models, and attempt to extract various forms of explanations of their decisions. These can range from diagnosing which parts of an input were the most relevant to a particular decision, to generating adversarial examples, which are carefully crafted to help reveal weaknesses in a model. We explore a novel type of approach, in parts allowed by the highly-performant but opaque recent Transformer architectures: instead of using a separate method to produce explanations of a model's decisions, we design and fine-tune an architecture which jointly

learns to both perform its task, while also producing free-form Natural Language Explanations of its own outputs. We evaluate our approach on a large-scale dataset annotated with human explanations, and qualitatively judge some of our approach's machine-generated explanations.

Résumé

Avec l'avènement des architectures Transformer en Traitement Automatique des Langues il y a quelques années, nous avons observé des progrès sans précédents dans diverses tâches de classification ou de génération de textes. Cependant, l'explosion du nombre de paramètres et de la complexité de ces modèles "boîte noire" de l'état de l'art, rendent de plus en plus évident le besoin désormais urgent de transparence dans les approches d'apprentissage automatique. La capacité d'expliquer, d'interpréter et de comprendre les décisions algorithmiques deviendra primordiale à mesure que les modèles informatiques deviennent de plus en plus présents dans notre vie quotidienne. En utilisant les méthodes de l'IA eXplicable (XAI), nous pouvons par exemple diagnostiquer les biais dans des ensembles de données, des corrélations erronées qui peuvent au final entacher le processus d'apprentissage des modèles, les conduisant à apprendre des raccourcis indésirables, ce qui pourrait conduire à des décisions algorithmiques injustes, incompréhensibles, voire risquées. Ces modes d'échec de l'IA peuvent finalement éroder la confiance que les humains auraient pu placer dans des applications bénéfiques. Dans ce travail, nous explorons plus spécifiquement deux aspects majeurs de l'XAI, dans le contexte des tâches et des modèles de Traitement Automatique des Langues : dans la première partie, nous abordons le sujet de l'interprétabilité intrinsèque, qui englobe toutes les méthodes qui sont naturellement faciles à expliquer. En particulier, nous nous concentrons sur les représentations de plongement de mots, qui sont une composante essentielle de pratiquement toutes les architectures de TAL, permettant à ces modèles mathématiques de manipuler le langage humain d'une manière plus riche sur le plan sémantique. Malheureusement, la plupart des modèles qui génèrent ces représentations les produisent d'une manière qui n'est pas interprétable par les humains. Pour résoudre ce problème, nous expérimentons la construction et l'utilisation de modèles de plongement de mots interprétables, qui tentent de corriger ce problème, en utilisant des contraintes qui imposent l'interprétabilité de ces représentations. Nous utilisons ensuite ces modèles, dans une configuration nouvelle, simple mais efficace, pour tenter de détecter des corrélations lexicales, erronées ou non, dans certains ensembles de données populaires en TAL. Dans la deuxième partie, nous explorons les méthodes d'explicabilité post-hoc, qui peuvent cibler des modèles déjà entraînés, et tenter d'extraire diverses formes d'explications de leurs décisions. Ces méthodes peuvent aller du diagnostic des parties d'une entrée qui étaient les

plus pertinentes pour une décision particulière, à la génération d'exemples adversariaux, qui sont soigneusement conçus pour aider à révéler les faiblesses d'un modèle. Nous explorons un nouveau type d'approche, en partie permis par les architectures Transformer récentes, très performantes mais opaques : au lieu d'utiliser une méthode distincte pour produire des explications des décisions d'un modèle, nous concevons et mettons au point une configuration qui apprend de manière jointe à exécuter sa tâche, tout en produisant des explications en langage naturel en forme libre de ses propres résultats. Nous évaluons notre approche sur un ensemble de données de grande taille annoté avec des explications humaines, et nous jugeons qualitativement certaines des explications générées par notre approche.

Acknowledgements

I would like to acknowledge the support and guidance of a number of people, without whom the completion of this dissertation would not have been possible.

I would like to express my deepest gratitude to Dr. Aurélie Névéol and Prof. Benoit Favre for the interest they have shown in my work, and agreeing to review this dissertation. I am very thankful for their time, and the attention to detail in their reviews. Their questions and suggestions were followed by rich and stimulating discussions during and after the defense of this thesis. I would also like to extend my sincere thanks to Dr. Leila Amgoud and Dr. Pascal Denis for being part of my dissertation committee and participating in the evaluation of my work.

I am deeply indebted to my supervisors, Dr. Philippe Muller and Dr. Tim Van de Cruys, for their support and guidance all throughout those three years. I could not have undertaken this journey without your assistance and encouragement, particularly in the difficult conditions we all had to go through during the COVID-19 pandemic. Your careful examination of the various drafts of this dissertation was invaluable in its completion. Our various discussions on the topics of explainability and interpretability were (and still are) intellectually stimulating, and I owe a majority of the things I learned about, and the experiments I have been able to perform, to our mutual interest in these subjects.

Many thanks to the members of team MELODI and my labmates at IRIT, for the various exchanges we had about our respective subjects and work, and to my fellow Doctors and PhD students who I spent parts of those three years with, for being able to share about the challenges and difficulties of PhD life with you.

Finally, I would like to thank my parents for supporting me and encouraging me through this journey, particularly when I was doubtful or anxious, and for giving me space and calm to write in, when I needed it.

Contents

Chapter 1

Setting The Scene

Over the last decade or so, we have observed an accelerating growth in the complexity of state-of-the-art machine learning models, in pretty much all fields of AI, but in particular in NLP, mainly due to the recent development of the Transformer [Vaswani et al., 2017] architecture, illustrated by the famous BERT [Devlin et al., 2019] and GPT-n [Radford et al., 2018, 2019; Brown et al., 2020] series of models.

Practically all popular models actively used in NLP nowadays are so-called *black-box* models, that is, architectures which are inherently opaque to direct human analysis, usually due to the sheer number of variables and non-linear interactions present in the underlying mathematical models involved (see Table 1.1). These architectures are usually trained in an end-to-end fashion, in which the machine learning practitioners and end-users both effectively limit their direct interactions to the inputs, outputs, and hyper-parameters of the models, whether at training or inference time. Once the task has been specified, usually through the definition of an objective (or inversely, cost) function to be optimized, the input and target data is fed to the training or inference algorithm, and the necessary intermediate steps of the computation are effectively left to the model's discretion. This is arguably the main strength of deep learning approaches, as they not only skip over most of the often heavy human requirement of engineering a pipeline appropriate for a given task, but they also seem to do this job significantly better, in terms of evaluation metrics, than humans. It is however also a weakness, in that it significantly reduces the ability for humans to understand what exactly these architectures have learned and executed in those intermediate steps. Some of these large architectures, usually pre-trained on massive amounts of unannotated text, achieve performance on difficult evaluation benchmarks, like GLUE or SuperGLUE [Wang et al., 2018, 2019], close to or sometimes surpassing [Sun et al., 2021] human performance on those same tasks.

Yet, despite these numeric achievements, some of those same models have been repeatedly shown to display a wide variety of overall undesirable behaviors, whether they be weaknesses

Model	Parameters	Depth
InferSent [Conneau et al., 2017]	∼ 50M	4
ELMo [Peters et al., 2018]	∼ 100M	4
GPT [Radford et al., 2018]	∼ 117M	24
BERT [Devlin et al., 2019]	∼ 336M	24
GPT-2 [Radford et al., 2019]	∼ 1.5B	48
GPT-3 [Brown et al., 2020]	∼ 175B	96

Table 1.1: Evolution of the sizes and depths (for the first two models, in number of layers, for the remaining Transformer models, in number of Transformer blocks) in recent popular NLP architectures over the years.

to non–semantically-destructive adversarial perturbations [Jia and Liang, 2017; Ribeiro et al., 2018b; Jin et al., 2020; Li et al., 2020], failures in handling basic and common linguistic phenomena, such as negation or antonymy [Naik et al., 2018; Hossain et al., 2020; Kassner and Schütze, 2020; Aspillaga et al., 2020], gender and lexical biases [Vig et al., 2020], and many more. These can ultimately translate into risks or costs at a more societal level, such as negative environmental or financial impact [Bender et al., 2021]. These shortfalls are particularly worrying when one considers that some of these models are already at this time being used in production environments, facing actual human end-users, such as in translation or text prediction systems: such behaviors, even if they occur infrequently, could ultimately erode the *trust* humans may have in these systems and the algorithms behind them, even if they could have eventually turned out to be overall beneficial to society. This is compounded by a certain amount of miscommunication and overall over-promotion of state-of-the-art model capabilities, which can lead both end-users and researchers to have an over-estimated impression of the actual abilities of these approaches [Bender and Koller, 2020].

We can impute many of these issues to the drawbacks of relying on simple metrics like accuracy on benchmark tasks to evaluate actual understanding of natural language in computer models. Indeed, while many of these tasks are crafted under the hypothesis that they each require the learning of some set of logico-linguistic skills and knowledge to be accurately solved, there are usually no strong guarantees that:

1. the distribution of the instances in the dataset(s) associated with a given task actually matches the distribution associated with the system which is intended to be modelled: when this is not the case, it could be classified as different forms of *dataset biases* (which are comprehensively explored for Computer Vision in Torralba and Efros [2011]), usually either covariate shift (when the distribution of the inputs, $p(x)$ may have shifted, while the conditional distribution of the labels $p(y|x)$ remains fixed) or its converse, label shift ($p(y)$ shifted, $p(x|y)$ fixed), respectively associated with so-called causal (predicting effects from their causes) or anticausal (predicting causes from their effects) learning

[Schölkopf et al., 2012; Lipton et al., 2018];

2. the specified objective function of the task, when optimized, actually guides the trained model towards its intended goal(s): when this is not the case, it could be classified as mismatched, conflicting or underspecified objectives [D'Amour et al., 2020], the primary symptom of which being found when training similar models on identical tasks, but ending up with completely different generalization behaviors on out-of-distribution instances (often both in training and testing sets, as these are unfortunately quite commonly drawn from the same source distributions), despite similar or even identical performance in-distribution [McCoy et al., 2020].

With either (or both) of these lacking, one will very probably end up with a model prone to some level of shortcut learning [Geirhos et al., 2020], in essence, a model which is deceptively well performing on a particular set of tasks and datasets, but which ends up behaving inappropriately, possibly in subtle ways, once presented with real-world data.

In any case, even assuming these issues have been somehow taken into account in a particular application, its end-users will almost always wish for and benefit from explanations being provided, particularly in the case of algorithmic decision-making systems which are bound to become more and more prevalent in society as time and technology progresses. There even already seems to be a growing consensus that this requirement should become a matter of law, as some have argued that the *General Data Protection Regulation (GDPR)*, which has recently become law in Europe, provides some level of "right to explanation" [Goodman and Flaxman, 2017] concerning decisions taken by automated systems, though this has also been disputed [Wachter et al., 2017]. Explainability and interpretability, as these concepts are often referred to, also plays an important role in AI Ethics and Fairness, to better try to ensure machine learning models do not unfairly discriminate on the basis of undesirable biased features, which often can ultimately be recovered and extracted by these approaches, even when some preliminary attempts were made to prevent this from occurring [Dodge et al., 2019; Aivodji et al., 2019; Arrieta et al., 2020; Sharma et al., 2020].

For all these reasons, it is apparent there exists a strong need for methods which would enable more transparency and human understanding of algorithmic decisions, particularly from the very opaque deep learning models which are employed nowadays. While a lot of work and effort has been deployed to attempt to reach these goals, the whole endeavor is very challenging, in parts due to the number of pre-existing or completely novel concepts which are involved, even just to define in more concrete terms what those very goals are: What is an explanation? What form(s) should they take? What makes a good explanation? Even these basic questions have not been fully explored or definitely answered yet, let alone the more challenging technical aspects of how to actually implement these concepts in practice.

1.1 Concepts

We will mainly focus here on aspects of these fields which are pertinent to Natural Language Processing. For a more complete and exhaustive overview of the fields of eXplainable AI (XAI) and Interpretable Machine Learning (IML) in general, we refer the reader to the IML book from Molnar [2019], which at the present time, represents one of the most exhaustive but also easy-to-grasp overview and review of these fields.

If we look at the relevant literature, terms like *explainable*, *explainability*, *explanation*, *interpretable*, *interpretability*, etc., are often employed in relation to the previously discussed concepts, without being explicitly defined. Unfortunately, whether one looks inside or outside the fields of eXplainable AI (XAI) or Interpretable Machine Learning (IML), it is difficult to find consensus on the definition of those terms, and in particular, *interpretability* and *explainability* are often used interchangeably when not precisely defined. If we look at the relevant definitions of the verbs *explain* and *interpret* in the online Merriam-Webster dictionary for instance, we can find respectively "(a) to make known | (b) to make plain or understandable" and "to explain or tell the meaning of : present in understandable terms", which does not help much in differentiating those two terms, outside of highlighting the common meaning of "to make understandable", and that this action might be done through the medium of explanations. And indeed, in the context of machine learning in particular, it appears that the terms explainability and interpretability are both commonly used to designate essentially "the ability to explain or present in understandable terms to a human" [Doshi-Velez and Kim, 2017; Guidotti et al., 2018; Lipton, 2018; Molnar, 2019; Arrieta et al., 2020]. However, one important aspect that often accompanies these two terms, concerns the *temporality* in which this action of explaining occurs: is the process-to-be-explained, the *explanandum*, already understandable in and of itself, and thus *a priori* its own source of explanations? or, is the *explanandum* a system too complex to be grasped in its entirety, thus requiring the intervention of another process, a *post-hoc* method, to act as a more understandable proxy of the first? This distinction is usually made explicit by using the qualifiers *a priori* or *intrinsically*, for the former, and *post-hoc*, for the latter [Rudin, 2019; Molnar, 2019], though the words "interpretable" and "explainable" also seem to implicitly carry these notions, and as such we can often find together "intrinsically interpretable", and "post-hoc explainable", to refer to the two distinct concepts. More concretely, intrinsic interpretability seems to refer to an inherent property of a type of architecture, whereas post-hoc explainability seems to encompass methods which produce explanations as separate processes from the models in question, either globally, after training, or locally, after a decision. While the particular use of these terms as such definitely does not make consensus at this time in the literature, it is nonetheless somewhat useful as it avoids the interchanging use of "interpretable" and "explainable" (and other derived terms). However, if strictly used to refer to distinct concepts, because both a priori interpretability and post-hoc explainability are often placed under the

same umbrella when discussing explanations, regardless of their sources, we would have to use both terms together quite often. As such, in the rest of this document, "**interpretability**" and "**interpretable**" will always refer to **intrinsically interpretable** models as described above, whereas "**explainability**" and "**explainable**" will refer to explanation producing methods in general, unless used in the phrases **"post-hoc explainable"** or **"post-hoc explainability"**, in which case they refer only to such methods.

1.1.1 Desiderata for Explainability/Interpretability Methods

Outside of purely definitional issues, there also exist numerous questions about what is desired out of these methods. In the relevant literature[Doshi-Velez and Kim, 2017; Guidotti et al., 2018; Lipton, 2018; Robnik-Šikonja and Bohanec, 2018; Molnar, 2019; Arrieta et al., 2020], many desiderata have been proposed, often in different terms and at different levels of analysis, which makes exhaustively listing them in a coherent manner practically impossible. These desired properties can target the explained models, the explanation methods, or the explanations they produce themselves. As the latter concerns the not-yet–discussed point of explanations, here is a non-exhaustive list of desiderata which might directly concern models and explanation methods (largely inspired by Molnar [2019]):

- One of the primary, if not the core desiderata of any machine learning model, explainable or not, is **accuracy**, that is, how good the model is at predicting seen or unseen instances. In the first case, this simply indicates how well a model is learning for its training distribution. In the second case, however, it also indicates how well a model is *generalizing* to Out-of-Distribution (OOD) instances, and by extension, how well it has learned to model the process or system *represented* by the data, instead of just the data itself. It is especially important to keep in mind, as it is often unfortunately a property which has to be traded off in some capacity to obtain more transparent and thus easier to explain models, though this has been debated [Alvarez-Melis and Jaakkola, 2018; Rudin, 2019; Aivodji et al., 2019].

- **Translucency** of an explanation method refers to the degree to which it derives its explanations directly from the explained model's internals, architecture and parameters. On the higher end of this scale, we can find intrinsically interpretable models, whereas model-agnostic methods would rate lower. This property may be desirable in that it makes the explanation method easier to understand and check with regard to the analyzed model, however it also naturally makes it more architecture-specific and thus less **portable**.

- **Portability** designates the range of architectures an explanation method may be applied to: a model-agnostic method, which makes little to no assumptions about the internals of a given model, and interacts with it purely within the black-box paradigm, would be

highly portable and thus useable against a wide range of existing models. In practice, one cannot have both high portability and high translucency at the same time, and so it might be necessary to compromise on either or both of those desiderata in some capacity.

- **Complexity**, whether it is algorithmic (in processing time and/or memory requirements) or conceptual/architectural (in number of parameters, hyperparameters, depth, or degrees of non-linearity), applies to both explanation methods and the models they might target. In either case, this is a property one would wish to minimize as much as can be afforded, since lower algorithmic complexity enables faster execution (both in training and at inference time), on smaller and/or more affordable hardware, and at lower energy costs. Similarly, lower conceptual complexity allows for easier deployment (mainly due to the fewer number of hyperparameters to fine-tune) and easier debugging/analyzing by humans. For explanation methods more specifically, it is generally considered undesirable to have similar or greater algorithmic complexity than the models which are to be explained, as this would make their use as diagnostic tools impractical.

The second, possibly even more important, set of desiderata to consider, are those which concern the produced explanations themselves.

1.1.2 Desiderata for Produced Explanations

In the eXplainable AI (XAI) and Interpretable Machine Learning (IML) literature, it is difficult to find a unified definition of the term "explanation". While most people will intuitively have a practical understanding of this term, figuring out a pertinent and more formal definition is very challenging [Doshi-Velez and Kim, 2017; Arrieta et al., 2020].

Similar to explanation methods, it may be that the easiest way to define explanations is through exploring what is desired out of them. Looking back at the previous list of desiderata, we can turn again to [Molnar, 2019] who compiles a fairly exhaustive list of explanations-specific desiderata, the following three of which (slightly reformulated) we believe are the most pertinent:

- **Faithfulness**, or **fidelity**, is probably the most important feature of an explanation: it represents how accurate an explanation is, as a model itself, at modeling its target *explanandum*. In other words, a faithful explanation is one which accurately reformulates the internal behavior of a model, and thus shares the same behaviors it has. It is extremely important to ensure we can *trust* these explanations: indeed, an unfaithful explanation of a particular prediction may actually look like a perfectly plausible behavior a theoretical model *could* have, but it will not actually accurately represent what went on internally in the target model we are interested in. A method which produces unfaithful explanation would most likely at some point stop making sense, when exploring parts of the input

space for which it diverges with the target model.

- **Stability** represents how consistent the produced explanations are when making small variations to the input instances fed to the target model and then explained. Ideally, an explanation method should not produce wildly differing explanations for very minute perturbations of an input. Low stability explanations may cause frustration in end-users, as attempting to exploit them to plan modifications of the data or models themselves would be made difficult, due to these inconsistencies appearing after every small modifications.

- **Comprehensibility** represents how easy it is for the end-users to parse the produced explanations, as well as act on them. This property will be highly dependent on the target audience of the method, though ultimately, a more comprehensible explanation format in an absolute sense will be more helpful to any audience, expert or non-expert. This property also depends on the *"size"* of the produced explanations: even simple *if-this-then-that* rules will become difficult to parse if there are hundreds of them to look for each prediction explanation. This also illustrates why even very basic models, such as linear models or Decision Trees, are not interpretable in an absolute sense, as the comprehensibility of the explanations they intrinsically allow to generate directly depends on the number of parameters/nodes in them.

To illustrate these concepts more concretely, let us now discuss one of the main use cases of explanations: detecting and understanding problematic behaviors learned by models. Of course, no architecture is inherently created flawed, rather, in the majority of cases, these behaviors stem from issues in the training data that is fed to them.

1.2 Garbage in, garbage out: Dataset Biases and Spurious Correlations

One of the main use-cases for explainability is to detect undesired behaviors in trained or in-training models, in order to correct, or at least be aware and wary of them. Most often, the behaviors are not inherent to the model architectures employed, but are rather distilled from the training data. The issue of *dataset biases* [Torralba and Efros, 2011] which plagues machine learning as a whole, whatever the subdomains and types of data manipulated are, can be viewed as an extension of the famous "garbage in, garbage out" principle in computer science: since current machine learning architectures essentially implement automated inductive and/or deductive reasoning using a set of training data as its basis, if the distribution of this data happens to differ in non-random ways from the distribution associated with the process which is to be modelled, these differences may introduce exploitable biases and spurious correlations with regards to the intended task, and thus lead to shortcut-learning models [Geirhos et al.,

Entailment			*Neutral*			*Contradiction*		
Word	Prob	Occ	Word	Prob	Occ	Word	Prob	Occ
instrument	0.90	20	tall	0.93	44	sleeping	0.88	108
touching	0.83	12	competition	0.88	24	driving	0.81	53
least	0.90	10	because	0.83	23	Nobody	1.00	52
Humans	0.88	8	birthday	0.85	20	alone	0.90	50
transportation	0.86	7	mom	0.82	17	cat	0.84	49
speaking	0.86	7	win	0.88	16	asleep	0.91	43
screen	0.86	7	got	0.81	16	no	0.84	31

Table 1.2: Examples of single-word biases in the SNLI dataset, found and reported by Poliak et al. [2018] (Figure 3 in the original publication), for each class: *Prob* lists the empirical conditional probabilities $p(\text{label}|\text{word})$ of predicting the given label when the given word is present in the hypothesis sentence; *Occ* lists the number of occurrences in the dataset of instances that contain the given word. While each of these artifacts independently does not impact a great number of instances, the fact some of them are such strong predictors for their respective class, means a model trained on SNLI may undesirably learn these as shortcuts.

2020].

Many such issues have been discovered over the years in NLP datasets, for instance, in visual question answering [Jabri et al., 2016; Zhang et al., 2016], reading comprehension [Chen et al., 2016; Jia and Liang, 2017; Kaushik and Lipton, 2018], or paraphrase identification [Zhang et al., 2019], using various methods: standard error analysis, "stress testing" models on adversarial generated or filtered examples, etc.

1.2.1 A concrete example: the cases of SNLI and MNLI

A famous example of such dataset biases can be found in the **Stanford Natural Language Inference (SNLI)** corpus [Bowman et al., 2015] and also its multi-genre variant, the **Multi-genre Natural Language Inference (MNLI)** corpus [Williams et al., 2018]. The Natural Language Inference task in NLP consists in predicting whether a premise text (e.g. *"A soccer game with multiple males playing."*) logically entails a hypothesis text or not (e.g. *"Some men are playing a sport."*, which is indeed entailed here, whereas *"Some men are sleeping."* would be contradicting the premise).

To illustrate, here are three example instances from the SNLI test-set, one for each of the possible inference labels (*entailment*, *contradiction*, or *neutral*). As all three share the same premise sentence, it is only displayed once. We also add a short comment for each which more explicitly explains the label:

Premise:	"*People waiting in line in a snowstorm.*"
Hypothesis:	"*People are in a snowstorm.*"
Ground-truth Label:	*Entailment*
Comment:	This hypothesis is less specific than the premise (by removing "waiting in line"), and so is entailed by it.
Hypothesis:	"*People are waiting in a line during a scorching drought.*"
Ground-truth Label:	*Contradiction*
Comment:	A scorching drought is not a snow storm, hence the contradiction.
Hypothesis:	"*People are waiting in line for food to get through the snowstorm with.*"
Ground-truth Label:	*Neutral*
Comment:	This hypothesis is more specific than the premise, and is therefore not entailed by it, but it also does not contradict any of its elements. It is therefore neutral.

These large-scale datasets, which were annotated through crowdsourcing (discussed in more details in the dedicated paragraph of Section 2.4), using the Amazon Mechanical Turk platform, are very popular, and have been used to evaluate numerous models, as part of the famous GLUE benchmark [Wang et al., 2018] for example, and also to train NLP models. The well-known InferSent [Conneau et al., 2017] sentence-encoding architecture, for example, was pre-trained to classify the inference relation between pairs of premises and hypotheses from the SNLI corpus.

Unfortunately, a few years after these datasets were introduced, a number of unwanted statistical annotation artifacts were discovered in them. One would expect such tasks to require a relatively high level of Natural Language Understanding (NLU) capabilities, as the relation between the two sentences to be predicted involves both learning the ability to parse and understand each premise and hypothesis in and of itself, separately, but also learning how two such sentences may relate to each other, in a logical and semantic manner. Compared to just evaluating structural or grammatical similarity, this task requires a lot of implicit world-knowledge and *commonsense*. However, McCoy et al. [2019] showed for example that models trained on these datasets tend to pan in on makeshift heuristics, such as the amount of word overlap between the two sentences, which is evidently not a property that should be learned in order to perform proper natural language inference. Perhaps even worse, Gururangan et al. [2018] and Poliak et al. [2018] both found that hypothesis-only models, that is, models that only look at the hypothesis sentences as part of their inputs, can correctly predict the labels of significant portions of those datasets, at least 67% of SNLI, and 53% of MNLI, according to Gururangan et al. [2018]. The main cause for this was attributed to the crowdsourcing process:

indeed, for these datasets, crowd workers were tasked with generating three hypotheses sentences, one for each of the possible inference classes, given a premise sentence collected from image captions, which are thus often nominal sentences (with no verbs; e.g., premise example above). While the workers were not presented with the actual images, they were informed that the premises were sampled from image captions, which might have introduced some implicit biases towards writing hypotheses describing situations which may be more likely to find as a caption for a picture. But more importantly, due to the nature of the crowdsourcing platform, in which workers are ultimately economically rewarded depending on the number of annotations they can produce per unit of time (over multiple annotation campaigns), a form of "least effort"-bias appears to have tainted the produced premises: mainly, for the contradiction class, a "least effort" strategy is to simply repeat the premise, with a minimum of one negation of an element added, which in English, can be done very easily in most situation by inserting the words "not" or "no" in the correct position. For instance, in the example shown above, a "least effort" contradiction of the premise *"People waiting in line in a snowstorm"* may simply be *"People **not** waiting in line in a snowstorm"*. Other similar single-word cues were found for each of the three classes, with varying importance, and were reported by Gururangan et al. [2018] and Poliak et al. [2018] (see Table 1.2, adapted from Figure 3 in their publication), with a high degree of overlap, using similar methods.

1.2.2 The Need for Explanations

These two datasets, as such, perfectly exemplify the need for explainability in Natural Language Processing, if not in machine learning in general. In particular, the fact such spurious correlations were only detected years after these datasets were made available, and used as part of the training or evaluation of, at the time, state-of-the-art architectures, is quite worrying. Improving the variety of explainability methods available, as well as their ease of implementation and usage, should be paramount to ensuring that, ultimately, end-user–facing machine learning models do not rely on such undesirable biased shortcuts, which may lead to deceptively unfair or even dangerous algorithmic decisions.

While explainability methods may be used to detect these types of issues, other related approaches have been proposed to attempt to then fix them, in various ways. For example, He et al. [2019] devise a method to train debiased models on these corpora: to do so, they first train an intentionally biased model which mainly exploits unwanted shortcuts such as those detected in the contributions discussed above, more specifically, by feeding it only incomplete information, as in the case of hypothesis-only models (see above). They then subsequently train a new model on the residuals of the previous one, that is, by down-sampling the instances which were confidently classified (with low loss function values) by previously obtained biased classifier, and which are thus likely to be biased and easy-to-learn themselves. Another way

to “repair” such biases in a dataset was proposed by Swayamdipta et al. [2020], in the form of *Data Maps*, which compile the *training dynamics* of a model on a dataset, mainly, the *confidence* (mean) and *variability* (standard deviation) of its predictions across epochs, allowing to visualize *easy-to-learn* instances (low variability, high confidence), *hard-to-learn* instances (low variability, low confidence), and finally *ambiguous* instances (high variability). By splitting the training dataset into these three parts, and adjusting the quantities of instances in each, the authors showcase how the final capabilities and performance of a model can be tweaked. They also showcase how this method can be used to detect mislabeled instances.

Inspired by these now famous examples of hidden dataset biases, detected through the use of methods related to explainability, we decided to focus this work on the exploration of more dedicated approaches, which might enable detecting similar issues in other datasets. This remainder of this work is divided into 4 Chapters, divided into two parts: Chapters 2 and 3 first explore intrinsic interpretability in the context of Natural Language Processing, with a particular focus on Interpretable Word Embedding models; Chapters 4 and 5 then explore post-hoc explainability methods, with a particular focus on Natural Language Explanations. Chapters 2 and 4 discuss and review the state-of-the-art respective to each part, as well as introduce the necessary concepts, models, and datasets, which are then used in the experimental setups showcased in Chapters 3 and 5.

Chapter 2

Intrinsic Interpretability in NLP

In this first part, we explore the specifics of intrinsic interpretability in Natural Language Processing. Beginning with a discussion of interpretability is particularly important when dealing with natural language as inputs or outputs, because it is in and of itself a challenging information-carrying medium to tackle: indeed, language can be used in a wide variety of forms, a lot of which are intended to carry significant amounts of implicit information, for example, in poetry, humor, or fiction. Even more "formal" uses of natural language, such as scientific publications, technical or administrative reports, etc., require a significant amount of contextual information, the majority of which is often not present in the documents that are to be parsed, but implied to be already be a part of the reader or listener, usually referred to as *commonsense* or *world-knowledge*. While not exclusive to human text or speech, these properties are of particular importance in NLP because the phenomena we want to model, such as the sequential generation of language, referred to usually as *language modeling*, very strongly depend on them, perhaps more so than in other types of data. Tabular or categorical data mostly stand on their own, by definition. Even a picture, though it would be more challenging, could have semantic information extracted from its internal, spatially-based correlations. But an arbitrary sentence, taken out of its surrounding explicit or implicit context, is mostly meaningless.

As such, we first discuss interpretability and the challenges related to the representation of language in NLP.

2.1 Context and Problematics

The common first step in digitally encoding any type of data, is to first decompose it into smaller, individually meaningful parts. For initially continuous data types, such as signals, this step is usually referred to as *quantization*. This can be done in various more or less systematic ways: images are usually decomposed into equally-sized pixels, whereas audio

may be similarly decomposed into discrete time-steps, or, if it contains human speech and this is what we actually wish to encode, into sequences of individual *phonemes*, the elementary semantic units of spoken language. Written text is no exception, however, deciding on an adequate decomposition scheme the outputs of which may then be used with machine learning models, happens to be quite difficult.

2.1.1 Elementary Semantic Units in Natural Language

One of the main conceptual difficulties in NLP regarding explainability is the abstractness of the manipulated data. In machine learning for Computer Vision or audio processing for instance, the numeric quantities involved usually have a clear link to their respective modelled real-world systems: whether it is images or sounds, the processes to encode or decode them to and from their canonical digital representations, usually, multi-channel integer matrices or vectors, are well established and mostly transparent to use, which allows models that operate on these representations to exploit as little or as much information about the real-world as is necessary, often way beyond human sensing capabilities (in terms of modalities or sensor/data resolution). When generating explanations for these types of media, one can point at and relate to parts, or even individual components, of a digital representation in a meaningful way with regard to the real-world source of that representation: even though the encoding process is not perfectly reversible (mainly due to quantization and sensor limitations), if a particular patch of pixels was highlighted as an explanation for a particular decision in an image, one could reasonably associate it with a corresponding visual feature of the represented scene, and draw conclusions regarding which real-world features might have been the cause of that decision.

When modelling more abstract systems, like those stemming from human behavior, these encoding/decoding processes are much more difficult to derive in a way that is equally natural. In the case of written natural language, once typically digitized using some character encoding scheme, the first available representation format for a piece of text is that of a sequence of symbols, which can be trivially converted to their corresponding integer ordinal code in the chosen character encoding table (such as the *ASCII* or *Unicode* standards). So-called character-level (or character-based) models are designed to function at this level of representation, and while not overall as popular as word or subword level models, they have been and remain actively explored [Zhang et al., 2015; Lee et al., 2017; Liang et al., 2017], even with more recent Transformer architectures [Al-Rfou et al., 2019; El Boukkouri et al., 2020].

However, the same way it is difficult to refer to the semantics of a single decontextualized pixel in an image (in contrast to a patch of pixels corresponding to a particular visual object), characters dealt with as separate entities do not seem like the natural minimal semantic unit for human language, and indeed, the vast majority of models in NLP deal with input texts as sequences of word-tokens. The two conceptual approaches each have their pros and cons,

mainly, dealing with characters directly bypasses the need to construct a fixed word vocabulary, which is helpful when one has to work with a large number of terms (such as with domain-specific languages, like medical terminology), at the cost of requiring a larger contextual window (in number of tokens) to attend to the same span of text compared to a word-based approach, which in practice limits the lengths of texts which can be handled comprehensively. Hybrid methods, based on subwords, attempt to compromise between these advantages and disadvantages, using, for example, the *Byte Pair Encoding (BPE)* scheme to construct efficient representations of frequent word pieces [Sennrich et al., 2016], which are for instance used in the popular GPT-n [Radford et al., 2018, 2019; Brown et al., 2020] series of models.

Whichever approach is selected to turn a piece of text into a sequence of tokens, these cannot be passed in directly as input into deep-learning models as they constitute categorical (non-numeric) features, and they thus have to be vectorized in some way first. Fortunately, we can easily use a *one-hot encoding* scheme to turn a given sequence of tokens into a sequence of binary vectors, and this type of representation is often used when a token (or sequence of tokens) is to be outputted by a model, in so-called *"Seq2seq"* (lit. "sequence to sequence") approaches, such as machine translation, text summarization, etc. However, when considering the inputs of a model, this process has many disadvantages: first, this type of encoding scheme is very inefficient and expensive dimensionality-wise, as the size of these vectors will grow linearly with the number of distinct possible tokens, which, while not necessarily too problematic with current available computing hardware, may still hinder proper learning due to the much larger input feature-space to explore. While one could in theory circumvent this issue by instead using an ordinal encoding scheme, this would require imposing an arbitrary ordering of the tokens along a single dimension, which is usually undesirable with categorical features, unless they happen to be naturally mappable onto a single axis in this way. Unlike vision, audio, or other numeric feature types, words (or any other type of linguistic token) generally cannot be naturally encoded into a vector space which preserves their original semantics completely, unfortunately (perhaps with the exception of linguistic numerals, like "two" mapping onto the integer value 2 for example).

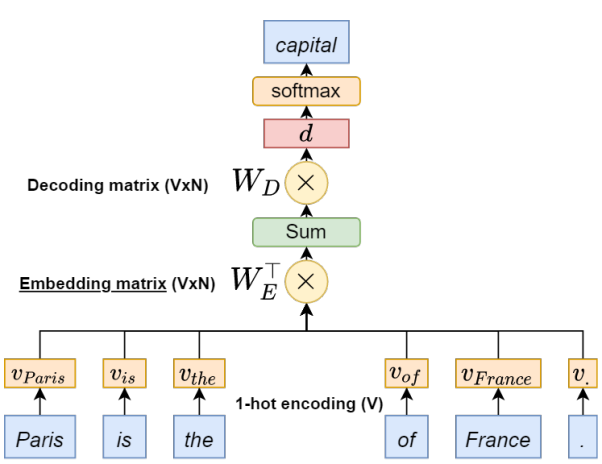
2.1.2 Distributional Semantics and Word Embedding Models

One way this problem was solved, and which was one of the largest factor in enabling the rapid evolution of deep learning approaches for NLP, was through the development of methods based on distributional semantics, first popularized by Harris [1954] and Firth [1957] as the hypothesis that one of the best and most fundamental way, absent any prior knowledge, to characterize a word is "by the company it keeps", in essence, by the contexts in which this word occurs in natural language. Based on this hypothesis, using statistical and probabilistic analysis of large text corpora, and, later on, dimensionality reduction techniques [Bengio et al., 2003, 2006], one

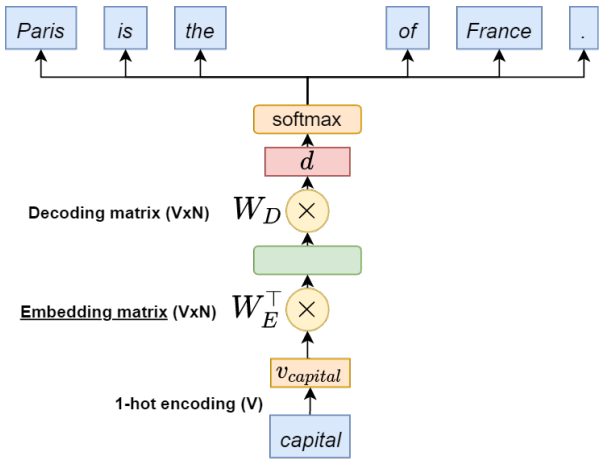
can construct a dense latent semantic space, in which the words in a vocabulary are embedded as vectors whose positions reflect the captured distributional properties of the text corpus used. One big advantage of this approach is that it does not require any human-produced annotations or supervision of any kind, other than large quantities of reasonably varied natural language texts, which are easily available nowadays thanks to the massive amounts of content available on the internet.

Initially, this idea was implemented via global matrix factorization techniques (such as singular value decomposition), for example in the so-called *latent semantic analysis* (LSA) (also known as latent semantic indexing, or LSI, in the field of information retrieval) [Deerwester et al., 1990] method, which, when applied on collected context-term co-occurrence statistics (where the context can be other terms, or entire documents, which was the initial point of interest of the method, in the field of information retrieval), could be used to construct a low-rank, thus in theory less sparse and de-noised, matrix approximation of those statistics. The produced vectors could then be used as numeric representations for their corresponding terms, with very interesting properties: indeed, the derived relatively low-dimensional (usually a few hundred dimensions) vector space, by construction, geometrically encodes the distributional hypothesis, such that terms which occur in similar contexts will have their corresponding vector representations appear "close" (usually, in terms of cosine similarity, as the magnitude of those vectors is more closely related to their a priori, uncontextualized probabilities, which is not of as much interest [Levy et al., 2015]) in the embedding space. This property can thus be immediately useful as a word similarity metric. It can also then later be extended as a document-level metric, by the weighted averaging of the relevant term-vectors.

However, in addition to this captured notion of similarity, it was later found that these constructed embedding spaces also geometrically encode various syntactic and semantic properties [Mikolov et al., 2013c]: mainly, when looking at the representations of a pair of words related in some specific way, for instance, $(v_{\text{apple}}, v_{\text{apples}})$ for a singular/plural-based relationship, or $(v_{\text{man}}, v_{\text{woman}})$ for gender-based relationship (where v_{word} is the vector representation of "word"), if one takes the offset between the pair and transposes it onto another term-vector, the result often happens to be closest to the term-vector one would expect to find if this offset somehow encoded the given relationship, e.g. here, we could expect to find that $v_{\text{apples}} - v_{\text{apple}} + v_{\text{car}} \approx v_{\text{cars}}$ or $v_{\text{woman}} - v_{\text{man}} + v_{\text{king}} \approx v_{\text{queen}}$. The reasons behind such linear relations appearing in spaces produced by non-linear methods, and why these allow these kinds of analogical structures ("*A* is to *B* what *C* is *D*") to be encoded as such, was explored by Arora et al. [2016], by modelling natural language corpora production as a generative process involving a random walk over a "discourse" space, which informs what types of word are likely to be generated at each time-step. The fact that such details can be captured in the dimensions of these embedding matrices means that downstream neural models can exploit these representations through different



(a) *CBOW* model architecture: the words in the context window of the target word (*"capital"*) are first 1-hot encoded (with a vocabulary of size V), then individually projected by the learned embedding matrix W_E, added together, before being decoded by the learned W_D matrix (which is discarded after training) while maximizing the probability of predicting the target word from output distribution of the word vocabulary d.



(b) *skip-gram* model architecture: the target word (*"capital"*) is first 1-hot encoded (with a vocabulary of size V), projected into the word embedding space by W_E, then independently maximized to be decoded into the words in the target's context window.

Figure 2.1: Schematic Illustrations of the two encoder-decoder architecture variants typical of the WORD2VEC word embedding model.

relatively simple computations (in terms of non-linearity requirements in particular) to extract rich types of semantic or syntactic information. For example, Kober et al. [2017] show that even when polysemic and context-dependent terms only have a single vector representation, it is nonetheless possible to disambiguate their senses contextually through simple vector composition operations (such as pointwise addition), which can be easily learned by most neural architectures. Arora et al. [2017] show similar results on different tasks and datasets, by using a weighted average of word representations to construct sentence embeddings, to achieve quite decent performance at very little additional costs. Arora et al. [2018], on the other hand, demonstrate that polysemous term-vectors can be disambiguated in and of themselves, as they seemingly are linearly composed of multiple *sense* component vectors, which can be recovered more or less successfully through sparse coding.

Later on, with the advances in and revival of neural network models, efficient implementations of these henceforth called *word embedding* approaches for very large corpora became possible [Morin and Bengio, 2005; Mnih and Hinton, 2009], and in 2013 at Google, Mikolov et al. [2013a; 2013b] introduced the now-famous WORD2VEC model, with its two encoder-decoder architecture variants (see Figure 2.1): *CBOW* (Continuous Bag-Of-Words), in which a neural model is trained to predict a word given a surrounding window (usually of size 5, i.e. two words on each side) of context words as input, and *skip-gram*, in which the model is trained

with the opposite objective. In both variants, once trained on large quantities of text, one of (or potentially, both, combined in some fashion, such as averaging) the learned parameters matrices is saved and then used as the output word embedding matrix. While the architectures and implementations for these neural models is fundamentally different from the earlier global matrix factorization techniques, Levy and Goldberg [2014] have shown that they in theory effectively optimize very similar objectives, and each method only really differs by its choice of defaults for a set of (potentially implicitly defined) hyperparameters. After the introduction of this model almost a decade ago, many variants and improvements were proposed and explored, whether they be combinations of global statistics factorization techniques with neural local context-window–based approaches, as found in the GloVe [Pennington et al., 2014] model, or later on attempts to further enable term disambiguation using a context-aware model, such as ELMo [Peters et al., 2018]. The general idea was also expanded to construct vector representations for other types of linguistic semantic units, such as subwords or character *n-grams*, to allow word embedding models to handle out-of-vocabulary tokens by composing already encoded subword components, as shown in the fastText model [Bojanowski et al., 2017] for instance. In the other direction, different approaches were proposed to encode and vectorize entire sentences through similar means (predicting context sentences given a particular sentence, or vice versa), producing general-purpose sentence embeddings, such as with the Skip-Thought [Kiros et al., 2015] or InferSent [Conneau et al., 2017] models, and later on with the various Transformer-based approaches like BERT or GPT.

2.1.3 Lack of Interpretability

While these approaches have enabled state-of-the-art results in many NLP tasks by providing efficient and easy to exploit numeric representations for words, subwords, or whole sentences, one big limitation is the lack of human interpretability of those vectors: in contrast to a vectorized picture, usually encoded as a 3-channel integer matrix, where each 3-tuple component can be interpreted as the tricolor value of the corresponding pixel in the image, the values in the dimensions of a word embedding vector usually have no discernible independent meaning to them. Indeed, in these traditional construction processes, no additional structural constraints are applied to the produced embedding space (other than very often a normalization of the word embedding vectors [Levy et al., 2015]), and effectively, the vector representations in most dense approaches can be freely unitary-rotated without losing their effectiveness as inputs in downstream neural models. This poses an issue with regard to explainability and interpretability in NLP models which make use of these types of representations, as this would limit the use and capabilities of explanation methods, in the sense that explanations referring to individual components of vectorized inputs, as they are what a model's internals ultimately manipulate, would be more or less meaningless: word embedding vectors would most likely have to be considered atomic and opaque components of an input sentence, for the purpose of

word2vec	
science	insult, rivalries, reactors, mw, armistice, massacre editing, airplay, cds, professionally, songwriter, screenplay ss, rbi, viii, 2d, xiii, shortstop
car	cox, colonists, expedition, ibm, usb, abduction bt, capt, ss, sr, casa, xiii dec, mw, cartoonist, poker, sketch, rapids
teacher	ore, greens, badminton, hymns, clay, gardener torture, abduction, executions, hostages, deportation, kidnapping heiress, mistress, photo, granddaughter, ap, latin
GloVe	
science	carbon, emissions, malayalam, dioxide, gases, revised algorithms, computational, anime, manga, renaissance, jens binary, finite, algorithm, algorithms, discrete, circuits
car	lama, apartment, mg, dalai, cream, milk propeller, tractor, barrel, mounted, rbi, drum championship, champion, championships, cup, debut, tournament
teacher	amateur, practitioner, archaeologist, educator, non-profit, physician sanskrit, canton, aristotle, breeding, pupil, polytechnic championship, champion, championships, cup, debut, tournament
fastText	
science	uavs, badminton, gridcolor, loneos, boldklub, medalists sportspeople, zl, gmina, colspan, camogie, njn romanized, laude, idaea, ploceus, hoseynabad, pygmaea
car	agung, setiawan, sjk, mhk, tenggara, sutil gunboats, cruisers, eprix, frigates, tramways, autódromo fsk, stratigraphic, ukr, scalemajor, altai, plesetsk
teacher	sofla, mowtowr, woredas, darreh, pā, plíšková sportspeople, zl, gmina, colspan, camogie, njn reductase, västerbotten, pär, åberg, purine

Table 2.1: Illustration of the non-interpretability of dense word embedding models: for three well-known models, we display the top-6 words with the largest values in the top-3 most active dimensions for each of three arbitrary words ("science", "car", and "teacher"). As can be qualitatively observed, it is practically impossible to determine any lexical correlations between the most active words of a dimension.

input feature attribution, or local perturbation methods.

To illustrate this lack of intrinsic interpretability in common dense word embedding models, we can look at "active" dimensions for particular words, that is, the dimensions with the largest values (positive, though negative factors could be looked at too, but it would yield similar results) in the corresponding vector representation. Each of these dimensions can then be characterized by the most active words in them, which, similarly, are the words whose vectors have the largest values in these dimensions. As we can clearly see in Table 2.1, this type of analysis reveals there is little to no interpretable meaning to individual dimensions in these types of embedding models.

Ideally, it would be useful to find a basis for these embedding spaces, such that each dimension is intrinsically meaningful for the purpose of human analysis, while preserving the geometric properties which make these representations effective at downstream NLP tasks.

We will now see how a significant amount of work has been done to attempt to achieve this goal of obtaining more interpretable word embeddings.

2.2 Interpretable Embedding Models

To achieve better interpretability in vectorized word representations, different approaches have been explored, which can be roughly divided into two classes: constraint-based embedding models, and models enriched with prior information.

The latter essentially consists in attempting to inject *a priori* semantic information into the embedding models to improve their interpretability. For example, Hurtado Bodell et al. [2019] attempt to use information priors in the form of word vocabulary pairs displaying a particular semantic relationship (e.g. {*man*, *brother*, *king*} and {*woman*, *sister*, *queen*}, for a gender-based relationship), to restrict and guide the learning model such that a given specified dimension discriminates (more or less strictly) these words along its axis. With this type of approach, one can carefully control and handcraft specific meaningful dimensions, while still leaving the models enough degrees of freedom to learn efficient representations. However, that is also a limitation in that it is necessary to construct these vocabularies for each semantic relationship desired, which may have a high annotation cost. Somewhat similarly, Fyshe et al. [2015] modify the scheme proposed in the NNSE [Murphy et al., 2012] approach (which we will discuss more in detail later on), by adding a term to the objective function which enforces a certain vector representation compositionality with regards to annotated phrases (in this case, adjective-noun or noun-noun pairs), extracted from dependency parsing features found in an earlier study on semantic composition of vectors [Fyshe et al., 2013]. The latter study also provides an evaluation task based on brain activity measurements in participants reading these phrases, a type of external data which, in Fyshe et al. [2014], is used to create a joint model (with the same starting word embedding approach, NNSE), thus combining two different data modalities to attempt to constraint and improve the interpretability of word representations.

The approaches on which we've focused on here however are of the first variety, constraint-based interpretable embeddings, as they seem to be the most well-studied and are used as a conceptual basis or starting point for numerous approaches of the second class. The main two constraints which are employed by these approaches to enhance interpretability are sparsity and non-negativity. Indeed, a wide range of contributions [Lee and Seung, 1999; Fyshe et al., 2014; Faruqui et al., 2015; Dahiya et al., 2016; Trifonov et al., 2018; Subramanian et al., 2018] have shown that these two properties are essential to produce distributional models where each dimension is independently semantically meaningful. Sparsity of the representations, that is, an upper-bound constraint on the number of non-zero (or occasionally relaxed up to a threshold above zero) dimensions for each word in the model's vocabulary, is desirable for

many reasons: first of all, experiments and evaluations have shown [McRae et al., 2005; Murphy et al., 2012; Fyshe et al., 2014] that humans have a preference towards describing objects and concepts with a few specific and strongly related words, rather than numerous but more weakly related associations. Similarly, models exploiting such constrained representations may benefit from this sparsity: indeed, for many NLP tasks, only a limited set of semantic features are actually relevant, for example, in sentiment analysis, while the semantics of an entire sentence may require a contextualized understanding of its constituting terms, in theory only their sentiment value should be taken into account, in the spirit of the task. Equivalently, this may also help produce less biased models, as there are properties of words which should ideally not be taken into account, say, gender, when performing tasks which do not in theory require them. Sparsity also lends itself to easier exhaustive analysis of the relevant components of an input, for the purpose of explanation methods. On the other hand, non-negativity is mainly useful as it allows to reason about the non-zero components of a representation as a degree of "participation" of different parts, in this case, abstract *senses*, in the whole object that is the word in question. Many have argued [Lee and Seung, 2001, 1999; Hoyer, 2002] that allowing different features of an object to cancel out through subtraction is generally undesirable, and indeed, when dealing with words, it is difficult to imagine what a negative value for a particular semantic component could mean, in addition to conflicting with the naturally intuitive process of constructive description which we mentally and verbally employ for most abstract objects.

Such constraints can be applied to word embedding approaches in essentially two different ways: either by starting from scratch and modifying or creating a new encoding process which includes these constraints in some fashion, or, by starting from an existing (usually dense) word embedding model, and transforming its vectors *a posteriori*, such that they acquire the desired properties, through matrix factorization or basis rotation, for example. In the next two sub-sections, we present and discuss different such approaches which can be found in the relevant literature, with varying levels of complexity, and which have experimented upon here. We also include our own very simple baseline model (NMF300), which illustrates how even just using non-negative matrix factorization techniques can already yield decent interpretable embeddings.

2.2.1 A Priori Constrained Interpretable Embeddings

These first type of approaches rely on a construction method which implicitly or explicitly imposes these constraints, sparsity and/or non-negativity (ideally both), on the produced embedding space. One relatively simple way to achieve this is by using *non-negative matrix factorization* (NMF) techniques, in a setup otherwise similar to the global matrix factorization approaches as can be found in LSA (see 2.1.2), that is, by applying this factorization on term-context co-occurrences statistics, collected on large text corpora. Thanks to the work by Lee

and Seung [1996; 1999; 2001] on these techniques, efficient algorithms exist to compute such factorizations, using multiplicative update rules which come in two variants, one minimizing the conventional least squares error, and the other the Kullback-Leibler divergence metric, with both having been proven to converge. Different approaches found in the literature use these objective functions as a basis, usually adding some additional term or terms to strengthen or enforce additional constraints on top of the explicit non-negativity and implicit sparsity, but, as we have experimented with our own basic model, NMF300 (which is discussed in more details in Section 3.1), these are not necessarily required to attain interpretable dimensions in the resulting embedding matrix.

Notation: For reasons of consistency, we have adapted the formulation of each objective function so that they make use of the same notation, when possible. In the following, $X \in \mathbb{R}^{V \times C}$ represents input statistical observation data, usually, the term-context co-occurrence matrix computed on the model's training corpus, where V is the size of the model's vocabulary and C is the number of context features considered for the co-occurrences statistics. $W \in \mathbb{R}_{\geqslant 0}^{V \times d}$ is the resulting sparse non-negative word embedding matrix, where d is the chosen embedding dimension size (usually 300, if not otherwise specified), $H \in \mathbb{R}^{d \times C}$ the second part of the factorization of X, often called the "dictionary" or basis matrix, when applicable, or some other learned parameter (usually discarded at the end either way, as W is intended to be used alone as the output embedding matrix). For any matrix A, $A_{i,j}$ corresponds to a single element, and $A_{i,:}$ and $A_{:,j}$ to its i-th row and its j-th column as a whole, respectively. Thus, the word embedding vector for the i-th word in the vocabulary will be found in $W_{i,:}$. $\|A\|_F$ is the Frobenius norm of A : $\|A\|_F = \sqrt{(\sum_{i,j} |A_{i,j}|^2)}$. $\otimes$ and $\oslash$ are respectively the element-wise product and element-wise division operators.

To qualitatively compare these interpretable approaches , both with each other as well as against the more traditional dense models discussed before, Table 2.2 showcases the most active dimensions for various words, similar to Table 2.1 (see Section 2.1.3).

NNSC: While not specifically intended to be used in NLP, ***Non-Negative Sparse Coding*** (**NNSC**) [Hoyer, 2002] was one of the first methods proposed which explicitly attempts to enforce both non-negativity and sparsity in the produced representations, by using non-negative matrix factorization under some additional constraints. The author proposes the following

objective function, $C_X(W, H)$, for the factorization[1]:

$$\arg\min_{W,H} C_X(W, H) = \frac{1}{2}\|X - HW\|_F^2 + \lambda \sum_{i,j} W_{i,j}$$

under the non-negativity constraints applied to both matrices $\forall i, j : W_{i,j} \geqslant 0, H_{i,j} \geqslant 0$, as well as the unit rescaling constraints applied to the columns of H, $\forall j : \|H_{:,j}\|_2 = 1$, and where λ is an positive hyperparameter controlling the trade-off between the accuracy of the factorization and the sparsity of the output embedding matrix W. The rescaling constraint is necessary to ensure the second term of the objective, which enforces the sparsity of W, does not lead to the uncontrolled scaling up of H and scaling down of W by the factorization algorithm to minimize the objective further and further. The algorithm used to minimize this objective is inspired by the iterative multiplicative update rules from Lee and Seung [2001] for the least squares error metric, with mainly one significant variation. Where both W and H could be updated with the following iterative update rules under the sole non-negativity constraint (where $\leftarrow$ is the "update" operator):

$$W \leftarrow W \otimes (XH^\top) \oslash (WHH^\top) \qquad\qquad H \leftarrow H \otimes (W^\top X) \oslash (W^\top WH)$$

these cannot be used as is alongside the unit rescaling constraint introduced above. To solve this issue, the author proposes modifying the update steps for the H matrix (on which the constraint is applied) into a projected gradient descent setup, where the following free descent step is first applied (where H' is a temporary variable, and α the gradient descent step-size):

$$H' \leftarrow H - \alpha(HW - X)W^\top$$

followed by a negative-clipping and a column-rescaling steps on H', before finally setting $H \leftarrow H'$ to continue with the multiplicative update of W as usual. The author experimented with this approach on image data, showing that while a purely NMF-based method can successfully and efficiently extract the base features of artificially generated instances, the addition of the sparsity constraint appears to enable overall more efficient representations as well as adding more robustness when the embedding space is *overcomplete* (when d, the chosen dimensionality for the output vector representations is larger than is necessary to encode all the given data).

[1] The author chose to use the second matrix in the product of the factorization as the one to keep as the encoding of the original data, hence the HW term instead of the more common WH.

NNSE: Inspired by the NNSC approach, Murphy et al. [2012] proposed the ***Non-Negative Sparse Embedding*** (**NNSE**) model[2] for NLP, with a slightly different objective function:

$$\arg\min_{W,H} C_X(W,H) = \sum_{i=1}^{V}(\|X_{i,:} - W_{i,:} \times H\|_F^2 + \lambda\|W_{i,:}\|_1)$$

under the single non-negativity constraint $\forall i,j : W_{i,j} \geqslant 0$ (different from NNSC), and the soft unit rescaling constraint $\forall i : \|H_{i,:}\|_2 \leqslant 1$. Similar to NNSC, this training objective is convex when doing alternating update steps on W and H while keeping the other matrix fixed, and the authors use the online dictionary learning algorithm presented in Mairal et al. [2010] to optimize it. NNSE embeddings were computed on dependency parsing statistics extracted from a large English web-corpus, the ClueWeb09 [Callan et al., 2009] dataset[3], of which 16 billion words and 10 million documents were used, with a vocabulary size of $V = 40\,000$ ($35\,560$ after frequency cutoff and positive pointwise mutual information filtering), and a feature size of $C = 2\,000$, resulting from the concatenation of two $1\,000$ dimensional singular value decomposition (SVD) matrix factorizations on both term-term co-occurrence counts and term-document co-occurrence counts, which was intended as a way to simplify the complexity of the optimization problem, by prefiltering some of the noise in the data.

Word2Sense: Panigrahi et al. [2019] have proposed ***Word2Sense*** (lit. "word to sense"), an LDA-based (Latent Dirichlet Allocation) interpretable word embedding method. Contrary to the previous methods, based on non-negative matrix factorization, in this approach, the term-term co-occurrence matrix is assumed to follow a generative model (similar to [Arora et al., 2016]), where a *sense model* is inferred (in a somewhat similar fashion to topic modelling), as a set of d' Dirichlet distributions over the words which can appear in the context window of any given word. Any word can then itself be encoded as a sparse d-dimensional Dirichlet distribution over these learned senses. Because this method was found to be prone to returning a large number (d') of redundant senses (and thus leading to a large d number of final dimensions for the produced embeddings), a merging step is applied, using the Jensen-Shannon (JS) divergence metric ($D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M)$, where $M = \frac{1}{2}(P+Q)$ and $D_{KL}(P||Q) = \sum_x P(x)log(\frac{P(x)}{Q(x)})$ is the Kullback-Leibler divergence) applied on each pair of sense-distributions to drive an agglomerative clustering algorithm. Starting from an initial $d' = 3\,000$ senses, the authors find that merging a quarter of the most redundant senses was optimal, resulting in a final embedding dimension size of $d = 2\,250$. The original model is trained on 3.5 billion words from the UKWAC [Ferraresi et al., 2008] and Wackypedia [Baroni et al., 2009] corpora[4], with a vocabulary of $255\,434$ words.

[2]available at: http://www.cs.cmu.edu/~bmurphy/NNSE/
[3]available at: https://www.lemurproject.org/clueweb09.php/
[4]available at: https://wacky.sslmit.unibo.it/doku.php?id=download

2.2.2 A Posteriori Constrained Interpretable Embeddings

In this second type of approach, instead of starting from scratch from text corpora statistics, a method is used to transform an existing dense word embedding matrix into a non-negative and/or sparser form, in order to increase its interpretability.

One way to achieve this is to consider this task to be a basis rotation problem: for instance, Park et al. [2017] propose using an algorithm from the Exploratory Factor Analysis literature, which attempts to minimize the given embedding matrix's row and column complexity (leading to fewer large values appearing in each row or column) to induce better interpretability without fundamentally modifying the structure of the embedding space, which preserves its exploitable geometric features. Similarly, Dufter and Schütze [2019] explore different rotation methods, such as linear SVMs and variations of the DENSIFIER algorithm introduced by [Rothe et al., 2016], in order to impose correlations between chosen linguistic features and a portion of the embedding matrices' dimensions. While these rotation-based approaches offer multiple advantages, they also more strongly rely on the quality of the initial dense embedding vectors to produce good representations, as the transformations (and therefore constraints) they apply on them is by-design more limited. Some other methods have been proposed which do not follow these restrictions, and which can therefore modify the structure of the starting space as needed to better enforce new constraints on it.

SPOWV: Faruqui et al. [2015] have proposed the ***Sparse Overcomplete Word Vector*** (**SPOWV**) Representations method, based on sparse coding, akin to NNSC and its variants, but applied to existing dense word vectors instead of corpora statistics, with the following objective function (in which, as such, x refers to a dense embedding matrix of size $V \times C$):

$$\arg\min_{W,H} C_X(W,H) = \sum_{i=1}^{V}(\|X_{i,:} - W_{i,:} \times H\|_F^2 + \lambda\|W_{i,:}\|_1 + \tau\|H\|_F^2)$$

where λ and τ respectively control the l_1-norm sparsity constraint on W and the l_2-norm soft bounding constraint on H. Unlike the previous methods, this method uses a specialized variant of online adaptive gradient descent (ADAGRAD) [Duchi et al., 2011], specifically adapted to handle the l_1 regularization term [Xiao, 2009], while also clipping the negative terms in W to 0 to enforce non-negativity. The authors experiment both with adding or removing the non-negativity constraint, as well as with the binarization of the non-negative embedding vectors. Binarization is argued to even further increase the interpretability of the word vectors, though at the cost of an even higher loss in information density. To partially combat this, the constructed representations are so-called *overcomplete*, because the size (d) of the output vectors are intended to be significantly larger than the input embeddings' dimension (C), to account for the information capacity loss due to the various constraints applied. In the original

publication, the authors find that a 10-fold increase is the most effective (that is, starting from $C = 300$ wide dense vectors, ending with $d = 3\,000$ wide sparse vectors), when experimenting with GloVe or word2vec initial embeddings.

SPINE: Subramanian et al. [2018] introduced the ***Sparse Interpretable Neural Embeddings*** (**SPINE**) model, based on denoising k-sparse auto-encoders (AE) which, similarly to SPOWV, are applied to existing dense vectors from methods like GloVe or word2vec, trained with a three-part objective function:

$$\begin{aligned}\arg\min_{W,H} C_X(W,H) = &\underbrace{\frac{1}{V}\sum_{i=1}^{V}\left(\|X_{i,:} - \mathrm{Dec}\big(\mathrm{Enc}(X_{i,:})\big)\|_F^2\right)}_{\text{RL}} \\ &+ \underbrace{\lambda_1 \sum_{h=1}^{d} \max\left(0, \left(\frac{1}{V}\sum_{i=1}^{V}\mathrm{Enc}(X_{i,:})_{:,h}\right) - \rho\right)^2}_{\text{ASL}} \\ &+ \underbrace{\lambda_2 \frac{1}{m}\sum_{i=1}^{V}\sum_{h=1}^{d}\Big(\mathrm{Enc}(X_{i,:})_{:,h} \times \big(1 - \mathrm{Enc}(X_{i,:})_{:,h}\big)\Big)}_{\text{PSL}}\end{aligned}$$

where $\mathrm{Enc}(X_{i,:}) = f(X_{i,:} \times P_e + b_e)$ (where f is the activation function of the encoder, discussed later) and $\mathrm{Dec}(Z_{i,:}) = Z_{i,:} \times P_d + b_d$ are respectively the encoding and decoding functions of the auto-encoder, with $P_e \in \mathbb{R}^{C\times d}, b_e \in \mathbb{R}^{1\times d}, P_d \in \mathbb{R}^{d\times C}, b_d \in \mathbb{R}^{1\times C}$ respectively the parameters and biases of the encoder and decoder, and $\lambda_1, \lambda_2, \rho$ respectively the hyperparameters controlling the **A**verage **S**parsity **L**oss (ASL) term, the **P**artial **S**parsity **L**oss (PSL) term, and the desired sparsity factor for the columns ($h \in 1..d$) of the encoded vectors. The **R**econstruction **L**oss (RL) term is similar to those seen previously, except the matrix factorization has been replaced with the neural encoder-decoder model. At the end of the training, the output sparse interpretable vectors are found by running the trained encoder module on the input dense vectors, $W_{i,:} = \mathrm{Enc}(X_{i,:})$. The **A**verage **S**parsity **L**oss (ASL) term pushes each dimension (column) in the output representations towards a ρ sparsity factor (or lower). However, alone, this term is not enough to enforce sparsity in the embedding dimensions, as it is only concerned with the averages of the columns (which might thus all take low, but non-zero values), so, in order to further push the close-to-zero values further towards zero, and at the same time attempt to produce a "soft" binarization (by equivalently pushing the higher values towards 1), the **P**artial **S**parsity **L**oss (PSL) term is added. To enforce a non-negativity constraint within this approach, one only needs to select an appropriate positive activation function f, such as a *Rectified Linear Unit* $\big(\mathrm{ReLU}(x) = \max(0, x)\big)$ or sigmoid/logistic function $\big(S(x) = (1 + e^{-x})^{-}1\big)$. The authors however discard the latter option (because of its asymptotic nature towards 0 values), and propose a variant to the former, *capped* ReLU $\big(\text{cap-ReLU}(x) = \max(0, \min(x, 1))\big)$, to further

NNSE	
science	genetics, biology, physiology, microbiology, ecology, biochemistry low-budget, time-lapse, live-action, science-fiction, zapruder, art-house cracks, seams, holes, scratches, bumps, scars
car	nissan, chrysler, volkswagen, mazda, chevrolet, buick tiers, subgroups, subfamilies, subtypes, genders, subsections berlin, munich, frankfurt, hamburg, bonn, mannheim
teacher	educator, historian, lecturer, researcher, scientist, essayist dermatologist, pathologist, veterinarian, psychiatrist, pediatrician, neurologist first-year, pre-med, vocational, tertiary, after-school, undergrad
SPOWV	
science	bone, adult, hans, science, cm, honolulu bowie, aired, licensed, ncaa, freely, broadcasting innovation, develops, horsepower, ridge, collaborations, futures
car	card, passport, mechanic, driver, thief, stabbed ferrari, yamaha, chevrolet, cavalry, fia, caste angles, curve, speeds, disk, motorsports, sheer
teacher	theatrical, conservatory, attic, painter, fairs, venetian verde, de, nord, emilio, kashmir, casa motorsports, old, tourists, waugh, healy, hostage
SPINE	
science	graduate, doctorate, phd, bachelor, anthropology, degree honorary, 1879, 1826, 1893, faculty, sciences institute, psychiatry, sciences, physics, laboratory, research
car	motor, engine, mazda, chrysler, coupe, chevrolet lanes, speeding, freeway, drivers, highway, brakes deck, asphalt, floating, leaf, door, bottles
teacher	diploma, curriculum, students, school, grades, exams sculptor, inventor, physicist, collector, playwright, businessman sawyer, jared, writer, cameron, leslie, moe
Word2Sense	
science	anthropology, sociology, sciences, humanities, science, geography ap, calculus, mathematics, placement, classes, excelled quantum, physics, mechanics, astronomy, einstein, chemistry
car	leasing, hire, car, dealer, rental, dealers parking, bays, ample, parked, attendants, spaces auto, insurance, quote, broker, cheap, owner
teacher	teachers, classroom, teacher, teaching, teach, english portfolio, pupils, stage, lesson, key, teacher secondary, lea, primary, pupil, pupils, academies

Table 2.2: Illustration of the much better qualitative interpretability of interpretable word embedding models, compared to dense models (see Table 2.1): with varying degrees, each model seems to capture different semantics in the active dimensions for each word.

enforce the soft binarization constraint.

2.3 Evaluating Interpretability

While a purely qualitative analysis of the active words in the dimensions of such interpretable embedding models is sufficient to showcase their differences compared to dense models (see Table 2.1 and Table 2.2), a more quantitative and methodical evaluation of their practical interpretability is necessary, in particular to compare these models to each other. One of the main desiderata for such interpretable representations concerns the *discriminative power* of their associated embedding spaces' dimensions: as each of these dimensions seems to encode a particular semantic attribute or class of the terms in the model's vocabulary, it would be

desirable if any given single dimension could clearly be used to discriminate between terms where this dimension is active and those where it is not.

One way this property can be comparatively evaluated is through the *Word Intrusion Detection* task, first introduced by Chang et al. [2009], and which can be summarized as such: for a given dimension of a non-negative sparse embedding model, we can select a small shuffled sample of words (4 or 5, for example) which are all highly active in this particular dimension. These should thus share at least one semantic commonality, associated with the dimension being studied. If one then selects and randomly inserts into this sample an "intruder" word, which is comparatively not active in this dimension, can a human evaluator accurately identify this intruder, given no other information than the list of words? This task initially designed to examine the semantic coherence of topics discovered by latent semantic analysis and other topic modeling methods, and as word embedding models, especially the interpretable ones, are conceptually quite similar to these approaches, this type of evaluation fits the problem well, so much so that it has seemingly become the de facto standard method over the years [Murphy et al., 2012; Fyshe et al., 2014; Faruqui et al., 2015; Subramanian et al., 2018]. Usually, the intruder is drawn from the bottom $b\%$ of the words sorted by their corresponding vector's value in the given dimension, and also in the top $t\%$ words in another different dimension (presumably to avoid picking overall "low activations" words, which usually correspond to stop-words or equivalents, such as *"the"*, *"of"*, *"a"*, etc.). Subramanian et al. [2018] for example suggest picking $b = 50\%$ and $t = 10\%$.

2.4 Downstream performance Evaluation

Outside of evaluating the interpretability of these models, it is also necessary to ensure they still perform adequately compared to their denser alternatives when used as part of standard NLP tasks and architectures. Due to their popularity, few linguistic tasks have not yet seen a number of neural models being proposed and evaluated to solve them, and most of these models use some type of word embedding representations as part of the processes to encode their inputs. In particular, many tasks fall into the sequence classification template, where each instance is composed of one or multiple pieces of input text, associated to one (or more rarely, multiple) categorical output label(s). Fit into this template (non-exhaustively): sentiment analysis, topic labeling, email spam or hate speech detection, closed-ended question answering, relation classification, etc. In essence, any categorical property or relationship which can be associated (manually or automatically) with one or multiple pieces of text may constitute a sequence classification task, as long as enough data can be collected and labelled appropriately. The principal advantage of using such downstream tasks to evaluate the performance of a given interpretable word embedding model is that their particular properties, mainly sparsity and non-negativity, should in theory not only not be detrimental (depending on how much

information loss has been traded off in the constraints enforcing process, of course), but may enable simpler models to more easily find exploitable correlations between semantic or lexical features and the tasks' labels, as those will be represented in individual dimensions of the input vectors, most likely in the form of sets of sparse linear combinations of specific dimensions.

We now succinctly present a few such tasks, with their accompanying datasets, from the relevant literature.

BoolQ: Introduced by Clark et al. [2019], **BoolQ** (for "Boolean Questions") is a yes/no question answering (QA) dataset, with the specificity that these questions are so-called "naturally occurring", meaning that they were collected in an unprompted fashion, in this case, from anonymized Google search engine queries: first, aggregated queries which have been heuristically identified to be candidate yes/no questions are gathered, then, for each query which returns a relevant Wikipedia article within the 5 first results, a human annotator is tasked with finding a short passage (on average, around 100 words) in the linked article which contains enough information to correctly answer the question, according to the gold answer which they also annotate. This pipeline is inspired by the one used for the Natural Questions [Kwiatkowski et al., 2019] benchmark dataset, which **BoolQ** incorporates $\sim$ 3 000 yes/no questions and passages from, for a total of $\sim$ 16 000 instances, split into a $\sim$ 9 400 train set, and both $\sim$ 3 200 development set and test set (the latter being not available at this time, unfortunately). The authors consider this task to be unexpectedly challenging, despite the boolean answers, as they require looking for potentially complex information in the accompanying passage with regards to the question, which is akin to the skill-levels required for Textual Entailment (TE, or Natural Language Inference, NLI) tasks, which the authors empirically confirm by showing that transfer learning from TE-trained models leads to better performance than starting from models trained on paraphrase or extractive question answering tasks.

Emergent: Ferreira and Vlachos [2016] propose the **Emergent** dataset for stance classification, in which the task is to classify the journalistic stance with regards to a claim sentence, from articles' headlines related to the claim, in which each article can be labelled either *for*, *against*, or simply *observing* (repeating without taking a position) the claim. This dataset contains 300 claims and 2 595 associated news articles, from the homonymous news rumor-debunking Emergent Project [Silverman, 2015], created and annotated by journalists as part of their work. This type of task and data is of particular interest nowadays, with the recent concerns and upsurge of so-called "fake news", making apparent the need for better fact-checking tools and services, which automation in the form of machine-learning may help to provide in a scalable fashion.

IMDB: Maas et al. [2011] make available a dataset for sentiment analysis, consisting of collected user movie reviews from the popular *Internet Movie Database* website (often abbreviated to *IMDb* or *IMDB*), with binary sentiment labels ("positive" or "negative") automatically mapped from the user-given review scores (on a 10 "stars" scale, with the following mapping: score $\leqslant 4 \rightarrow$ negative; score $\geqslant 7 \rightarrow$ positive; "neutral" reviews are not included).

SST: The **Stanford Sentiment Treebank** dataset [Socher et al., 2013] is one of the few fully labelled large-scale parse trees corpus for sentiment analysis, based on the movie reviews dataset from Pang and Lee [2005] and parsed by the Stanford Parser [Klein and Manning, 2003]: it contains $11\,855$ single sentences, which are broken down into a total of $215\,154$ unique phrases constituted into parse trees, which have all been annotated by 3 human judges into one of five polarity classes ("very negative", "negative", "neutral", "positive", "very positive"). This level of fine-grained annotation is necessary to study the compositional aspects of natural language, in this case, sentiment polarity. To handle this type of structured data, the authors propose the Recursive Neural Tensor Network (RNTN) architecture, which improves upon the Recursive Neural Network [Goller and Kuchler, 1996; Socher et al., 2011] (not to be confused with Recurrent Neural Networks) and Matrix-Vector Recursive Neural Network [Socher et al., 2012] architectures, by implementing a learnable tensor-based recursive composition operation, allowing the model to compute the representations for higher-level elements in the hierarchical parse-trees from lower-level elements.

Sarcasm: Oraby et al. [2016] provide the **Sarcasm** dataset, constituted of internet forum posts from the *Internet Argument Corpus (IAC) 2.0* [Abbott et al., 2016], which have been annotated for the presence of sarcasm (as well as the more specific class of sarcasm, mainly either *generic* sarcasm, *rhetorical questions*, or *hyperbole*), in parts automatically, using the *AutoSlog-TS* [Riloff, 1996] weakly-supervised lexico-syntaxic pattern-learner model to identify the more accurately non-sarcastic instances, with the remainder of them being manually annotated by workers on the crowdsourcing platform Amazon Mechanical Turk. As sarcasm is noted to be a somewhat uncommon occurrence in online debates (estimated to represent approximately 12% of the IAC [Walker et al., 2012]), the authors choose a conservative 20% ratio for the sarcastic class, representing a 6 out of 9 annotator agreement threshold, resulting in a total of $3\,260$ labelled posts for each class (sarcastic, non-sarcastic).

UR-FUNNY: The **UR-FUNNY** dataset [Hasan et al., 2019] is a multimodal humor detection (more precisely, punchline detection) dataset, incorporating textual, visual and acoustic modalities, extracted from publically available TED talks videos, which are remarked to be an excellent source of data for this type of task in particular for multiple reasons: indeed, not only do these digital talks include very reliably transcriptions, allowing for accurate audio-text alignment, but these also contain annotated audience markers which reflect the behavior of the audience,

in particular, laughter, which is here used to detect potential punchlines and their preceding setup/context (inspired by Chen and Lee [2017]). For the negative class instances, the authors simply sample an equivalent number of random video, audio and transcript intervals which are not immediately followed by an audience laughter marker, resulting in a balanced dataset containing $8\,8257$ multimodal (*punchline*, *context*, *label*) instances for each class (humorous, non-humorous), spanning over $1\,741$ speakers, $1\,866$ videos and 417 topics.

SNLI: [Bowman et al., 2015] provide the **Stanford Natural Language Inference** corpus for the Natural Language Inference (NLI) task, often interchangeably referred to as Recognizing Textual Entailment [Poliak, 2020]. This task consists in predicting whether a hypothesis sentence (e.g. *"There are children present."*) logically and semantically follows from a premise sentence (e.g. *"Children are smiling and waving at a camera."*), with three possible classes of relationships to predict: *entailment*, if the hypothesis follows from the premise (which is the case for the given examples previously); *contradiction*, if the hypothesis contradicts the premise; *neutral*, if the hypothesis does not relate directly to the premise. This type of task is considered particularly difficult, as it not only requires understanding the semantics of the two propositions separately (which often require some level of commonsense or world-knowledge), but it also requires abstract logical reasoning skills to assess the nature of the inference relationship between them. The data for this very large corpus ($\sim 570\,000$ instances) originates from the Flickr30k corpus [Young et al., 2014], which contains $\sim 160\,000$ image captions (spanning over $\sim 30\,000$ images), which were used as starting premise sentences. These were then manually augmented by workers on the crowdsourcing platform Amazon Mechanical Turk, who were tasked with writing three alternative hypotheses captions for each given premise caption, appropriately for each of the inference classes (*entailment*, *contradiction*, *neutral*), without being shown the actual picture associated with the premise caption (which may have caused the workers to unintentionally exploit non-textual information to craft the hypotheses, see Section 1.2.1). This dataset, and its multi-genre variant, MNLI [Williams et al., 2018], were found later on to be tainted with many annotation biases, as discussed previously in Section 1.2.1.

PDTB: The **Penn Discourse TreeBank** [Prasad et al., 2008] is built upon the same data as the Penn TreeBank corpus [Marcus et al., 1993], and constitutes a large-scale dataset manually annotated for discourse relations, on texts collected from the Wall Street Journal (WSJ) corpus. Discourse relations (also known as rhetorical relations), attempt to more or less formally characterize the textual relations between two (usually adjacent) segments of a discourse. These relations can either be realized explicitly, through some type of linking word or phrase, most often conjunctions (*and, or, because, when, although*, etc.), in which case the second argument of the relation is usually a syntactic dependent clause of the first, or, they can also hold implicitly, and must therefore be contextually inferred from the content of the two thus

Corpus	Train	Test	Classes	Class balancing
IMDB	25000	22500	2	eq.
BoolQ	9427	2943	2	true=62.3%, false=37.7%
Sarcasm	3754	469	2	eq.
UR-FUNNY	8074	1058	2	eq.
SST	8544	1989	3	positive=42.0%, negative=39.2%, neutral=18.8%
SNLI	549367	9824	3	entailment=33.4%, contradiction=33.3%, neutral=33.3%
Emergent	2076	259	3	for=47.7%, observing=37.0%, against=15.3%
PDTB	12907	1085	11	cause=26.5%, conjunction=22.1%, restatement=19.1%, contrast=12.4%, *reste*=19.9%

Table 2.3: Statistics about the different corpora used. For the PDTB corpus, only the 4 most major classes are listed (this corpus presents high class imbalance, especially in the test set, where some classes are not represented at all).

separate but generally consecutive sentences. In the PDTB, a relation is identified by its type (*Explicit*, *Implicit*, etc.) and by the connective expression linking its two arguments (spans of text): in the case of *Implicit* relations, an explicit connective which best expresses the inferred relation is manually inserted and annotated, as though the two argument sentences were linked clauses. In addition, a hierarchical *sense* annotation is provided, which makes more explicit the exact nature of the relation (within a 3 level hierarchy of tags, 4 coarse-grained *classes*, ~ 15 intermediate-level *types*, and ~ 40 fine-grained *subtypes*). While models fine-tuned for the prediction of explicit relations achieve accuracies above and around 95%, detecting and classifying implicit relations is still a much more difficult problem, with accuracies under 60% [Dai and Huang, 2019; Kim et al., 2020].

To fully exploit the interpretable nature of non-negative sparse embedding vectors while solving such tasks, however, the choice of a machine learning model is very important.

2.5 Intrinsically Interpretable Models

Indeed, while it would be possible in theory to simply replace the dense embeddings from any state-of-the-art NLP setup with interpretable embeddings, with just a few potential shape adjustments, the resulting architecture would very likely not become interpretable as a result: indeed, most machine learning models, in particular neural architectures, make use of a number of layered non-linearities, which when all combined, can in theory learn and simulate any mathematical function (depending on the depth and number of parameters available), which is what gives them their modeling strength, but also their major drawback of being opaque and difficult to interpret. By themselves, interpretable word embeddings would behave no

differently in such models than any other type of embeddings, i.e., their sparsity and non-negativity would most likely be "lost", probably right after passing through the first input layer. While one could consider using post-hoc explainability methods (which will be discussed later, in Chapter 4) alongside these interpretable representations to generate explanations for such models, an alternative would be to use models which are themselves intrinsically interpretable.

In practice, only a few types of architectures can be said to be truly intrinsically interpretable, with mainly two large categories:

- the Generalized Linear Models [GLMs; Nelder and Wedderburn, 1972] family, which encompasses and generalizes linear or logistic regression models, for which predictions always depend on the weighted sum of the inputs, making them easily interpretable: the "importance" of an input feature will always be proportional to the corresponding weight learned by these models. If the number of input features and thus learned weights is relatively large however, it may be difficult to exhaustively analyze all of them.

- Rules-based models, such as Decision Trees, which learn a finite number of single-feature branching decision rules, arranged in a tree, recursively dividing a dataset into parts, until each part can be accurately labelled. Other algorithms exist to extract such rules, not necessarily using a tree structure, such as Bayesian Rule Lists [Yang et al., 2017]. These rules can be exhaustively listed and are easy to understand, however, if the number of nodes in a tree becomes too large, it can become difficult to parse and understand all the learned rules.

In NLP, the latter rules-based models have fallen out of use, in favor of neural architectures, which are themselves networks of generalized linear models (a single neuron is a GLM of its inputs). Simple linear models are still in use in NLP, in parts thanks to word embeddings, which can enable computing decent quality sentences representations by a simple linear composition (such as a point-wise addition or mean) of their words' embedding vectors, as shown by Kober et al. [2017]. For example, Joulin et al. [2017] exploit this fact, using the fastText [Bojanowski et al., 2017] embedding model to very cheaply and quickly create a small linear sentence classifier, which simply averages the input fastText embeddings, passes them through a single hidden layer, which then projects them into a softmaxed distribution over the class labels, achieving close or even surpassing state-of-the-art of results, at the time it was presented, even beating significantly larger models.

Taking inspiration from this approach, and noticing a lack of concrete usage of interpretable word embeddings, we decided to explore their use as part of an a priori interpretable machine learning setup, with the objective of exploiting their interpretable dimensions to attempt to diagnose potential hidden biases in various NLP datasets, and will present it in the next Chapter.

Chapter 3

Interpretable Word Embeddings for the Detection of Hidden Biases — Experiments

While many methods have been proposed and evaluated to construct interpretable word embedding models, such as showcased in the previous Chapter, we noticed a distinct lack of experiments actually exploiting their intrinsically interpretable features as part of a concrete task. We thus propose to use the interpretable dimensions of non-negative sparse word embedding models to diagnose potential lexical biases in various datasets: by training an interpretable model, such as a linear classifier, on a to-be-diagnosed task, whose inputs are encoded by using an interpretable word embedding model, we could exploit the learned weights of this model as global explanations for each of the task's classes, in the form of a numeric "feature-attribution" associated to each of interpretable dimensions, and thus in theory, to the lexical family of words it is associated to.

We first showcase a baseline interpretable word embedding model we built from scratch, which we then evaluate alongside the previously discussed (see Section 2.2) existing models, both in a qualitative interpretability evaluation, using a novel variant of the *Word Intrusion Detection* task, and then quantitatively on a set of downstream tasks (discussed in Section 2.4) using a simple interpretable setup. We finally analyze some of the trained models weights to showcase how these embeddings can be used, in the previously described fashion, to potentially detect hidden lexical dataset biases.

NMF300	
science	neurology, ophthalmology, oncology, radiology, microbiology, cardiology courses, curriculum, undergraduate, students, vocational, teaching harvard, phd, doctorate, caltech, yale, swarthmore
car	locomotives, locomotive, wagons, trucks, carriages, cars sedan, hatchback, coupé, roadster, convertible, coupe nascar, waltrip, speedway, earnhardt, daytona, racing
teacher	courses, curriculum, undergraduate, students, vocational, teaching educator, linguist, scientist, writer, educationist, mathematician assistants, assistant, electrician, clerk, trainee, salaried

Table 3.1: Illustration of the interpretable word embeddings produced by our baseline approach, NMF300. While very simplistic, the dimensions produced still appear to display comparable levels of qualitative interpretability (see Tables 2.1 and 2.2 for comparisons).

3.1 A Baseline Interpretable Embedding Model: NMF300

While there is a wide variety of trainable and pre-trained interpretable embedding models to choose from within the literature (see section 2.2), we felt it necessary to have a common baseline model to compare them to. For the comparison to be fair, however, this baseline model still needs to display some level of interpretability, as manifested by the properties of sparsity and non-negativity. Fortunately, it happens that non-negative matrix factorization techniques, by their very nature, produce output matrices which display these properties. We therefore constructed the **NMF300** baseline model, as a point of reference to compare more complex models from the literature against, by directly using the non-negative factorization algorithm proposed in Lee and Seung [2001] for the Kullback-Leibler divergence (D_{KL}) reconstruction cost function (as opposed to the more common least squares variant used as a basis in other models):

$$\arg\min_{W,H} D_{KL}(X\|WH) = \sum_{i,j=1,1}^{V,C} (X_{ij} \log \frac{X_{ij}}{(WH)_{ij}} - X_{ij} + (WH)_{ij})$$

under the non-negativity constraints applied to both matrices $\forall i, j : W_{i,j} \geqslant 0, H_{i,j} \geqslant 0$, as well as the unit rescaling constraints applied on the columns of W, $\forall j : \|W_{:,j}\|_1 = 1$. This baseline model is constructed on a term-term co-occurrence statistics matrix (with a sliding window of size 5), collected on over a 2.2 billion word Wikipedia dump (from May 2017), with a vocabulary size of $V = 100\,000$ words (99 763 after frequency cutoff), $C = 20\,000$ features. Examples of the produced embeddings can be found in Table 3.1, similar to those discussed in sections 2.1.3 and 2.2.1 (Tables 2.1 and 2.2).

3.2 Comparing Interpretability

For our experiments, we thus chose to use our baseline interpretable embedding model, NMF300, alongside four other models from the literature (discussed in the previous Chapter, see Section 2.2): NNSE [Murphy et al., 2012], SPOWV [Faruqui et al., 2015], SPINE [Subramanian et al., 2018], and Word2Sense [Panigrahi et al., 2019]. Before using those models as part of a machine learning model in various downstream tasks, we first wished to qualitatively compare and evaluate the interpretability of their dimensions, and as such we chose to use a *Word Intrusion Detection* evaluation setup (see Section 2.3), as was commonly in the original publications for these models. The evaluators were this author and his two supervisors, and were, following a blind process, given 50 intrusion detection instances per model (for a total of 250 instances to evaluate), with 5 words per instance (including the intruder). All instances were shuffled (both their order of presentation in the task, and the words in each instance), and sampled according to the following random process: we first computed the intersection of all five models' vocabularies (giving us a common vocabulary of size 12 726), then iteratively selected 50 random words within it, under the constraint that a word can only be picked if it does not share its most active dimension with another already picked word, for all five embedding models. Finally, the most active dimension for all these words within all five models were chosen as the target dimensions for the intrusion detection task. After a first few test runs to tweak the hyperparameters of the method, mainly the sampling threshold for the selection of the least active words in a dimension, we made two significant adjustments to our evaluation: first of all, as was done in Subramanian et al. [2018] and Panigrahi et al. [2019], we initially considered also comparing these models against a dense embedding model, such as GloVe [Pennington et al., 2014] or word2vec [Mikolov et al., 2013a,b], however, similar to the results obtained in these publications we found that, as can be expected, it is basically impossible to tell the intruder apart from the most active words for such dense representations. As such, and because we did not wish to increase the workload of the evaluators to just confirm this fact an n-th time, we only ran this evaluation of the 5 previously mentioned interpretable embedding models. The second observation we made was that the task was too "easy": for many of the models, using the implementation of the task as described in Subramanian et al. [2018], even when tweaking the threshold hyperparameters b and t (see Section 2.3), we found the task not challenging enough, to the point where some samplings of dimensions following the original process lead to almost perfect accuracies. Conceptually, we argue that the act of picking the intruder in a random dimension (other than the one that is being evaluated) does not allow to properly test its lexical *discriminative power*: ideally, we would like to ascertain whether the particular selected dimension can help differentiate between words who otherwise share similar values in their other dimensions. In other words, an interpretable dimension should ideally target a specific aspect of a lexical or semantic family. Thus, to enable this, we propose

a modification of the sampling process for the intruder, which we found significantly increases the difficulty of the task: just like in the "classic" process, we first select the bottom $b\%$ (in our test runs, we found $b = 10\%$ to be a good value, vs. the 50% proposed by Subramanian et al. [2018]) least active words in the target dimension. However, instead of then refining this selection by picking from those words that are in the top $t\%$ of another random dimension, we instead specifically pick the "second most active common" dimension to the four ground-truth words in the intrusion detection instance. After experiments, we find that picking one word from the five most active in the dimension which has the second-highest median of the four ground-truth words gave satisfactory results. To better illustrate this process concretely, here is an example of the steps which might be taken when generating a word intrusion detection instance for dimension n°110 of the SPINE model:

Example: Looking at the top 10 most active words in SPINE's 110-th dimension (indexed starting from 0), we find, from most active to least active:

"pius, pope, diocese, bishops, basilica, archdiocese, benedict, vatican, catholic, bishop".

After picking the first four words as our ground-truth words for the instance (*"pius, pope, diocese, bishops"*), we then compute their second most active common dimension by taking the median of all their components, and picking the second largest, which in this case is the 178-th dimension, with the following top-10 words:

"baptist, jesus, christians, holy, lutheran, religious, judaism, believers, prayers, baptism"

As can be observed, these qualitatively look quite close to the active words in the 110-th dimension, lexically speaking. In the "classic" variant of the task, we would then pick the intruder within the bottom $b\%$ of the 110-th dimension, and from the top $t\%$ of another random dimension, which might give us one of the five following sampled words for example: *"baseline, sculptures, feedback, armoured, modeled"*. We can see that picking one of these words as the intruder alongside our four ground-truth words (*"pius, pope, diocese, bishops"*) would definitely make for a trivial instance. Instead, in our proposed more difficult variant, sampling five random words with the same constraint (picking from those that are also in the bottom $b = 10\%$ of the 110-th dimension) from the 178-th dimension instead gives us: *"judaism, mormon, preacher, buddhism, meditation"*. We can see that this sample is much closer lexically to the four ground-truth words, making the task significantly harder if one were to be picked as an intruder. Yet, they also would better illustrate the differences between the 110-th and the 178-th dimension: indeed, the former seems more strongly associated with terms specific to the Catholic religion, whereas the 178-th dimension seems more strongly associated with terms related to religions in general.

Table 3.2 displays both the evaluators' average accuracy as well as the inter-evaluator

Model	Average Evaluator Accuracy	Inter-evaluator Agreement	Fleiss' Kappa
NMF300	76%	**94%**; 72%	0.74
NNSE	**79%**	90%; **74%**	**0.76**
SPOWV	38%	84%; 34%	0.43
SPINE	**79**%	92%; 60%	0.63
WORD2SENSE	65%	88%; 56%	0.61

Table 3.2: Results of our variant of the Word Intrusion Detection task on the five interpretable embedding models compared here. The agreement is formatted as: `majority agreement(2 out of 3) ; unanimity agreement (3 out of 3)`. Best results in each column are highlighted in **bold**.

agreement metrics on our variant of the Word Intrusion Detection task. We first note that our results are relatively similar to those obtained by Subramanian et al. [2018] and Panigrahi et al. [2019], taking into account the difference in evaluation protocols, and the somewhat subjective nature of this task. Surprisingly, our baseline model, NMF300, performs quite decently on this evaluation, despite its simplicity. This seems to showcase once again that the simple addition of the sparsity and non-negativity constraints is very effective at producing mostly interpretable dimensions. While almost all models on average performed relatively well in this more difficult evaluation task (with the exception of SPOWV), we noted during the evaluation that qualitatively, the interpretability of these models' dimension is very heterogeneous: indeed, while the majority of dimensions were relatively easy to associate to a given lexical aspect, some on the other hand seem to capture pseudo-lexical phenomena, which seem to depend on the corpus they were created from. For example, models trained on Wikipedia articles, such as our own NMF300, seem to have captured a few frequency artifacts, caused by the presence of highly repetitive tabular data. Similarly, some highly specific families of proper nouns, such as names of sports players, teams, or brands, seem to occupy a disproportionate importance in some models' learned interpretable dimensions, which we believe is also caused by frequency artifacts, due to the number of distinct articles which discuss these aspects at different levels (e.g., an article on a particular sport may refer to articles discussing individual teams, which themselves may refer to articles discussing individual players, etc.), and with highly specific vocabulary overlap between them.

We argue, however, that a better test for the interpretability of these representations is to use them in a concrete setup, as inputs to a machine learning model performing a task. We thus designed an experimental setup that would not only evaluate their effectiveness on a given task's objective, but in theory also showcase their interpretability, to gain some insights on the task's associated dataset and its potential hidden lexical biases.

3.3 Downstream Tasks Evaluation

In theory, an interpretable word embedding model can be used as part of an NLP machine learning setup just like any other (dense or not) word embedding model, simply as a way to provide vector representations to encode the tokens in a text. However, if one wants to make full use of their intrinsically interpretable dimensions, care must be taken as to the choice of model to use. As noted in Section 2.5, using an intrinsically interpretable model, such as a linear classifier, would allow directly exploiting its learned weights, each associated to an input interpretable dimension, as an explanation for a given predicted class. These may then allow us to detect if any undesired lexical correlations may be present in training datasets, which will be actually helped by the fact we are using a simple classifier, who can be more prone to fine-tune on easy-to-learn, often biased features in datasets.

Thus, for each of the tasks and their datasets presented in Section 2.4, and for each of the interpretable embedding models discussed before, we trained an intentionally *elementary* Continuous-Bag-Of-Words (CBOW) softmax regression classifier, with a parameters matrix of size $H \times C$, where H is the size of the embedding vectors used, and C the number of classes in the task. Taking inspiration from the simple fastText-based linear classifier presented in Joulin et al. [2017], we simply use the average of the interpretable embeddings of all the words in an instance as an input to our model. For tasks which have two separate input texts in their instances, we take inspiration from the sentence-encoding architecture InferSent, from Conneau et al. [2017], who propose the following composition pattern to combine two separate sentence representations u and v: $(u; v; |u - v|; u * v)$, where $*$ is the element-wise product operator, and $(a; b)$ represents the concatenation of vectors a and b. For each $(\text{model}, \text{corpus})$ pair, we train one such classifier for a maximum of 200 epochs, using the ADAM optimizer, with 50 preceding epochs of automatic hyperparameters fine-tuning, using the *Tree-structured Parzen Estimator* algorithm [Bergstra et al., 2011, 2013], via its implementation in the `optuna` library [Akiba et al., 2019].

We then evaluate each produced classifier on its respective task's test set, and display the results in Table 3.3. In addition to the five interpretable embedding models we used, we also trained and evaluated in the same way described above a classifier using the fastText dense embedding model (without sub-words), to compare the performance of interpretable embeddings against those of a dense model. We also display the results of a "dummy" classifier, who merely generates predictions at random, weighted by the task's class distribution, as a sanity check.

We notice that, quite surprisingly, considering the simplicity of the approach, the accuracies of the trained classifiers are quite high, and even, for the majority of the tasks, better with some of the interpretable embedding models than with the dense fastText model. This seems

Model \ Corpus	IMDB	BoolQ	Sarcasm	UR-FUNNY	SST	SNLI	Emergent	PDTB
NMF300	67.8	62.6	60.5	57.7	54.6	58.6	50.9	33.2
NNSE	78.7	63.6	63.9	59.9	60.6	56.3	66.8	31.1
SPOWV	81.9	**66.9**	**70.5**	**65.0**	62.9	62.9	**72.2**	**36.6**
SPINE	81.3	65.9	67.8	63.6	59.9	64.1	**72.2**	34.5
Word2Sense	**82.2**	66.2	67.3	63.9	61.4	**65.5**	69.8	34.2
Dummy *(baseline)*	50.5	53.5	53.0	52.5	39.5	33.6	41.3	19.3
fastText	82.0	63.7	70.1	64.5	**64.4**	61.3	69.5	33.4
*Dedicated models**	96.8	76.9	74[†]	64.4	96	91.5	73	48

Table 3.3: Results of the approach on the downstream classification tasks evaluated. Accuracy scores for each model-corpus pair are reported in percentages (best scores in bold). *We additionally list results for popular task-dedicated models found in the literature which achieve (or come close to) state-of-the-art performance, as a comparison (IMDB, SST: Yang et al. [2019]; BoolQ: Clark et al. [2019]; Sarcasm: Oraby et al. [2016]; UR-FUNNY: Hasan et al. [2019]; SNLI: Liu et al. [2019b]; Emergent: Ferreira and Vlachos [2016]; PDTB: Dai and Huang [2019]). These are only indicative as the conditions in the cited papers differ: SST is reported for two classes with a larger training set; PDTB is trained on the 11 classes only present in the test set. [†]F1 score for the positive class (accuracy not available).

to imply that many NLP tasks have a more or less important purely lexical aspect to them, and this seems to be especially true for sentiment analysis for instance, on which we get the overall highest scores. This may to some degree be problematic, as this means a large proportion of instances are "easy" to label, which may be a source of problem for more capable models training on these datasets, as they might be learning easy lexical shortcuts, instead of skills required for proper natural language understanding. While it is to be expected that some lexical features would be associated to some task-relevant aspects, for example, that certain family of words may have a more or less intrinsic sentiment value, a model relying on these cues may not properly learn to handle more complex structural phenomena, such as negation. Various contributions have shown that even more complex state-of-the-art models can fail at handling negation, when instances are specifically crafted to test for it [Naik et al., 2018; Kassner and Schütze, 2020; Hossain et al., 2020]. Even if these results cannot by themselves demonstrate the presence of spurious correlations in those datasets, the fact such a simple approach achieves overall relatively high accuracies may be a sign that some training datasets might not be ideally designed to teach models the NLP skills intended in their task specification. Overall, all of the interpretable embedding models perform relatively well, with SPOWV, SPINE, and Word2Sense being the most performant, while NMF300 and NNSE significantly below, especially on some tasks (BoolQ and SST for NMF300) where some classes were not predicted at all.

For the last step in our experimental setup, with our models now trained, we can then simply analyze their learned weights to produce global lexical explanations.

3.4 Explanations Analysis

For each of our trained models, we generate global explanation reports[1], consisting of, for each class in the task: the top-10 most positive and top-10 most negative weights, associated to the interpretable embedding model's dimension index, for which we then list the top-10 most active words. For tasks with multiple input sentences, we specify to which part of the composition pattern $(u; v; |u - v|; u * v)$ the listed weight belongs, as this can also be used to qualitatively judge these learned correlations. We qualitatively review these reports, and showcase a few notable examples in Table 3.4.

IMDB: This is one of the datasets for which the performance of the trained elementary models are the highest. Not too surprisingly, a significant portion of the most active dimensions for the "positive" and "negative" classes seem to correspond to lexical families of words containing appropriate sentiment markers, for most models (see the first and second rows in Table 3.4). However, for the NMF300 model in particular, we noticed several dimensions associated with a large number of surnames and first names (e.g., 4th row in Table 3.4) that appear to be strong predictors of the "positive" class. To analyze this potential bias, which does not seem intuitively very relevant to sentiment, we used the Named Entity Recognition (NER) module of the `spaCy` library to count the numbers of Named Entities of type "PERSON" in the movie reviews of the dataset, and we found a weak linear correlation (Pearson coefficient $r = 0.124$) between these counts and the classes of the instances. Further analysis would be needed to see whether a non-linear model could also exploit this aspect, or possibly an even more specific one : indeed, it seems that several dimensions created by NMF300 are associated with famous artists' or celebrities' names, in particular, the second most contributing dimension for this task in this case. Taking this into account, one possible explanation for how such a spurious correlation may have occurred may be that review authors could be more inclined to mention the actors of a movie by name when leaving a positive review than when leaving a negative one. We can note that 80.68% of the reviews of the dataset contain at least one named entity of this type, which is also consistent with the high weighting of the parameter corresponding to this dimension. We also illustrate this analysis graphically, in Figure 3.1.

BoolQ: On this dataset, the most contributing dimensions seem to focus on particular themes: for the "false" answers, these seem to point towards questions about themes which are

[1]These reports are publically available and can be found at: https://github.com/TomBourgeade/InterpEmbsForBiasDetection/tree/main/experiments_results

often debated (dieting, laws, etc.), often subject to conspiracy theories (intelligence agencies, space exploration, etc.), or emotionally marked language (with adjectives such as *"dignified"* or adverbs such as *"dramatically"*). For the "true" answers, these seem to point more towards questions about science, history, geography, or politics, or to numerical values (dates, ordinals, or miscellaneous numbers). All of this may indicate a slight bias in the data collection (which relies on user queries from a search engine), which is perhaps exploited by the models without the need to analyze the response. Nevertheless, since this is a two-input task (question and passage), we can also observe in which part of the composition vector $(u; v; |u - v|; u * v)$ (see Section 3.3) the most important weights are found: for most models (with the notable exception of NMF300 and WORD2SENSE), we observe that these are in the term-to-term product part of the composition, indicating that these elementary models are probably mainly looking at interactions between the question and the passage in the input, which is expected, considering the task. The remaining important weights are on the other hand mostly located in the question part of the composition, which could indicate the presence of more or less "rhetorically" biased questions (that is, questions which more or less strongly imply their own answer, regardless of the passage).

EMERGENT: For this dataset, we again notice a number of thematic consistencies in the dimensions that contribute the most towards the best predictions, related to uncontroversial topics. For instance, one dimension in NMF300 which is related to animals is highly correlated to the *for* stance. An inspection of the dataset confirms that newspapers almost exclusively take a positive stance on stories about animals (mostly, cats and dogs). This kind of bias seems inherent to the way the dataset was constructed from newspaper headlines.

SARCASM AND UR-FUNNY: The NMF300 model reveals some popular subjects in the SARCASM dataset, with positive-class dimensions seemingly associated with music and musical artists (dimensions with top words: "*burnin, dreamin, rmx, blowin, movin*" and "*lil, ludacris, rapper, dogg, snoop*"). Negative-class dimensions focus more on medical ("*neurology, ophthalmology, oncology*") or legal themes ("*plaintiffs, plaintiff, court, appeals*"), and a lot of technical-themed dimensions. This should be investigated at the instance level, since it could be an indicator of a lack of diversity in the corpus. The UR-FUNNY dataset shows similar important dimensions (with also a lot of proper nouns) for NMF300, but focusing more on the punchline than the context in the inputs. NNSE shows more variety and less important weights, also focusing on the composed representations. Negative-class dimensions still include technical-themed dimensions.

PDTB: The PDTB implicit relation task is interesting because it is a difficult problem, mixing quite different semantic/pragmatic relations. The simplest model, NMF300, predicts only 4-5 relations out of those appearing in the test set, focusing mainly on the most frequent: *Cause*,

Contrast, *Conjunction*, *Restatement* and *Instantiation*. As an example of the kind of information it reveals, we found that the dimension with the largest weight associated with the *Instantiation* class in the NNSE model is one where the top words are "*educator, historian, lecturer, researcher, scientist*", in the part of the composition vector corresponding to the second argument of the input relation. Upon inspection in the training set, we found that only three of those words appear with that relation type in about 20 instances. It seems to indicate these are mostly citations illustrating a point made in the first argument of the relation, something confirmed when looking for other citations cues, and observing that they are in about a third of all *Instantiation* instances, pointing arguably at a quite specific journalistic aspect of the PDTB. Similarly, some dimensions important for predicting the other relation types seem specific enough to warrant a closer inspection of instances in this dataset.

SNLI: This dataset seems to be a special case, with many different, seemingly unrelated dimensions being top contributors, for all models. This could be partly explained by the large size and thus the larger variety of instances in this dataset. SNLI has known biases (see Section 1.2.1), which are partly associated with syntactical or structural aspects (negation, additional prepositional phrases, etc.). These are obviously more difficult to discover with the interpretable embeddings used here, which are mainly lexical in nature, combined with the elementary classifier architecture we used, which cannot directly model structural aspects, on account of the sentence representations being averages of word embedding vectors.

Dataset	Model	Class	h	C	Most active words in h-th dimension
IMDB	NNSE	*pos*	192	1.0	*utmost, sheer, immense, tremendous, newfound, unparalleled, ...*
IMDB	NNSE	*neg*	217	1.0	*debris, trash, garbage, lint, rubbish, sludge, dust, dirt, manure, ...*
IMDB	NMF300	*pos*	100	1.0	*imaginative, vivid, lyrical, poetic, realistic, imagery, subtle, ...*
IMDB	NMF300	*pos*	131	0.76	*shakira, lauper, mcentire, yearwood, parton, estefan, streisand, ...*
BoolQ	SPINE	*false*	575	1.0	*leaked, confidential, libby, fbi, classified, memo, leak, intelligence, ...*
BoolQ	SPINE	*false*	841	0.79	*astronaut, soyuz, spacecraft, iss, nasa, astronauts, shuttle, mir, ...*
BoolQ	SPOWV	*true*	758	1.0	*cyclone, katrina, hurricane, disaster, ike, flooded, shear, dolly, ...*
BoolQ	SPOWV	*true*	173	0.83	*tong, lumpur, myanmar, singaporean, kuala, chung, penang, ...*

Table 3.4: Examples of explanations generated by our approach, where C is the "contribution" of the h-th dimension for the given class, that is, its corresponding weight in the trained linear classifier, normalized by the largest weight (in magnitude) sharing the same sign for that class. A contribution of 1.0 thus indicates the dimensions that were the most influential. We can notice that with NNSE, the most influential dimension for each class are not too surprising, being associated with respectively strong positive and negative markers. With NMF300 however, while the most contributing dimension to the positive class is similarly unsurprising, the second one is more questionable, as it appears to be associated with the names of public celebrities.

This approach to generating global explanations for datasets is overall quite interesting:

it is relatively easy to implement, and while it cannot be used alone to definitely prove the presence of hidden dataset biases, we believe it may be a good starting tool in a more complete suite of explainability methods. It could be used to quickly check for potentially easy-to-miss correlations, and to help direct the focus of more expensive to deploy methods, in terms of compute, time, implementation, or expert costs, towards potentially problematic aspects of a dataset.

An example of follow-up explainability method which one may use in conjunction with this approach are *feature-attribution* methods, such as gradient-based *saliency maps* (which will be discussed in more details later in Section 4.2). These methods allow computing a score for each feature of an input, depending on how much it contributed to the overall model's prediction (see Figure 3.2). When used alongside our proposed approach, one may also exploit the interpretable dimensions of the chosen embedding model to perform this type of analysis at a more granular level, which would normally not be relevant to explore with dense embeddings (see Figure 3.3).

3.5 Conclusion and Perspectives

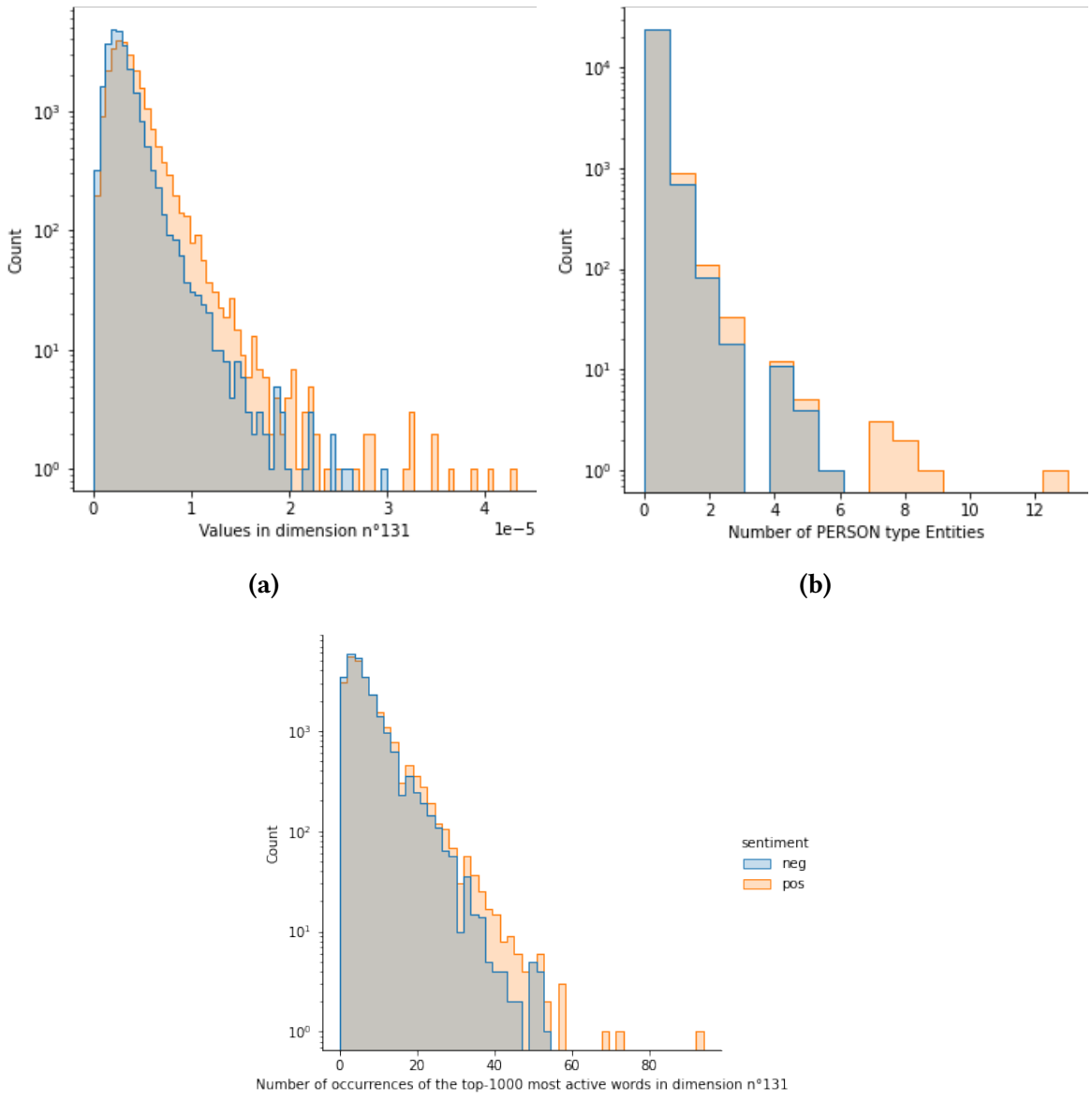
We demonstrated here how a simple but effective method can be used to help detect potential spurious biases in NLP datasets, exploiting interpretable word embeddings to qualitatively diagnose unwanted artifacts which might negatively influence more complex models on downstream tasks. We intrinsically and extrinsically compared different interpretable embedding models for this purpose, and found that the newer and slightly more involved methods perform better on downstream tasks, but that this does not necessarily translate to their dimensions' interpretability. This part was the basis for a paper [Bourgeade et al., 2021] presented at the *TALN-2021* conference.

Various improvements and additions to this approach warrant exploration: to be able to better discern which embedding models provide the best insights into how instances of a dataset might be classified, a human evaluation of the explanations produced by this approach could be performed, in a manner similar to the evaluation method presented in Strout et al. [2019]. One important limitation with the type of interpretable embedding models used here is that their dimensions mostly encode lexical information, which means that structure-based biases, for instance, may be more difficult to detect this way. An avenue worth exploring might thus be to use embedding models which encode different linguistic features, which would allow detecting a wider range of biases. Negation and discourse markers, for example, are two types of phenomena that have been shown to strongly correlate with other, often unrelated features in datasets. Similarly, the ease of interpretability granted by the use of an elementary CBOW classifier comes at the loss of precious information carried by word order, mainly contextual

semantic information. This could potentially be improved without compromising on model complexity, by using contextual word embedding models, such as BERT Devlin et al. [2019] or ELMo Peters et al. [2018], trained in an interpretable fashion. Another improvement would be to combine this approach with example-based interpretability methods: in addition to detecting unwanted overall behavior resulting from biases in datasets, it would be helpful to be able to locate the specific subsets of instances that are responsible. We briefly showcased in at the end of Section 3.4, how this approach may be combined with post-hoc explainability methods, to open up other modes of analysis, but this could be extended to more complex methods. For example, Layer-wise Relevance Propagation [LRP; Montavon et al., 2018, 2019] could allow diagnosing a suspect non-linear model on instances which have been identified, through our approach, as potential sources of biases. This could then be used to potentially correct the issue, either in the dataset (by removing or balancing out the offending instances), or in the model itself (by modifying its architecture and/or hyperparameters until the spurious behaviors have been diminished or eliminated).

Overall, intrinsic interpretability is challenging to implement, and especially so in NLP. The sets of properties required to maintain naturally interpretable processes at all levels of a machine learning pipeline impose strong constraints on both the models, and the data they manipulate. While it is a worthwhile goal, unfortunately, the recent trends in state-of-the-art NLP architectures are not conducive to intrinsic interpretability, on the contrary: models are becoming larger, more complex, and ever more opaque (see Table 1.1). As much as we hope that some of the concepts explored in this first part may find their way into future state-of-the-art architectures, it may turn out to be necessary to compromise, at least in the short term, on the desiderata of intrinsic interpretability. In the next part, we will thus explore the other side of the spectrum, with post-hoc explainability methods, and how they apply to Natural Language Processing.

Figure 3.1: Histograms for the NMF300 model used on the IMDB dataset, showing (a) the distributions of values in the 131-th dimension of the embeddings of the words present in the dataset, (b) the distribution of the number of PERSON-type Named Entities found in the dataset, and (c) the distribution of the number of top-1000 most active words in the 131-th dimension found in the dataset, all in log-scales. As we can observe, there seems to be a slight bias towards the *positive* class, associated to proper names, seemingly of celebrities, represented by the 131-th dimension of NMF300.



(a) **(b)**

(c)

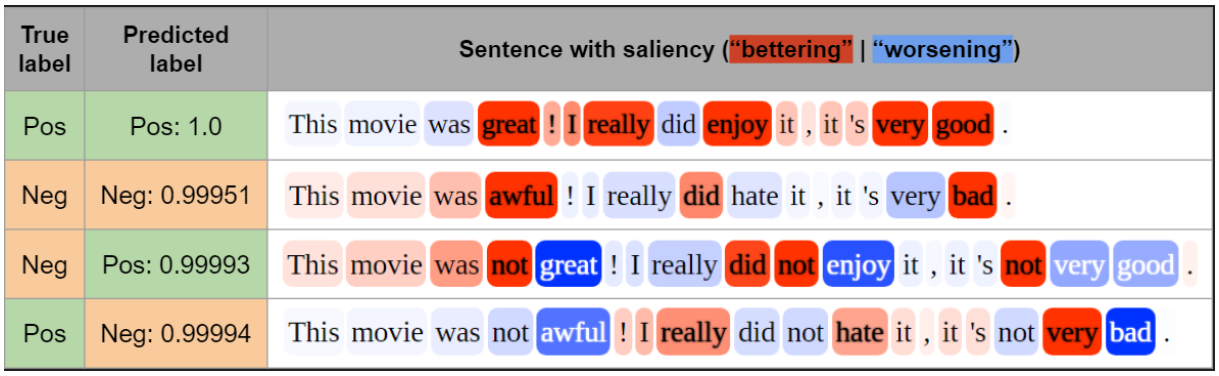


Figure 3.2: Example of a *saliency* maps obtained by using a simple gradient-based feature-attribution method on a trained elementary linear classifier with SPINE interpretable embeddings, trained on the IMDB dataset. The ground-truth ("true") labels, the model's predicted labels (alongside the output probability), and a saliency map over each input's tokens are displayed. The saliency map are color-coded: the more red a token is highlighted in, the most beneficial to the model attempting to predict the correct (ground-truth) label, and inversely for blue tokens. These four instances were handcrafted to demonstrate how such a model trained with this data cannot learn to correctly handle negation: the first two instances intentionally make use of antonymic verb and adjectives, which the model has no trouble correctly classifying. The last two instances make use of multiple inserted "not" negation markers to confuse the model: as can be seen from the saliency maps (note again that the "direction" of the colors depends on what the true label for the instance is, not the predicted one), the word "not" appears to have been simply associated with the *Neg*ative sentiment class, regardless of the context.

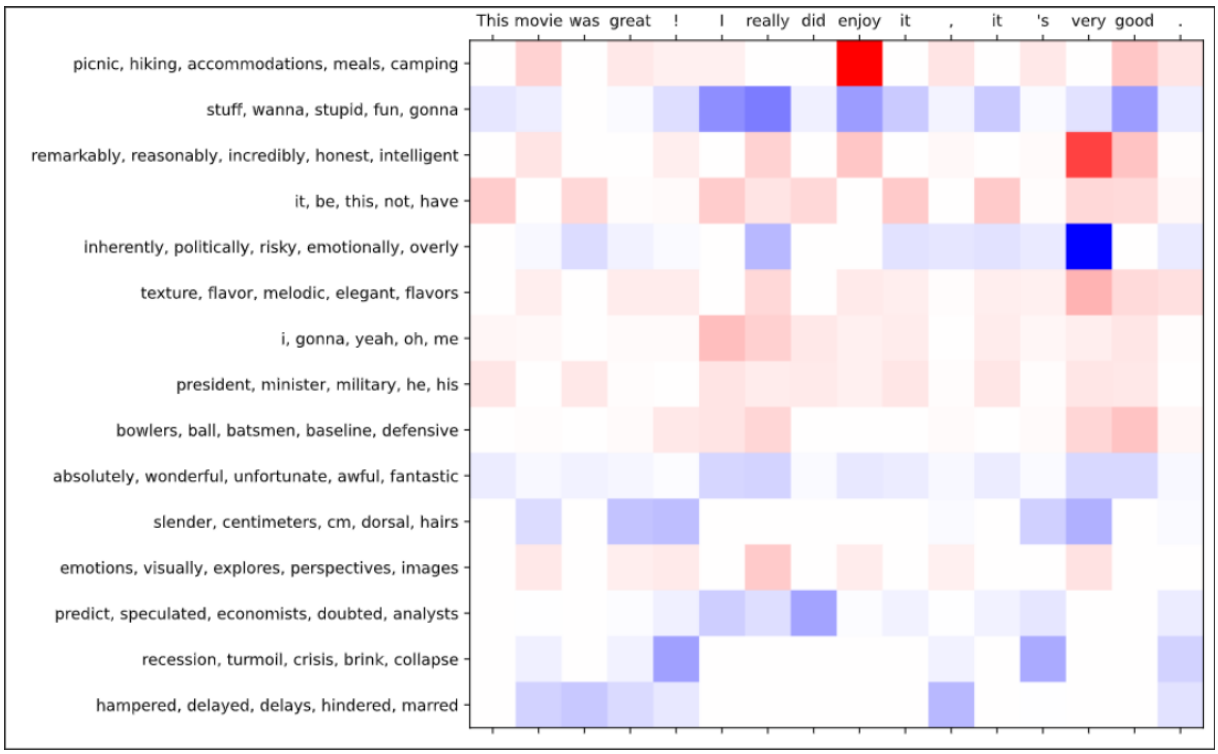


Figure 3.3: Example of a detailed *saliency* map, corresponding to the first example instance in Figure 3.2): here, we associate the most relevant dimensions of the feature-attribution vectors for each token, to their corresponding first few most active words. As can be seen, the values of the loss function's gradient with respect to the inputs are sparse for interpretable word embeddings, and the sign of those values can allow us to gain some insights for individual tokens: for example, we can see that "very" has both a strong "positive" and "negative" component to it, which may accurately indicate it is a potentially polarizing word, for either sentiments. On the other hand, "enjoy" only displays a single strong positive component.

Chapter 4

Post-hoc Explainability in NLP

4.1 Context

While intrinsic interpretability is a difficult goal to achieve, especially in NLP, *post-hoc explainability* methods are much more numerous in the eXplainable AI (XAI) and Interpretable Machine Learning (IML) literature, for multiple reasons: firstly, interpretability and explainability of models are aspects which are unfortunately often considered after-the-fact in machine learning research. As such, there are many established tasks for which high-quality trained black-box models exist, which would benefit from methods that could produce explanations for their decisions, without having to "open the black-box". Indeed, many models are expensive to train, in terms of time and compute, and could thus not realistically be re-trained with modifications aimed at making them *a priori* interpretable, assuming those modifications even exist and are possible to implement for those models. While intrinsic interpretability would in theory offer better guarantees on many of the traditional explanation desiderata (faithfulness, mainly), extracting or generating *post-hoc* explanations may ultimately be the only feasible option in many cases, though many have argued and shown that post-hoc explainability methods can be easily incorrectly used to provide deceptively rational-looking explanations [Alvarez-Melis and Jaakkola, 2018; Aivodji et al., 2019; Rudin, 2019]. However, at this point in time in NLP, with the evolution of state-of-the-art architectures towards seemingly ever-growing numbers of parameters (see Table 1.1) and opaqueness, which are in practice antithetical to the concept of intrinsic interpretability, we believe it is necessary to explore post-hoc explainability methods which may, at least in theory, be applicable to them, even if in this process, some compromises have to be made on explanation desiderata.

As such, we showcase below a limited selection of post-hoc explainability methods from the relevant literature, though many more are available, albeit not necessarily all applicable or well-suited to NLP. Please see Molnar [2019] for a much more exhaustive overview of post-hoc

explainability methods in general, as well as for more detailed illustrations of the approaches discussed below, in addition to the relevant cited publications.

4.2 Local Explanations

A first popular type of post-hoc explainability method are so-called *local* methods, which attempt to produce explanations for individual instances and predictions, rather than for entire models at once. Most of them are designed to be as model-agnostic and portable as possible, though some methods are more specifically designed to be used with neural architectures trained with gradient descent. An example of this latter type are gradient-based attribution methods, also referred to as *saliency* methods.

Gradient-Based Feature-Attribution: In these methods, the objective is to determine which components of an input instance were the most relevant to a model's prediction for it. In other words, we use a particular *saliency* measure, at a specific granularity level of an input's decomposition, to attribute to each component of an input a value, indicating how much and in which direction it contributed to the model's overall prediction. The form explanations will take will thus generally be *saliency maps*, which for NLP tasks may look like the different examples in Figures 3.2 or 3.3. One method proposed initially by Simonyan et al. [2014] (for Computer Vision models) to do this is to simply sample the gradient of the training loss function with respect to the inputs at the instance which is to be explained, at the desired granularity level (for example, in Figure 3.2 the magnitudes of the gradient vectors for each token are used, whereas Figure 3.3 presents each vector's dimensions independently). This is usually trivial to compute, since learning via gradient descent already imposes being able to compute this gradient, to enable the back-propagation of prediction errors to all learnable parameters in the network. A few issues were observed with this simple approach, however. The main shortcoming of this approach is that it tends to be sensitive to small, insignificant perturbations, which can lead to unstable explanations. Similarly, using the simple gradient as saliency fails to correctly model saturation in neural architectures [Shrikumar et al., 2017], where an output does not change when individual components of the input are perturbed, which can cause inconsistent explanations to be produced. To attempt to fix some of those issues, variants of this simple approach were proposed. For example, Smilkov et al. [2017] propose SmoothGrad, which as its name implies, attempts to improve the stability of the saliency measure by smoothing out the noise in the computed gradients, by sampling at multiple points around the target instance according to some level of Gaussian noise and averaging the results. Similarly, Sundararajan et al. [2017] propose *integrated gradients*, modifying the basic gradient-as-saliency measure by instead taking the average gradients sampled along a path from a reference instance (e.g. an all-black image in Computer Vision, or the zero

embedding vector in NLP) to the target to-be-explained instance. While cheap to compute, other contributions argue that using the gradient of the loss function poses too many issues, and so propose using other functions as saliency measures. For example, Shrikumar et al. [2017] propose DeepLIFT, in which saliency is instead computed by back-propagating the contribution to a prediction of all neurons in a network, where this contribution is defined as the difference in a neuron's forward-pass activation and its reference activation, which is itself the activation for a reference instance, depending on the task and type of data (similar to *integrated gradients* above). Murdoch et al. [2018] instead explore contextual decompositions (CDs) of LSTM architectures, using linearized activation functions to study how each input token in a sequence contributes to a model's ouputs at each timestep, and then for the overall sequence. Finally, Montavon et al. [2018, 2019] propose the Layer-wise Relevance Propagation (LRP) method, in which the forward-pass activations of neurons are collected for the target instance, then back-propagated through the network all the way back to the input layer, as a measure of input "relevance", with the possibility of using different propagation rules depending on the types and depths of the layers encountered. Overall, saliency methods have been criticized for being often misleading, as many of them have been shown to be, to some degree, independent of the models they are supposed to help explain [Adebayo et al., 2018]. They may also give a false sense of understanding, especially to non-expert users: they may show *where* a model is looking at, this does not necessarily translate into an accurate display of *what* the model is doing with these parts of the inputs [Rudin, 2019; Alvarez-Melis and Jaakkola, 2018].

Most popular post-hoc explainability methods which have been proposed on the other hand usually are *model-agnostic*, that is, they do not directly rely on a model's internals to generate explanations, but rather only require access to these model as black-box, and extract information about their behaviors through the appropriate analysis of predictions on specific input data.

LIME: The ***Local Interpretable Model-agnostic Explanations*** framework from Ribeiro et al. [2016] is a local post-hoc explainability method which proposes using *local surrogate models* to explain algorithmic predictions. A local surrogate model in this case is an intrinsically interpretable model, such as a sparse linear classifier (also known as a "Lasso" model), which is trained to imitate the target black-box model, but only locally, in a neighborhood close to the target to-be-explained instance. This is proposed to be done through sampling of the original target model at various data points in the vicinity of the target instance, weighted by their distance to it, which then serve as the training corpus for a simple linear model, whose learned weights will serve as local explanations of the original model's decision for the target instance. The first step is to first choose an *interpretable representation* space into which the original model's input vectors may be projected: for example, for NLP models, it is likely the target models will take as inputs sequences of word embedding vectors, which are not in and

of themselves interpretable by humans. In this case, the authors propose using the space of binary vectors, the components of which correspond to the presence or absence of each word in the target input. For images, the authors propose using a similar binary vector, denoting the presence or absence of patches of similar pixels (so-called *super-pixels*) in the target input. The local surrogate model will be trained in this space rather than on the original one, as otherwise, despite being an intrinsically interpretable model, the learned weights attributed to each input component may likely not be directly understandable, even by expert users. If we denote x the target instance (in original input space $X = \mathbb{R}^d$ with d dimensions), C the shape of distributions of probabilities over the set of possible labels in the classification task, $f : \mathbb{R}^d \rightarrow C$ the original black-box model's function, then $x' \in \{0,1\}^{d'}$ (with d' the number of interpretable dimensions, i.e., for text, the number of words in sequence x) is the interpretable representation of x, and $g : \mathbb{R}^{d'} \rightarrow C$ is the surrogate interpretable model's function. g is thus trained by minimizing the weighted square loss function:

$$\mathcal{L}(f, g, \pi_x) = \sum_{z,z' \in Z} \pi_x(z)(f(z) - g(z'))^2$$

where $z' \in Z$ are perturbed instances derived from x' (thus, for text, by removing a random quantity of words from the original target instance), z are the recovered corresponding points to z' in the original input space, and $\pi_x(z) = \exp(-D(x,z)^2/\sigma^2)$ is a proximity measure between x and z (with $D(x,z)$ being cosine similarity for text, and σ a width factor). See figure 4.1 for a schematic illustration of this process. In addition to providing this framework to generate explanations for individual decisions, the authors also propose the LIME-SP (Submodular Pick) algorithm, which, for a set of instances and their accompanying generated explanations, allows selecting the most relevant arbitrarily sized sub-set of instances and explanations which best illustrate a model's overall behavior. This algorithm is based on estimating global feature importance values for that set, which allows iteratively selecting examples which differ in the most important features, while avoiding redundancies in the other already seen features. This method has the advantages of being relatively easy to implement. However, the interpretable perturbation scheme it proposes when working with text is not ideal: indeed, removing words, even just one, from a sentence is a highly significant perturbation which may very likely completely change the meaning of the input, but most likely will simply lead to sampling the original target model on nonsensical inputs. This may then lead to surrogate model explanations that are not very informative or robust. This method is notably prone to adversarial attacks [Slack et al., 2020], which may put into question how faithful its explanations actually are.

Anchors: Some time after LIME, the same researchers introduced *Anchors* [Ribeiro et al., 2018a], a post-hoc explainability method also based on perturbations, but this time to extract a set of *if-then* rules, the so-called *anchors*, which almost sufficiently (in the mathematical

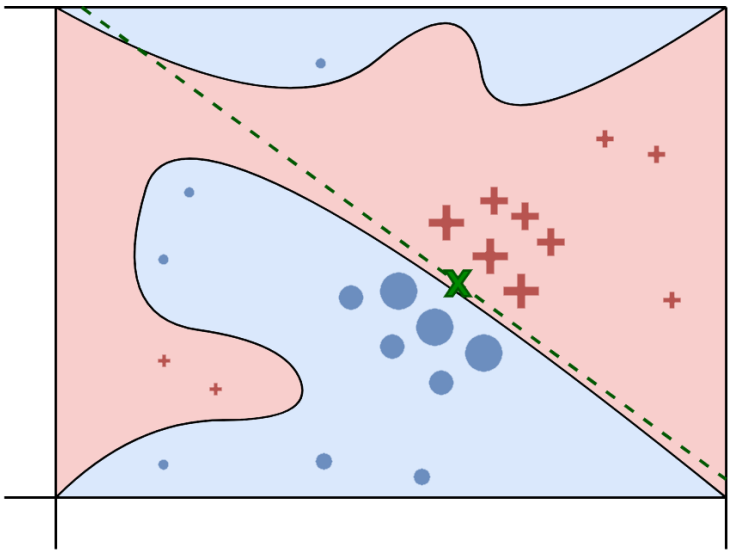


Figure 4.1: Illustration of how the LIME method creates a local linear surrogate model for a particular target instance (green cross). The approach does not have complete access to the original model's input prediction space (with the two classes shown in blue and red, separated by their decision boundaries), but the target instance is perturbed in order to sample various points (blue dots and red crosses), weighted by their distance to the target instance (shown by the size of the markers). These samples are then used to train a linear surrogate classifier (green dashed line), whose learned weights will serve as an explanation for the original model's behavior on the target instance.
(Figure adapted from Figure 3 in Ribeiro et al. [2016])

sense) explain a target model's decisions on a subset of instances. In other words, an *anchor* is a rule over instance features, which, when matched, indicate those instances have a high probability, above a parameterized threshold τ (for example 90%), to be predicted as having a given label by the target model. For example, on a sentiment analysis model and dataset, an extracted anchor rule might look like: $\{$*"not"*, *"bad"*$\} \rightarrow$ *Positive*, and would mean that any instance which contains the words *"not"* and *"bad"* has an above τ probability to be predicted as the *Positive* sentiment class by the target model. Unlike LIME, which only attempts to locally approximate the target's model behavior, anchors are much more faithful by design, because they explicitly indicate for which parts of the instance prediction space they are valid, and to which degree of precision, though, just like LIME, it only requires query access to the target model, as a black-box. More formally, an anchor $A : \mathbb{R}^d \rightarrow \{0, 1\}$ is a set of predicates in conjunction, which correspond to the logical rule of the anchor, outputting 1 when an input instance matches the rules and 0 otherwise. $A(x)$ is an anchor of a target model on instance x, if and only if:

$$\mathbb{E}_{\mathcal{D}_x(z|A)}[1_{f(x)=f(z)}] \geqslant \tau, A(x) = 1$$

where $x \in R^d$ is the target to-be-explained instance, $f : \mathbb{R}^d \rightarrow C$ is the prediction function of the target model (with C the shape of distributions over the possible predicted labels), $\mathcal{D}_x(\cdot|A)$ indicates a distribution of perturbed instances derived from x, also matching A. In simpler terms, A is an anchor if and only if, given a distribution of perturbed instances $\mathcal{D}_x$ around a target instance x, which all match A's predicates, at least τ percent of those instances match x's predicted label. In practice, because testing every single instance in distribution $\mathcal{D}_x$ would be difficult in large input spaces, this theoretical definition is instead relaxed to be probabilistic, and approximated by iteratively generating samples, until a certain statistical confidence threshold is reached. An additional desideratum for these anchors is to have their *coverage*, that is, the quantity of instances on which they apply, to be as large as possible, as otherwise, the best strategy to maximize precision only may result in the extraction of a huge quantity of highly specific anchors, which would severely reduce the practicality of this explainability method. For text classification tasks, the authors propose using a perturbation scheme where, unlike in LIME, tokens are not removed from the target instance but instead replaced by random words with both the same Part-Of-Speech tag (*Noun*, *Verb*, *Adjective*, etc.) and a high cosine similarity to the original token, as per the word embedding model used. The process for the extraction of anchors begins with the generation of single-feature candidate rules, for example, with text, single word-presence rules, for each word in the target instance. Using these candidates, perturbed neighbors are sampled using the given perturbation function, fixing the candidates' affected tokens in place so that the sampled instances necessarily match their corresponding candidate anchor rule. However, because this would otherwise require numerous calls to the underlying target model, which may be costly in some cases, a pure-exploration Multi-Armed-Bandit (in this case, the *KL-LUBC* algorithm from Kaufmann and Kalyanakrishnan [2013])

is used to more efficiently explore and estimate each candidate's precision. Once the best candidate (with the highest precision, as defined above) is found, if it passes the τ threshold, then an anchor has been found. Otherwise, the candidate becomes a predicate of the new iteration's generation of rules candidates, thus looking at two-words anchors, then three-words anchors, etc., until a τ precision anchor is found. While this approach has many advantages, especially over LIME, mainly with anchors being very easy to interpret, a number of issues have been raised: the method requires a heavy amount of non-trivial setup and design choices, primarily for the perturbation function, and, as noted above, despite the steps taken to diminish this issue, the approach may still generate too-specific sets of rules which may not be very informative of a model's overall behavior outside of the associated specific instances.

Shapley Values: Originally coming from the field of cooperative game theory, *Shapley Values*, invented by Shapley [1953], is a method originally proposed to compute an ideal distribution of the total reward among players who cooperated as a coalition to gain this reward, based on how much each player contributed to the coalition. It was proposed to be adapted as a prediction explainability method, where the "players" are constituted by the features of the target instance under investigation, the "cooperative game" by the prediction task, and the "total reward" by the actual predictions/probabilities output, compared to some reference, or set of reference "empty" instances, in which none of the features of the target instance can be considered "present" (e.g. an empty sentence, zero embedding, or equivalent, in NLP), or to a randomly selected instance. In theory, the Shapley value of a feature should be computed by measuring the average marginal contribution of said feature, that is, how much "adding" or "removing", by swapping its value with that of a random or "empty" instance, said feature affects the prediction score, over all possible coalitions of features, that is, all other combinations of the other target instance's features being similarly added or removed. In practice, computing this value exhaustively is intractable for large numbers of total features, and so approximated methods have been proposed. For example, Štrumbelj and Kononenko [2014] have proposed using Monte-Carlo sampling to approximate the Shapley value $\hat{\phi}(j)$ of a feature j of an input x, using the following process:

$$\hat{\phi}(j) = \frac{1}{M} \sum_{m=1}^{M} \left(f(x_{+j}^m) - f(x_{-j}^m) \right)$$

where M is the number of Monte-Carlo sampling steps, $f(x_{+j}^m)$ and $f(x_{-j}^m)$ are the target model's predictions for x, but where a random number of random features have had their values replaced by the values of a random (or reference) instance z, respectively with feature j preserved ($+j$) or also replaced by drawing from z ($-j$). The advantages of this method is that it is relatively simple to understand and implement, though managing to do so in an efficient fashion is one of the main drawbacks (due to the number of sampling steps required from the

original model to obtain decent enough approximations, as the process must be repeated for each and every feature). It is also heavily dependent on how the reference or random instances are selected, as that will affect the marginal contribution estimation of each feature, which may actually be cleverly used to perform contrastive explanations in some situations.

SHAP: ***SHapley Additive exPlanations***, proposed by Lundberg and Lee [2017], is an additive feature attribution method, which proposes combining all the previously discussed methods: indeed, the authors note many of these approaches, in particular LIME, DeepLIFT, Layer-wise Relevance Propagation, and Shapley Values, all more or less closely fit the same additive *explanation model*, that is, an explainability method applied to a target model, seen as a machine learning model itself, which SHAP formulates as:

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j$$

where $g : \{0,1\}^M \rightarrow C$ is the explanation model (with C the shape of distributions over class labels), $z' \in \{0,1\}^M$ can be seen as an *interpretable representation* of an original input z, as in LIME, or a *coalition vector*, as with Shapley Values, where each of the M interpretable features that is "present" is marked by a 1, or if "absent" by a 0. As in LIME, these interpretable representations will usually depend on the target to-be-explained instance, x, and can be mapped back to the original input space through the mapping function h_x, where $h_x(x') = x$. ϕ_j corresponds to the feature attribution for feature $j \in [1..M]$, essentially a generalization of the Shapley Value of j (with ϕ_0 the original model's output for an "empty" reference instance, where none of the M interpretable features are present). For a target model f and a target instance x, local explainability methods that fit this type of explanation model usually try to ensure that, for samples z' drawn close to x', $g(z')$ predicts the same output as $f(h_x(z'))$. In particular, the explanation model must match the original model exactly on the target instance x, that is $g(x') = f(x)$, which the authors denote as the *local accuracy* property. They argue for two other desirable properties for such explainability methods: *missingness*, which states that an "absent" interpretable feature $x'_j = 0$ should have an associated feature attribution ϕ_i of 0; and finally, *consistency*, which states that if the marginal contribution (see Shapley Values above) of a feature j increases or stays the same, so should its associated feature contribution ϕ_j. The authors then show that the only solution that satisfy these three properties are Shapley values, and propose a novel algorithm to approximate them, KernelSHAP. This proposed method combines steps from the LIME and the Monte-Carlo sampling Shapley values algorithms, but using a novel way to weight the samples, the *SHAP kernel*: first, random coalition/interpretable features vectors $z' \in \{0,1\}^M$ are sampled, using the same replacing of "absent" feature values with values from either a randomly selected, or a specifically chosen reference "empty" instance. Then, the predictions by the original model $f(h_x(z'))$ are queried, which are then combined

with a weight attributed to each vector by using the *SHAP kernel* π_x:

$$\pi_x(z') = \frac{(M-1)}{\binom{M}{|z'|}|z'|(M-|z'|)}$$

where $|z'|$ is the number of non-zero values in z' (corresponding to the number of "present" interpretable features in the vector), and M is the total number of possible interpretable features. This weighting function is derived from the weighting factor used in computing Shapley values for regression models [Lipovetsky and Conklin, 2001]. Then, similar to LIME, a linear model is fitted onto these weighted samples with the collected labels produced by the original model as targets, whose trained weights then contain the approximated feature attribution values ϕ. These steps are repeated iteratively, as in Monte-Carlo sampling (see above), with a sampling strategy which prioritizes selecting "extreme" values of $|z_i|$ (that is, which have a high number of "present", or a high number of "absent" interpretable features), which are weighted higher by the *SHAP kernel*. The authors propose different variants of this algorithm, better adapted to different types of target models. In Lundberg et al. [2019], the authors adapt the SHAP to Tree Ensembles models. In theory, SHAP is the culmination and unification of many of the previously discussed approaches, and thus combines many of their advantages. It also unfortunately combines many of their disadvantages, mainly the high compute overhead necessary to obtain good approximations of Shapley values, especially for large target models which are costly to sample predictions for. Just as with Shapley values, the choice of reference (or non-choice of a random) instance has a great impact on the explanations which are generated, and this may be a disadvantage as it adds an important hyperparameter to tweak which non-expert users may have difficulty with. Finally, it is also prone to the same type of adversarial attacks as LIME [Slack et al., 2020], which puts into question the faithfulness of explanations produced by using it.

While most of these methods usually produce explanations with a relatively easy to parse format, that is, a set of numeric score associated to each *interpretable feature* in an explained instance, corresponding to how much each particular feature "contributed" to the model's decision, there has been some debate as to whether this type of explanations are actually useful to end-users, particularly to non-experts. Indeed, a number of contributions [Miller, 2017; Dodge et al., 2019; Verma et al., 2020; Kaur et al., 2020; Kumar et al., 2020; Sharma et al., 2020] seem to point towards the fact that human users prefer *example-based explanations*, and in particular, *counterfactual explanations*.

4.3 Example-Based Explanations

Contrary to local feature-attribution–based explanations, which may potentially present a user with a large number of features all contributing in a limited way to the algorithmic decision, a *counterfactual* presents a *causal* explanation of a prediction: "Had feature j not been present in this input instance, this decision would not have been taken by the model." is more directly actionable for end-users, as it presents an example of how to "fix" an undesired prediction, either by modifying the input, if the model's decision is judged to be acceptable, or by attempting to "repair" the model if it was not, potentially by using corrected or modified versions of this example as new training instances.

Counterfactuals: Counterfactual explanations and adversarial attack methods are fairly similar in overall concept: in either case, the goal is to find a minimal modification which can be applied to a particular target instance to cause a change in the target model's prediction. However, where an adversarial attack method attempts to find modifications which are not perceptible as such to human evaluators and annotators, a counterfactual explanation method should instead try to produce modified instances in a way that is as naturally contrastive with regard to the target instance as possible. While considered in the category of example-based explainability methods, note that a counterfactual may not necessarily be an actual instance drawn from the target model's training dataset. A wide variety of approaches have been proposed to generate counterfactuals x' with changed label y', given a target model f and instance x. For example, Wachter et al. [2018] propose minimizing the following loss function:

$$\arg\min_{x'} \max_{\lambda} \mathcal{L}(x, x', y', \lambda) = \lambda(f(x') - y')^2 + d(x, x')$$

where λ represents a "closeness" factor between the desired counterfactual output prediction y' and the actual model prediction $f(x')$ for the counterfactual, and $d(x, x')$ represents some distance function between the target instance and the generated counterfactual. In the original contribution, the authors propose using a Manhattan distance weighted by the inverse median absolute deviation over the entire dataset (X) for each feature:

$$d(x, x') = \sum_{j=1}^{p} \frac{|x_j - x'_j|}{\text{median}_{a \in X}\left(|a_j - \text{median}_{b \in X}(b_j)|\right)}$$

To allow users to set their preference with regard to the trade-off of generated counterfactuals x' between being closer to x, and the counterfactuals outputs $f(x')$ being closer to the desired output y', λ instead starts at a low value, and is then maximized until constraint $|f(x') - y'| \leqslant \epsilon$ is matched, with ϵ a user defined hyperparameter. Any suitable optimization algorithm can be used with this objective, depending on the type of target module used. With neural architectures,

the authors suggest using the ADAM optimizer, a popular gradient descent algorithm used in machine learning. One limitation of this approach is that the suggested distance measure does not enforce creating counterfactuals with few feature changes (due to the Manhattan distance), which is desirable to avoid overloading the end-users of such explanations. As an improvement, Dandl et al. [2020] propose a more complex multi-objective loss function, which, in addition to enabling taking into account categorical features (via Gower's distance [Gower, 1971]), also enforces sparsity in feature changes, as well as a higher similarity of the generated counterfactual to the target model's training data distribution, leading to more "realistic" counterfactuals in theory. The main limitation of counterfactual explanations are that, unlike local feature-attribution explanations, they are not exhaustive, that is, one could in theory create an infinite number of counterfactuals which may all distinctly explain one aspect of the target model's behavior on the target instance. In practice, similar to too numerous feature-attribution scores, users will only be able to focus on at most a few counterfactuals at once for each diagnosed instance, and the ones presented may not necessarily end up being the most actionable for them.

Another way to use examples to explain a model's behavior, is to find the most *influential* instances in the training dataset, that is, instances which, if removed from the training corpus of a model and then retrained, would impact the new learned parameters of the model the most, either in a positive direction (improving the total loss metric), or a negative direction (worsening the total loss metric). These are often referred to as *deletion diagnostics*. Unfortunately, for obvious reasons, actually retraining an entire model for each instance to diagnose would be in practice too costly. As such, one proposed solution is to approximate the influence of an instance, without actually retraining the model.

Influence Functions: Koh and Liang [2017] proposed using a well-known tool of robust statistics called *influence functions* [Cook and Weisberg, 1980], in which, instead of simulating the effects of the actual removal of target instance $\bar{x}$, the same effects are approximated by upweighting the loss value associated with $\bar{x}$ in the training process of a model with parameters θ and training dataset X:

$$\hat{\theta}_{\bar{x},\epsilon} = \arg\min_{\theta}(1-\epsilon)\frac{1}{|X|}\sum_{x\in X}\mathcal{L}_{\theta}(x) + \epsilon\mathcal{L}_{\theta}(\bar{x})$$

where $\hat{\theta}_{\bar{x},\epsilon}$ represents the approximated new parameters after the upweighting of $\bar{x}$, ϵ represents an infinitesimally small up/down-weighting factor, and $\mathcal{L}_{\theta}$ is the target model's loss function. To compute this, we can thus use the *influence function* $I_{\text{up,params}}$ of the learned parameters $\hat{\theta}$ to

find how they will be impacted when the instance $\bar{x}$ is unweighted:

$$I_{\text{up,params}}(\bar{x}) = \frac{d\hat{\theta}_{\bar{x},\epsilon}}{d\epsilon}\Big|_{\epsilon=0} = -H_{\hat{\theta}}^{-1}\nabla_\theta \mathcal{L}_{\hat{\theta}}(\bar{x})$$

where $\nabla_\theta \mathcal{L}_{\hat{\theta}}(\bar{x})$ is the gradient of the loss function with respect to the model parameters, and $H_{\hat{\theta}} = \frac{1}{|X|}\sum_{x\in X} \nabla_\theta^2 \mathcal{L}_{\hat{\theta}}(x)$ is the Hessian matrix (second derivative) of the loss function with respect to the model parameters. Using this influence function, we could approximate the new post-upweighting parameters $\hat{\theta}_{\bar{x},\epsilon}$ without actually retraining the model, starting from the learned parameters $\hat{\theta}$:

$$\hat{\theta}_{\bar{x},\epsilon} \approx \hat{\theta} - \frac{1}{|X|} I_{\text{up,params}}(\bar{x})$$

These new parameters however need not actually be computed, as we can use a similar influence function, this time of the *loss* function directly, to estimate what effects on the predictions of a model the deletion of instance $\bar{x}$ would have had. Given an instance x_{test} we wish to evaluate the imparted changes in loss value on, we can use the chain rule to obtain the following influence function:

$$\begin{aligned} I_{\text{up,}\mathbf{loss}}(\bar{x}, x_{\text{test}}) &= \frac{d\mathcal{L}_{\hat{\theta}_{\bar{x},\epsilon}}(x_{\text{test}})}{d\epsilon}\Big|_{\epsilon=0} \\ &= \nabla_\theta \mathcal{L}_{\hat{\theta}}(x_{\text{test}})^\top \frac{d\hat{\theta}_{\bar{x},\epsilon}}{d\epsilon}\Big|_{\epsilon=0} \\ &= -\nabla_\theta \mathcal{L}_{\hat{\theta}}(x_{\text{test}})^\top H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}_{\hat{\theta}}(\bar{x}) \end{aligned}$$

The best way to interpret and then actually approximate the computation of this function is to split it into two parts (highlighted in blue and red): $\nabla_\theta \mathcal{L}_{\hat{\theta}}(x_{\text{test}})^\top$ essentially represents how much the loss value for instance x_{test} is affected by the changes in the learned parameters post-upweighting of instance $\bar{x}$; $H_{\hat{\theta}}^{-1}\nabla_\theta \mathcal{L}_{\hat{\theta}}(\bar{x})$ on the other hand approximates the changes the target model's learned parameters if the model was retrained with instance $\bar{x}$ upweighted (see $I_{\text{up,params}}(\bar{x})$ above). By cleverly rearranging this formula into two new parts (in green and orange) in the following fashion:

$$\begin{aligned} I_{\text{up,}\mathbf{loss}}(\bar{x}, x_{\text{test}}) &= -\nabla_\theta \mathcal{L}_{\hat{\theta}}(x_{\text{test}})^\top H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}_{\hat{\theta}}(\bar{x}) \\ &= -H_{\hat{\theta}}^{-1}\nabla_\theta \mathcal{L}_{\hat{\theta}}(x_{\text{test}}) \nabla_\theta \mathcal{L}_{\hat{\theta}}(\bar{x}) \\ &= -(\text{IHVP}) \nabla_\theta \mathcal{L}_{\hat{\theta}}(\bar{x}) \end{aligned}$$

we can divide the problem into two distinct sub-problems: the first allows bypassing the computation of the inverse Hessian, by using implicit Inverse Hessian Vector Products (IHVP) techniques [Pearlmutter, 1994]. The authors discuss two algorithms from the literature, one based on conjugate gradients [Martens, 2010], the other on stochastic estimation [Agarwal

et al., 2017]. In either cases, this part of the influence function need only be estimated once for a model, and thanks to the automatic differentiation capabilities of most machine learning programming libraries (TensorFlow, PyTorch, etc.), it is not too difficult to implement for most models. The second part, for similar reasons, can also be obtained cheaply, as indeed, $\nabla_\theta \mathcal{L}_{\hat{\theta}}(\bar{x})$ is nothing more than a prediction of $\bar{x}$ by the trained model, with the loss function and its gradient with respect to the learned parameters instrumented. However, to implement this method, two properties must be respected by the model: firstly, its loss function must be convex (otherwise H may not be positive-definite, and as much possibly not invertible). This can in practice be enforced by using a l_2 (or weight-decay) regularization step on the learned parameters. The second constraint is that the loss function must be twice differentiable, which is slightly more problematic, as many modules used in modern architectures are not, for example, ReLU $\big(\text{ReLU}(x) = \max(0, x)\big)$ and other similar piece-wise activation functions. One solution proposed by the authors to this problem is to use smoothed variants of these functions for the purpose of approximating influence values. While more complicated to implement than most of the other methods presented above, this particular post-hoc explainability method is very powerful. Its most obvious application is to explain a model's prediction of a target instance, by finding and presenting the most influential (positively or negatively) training instances that led the model to learn this particular behavior, which may not only help to diagnose the source of mispredictions for example (mislabeled instances or ambiguous training instances possibly), but also potentially fix the issue, by actually removing/relabelling the negatively influential instances from the dataset, and/or by adding more variations of beneficial instances to the data set. The authors also show how influence functions can be used to construct *training-set attacks*, a human-imperceptible adversarial modifications of a single training instance, which causes the model to mislabel a specific selection of test instances, once retrained with the tampered with instance.

Feature-attribution and example-based explainability methods have many interesting properties, but also a few important drawbacks. Mainly, most of them do not scale well with especially large models, such as those that are becoming more and more common in state-of-the art NLP (see Table 1.1). On the other hand, alongside their increase in complexity comes an increase in capabilities, in particular, to generate natural-looking texts which often appropriately fit a given set of contextual information. Taking inspiration from how humans generally communicate explanations or justifications of their own behaviors, that is, through verbal or written natural language explanations, we could imagine a class of models which may be trained to explain their own decisions, in a clear and easy to understand manner, not with numeric feature-attribution scores or example relevant instances, but with free-form *natural language explanations* directly.

4.4 Natural Language Explanations

With the increasing size and complexity of modern state-of-the-art neural models[1], exemplified with the advent of Transformer-based [Vaswani et al., 2017] architectures in NLP, such as BERT [Devlin et al., 2019], and more recently Large Language Models (LLMs) such as GPT-3 [Brown et al., 2020], explanation methods based on pure model internals analysis may become less and less viable, due to the very large number of parameters and depth, which poses explanation size and computational tractability issues. As discussed previously, while a wide variety of contributions have explored different, mostly indirect methods to showcase the presence of, and/or attempts to extract, particular forms of learned knowledge and behaviors from these types of models, one popular example being the informal subfield of "BERTology" (see Rogers et al. [2020] for an overview), there is still a lot of debate as to the ability to accurately and faithfully directly interpret the behavior of components of these models, such as attention layers [Jain and Wallace, 2019; Wiegreffe and Pinter, 2019; Klein and Nabi, 2019; Serrano and Smith, 2019; Pruthi et al., 2020].

Another limitation of most interpretability or explainability methods is that they are not necessarily directly exploitable by non-expert users: indeed, while methods such as LIME [Ribeiro et al., 2016] or SHAP [Lundberg and Lee, 2017] claim to be usable by non-expert users (which, in the former, is evaluated by having human subjects use the method's feature importance outputs to select the qualitatively best performing model out of two, or, to perform feature engineering to improve a model's performance), they in practice require some level of expert knowledge to initially set up, as well as to know their limitations, in particular on the types of data and models they are best suited to, and further on to interpret their explanatory outputs [Horne et al., 2019; Hase and Bansal, 2020]. In the context of the increasing need for a "right to explanation" [Goodman and Flaxman, 2017] of algorithmic decisions, it would feel a little counterproductive for interpretability or explainability methods to themselves require experts-produced explanations, to be understood by the average, not necessarily tech-savvy end-users (or targets) of these algorithms.

As such, an alternative which has been proposed is to generate explanations which are closer in form and content to the common usage of the word, that is, more or less free-form natural language text, acting in a sense as human-understandable "justifications" of an algorithm's decisions. These would have many advantages: first of all, as mentioned before (and argued by Camburu et al. [2018] for instance), natural language explanations would be more easily directly understandable, and thus in theory actionable, by non-expert users. Secondly, being easier to parse by humans, for the same underlying reasons, also makes them equally easier

[1] For more details on the Natural Language Processing architectures discussed below, such as Recurrent Neural Networks (RNNs), Transformers, etc., see Section 5.1 in the following Chapter.

to get human annotations for, either manually, as a task-specific annotation process (where explanations might also be requested alongside the actual task's gold labels, for instance), or automatically, from already existing adequate natural language content which could be construed as explanations (such from as internet forums discussions or arguments, for instance). Because of this theoretical easier access to annotated data pertaining to explained decisions, it is possible to consider methods which may automatically extract, or more appropriately, generate these forms of explanations, as supervised NLP tasks in and of themselves. In particular, to that end, one could then exploit the advances in natural language generation capabilities provided by the now popular large opaque architectures, such as Transformers, those very same architectures which may have been otherwise difficult to explain through more formal methods.

We thus chose to mostly focus our exploration of explainability methods, at this other end of the spectrum, from mostly fully-transparent intrinsically interpretable but less powerful models discussed in Chapter 2, to "self-justifying" more performant, but more opaque by nature as a tradeoff, explanation-generating models. More specifically, we chose to focus on approaches which are as generic and non–domain-specific as possible, in the interest of attempting to deploy such systems in as many domains and tasks as they can be found to work well in.

4.4.1 NLE Datasets

In practice, we can find various specific forms these natural language explanations (NLE) can take (see Wiegreffe and Marasović [2021][2] for a review and overview of various NLE datasets), on a spectrum from relatively structured explanations, usually specific to the task or the type of data at hand, all the way to completely free-form text. Somewhere on that spectrum, we could also place textual highlights, that is, explanations that take the form of a set of potentially non-contiguous spans of text (usually individual words or pieces of sentences) from the input texts, which, while not necessarily as clear and explicit as a separate explanatory text, can be considered a form of implicit explanation through referencing of the relevant parts of the input (something which is often done explicitly in separate natural language explanations). If we look at the relevant literature, for the more structured NLE category, we can find for example formal reasoning chains, found in the eQASC multihop question answering dataset [Jhamtani and Clark, 2020], based on the QASC [Khot et al., 2020] dataset, in which each multiple-choice question instance has been augmented with a set of valid or invalid reasoning chains (for example, of the form: *"A is a B* ***AND*** *B has C* ***IMPLIES*** *A has C"*), for each of the possible answers, along with a gold valid reasoning chain corresponding to the correct answer. Text highlights can be found in many datasets, often referred to as "human rationales" in the more specific literature [Strout et al., 2019; Bastings et al., 2019], but also as an additional

[2] See also the `https://exnlpdatasets.github.io/` accompanying website.

type of data in other types of NLE datasets [Camburu et al., 2018; Rajani et al., 2019], or even can be a posteriori extracted from more traditional datasets, such as the *Stanford Sentiment Treebank* (SST) [Socher et al., 2013], as was done in Carton et al. [2020]. Finally, we can find fully free-form natural language explanations, where the annotators are left mostly free to formulate instances' explanations as they see fit (though quality controls are of course implemented to filter out undesirable annotated explanations). It is this latter type which we've mainly focused on in this exploration of explainability methods.

To illustrate the different types of natural language explanations discussed previously, we propose an SNLI [Bowman et al., 2015] instance, alongside an example of what could be one of each type of NLE generated each by a dedicated model:

Premise:	"*A girl playing a violin along with a group of people.*"
Hypothesis:	"*A girl is washing a load of laundry.*"
Ground-truth Label:	*Contradiction*
Predicted Label:	*Contradiction*
Reasoning Chain:	*"playing" is not "washing"* ***AND*** *"a violin" is not "a load of laundry"* ***IMPLIES*** *"A girl playing a violin" is not "A girl is washing a load of laundry".*
Text **[Highlights]**:	"*A girl* ***[playing a violin]*** *along with a group of people.*"; "*A girl is* ***[washing a load of laundry]***."
Free-form:	"*One cannot be playing a violin while washing a load of laundry.*"

Within the free-form category of NLE datasets, we can find various contexts and tasks, which influences the content and general form the accompanying explanations will take. For example, we can find some relatively domain-specific datasets, such as the algebraic word problems presented in Ling et al. [2017], and their associated answer rationales, made up of natural text interspersed with algebraic expressions detailing the precise steps one may take to solve each particular mathematical problem, which the authors use to automatically derive formal and executable *programs* which algorithmically produce the problem's answer. We can also find both categorical and free-form annotations of possible offensive implications in Sap et al. [2020], or explanations of internet arguments' persuasiveness in Atkinson et al. [2019], in the context of social media posts. While a lot of NLE datasets unfortunately do not necessarily have enough instances to use more data-intensive deep NLP architectures, often due to the requirements and costs associated with the crowdsourcing of annotations (which is in practice usually the only way to obtain the large quantities of data required by deep learning methods), these few example corpora do contain a significant amount of exploitable instances (in the order of magnitude of tens of thousands of instances, where other datasets are often more in the order of a few hundreds). However, because their explanations were on the more

domain-specific end of the spectrum, we chose not to explore them further for our experiments. On the other end of this spectrum, we can find more domain-generic datasets, associated with more broadly applicable tasks such as general question answering or classifying semantic relations such as inference/entailment. We also only considered purely-textual tasks, so we did not explore in details visual-language datasets for instance, such as those found in the E-ViL benchmark [Kayser et al., 2021].

As such, we focused mainly on two NLE datasets which we thought were most representative of the general idea of this approach to explainability: E-SNLI provided by Camburu et al. [2018], which, as its name implies, is a variant of SNLI [Bowman et al., 2015] augmented with both free-form text and highlighted text spans that serve as human-annotated explanations; and CoS-E, provided by [Rajani et al., 2019], an NLE dataset for **co**mmon**s**ense (that is, where "commonsense" world-knowledge and implicit semantic relation rules are required) question answering, derived from the CommonsenseQA dataset [Talmor et al., 2019], augmented with human-annotated explanations, in the form of free-form explanations and highlighted text spans. In particular, we mostly focused on the former, E-SNLI, as it is the largest of the two (being on based on the large-scale SNLI, which contains more than $500\,000$ instances, vs. approximately $10\,000$ in CoS-E), and because it also was a better fit, conceptually and in its intended usage/evaluations, where CoS-E's explanations are more intended as an intermediate augmentation step in order to improve downstream performance.

E-SNLI: Camburu et al. [2018] introduce E-SNLI, a variant of the SNLI dataset (see the relevant paragraph in Section 2.4 for more details), augmented with natural language explanations. More precisely, annotations were crowdsourced using the Amazon Mechanical Turk platform, where annotators were presented with SNLI instances (*premise*, *hypothesis*, and *gold label* of the inference relation), and then asked to provide natural, non-obvious (that is, ideally avoiding referring to elements which are verbatim overlapping between the premise and hypothesis), self-contained (that is, which stand on their own as sentences and do not necessarily require reading the premise and hypothesis) explanations, in two steps (to attempt to filter out inadequate annotations): first, for each instance, annotators were tasked with highlighting relevant words from the premise and/or hypothesis which help to explain and justify the gold inference relation; then, they were tasked with writing a free-form explanation, which referenced at least a part of the previously highlighted words. Specific instructions were designed at each step for each of the possible inference labels (*entailment*, *neutral*, *contradiction*), to guide the non-expert annotators, as well as to further attempt to filter out low-quality annotations. A single explanation per instance was collected for the training set, whereas three were kept for the test and development sets. The authors note that numerous annotations unfortunately fell into "template-like" patterns, in which the entire hypothesis and premise are inserted into an almost static sentence template, dependent on the inference label, for instance, for *neutral*

$(premise, hypothesis)$ pairs: *"Just because <premise> does not mean <hypothesis>."* The authors attempted to filter out and re-annotated explanations which fell into such patterns, but subtle variations in the templates used appear to have left quite a few of them in the data. The authors also then provide different architectures and setups which, using this collected NLE data, learn to both generate an explanation given the regular SNLI inputs, alongside predicting the correct inference relation for that instance. We will discuss these architectures in the next section.

CoS-E: [Rajani et al., 2019] propose the CoS-E dataset, which expands upon the CommonsenseQA (CQA) question answering dataset [Talmor et al., 2019], adding human-annotated explanations alongside the original input questions and answers. The CQA dataset originally contains so-called "commonsense" questions, which have been specifically designed to require background world-knowledge to answer, in contrast to most other work on question answering, in which questions are usually framed within an explicit context, often an accompanying text passage in which the answer can be found. Each instance, made up of a question, and three (five in the v1.11 of the dataset) possible answers, only one of which is the correct one, were themselves crowdsourced, using associated concepts extracted from ConceptNet [Speer et al., 2017] to guide crowd-workers. These natural language explanations were also collected through crowdsourcing using the Amazon Mechanical Turk platform. Annotators were, similarly to e-SNLI, asked to first highlight relevant parts of instances' questions which justify the gold answer, then to provide a short free-form explanation text to the same effect. Similarly, the authors also performed various filtering steps to improve the quality of the annotations, including checking for template-like explanations (e.g. *"<answer> is the only option that is correct/obvious"*). The authors then propose using these collected NLEs in a Commonsense Auto-Generated Explanations (CAGE) framework, training a language model (GPT, from Radford et al. [2018]) to predict these explanations, conditioned on the input questions and answer choices, in either one of two ways: an *explain-then-predict* (also called "reasoning") setup, where the explanations only depend on the inputs and not the predicted answers, which may themselves be derived from the questions, answer choices and just-produced explanations, in a later step, by a classification model (in this case, a BERT model); or, a *predict-then-explain* (also called "rationalization") setup, in which the language model which produces the explanations is conditioned on the predicted (or ground-truth during training) answer. Thus, similar to [Camburu et al., 2018], the authors also propose architectures which both learn to perform the main task as well as produce natural language explanations. An example of a CoS-E instance with its human-annotated explanation (drawn from the development set) is:

Question:	"*Where is one likely to find a fan for their stove?*"
Possible Answers (**Correct**):	"***appliance store***", "*sports stadium*", "*hot room*"
Explanation:	"*stove and other appliances are bought in the appliance store*"

To work with these datasets, different types of architectures, which we will refer to informally from here on as "Explainer" models, have been proposed, with a lot of conceptual similarities, which enables us to regroup them into a few different categories.

4.4.2 Explainer Models

As succinctly explored in the previous section, most NLE datasets can be viewed as a more traditional NLP dataset, which has been augmented with accompanying explanations. As such, if one considers these explanations as an additional output sequence to be predicted alongside the traditional task output, a naive type of architecture which could be proposed may simply consist in separately performing the two sub-tasks, using two separate models. This type of approach quite apparently and obviously sounds unsatisfactory in the context of any chosen desiderata for explainability: indeed, even in the best case scenario, where there would be no limitations in learning capabilities, one could expect the explanation generating sub-model to internally learn to solve the main task in order to better produce the corresponding explanation, in which case the second sub-model solving the main task would feel somewhat redundant. In the general case, not allowing the two obviously linked parts of the task (the explanation and its target, the *explanandum*) to be reflected into a similarly linked architecture would almost invariably lead to suboptimal performance, for both sub-tasks.

Both learning to solve a task *and* learning to explain one's reasoning to solve this task can be expected to be more beneficial than doing either separately, and this is somewhat confirmed by the results from Rajani et al. [2019] on the CoS-E dataset, where learning to intermediately generate explanations appears to have significantly improved the performance when solving the main task, compared to directly attempting to solve the task alone. As such, if we consider architectures which link both of these sub-tasks, there are various ways one may proceed to join them, with the main distinguishing factor being the order of operations: should the architecture first produce an explanation, and then attempt to solve the main task, conditioned on this explanation, or, should the explanation be generated post-hoc, once a prediction has been produced for the main task? This distinction could be seen more as a spectrum than a set of strict categories, but still, as it appears to be a common factor approaches are differentiated on in the relevant literature, we can try to more or less strictly categorize methods depending on whether they fit one of the following two temporal orderings between prediction and explanation:

- the first type of approach, which are often referred to as variations of **"explain-then-predict"** architectures, consider that the explanations to be generated should behave somewhat like *reasoning* steps, expressed as natural language. In Rajani et al. [2019], this corresponds to the CAGE-reasoning architecture, where an autoregressive language model (GPT) is trained using the CoS-E dataset to generate explanations conditioned

on the inputs (questions and answer choices, concatenated), which are then themselves concatenated to the inputs and used by a question answering (fine-tuned BERT) model to learn to find the correct answer. Somewhat similarly, though the tasks are different, Camburu et al. [2018] propose a ExplainThenPredict model, based on an InferSent [Conneau et al., 2017] universal sentence-pair encoder architecture, coupled to a Multi-layer Perceptron classifier, to predict the inference relation. A similar architecture, with a LSTM decoder instead of the classifier (with two variants, a bare one, while the other has had two attention modules added to it) is used to first generate the explanations which are then fed into this predictor. Compared to the previous approach, the predictor here is only allowed to look at the explanation and not the rest of the input, though that is justifiable considering the differences between the tasks and the forms the collected NLEs take: the authors of e-SNLI indeed note that in their dataset, the forms of the explanations (mainly, the linking words or expressions used to connect related elements in the premise and hypothesis) are highly correlated with the inference labels, and thus should be sufficient input for the predictor. This type of approach is in theory more desirable with regard to common desiderata for explanations, particularly, faithfulness: indeed, in this case, since the explanation/reasoning has to be generated prior to the solving of the main task, it is less likely that the model will be able to confabulate a valid-sounding justification for the prediction which would be completely unrelated to the actual internal reasoning of the model, though this is of course no strong guarantee that it will not be the case. One could for instance imagine the explanation generator "sneaking in" hidden, or at least not clearly apparent information within the explanatory text, to the predictor, by subtly toying with the distributional characteristics of the text, through obscure lexical cues for example.

- the second type of approach, often referred to as variations of **"predict-then-explain"**, invert the order of the steps described previously: instead of the explanations being treated as natural language reasoning towards a prediction, they rather behave like post-hoc *rationalizations* of an already taken decision, their generation being conditioned on the predicted (or, at training time, the ground-truth) label, in the case of classification tasks. In Rajani et al. [2019], this corresponds to the CAGE-rationalization architecture, which is identical to the previous one, except the prediction of the correct answer is done first, and this output is concatenated to the rest of the input normally fed to the language model. Camburu et al. [2018] correspondingly propose a PredictAndExplain model, where the same InferSent architecture is used to produce an encoding vector for the input sentences pair, which is then first fed into an MLP classifier predicting the inference label, that is then used to condition, alongside the same input encoding vector, an LSTM decoder, trained to produce the explanation. As its name would imply, this model could be seen as jointly performing both objectives at once (with two different

"head" modules), however conditioning the explanation generator on the predicted label we argue puts it more in the post-hoc rationalization category, though this distinction is again somewhat subjective. This second type of approach seems conversely less desirable with regard to the faithfulness of produced explanations, as they are generated after the decision has already been taken, so to speak, and may thus be more likely to resemble the human behavioral sense of "rationalization", that is, a plausible and/or convincing justification for a decision, which may not correspond in any way, shape, or form, to the actual reasoning process which led to the decision, which would be problematic. Similarly to the previous type however, there is no strong guarantee of this, and indeed qualitative analyses performed in Camburu et al. [2018] and Rajani et al. [2019] did not seem to showcase significant differences relevant to this criterion between output explanations from these two types of approaches.

Various other contributions have proposed approaches for generating natural language explanations, a lot of them fitting more or less closely within these two categories.

For example, Liu et al. [2019a] propose a model-agnostic generative explanation framework for datasets with fine-grained explanatory elements, such as detailed evaluation criteria in products or services reviews, for example "price", "quality", or "practicality". Their generic Generative Explanation Framework proposes a middle-ground joint *prediction-and-explanation* approach, with a common input encoder model, feeding into both a direct predictor, and an explanation generator, which itself feeds into a classifier. In addition to both of the chains' standard loss functions, the authors propose an additional explanation factor term, which combines distributional distances between ground-truth and generated explanations, and also between generated explanations and the original text.

Atanasova et al. [2020] propose an approach based on a transformer architecture which jointly predicts the veracity of claims and extracts relevant explanatory sentences from the associated ruling comments, akin to an extractive summarization task. They similarly propose a more joint approach, however the explanations take the form of a binary selection of a number of "ruling sentences" accompanying each claim, provided by the LIAR-PLUS [Alhindi et al., 2018] dataset.

Kumar and Talukdar [2020] propose NILE, an NLI (also trained and evaluated on the E-SNLI dataset) multi-Explainer system, which independently generates three explanations (using three GPT-2 architectures), one for each of the inference-relations' labels in the task (*entailment, neutral, contradiction*), which are then processed together and used to classify the instance (using a RoBERTa [Liu et al., 2019c] architecture). The explanation corresponding to the predicted label is then selected and passed-through as the instance's output explanation, alongside the predicted label. The authors argue that this approach has advantages over methods which

sequentially generate a single explanation and derive a label prediction from it (or vice-versa), as it allows better probing of the explanations' faithfulness to the model's actual internal reasoning process, since one can look at what each of the three label-specific explanation generators outputs for a given instance. Though it uses similar concepts, this approach intentionally differs significantly from the *explain-then-predict* and *predict-then-explain* categories presented above: indeed, while the overall architecture first generates explanations then makes a prediction based on them, each of the three individual explanation generators is fine-tuned to generate explanations for a specific label, hence one could argue they should be in the *predict-then-explain* category. While this allows for a sort of *counterfactual* probing of the generated explanations, these have in theory little basis on the actual content of the instance, as the individual generators are fine-tuned *assuming* every instance encountered is of their specific label.

Ultimately, the two Explainer models categories used here, as stated above, may be more accurately considered a spectrum, depending on what proportion of an Explainer architecture attends to either explaining or predicting. At the same time, one could also consider approaches where both objectives are in large parts attended to by the same base model, which is especially common for multi-objective training setups, particularly those using modern pre-trained Transformers [Vaswani et al., 2017]. Indeed, with these architectures, one can use and fine-tune the same "core" model for various tasks, potentially at once, in multitask learning setups [Kaiser et al., 2017]. The core may thus learn to encode inputs into general-purpose information-rich representations, which can then be used by different adequately designed "head" modules, who only need to learn relatively simple task-specific mappings from those representations to the required output format, extracting from them only the necessary information to solve their associated sub-task. It is this latter avenue which we decided to explore experimentally, which will be discussed in the next Chapter.

Chapter 5

Explanation Generating Classifiers — Experiments

Taking inspiration from the various contributions discussed previously, we have chosen to focus our experiments on the e-SNLI [Camburu et al., 2018] and CoS-E [Rajani et al., 2019] datasets, and on Transformer-based [Vaswani et al., 2017] architectures, which we argue are a good fit for this objective of jointly solving a task and generating explanations: indeed, these architectures have been successfully used in various tasks, such as summarization [Liu and Lapata, 2019], question answering [Talmor et al., 2019], and reading comprehension [Xu et al., 2019], all of which require learning skills which would be essential to producing good explanations. Moreover, most of these pre-trained models can be adapted and fine-tuned to accomplish these tasks simply by adding one or multiple appropriate task-specific "head" modules. With the correct configuration, such a model may be able to jointly learn the skills required to predict and to explain, each aspect hopefully helping to improve the other.

As these notions will be important to understand the experimental setups used here, the next section may serve as a short summary or refresher on Transformers, and the architectures that preceded them.

5.1 A Few Words on Transformers

Natural Language Processing is a particularly challenging field of AI and machine-learning research, because a "canonical" conceptual architecture to deal with natural language as an information medium has not yet been found. In Computer Vision for example, deep Convolutional Neural Networks (CNNs) [LeCun et al., 1990, 2004, 2010] have proved to have the ideal set of properties to solve most vision tasks (the ability to detect local patterns, initially at the scale of pixel neighborhoods, followed by larger scale and more abstract patterns with

each additional pooling and convolution layers), being themselves a rough approximation of how the natural visual perception mechanisms function in animal brains [Gu et al., 2018]. For natural language however, so far, finding a similarly suitable conceptual architecture has proven to be more difficult: first of all, there is no single self-evident decomposition into semantic units for language, like there is with pixels for image data. Furthermore, given some decomposition scheme, for example, tokenization of texts into sequences of individual words, there is no trivial or self-evident way to encode these tokens into numerical representations, though thankfully, over the years, state-of-the-art results were achieved using various word embedding models (see Chapter 2 for a quick overview). Finally, assuming all these previous decisions have been addressed, no single conceptual architecture evidently presents itself to deal with encoding entire sequences of tokens. Instead, various types of approaches have been experimented with over the years: initially, Bag-Of-Words (BOW) and related representations were used [Harris, 1954], in which a text or document was represented by the aggregation (sum or average usually) of one-hot encoded words (or *n-grams*) contained in it, irrespective of the order or structure of the text. These types of representations have many disadvantages, mainly linked to this last fact, as well as the inefficient number of dimensions required for the produced vectors (scaling with the size of the vocabulary), and the absence of geometrically encoded semantic aspects (such as can instead be found in word embeddings, with distance-as-similarity metrics). To tackle this problem of taking the order words and the general structure of documents, a first family of models was proposed to be used: Recurrent Neural Networks [Rumelhart et al., 1986] (see Figure 5.1), which were intended to model, and also were roughly inspired by various phenomena or systems with time-persistent features, including the functioning of short and long-term memory in the brain [Little, 1974; Hopfield, 1982]. In addition to learning a mapping of numerically encoded inputs to outputs, this family of models are able to do so in the context of an ordered sequence of inputs and outputs, possessing one or multiple connections through time, allowing such models to "remember" past inputs/outputs when iteratively processing a sequence. This in theory makes them particularly suited to dealing with natural languages, which have strong sequential and time-based components to them. The two most popular implementations in the RNN family are the Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997], and Gated Recurrent Unit (GRU) [Cho et al., 2014] architectures, and these have enabled achieving, at the time they were presented, state-of-the-art performance on numerous NLP tasks. The most successful architectures make use of bidirectional variants of these two architectures, where at least two layers of such RNN cells are used in parallel, each running in opposite directions on the input/output sequences (with their final outputs usually concatenated), allowing for the capture of both left-to-right and right-to-left temporal information.

In particular, Neural Machine Translation models, and other so-called sequence-to-sequence models (often abbreviated as *"seq2seq"*) [Sutskever et al., 2014], in which both the inputs and

outputs are textual documents decomposed as sequences of tokens, often employ a particular dual RNN encoder-decoder setup (see Figure 5.2), in which an input sentence is first iterated upon and encoded into the hidden state of an encoder RNN module, that is then used to condition a decoder RNN, tasked with producing the expected output sequence, in an autoregressive fashion, predicting the $(i+1)$-th token in the output, based on the i-th token and the encoded representation of the input (usually initializing, and then being combined with the regular decoder's hidden state forward-flow in the following iterations). With the successes of these types of approaches when fine-tuned on specific tasks, such as Machine Translation, an idea rapidly emerged, generalizing from the then popular pre-trained word embedding models, and their ability to be used "universally" out-of-the-box, as input encoding modules in a wide variety of tasks and setups: could similarly "universal" phrase, sentence, or document representations be generated by such encoding models, pre-trained (ideally in an unsupervised fashion) on large quantities of textual data, capturing a variety of semantic information which could then be exploited in a variety of setups downstream? To attempt to answer this question various approaches were proposed: Kiros et al. [2015] for example propose the Skip-Thought unsupervised approach, abstracting the *skip-gram* [Mikolov et al., 2013b,a] word embedding model to the sentence level, where an RNN encoder-decoder setup learns to encode sentences inside a text by conditioning two decoders, each respectively tasked with predicting the sentences directly preceding and following the encoded sentence; [Palangi et al., 2016] employ a similar approach with an LSTM encoder model, trained on weakly supervised data from a commercial search engine, learning to produce sentence embeddings for user search engine queries and their associated clicked document with a high cosine similarity, in the context of Information Retrieval. Other similar methods were also proposed which did not necessarily use RNN-based encoder-decoder setups, or even neural networks at all: the InferSent model from Conneau et al. [2017] for example is also intended to produce such "universal" representations, though it does not use an unsupervised encoder-decoder setup, but rather an encoder-classifier trained on the SNLI dataset; Hill et al. [2016] propose an overview and evaluation of a variety of these approaches which were available at the time, as well as a surprisingly simple yet performant FastSent method, not based on RNNs but on Bag-Of-Words and a simple log-bilinear additive model, learning word embeddings which are summed to produce a sentence's representation, with the objective of predicting the words from adjacent sentences.

However, while RNNs presented many interesting features for NLP tasks, a few major intrinsic issues were quickly noticed: first of all, while not specific to RNNs, the unrolling process used in training (see Figure 5.1) caused vanishing gradient problems [Hochreiter, 1998], due to the depths of the effective networks being dependent on the desired unrolled size, which, to allow the proper processing of sentences and texts, ideally needs to be as long as the longest sequence in the dataset. While the LSTM [Hochreiter and Schmidhuber, 1997] architecture was designed to try to circumvent this issue, by implementing mechanisms akin

to long and short term memory in animal brains, it was still noticed that for particularly long sequences, and especially in encoder-decoder setups such as were used in Neural Machine Translation, even LSTM-based models had issues "remembering" early parts of input sentences. In translation tasks, this meant that for natural languages with differing word orders for instance (the order in which the different syntactic elements are arranged, for example, subject-object-verb vs. subject-verb-object), mistranslations would occur frequently for somewhat long sentences, as the semantic information from earlier parts of the input sentences would be overwritten or forgotten by the time the encoder had parsing all tokens. While bidirectional models could somewhat mitigate these issues, another mechanism was proposed to better allow decoders to refer to specific parts of the input, which was referred to as **attention** [Luong et al., 2015; Bahdanau et al., 2016]: instead of using the final hidden state produced by iterating the encoder as the input sequence's vector representation, all intermediate hidden states, in theory containing information relevant to their respective position in the sequence, are kept. The decoder is then augmented with a so-called *attention* module, which, at each decoder iteration, assigns a score to each of the previous encoder hidden states, using some kind of scoring function (either static or itself a learning neural network), usually taking as input the decoder's current input hidden state (outputted from the previous iteration) and each respective encoder hidden state. These attention scores (once passed through *softmax*), can be interpreted to be a sort of soft (non-binary) *alignment* or mapping of the output tokens to the input tokens, and are used as weights to compute (for instance, through a weighted sum) a custom input sentence encoding vector for each decoding step. Using this mechanism, a decoder can therefore learn to *attend* differently to each element of the input, at each time step, no matter how long the temporal distance between the two. This is in theory also more advantageous compared to using Convolutional Neural Networks, which are also a common alternative to RNNs (for classification tasks, or as sentence encoders), as CNNs can only capture long-distance relations with higher numbers of layers. Using this attention mechanism, state-of-the-art results were obtained for a variety of NLP tasks, such as Machine Translation [Luong et al., 2015; Bahdanau et al., 2016] or Natural Language Inference [Wang and Jiang, 2016], and many others, usually outperforming non–attention-based models.

Noticing how well this attention mechanism was performing, and especially how it seemed to circumvent many of the issues of RNNs, especially in their encoder-decoder configurations, Vaswani et al. [2017] proposed an innovative idea: what if attention alone could be used as a basis for a sequence-to-sequence architecture? In their now very famous "Attention Is All You Need" publication, they propose the Transformer architecture as an alternative to recurrent (or convolutional) architectures, which it departs from in a few important ways: first of all, unlike RNNs, Transformers have no sequential or iterative components to them, which eliminates all issues related to unrolling, and allows for better parallelization of computations (especially on GPUs or other related dedicated hardware). This also means that a Transformer has a fixed

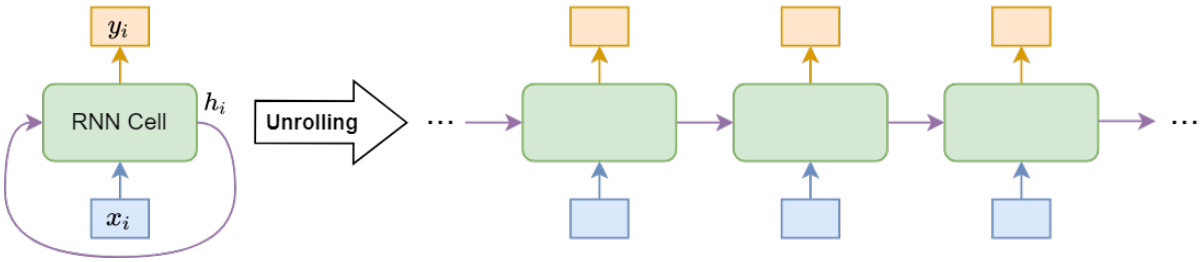


Figure 5.1: Illustration of a schematic Recurrent Neural Network architecture (RNN): an RNN cell is iteratively fed pairs of inputs (x_i) and outputs (y_i) from a sequence. To enable the learning of temporal patterns, information is allowed to flow between iterations, through the hidden state (h_i), which is passed "forward" (though one can also have a cell attend to a sequence in reverse order if needed, such as in bidirectional RNNs [Graves and Schmidhuber, 2005; Thireou and Reczko, 2007]) in time. In practice, a RNN cell is usually "unrolled" (see diagram) for a given number of iterations, behaving as a sliding window on the input/output sequences, to optimize computation and parameters fine-tuning.

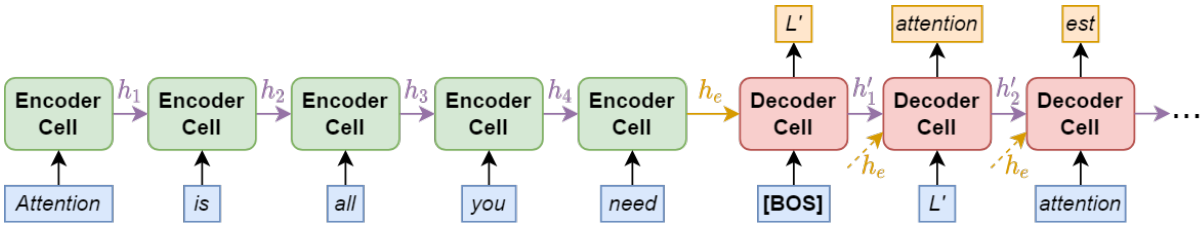


Figure 5.2: Illustration of a typical encoder-decoder setup using a Recurrent Neural Network architecture, such as can be found in Neural Machine Translation: a first RNN Cell module acts as an input sequence encoder (in green), iteratively feeding on each element of the input (*"Attention is all you need"*), while propagating forward through time a hidden state (h_i), until the end of the sequence is reached. At this point, the final hidden state (h_e) is considered to be an encoded representation of the first input sequence, and it is fed as part of the hidden state of a second RNN Cell module, acting as a decoder. This decoder is trained to generate the expected output (in this example, a translation of the input from English to French), in an autoregressive fashion, being iteratively fed as input the previous predicted output token, with the first iteration receiving a Beginning-Of-Sentence ([BOS]) marker. Usually, the input sequence encoding (h_e) is kept as-is as a part of every subsequent iteration's decoder hidden state, to help prevent the decoder from "forgetting" the input after many decoding iterations.

sequence length it can use as input and output, though in practice one may try to circumvent this by "compressing" text that falls out of the current window into a special token's embedding vector (such as the [CLS] token in BERT [Devlin et al., 2019]; see Figure 5.3). The Transformer architecture is based around a conceptually simple but generic and very powerful family of components which we can refer to as Transformer blocks, which are usually arranged in stacks, each block feeding its output as the input to the next in the stack, like in a regular neural architecture, where a complete model usually contains between one or two stacks. This stacking is made easier by the fact that a standard Transformer block intentionally uses the same dimensionality for its main input and output: as input, a block receives a matrix of a fixed width corresponding to a fixed-size sequence of embedding vectors, and outputs a similarly sized matrix, where each embedding vectors has been transformed and projected into a new embedding space, through an attention mechanism that spans the entire sequence (see Figures 5.6a and 5.6b).

Before a sentence can be presented as input to the stack however, a few important steps must be taken to turn it into a sequence of embedding vectors. First of all, while not absolutely mandatory, most popular Transformer architectures make use of particular tokenization schemes (transforming a contiguous string of characters into a sequence of separate linguistic tokens), usually based on *n-grams* or *word-pieces*: instead of splitting sentences into sequences of whitespace-separated word and punctuation tokens, which can require very large vocabulary sizes if one wishes to be able to handle rarely used words, rarer and harder to exhaustively include proper nouns, or even numerals, dates, and other miscellaneous elements, using sub-word tokens allows for a more efficient encoding of out-of-vocabulary elements, at the cost of slightly longer tokenized sequences compared to whitespace-based tokenization.The two most popular sub-word tokenization schemes in use currently are WordPiece [Wu et al., 2016], notably used in BERT, and Byte Pair Encoding [BPE; Gage, 1994; Sennrich et al., 2016], notably used in the GPT model family as well as many other Transformer architectures. Once the text is tokenized, each token can be projected into an initial embedding space, usually through a simple lookup matrix whose entries will be learned as part of the Transformer's training process. However, one issue that is essential to address at this stage is the encoding of positional information: indeed, due to their design, unlike RNNs, Transformer blocks do not have the intrinsic ability to model temporal information, since the attention modules used span indiscriminately over the entirety of their input sequences. Thus, a solution proposed by Vaswani et al. [2017] is to assign to each position in the sequence a *positional embedding* vector, which will be summed to the corresponding token embedding vectors, to add this positional information back into the input. These positional embeddings are usually constructed using multidimensional sinusoids, where each dimension corresponds to a different wavelength, allowing models to more easily learn relative positioning (which can be computed through simple linear transformations), also being a pattern that is easy to extrapolate for sequences length which may not have been encountered

during training. Some architectures also further add more miscellaneous information in this fashion, such as "segment embeddings", which encode which of a pair of two concatenated sentences a token is a part of (usually in addition to having a special separation marker, e.g. [SEP] in BERT).

At the end of a Transformer stack, one is left with a sequence of embedding vectors that have been repeatedly transformed through each block, and which can then be used in various ways, the first main one being to be decoded as another sequence of tokens, by simply projecting the embedding vectors (using a small linear model) into probability distributions over a token vocabulary, which is what is done as part of unsupervised Language Modeling (LM) tasks. Alternatively, one can project one of the output embedding vectors, or the entire output matrix, into any other type of distribution, over a set of labels for instance, to perform classification tasks. One can even perform multiple of these at once, using the same underlying Transformer model with a different set of "head" module(s) depending on the specific task to be solved, all taking as input the same Transformer-output embedding matrix, which would hopefully have captured enough semantic information from the input sentence(s), and encoded it in such a way that a simple shallow linear model may be fine-tuned to extract the parts relevant to its objective. Realizing this, two major implementations of the Transformer architecture, BERT [Devlin et al., 2019] and GPT [Radford et al., 2018], proposed a somewhat novel approach: pre-training large (compared to the comparable models of the time, see Table 1.1) Transformer models jointly on various unsupervised language modeling or classification tasks, using large quantities of unannotated (and thus more readily available) text data, in order to teach them to extract general purpose semantic information from arbitrary sentences. These pre-trained models may then be later fine-tuned on smaller datasets, alongside the appropriate smaller head modules, effectively performing transfer learning (see Pan and Yang [2010] for a general overview) or domain adaptation to specific tasks, without having to completely re-train a large model from scratch.

Devlin et al. [2019] thus proposed the now very famous pre-trained general-purpose Transformer model, called BERT (which stands for "**B**idirectional **E**ncoder **R**epresentations from **T**ransformers"). Using a single stack of Transformer blocks to form their model, the authors propose pre-training it using a novel Language Modeling objective, inspired by the Cloze task proposed initially by Taylor [1953], a language teaching or assessment tool, in which a student (human or, in NLP, machine) is presented with a piece of natural language text where a number of words have been erased or masked, and which must be filled back in with the help of the contextual information still present in the text: the authors thus propose a similar Masked Language Modeling (MLM) task (see Figure 5.3), in which an input sentence is perturbed with a number of tokens being masked (replaced by a [MASK] special token, or more rarely, with another randomly selected token), with the objective of predicting back the original tokens at

the masked positions. Additionally, as this model is intended to be used in various types of NLP tasks, and because many of these require understanding the semantic relations between two or more separate (but related) sentences, the authors propose adding another objective which will be learned jointly with the MLM one: given a pair of sentences, which are either following each other in a document of the training dataset (positive class), or two random unrelated sentences (negative class), the model is tasked with predicting which of these two classes the pair is from, which is called the Next Sentence Prediction objective. To do so, both sentences are first concatenated, with a special separator token (`[SEP]`) in between. Then, a special `[CLS]` ("Classification") token is prepended to the input sequence, and it is dealt with differently than the others on the Transformer's output side: where the rest of the output embedding matrix will be fed through the language modeling part of the joint task, the output vector corresponding to the `[CLS]` token (in the very first position) will instead be fed to a classification module training on the Next Sentence Prediction objective, as though it was an embedding vector for the entire input sequence. Combining these two objectives as part of the unsupervised pre-training process allows the BERT model to both learn intrinsically bidirectional language modeling (since masked tokens can occur anywhere, and in variable quantities in the MLM objective) as well as how to perform whole-sequence (potentially containing multiple `[SEP]`-separated sentences) classification tasks, through the `[CLS]` special token. Once pre-trained on a large quantity of unsupervised text, the authors showcase how the model can be fine-tuned to perform a more specific NLP task, such as Natural Language Inference for example, by concatenating the premise and hypothesis sentences, and using the `[CLS]` token to predict the inference label for each pair. They evaluate this model on the various tasks contained in the GLUE benchmark [Wang et al., 2018], and improve, at the time, on state-of-art approaches by a relatively wide margin on all tasks, using their largest model variant, $\text{BERT}_{\text{LARGE}}$, with approximately 345 million parameters. While pre-training such a large model requires a large amount of time and/or compute ($\text{BERT}_{\text{LARGE}}$ was reported to have taken 4 days to pre-train on 800 million words from the BooksCorpus [Zhu et al., 2015] and 2500 million words from English Wikipedia, using 16 Cloud TPUs simultaneously), once pre-trained, it can be fine-tuned for a specific task in a much more reasonable amount of time and compute (approximately a few hours on a GPU, depending on the dataset size and hyperparameters).

Radford et al. [2018] on the other hand proposed the Generative Pre-Training approach, with a similar overarching set of principles, but a few major differences, mainly in the pre-training setup: indeed, GPT uses a left-to-right autoregressive language modeling objective (see Figure 5.4), where, closer to RNN decoders than BERT's MLM objective, the Transformer is tasked with predicting the following tokens in a sentence, given all the previously predicted tokens. Due to the Transformer's non-recurrent nature, this is effectively implemented by modifying the attention modules (see Figures 5.6) to prevent any position from attending to positions to its right (which, in an iterative decoder, would not have been generated yet),

and then shifting the input sequence to the right, by usually, prepending a starting position Beginning-of-Sentence marker ([BOS]). At inference time however, once the model has been fine-tuned, such autoregressive models are actually iterated, initially feeding a starting input sequences (which may be at a minimum, a single [BOS] token), which will be run through the Transformer to predict the next token. The new expanded sequence is then fed back recursively, until some stopping condition is met, usually when the End-of-Sentence ([EOS]) token is generated, or once the maximum sequence length has been hit, whichever happens first. While simply picking the most probable predicted token at each decoding step is a possibility (usually referred to as "greedy" decoding), various other methods have been proposed which in practice produce much higher quality results, such as beam-search [Shao et al., 2017; Vijayakumar et al., 2018], in which multiple paths are explored in parallel in the probability-weighted decoding tree, selecting only the most likely complete path, as a whole product of all the decoding steps' weights. Unlike BERT, this is the only objective that GPT is pre-trained on. When fine-tuning on a specific whole-sequence (which may also be a sentence pair, using a similar separator special token) classification task, an appropriately sized linear classification head module is added, taking as input the entire Transformer output embedding matrix, instead of just a single dedicated position's vector, like in BERT. The autoregressive objective is also kept during fine-tuning as well, to prevent the fine-tuned Transformer from "forgetting" too much its pre-trained language modeling capabilities. This approach was also evaluated on downstream tasks from the GLUE benchmark [Wang et al., 2018], also beating the state-of-art performance of the time (though it was then beaten by BERT when it came out a year later).

In actuality, these two approaches can be viewed as two different specializations of the original Transformer encoder-decoder architecture initially proposed by Vaswani et al. [2017] (see Figure 5.5), each using one of the two Transformer stacks originally described (BERT, the encoder sub-Transformer, and GPT, the decoder sub-Transformer). This original architecture was itself tested and evaluated for Neural Machine Translation, where the original language sentence was first encoded by the encoder Transformer, whose output embedding matrix was then used in the cross-attention modules of the decoder Transformer, to condition the autoregressive generation of the target language sentence. In our own experiments, we generate explanations based on an input instance, for which we also need to condition the generation of a sequence on the representation of another; we therefore chose to use this more generic architecture. Specifically, we make use of the BART model [Lewis et al., 2020], which combines a Transformer-based encoder-decoder architecture with a specific denoising objective suitable for monolingual text.

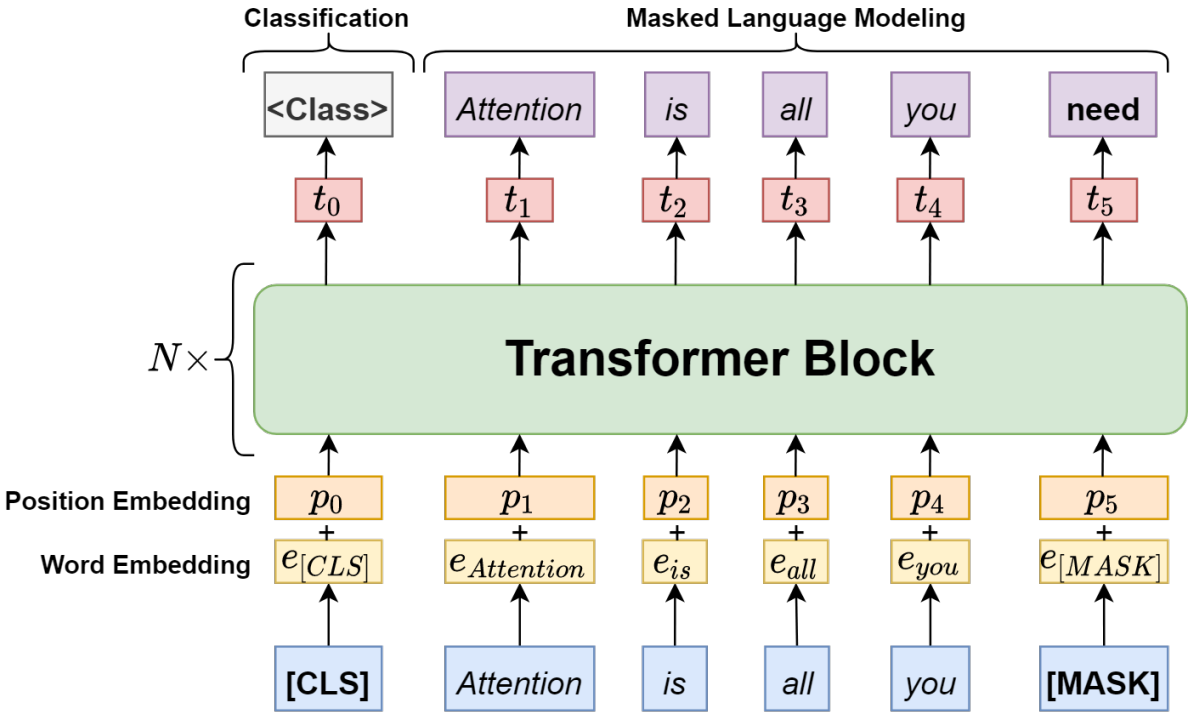


Figure 5.3: Schematic illustration of a typical Masked Language Modeling (MLM) training setup for a Transformer model, as can be found in BERT [Devlin et al., 2019] for instance: an input sentence (*"Attention is all you need"*) is perturbed by masking one of the words (here *"need"*) at any position in the sequence, replacing it with a special [MASK] marker. Each input token is then projected into an embedding space, however, because Transformers have no innate temporality modeling, the entire sequence being processed at once, with interactions between positions only taking place inside the attention mechanism of the Transformer blocks (see Figure 5.6), positional embeddings are added to form the final input embeddings. These are then processed by a given number N of Transformer blocks, with the final layer outputting a vector representation (t_i) corresponding to each input token. For the MLM task, these are decoded back into words, with the objective of restoring the unperturbed sentence, by correctly recovering the masked token. Usually, an additional [CLS] ("Classification") special token is prepended to the input: this token is intended to represent the input sequence in its entirety, by jointly being used on the output side to perform a whole-sentence classification task for example, such as predicting if a pair of (concatenated) sentences are following each other directly in a text or not.

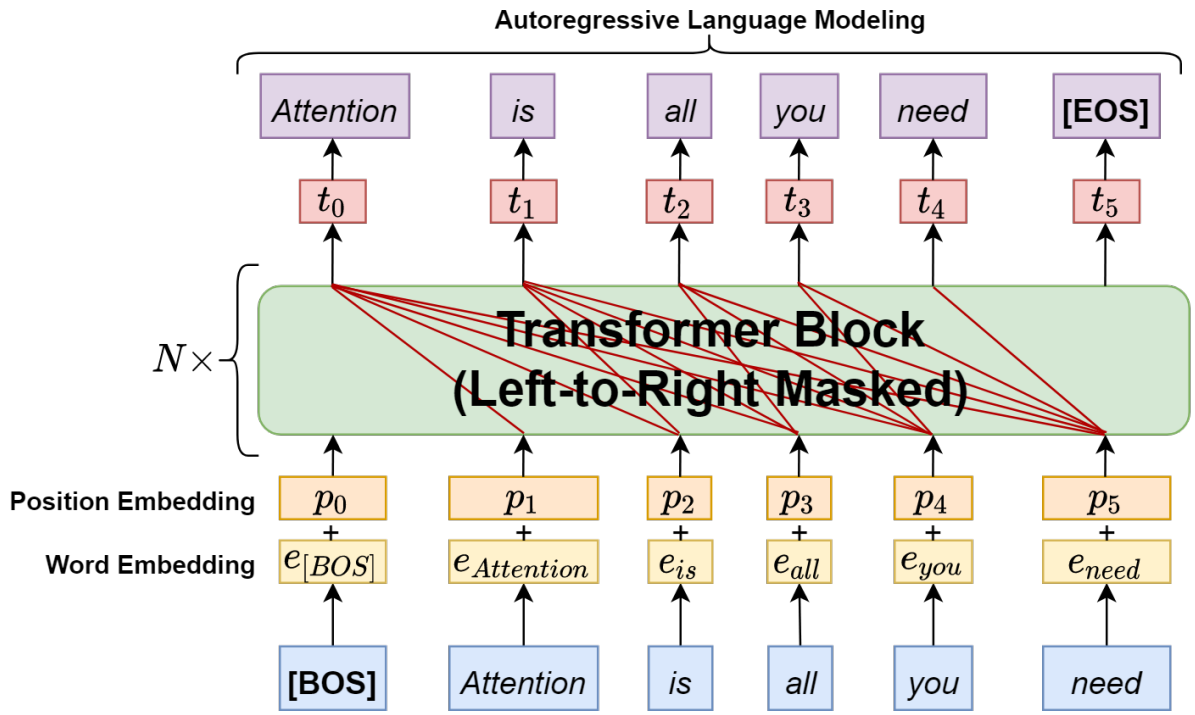


Figure 5.4: Schematic illustration of a typical autoregressive language modeling training setup for a Transformer model, as can be found in the GPT models [Radford et al., 2018, 2019; Brown et al., 2020]: this setup uses a modified Transformer block, in which the attention mechanism (see Figure 5.6) is blocked (indicated by the red lines) from attending to "future" (rightward) positions. Combined with shifting the input sequence to the right, by prepending a Beginning-Of-Sentence marker for example ([BOS]), this setup emulates an autoregressive decoder, in which the model conditions the prediction of the next token in the sentence on the previously predicted tokens only, until a terminating End-Of-Sentence ([EOS]) marker is produced. At inference time, such a model needs to be actually iterated according to some decoding algorithm, feeding the predicted sequence back in as the next input recursively.

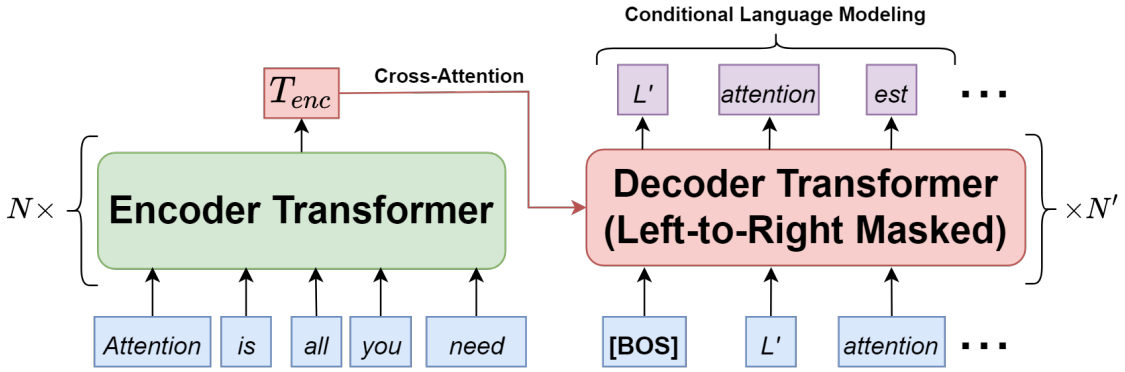
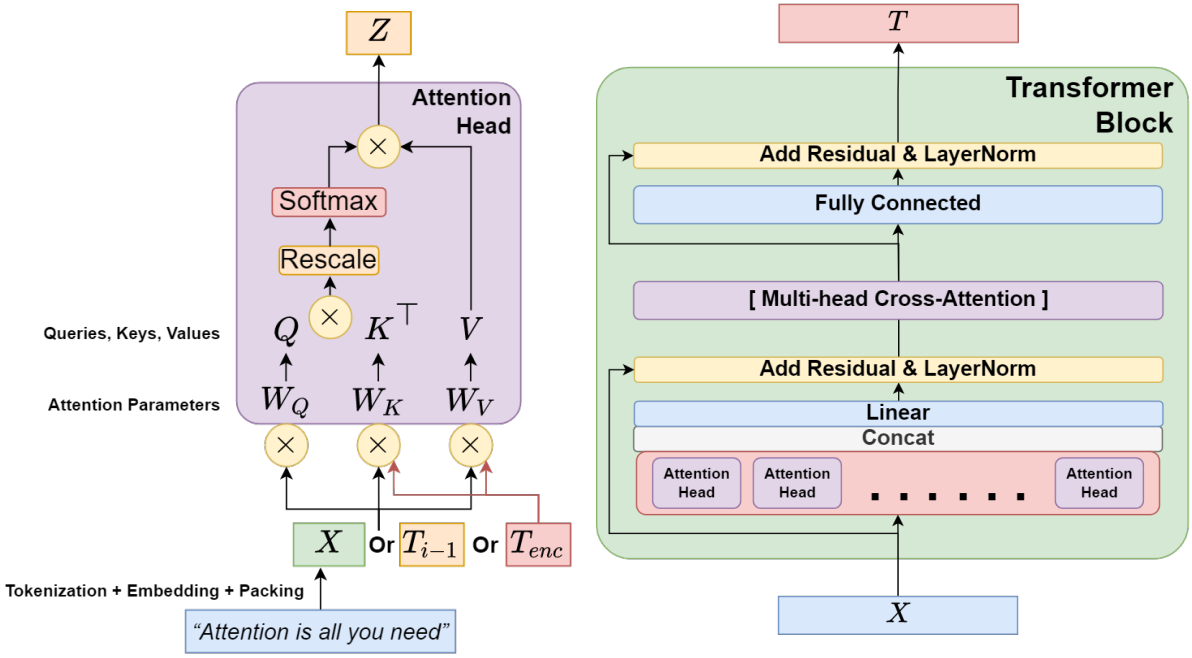


Figure 5.5: Schematic illustration of the canonical encoder-decoder Transformer architecture from Vaswani et al. [2017], and as can be found in BART [Lewis et al., 2020]: this setup emulates the most closely the typical RNN encoder-decoder architecture, as used in Neural Machine Translation tasks. It combines two Transformer block stacks, one acting as the encoder, whose role is to produce a matrix representation (T_{enc}) of the input sentence, which is then fed to the cross-attention modules (see Figure 5.6b) of a second decoder Transformer, which will then be able to condition its sequence decoding on this input sentence, in addition to the regular autoregressive process (see Figure 5.4), in this example, to perform an English to French translation.



(a) Schematic illustration of a single Transformer attention head module: an input sentence ("Attention is all you need") is first tokenized, embedded (see Figure 5.3 or 5.4) and finally packed into a single matrix X of token embedding vectors. Then, a query (Q), a key (K), and a value (V) matrix are computed by multiplying X with each corresponding learned parameter (W_Q,W_K,W_V). Q and K are used to compute attention scores which then weight the values V (which roughly correspond to the attended to hidden states in an RNN encoder-decoder setup) to obtain the intermediate output Z. Alternatively, for an attention head deeper in a Transformer block stack, the input will be the previous block's output T_{i-1}. In either case, these are referred to as "self-attention", as opposed to cross-attention module (see Figure 5.5), in which K and V will instead by computed from the output of an encoder Transformer stack, T_{enc}.

(b) Schematic illustration of a Transformer block module: multiple attention heads are used in parallel (each with their own parameters) to compute individual intermediate outputs. These are then concatenated, and fed through a linear layer, before going through a two-layer fully-connected layer with a non-linearity (ReLU). In between each submodule, residual connections [He et al., 2016] and layer normalization [Ba et al., 2016], which improve stability and overall help reduce the training time of the model. These blocks will usually be stacked on top of each other a given number of times to form a single model. Alternatively, in a decoder Transformer, a multi-head cross-attention module will be inserted in between the regular (or left-to-right masked) multi-head self-attention and the fully-connected module. It is identical to the preceding multi-head self-attention block (residual connection and layer normalization included), except it receives part of its input (see Figure 5.6a) from an encoder Transformer.

Figure 5.6

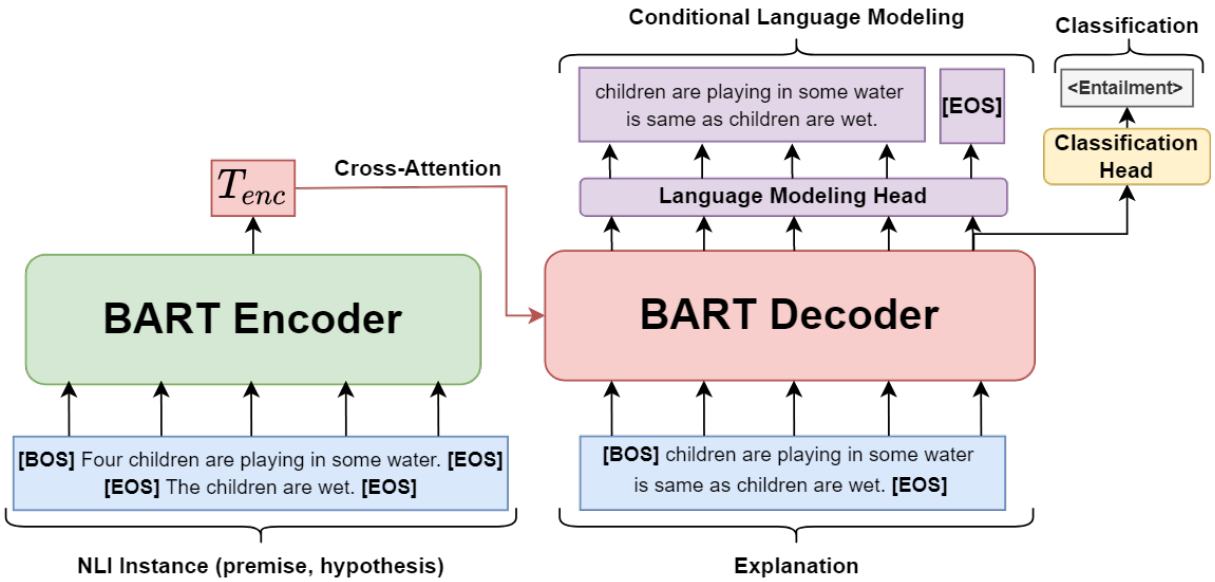
5.2 A Joint Classifier-Explainer using BART

For our experiments, we oriented ourselves towards the BART Transformer model [Lewis et al., 2020], because it possesses many relevant features and advantages for the purpose of jointly performing a classification task while also generating explanations. BART is a denoising sequence-to-sequence pre-trained auto-encoder, which essentially combines the capabilities of BERT [Devlin et al., 2019] and GPT [Radford et al., 2018], with a bidirectional "noisy" text encoder transformer, whose last hidden state is fed into the cross-attention layers (following the encoder-decoder setup initially proposed in Vaswani et al. [2017]) of an autoregressive left-to-right decoder transformer. This architecture can be pre-trained for any type of denoising task: indeed, while bidirectional models like BERT are designed and trained for token masking, where a certain number of words in an input text are replaced with [MASK] tokens which must then be filled back in contextually, and autoregressive models such as GPT for left-to-right language modeling, BART can be pre-trained with a variety of text corruption schemes, such as arbitrary span masking/deletion (where any span of text, including zero-length ones, can be masked by a single [MASK], or completely deleted), or arbitrary permutations of tokens or sentences. During pre-training, the noisy input is usually fed into the encoder, while the decoder is fed the original intact input in an autoregressive fashion. However, this need not be the case when fine-tuning for specific tasks: indeed, by using different "head" modules at the end of the decoder, and by feeding the input instances in the appropriate fashion to the encoder and decoder modules, this architecture has been used to perform various tasks, ranging in type from classification (using a classification head layer which is fed the last hidden state of the decoder for the final token of the output), to text generation (using a language modeling head layer), and even machine translation (by adding an intermediate small encoder network before the primary encoder, to allow the learning of a mapping to the target language's word embedding space), which was also expanded upon in a multilingual variant of BART, called mBART [Liu et al., 2020].

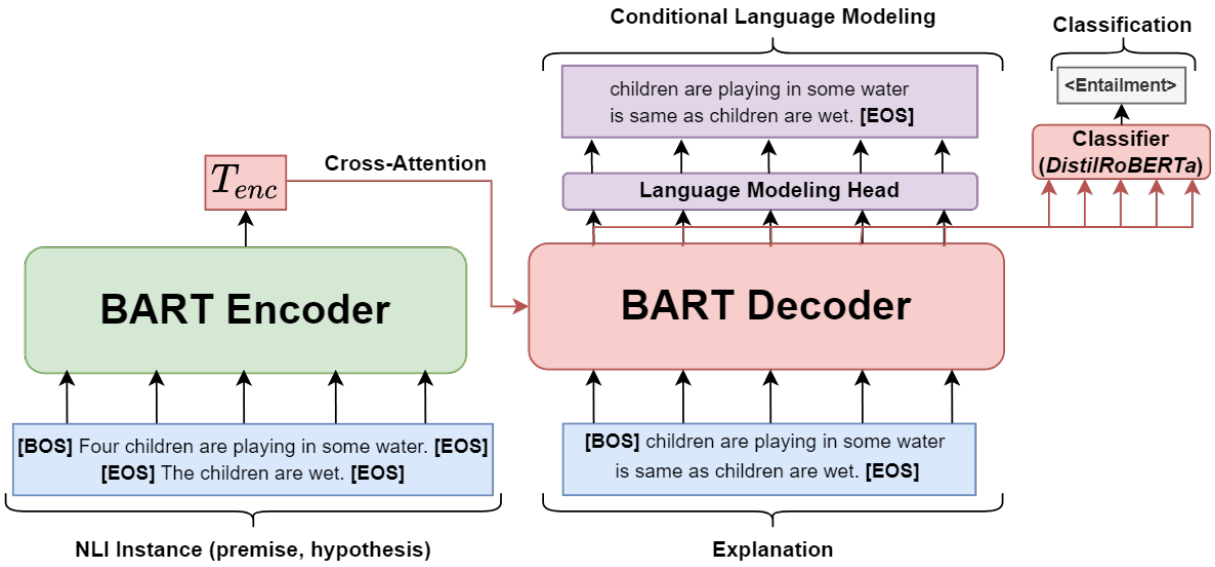
Taking inspiration from the conceptual frameworks showcased in the work discussed in Section 4.4, in particular the setups presented in Camburu et al. [2018] for the e-SNLI dataset, we propose fine-tuning a pre-trained BART architecture as a base to construct a joint natural language explanation generator and classifier model. We thus experimented with three different joint classification-explanation setup variants, using a common underlying BART architecture (see Figure 5.7):

- the BART encoder, which is fed the base SNLI (premise; hypothesis)-pair instances, tokenized, indexed, and concatenated as a single sequence, augmented with the appropriate beginning/end of sentence markers ([BOS]/[EOS]), in the following fashion: "[BOS] *<premise>* [EOS] [EOS] *<hypothesis>* [EOS]", following the scheme from

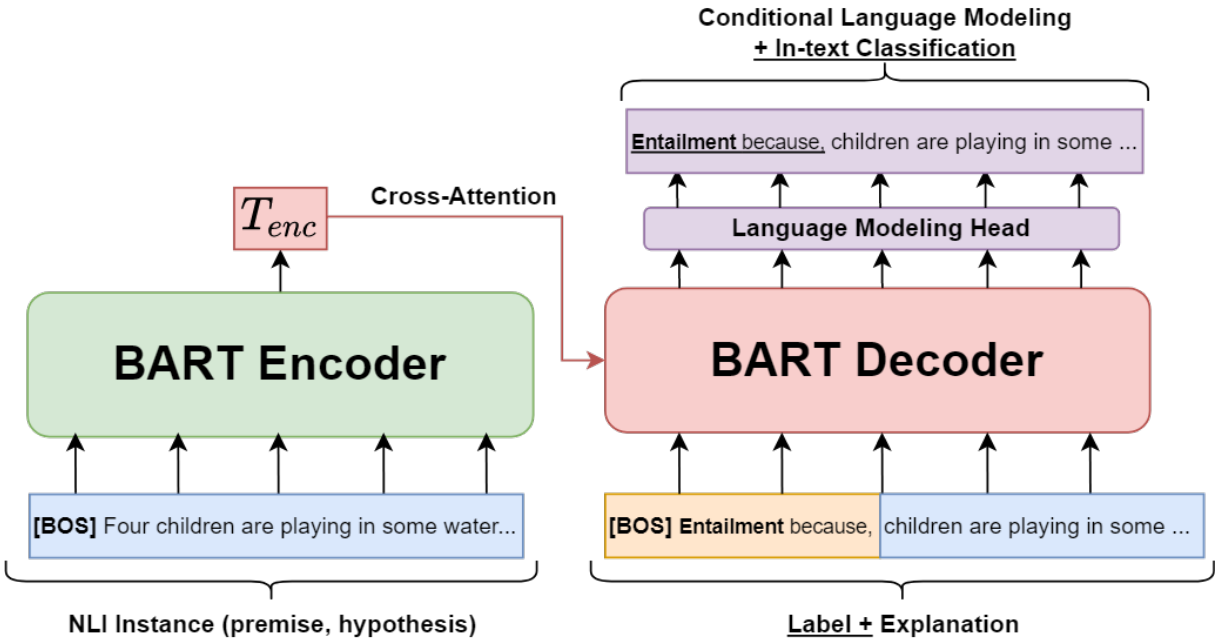
Figure 5.7: Schematic illustrations of our three BART-based classifier model variants. In all three cases, a BART encoder is tasked with producing an embedding matrix for the input instance (in this example, from E-SNLI), which is then used by the cross-attention modules in the BART decoder to condition the explanation generation and classification sub-tasks.



(a) The JointSmpl variant: in this Explainer model, the BART decoder is equipped with a language modeling head, tasked with learning to produce explanations autoregressively, conditioned on the input instance, and also a simple one-layer linear classification head, which takes as input the very last produced output embedding vector, corresponding to the End-of-Sentence ([EOS]) special token. This is similar to the way the [CLS] token is used in the BERT architecture (see Figure 5.3), except here the token used must be the very last one generated, so that it may have attended to the entirety of the sequence, due to the left-to-right masking present in the BART autoregressive decoder.



(b) The JointAux variant: this variant is similar to the JointSmpl one, except the classification task is performed by an auxiliary smaller Transformer model (DistilRoBERTa), which takes as input the entire decoder output embedding matrix.



(c) The ExplAsGen variant: in this final variant, no classification head module is present. Instead, the classification sub-task is merged with the explanation generation, by prepending the input explanations during fine-tuning with the pattern *"<Label> because,"*, where *<Label>* corresponds to the ground-truth label of the instance (in this example, the NLI inference class). The only objective function is the one associated to the conditional language modeling sub-task, however we still need to compute the classification accuracy metrics, so the same pattern is used to extract the generated label out of the decoded output, which is then matched to the task's possible labels by name.

the RoBERTa [Liu et al., 2019c] tokenizer-indexer as implemented in the Hugging Face transformers library [Wolf et al., 2020]. For CoS-E, we use the following scheme instead: "[BOS] *<question>* [EOS] [EOS] *<answer choice n°1>* ; *<answer choice n°2>* ; *<answer choice n°3>* [...] [EOS]". We use semicolons as separators as they never otherwise appear in question or answer choices and thus may be exploited as a task-specific separation marker.

- the BART decoder with its pre-trained language-modelling head module, which is by default fed the corresponding explanation for the current instance, as input and output (during training), autoregressively, inspired by the suggested text-summarization setup from Lewis et al. [2020].

Unlike some of these approaches however, we specifically then design each of our three variants such that there are no interruptions of losses' gradients flow in the entire architecture (for the first two two-headed models, the two loss functions are simply added together), such that the underlying core BART model is forced to learn both the classification and explanation aspects truly jointly:

- the JointSmpl variant performs both the inference-relation classification task and the

production of the accompanying explanation using the same common BART encoder-decoder, with a simple one-layer linear softmax classification head added, as suggested by Lewis et al. [2020], which is fed the decoder's final layer hidden state vector corresponding to the last End-of-Sentence ([EOS]) token output (which is roughly related to the similar approach used with the BERT architecture's [CLS] token, except BART's is located at the end and not the beginning, due to the decoder's autoregressive left-to-right nature). This setup is roughly inspired by the PredictAndExplain approach from Camburu et al. [2018], though unlike them, we do not condition the generated explanation on the predicted label by adding it to the initial decoder input. In fact, due to the autoregressive nature of the BART decoder, with the classification head module attending only to the last generated token's position in the hidden state, this variant, and the next one, could be more accurately placed into the *explain-then-predict* category. In the case of CoS-E (in both this variant and the next one), the classifier predicts a numeric label corresponding to the correct answer's position in the answer choices semicolon-separated list.

- the JointAux variant, employs an auxiliary transformer model to help perform the classification task, on top of the explanation-generation common architecture. We chose to use the pre-trained DistilRoBERTa model from the Hugging Face transformers library [Wolf et al., 2020] (inspired by the distillation method shown in Sanh et al. [2020]), which here is fed the last layer's hidden state produced by the common BART decoder, bypassing the transformer's input embedding layer. We chose this approach to in theory allow the BART decoder not to have to learn to potentially fully encode prediction related information in each generated token position: indeed, as the decoder does not directly know when the sequence of tokens will end (especially at inference time, since a step-by-step decoding algorithm is used to iteratively generate each encoding vector), it is in theory forced to either pack the relevant label-related information in every output embedding vector, or to more carefully learn to "expect" when the end of the generated sequence may occur (or partially both).

- the ExplAsGen variant reformulates the task as a purely generative one, where the input and output of the common decoder module combines both the label (as a single word corresponding to the inference relation for e-SNLI, or a group of words corresponding to the content of the predicted answer) and the explanation associated to each instance, using the following pattern: "*<Label/Answer> because, <Explanation>*". As this variant will thus generate the prediction first, autoregressively, it could be unlike the previous two considered a *predict-then-explain* approach.

5.3 Downstream Task Evaluation

For our experiments, we used the filtered training, development and test splits provided in the E-SNLI dataset[1] [Camburu et al., 2018], and the processed (filtered) training and development sets from the CommonsenseQA v1.0 version of CoS-E[2] (as the publically available CommonsenseQA test set does not provide ground-truth answers and was thus not annotated with explanations).

For the JointSmpl and ExplAsGen variants, we used the `facebook/bart-large` pre-trained model from the Hugging Face transformers library, which contains 12 encoder and 12 decoder layers, with a hidden unit size of 1024, for a total of approximately 406 million trainable parameters. For the JointAux variant, due to memory constraints caused by having to fit the `distilroberta-base` pre-trained classifier (6 layers, 768 hidden units, 82 million parameters), as well as to fit the differently sized hidden unit dimensions, we instead used the smaller `facebook/bart-base` pre-trained model (12 total layers, 768 hidden units, 139 million parameters). After initial experimentation, we found that freezing the pre-trained word embedding modules either slightly improved or at the very least did not harm the different models' performance, while producing a significant speed-up and reduction in memory usage. These experiments were performed on a 48-cores CPU, 250 GB of RAM local server, equipped with 3 GPUs (2 $\times$ Nvidia GTX 1080 Ti, 1 $\times$ Nvidia RTX 2080 Ti) with $\sim$ 12 GB of VRAM each. We used the AllenNLP [Wallace et al., 2019] framework[3] and PyTorch for all our experiments. All variants were trained with the Hugging Face AdamW [Loshchilov and Hutter, 2019] optimizer implementation, with a polynomial learning rate decay scheduler. For the decoding algorithm at inference time, we tried both beam-search (with 4 beams) as well as sampling with various hyperparameters, but found that sampling is much harder to get good subjective-quality outputs with (especially as there are more hyperparameters to tune), and thus chose to default to beam-search. As done by Lewis et al. [2020], we use label smoothing [Pereyra et al., 2017] with a parameter of $\epsilon = 0.1$ for the conditional language modeling objective during training, which smooths out the target 1-hot encoded tokens, replacing the value 1 by $1 - \epsilon$ and all the 0 by $\frac{\epsilon}{V-1}$, where V is the token vocabulary size.

We fine-tuned each of our variants on the E-SNLI train set for 10 epochs (or fewer, with early stopping), and 30 epochs (or fewer) on CoS-E. We evaluated our models using the simple classification accuracy (as is done in the respective original contributions) for the classification part of the task. To evaluate the explanation generation sub-task, we initially consider using perplexity, as is traditional in natural language generation tasks using language models. Perplexity is usually defined by the normalized inverse probability of a language model predicting the entirety of the test set. In practice, it is computed by taking the exponential

[1] available at: https://github.com/OanaMariaCamburu/e-SNLI
[2] available at: https://github.com/salesforce/cos-e
[3] available at: https://github.com/allenai/allennlp

of the average sequence cross-entropy loss on the test set. This metric can be interpreted as measuring how *surprised* a model is on average when having to predict the next word in a sentence of the test set, or in other words, it indicates the average weighted number of tokens the model considers likely to be the one appearing next in the test set, at each decoding step. A lower perplexity model is thus in theory "better" than a higher one, as it is more "sure" on average of the next word it should generate at each decoding step. While this metric is frequently used in language generation tasks, we had multiple issues with it. Firstly, due to the significant implementation differences between training time and inference time, caused in parts by the presence of the decoding algorithm (in our case, beam-search) as well as the label smoothing, our inference time loss calculations appear to behave inconsistently, reporting very high average losses at inference time (on either the dev or test set) for all three of our models: at training time we obtain losses in the range of approximately 2 to 2.5, and thus training perplexities in the range of ~ 7.3 to ~ 12.1, however, the calculated average losses on the test and dev sets are all close to approximately ~ 10.7, leading to perplexities of approximately ~ 44355.8, which is orders of magnitude higher than what would be expected of a model that ends up behaving subjectively comparably to those of Camburu et al. [2018] for instance. As we do not believe this measure we implemented to be accurate, further investigation would be required to determine where exactly the potential issues are located.

But more importantly, as a comparative metric, we find perplexity is not very informative of the quality of the produced explanations, as it is too dependent on the exact formulation of the human-annotated explanations. In summarization or machine translation, as well as other work similar related to language generation, the automated metrics BLEU [Papineni et al., 2002] and/or ROUGE [Lin, 2004] are often used in complement or as alternatives to perplexity.

Both of these metrics are based on comparing *n-grams* statistics between candidate generated texts and reference human-annotated texts. BLEU (**Bi**L**ingual** **E**valuation **U**nderstudy) is a metric function originally proposed by Papineni et al. [2002] to evaluate the quality of a machine-translated text by comparing them to one or more reference human translations, is computed using the following formula:

$$\text{BLEU} = BP \times \exp\left(\sum_{n=1}^{N} \frac{1}{N} \log p_n\right)$$

$$p_n = \frac{\sum_{c \in \text{candidates}} \sum_{n\text{-gram} \in c} \text{count}_{\text{clip}}(n\text{-gram})}{\sum_{c' \in \text{candidates}} \sum_{n\text{-gram}' \in c'} \text{count}(n\text{-gram}')}$$

$$\text{BP} = \begin{cases} 1 & \text{if CL} > \text{RL} \\ \exp(1 - \frac{\text{RL}}{\text{CL}}) & \text{if CL} \leqslant \text{RL} \end{cases}$$

where: N is the maximum size of *n-grams* considered (usually, if unspecified, $N = 4$);

p_n is the modified *n-gram* precision metric, which is computed by taking the sum across all candidate texts of the maximum counts of *n-grams* appearing in both the candidate text and any one reference text, clipped by the maximum number of times this *n-gram* occurs in any reference text, divided by the total number of *n-grams* in the entire candidates corpus;
BP is the Brevity Penalty, which is argued to serve as a better alternative to recall in this case, since this metric was intended to be used alongside multiple varied reference translations, and recalling more words from across multiple different translations would very likely diminish the translation quality while increasing the metric. It is computed using RL, the *best-match* References Length, which is the sum of the lengths of the best matching (closest) length-wise references for candidate (or shortest length on ties), and CL, the Candidates Length, which is just the total length of the candidates corpus. ROUGE (**R**ecall-**O**riented **U**nderstudy for **G**isting **E**valuation) on the other hand is a text summarization metric which was proposed later by Lin [2004], with similarities and differences to BLEU: both measures are intended to be usable with a number of different reference texts for each one candidate, however, where BLEU is a precision-based metric, ROUGE by default favors recall instead, arguing that it is more important in summarization to be exhaustive than to be accurate. However, in practice, both recall and precision are often computed as part of this metric, and may then be combined into a single F1-measure (the harmonic mean between precision and recall) as follows:

$$\text{ROUGE-}n\text{-R} = \frac{\sum_{r \in \text{references}} \sum_{n\text{-gram} \in r} \text{count}_{\text{overlapping}}(n\text{-gram})}{\sum_{r' \in \text{references}} \sum_{n\text{-gram}' \in r'} \text{count}(n\text{-gram}')}$$

$$\text{ROUGE-}n\text{-P} = \frac{\sum_{r \in \text{candidates}} \sum_{n\text{-gram} \in r} \text{count}_{\text{overlapping}}(n\text{-gram})}{\sum_{r' \in \text{candidates}} \sum_{n\text{-gram}' \in r'} \text{count}(n\text{-gram}')}$$

$$\text{ROUGE-}n\text{-F1} = \frac{2}{\text{ROUGE-}n\text{-R}^{-1} + \text{ROUGE-}n\text{-P}^{-1}}$$

where these scores are generally computed for $n = 1$ and $n = 2$. ROUGE also proposes additional measures, for example, ROUGE-L which is based on Longest Common Subsequence (LCS) of tokens between a candidate sequence c and any reference sequence r:

$$\text{ROUGE-L-R}(c, r) = \frac{LCS(c, r)}{|c|}$$

$$\text{ROUGE-L-P}(c, r) = \frac{LCS(c, r)}{|r|}$$

$$\text{ROUGE-L}(c, r) = \frac{2}{\text{ROUGE-L-R}(c, r)^{-1} + \text{ROUGE-L-P}(c, r)^{-1}}$$

where the global metrics are obtained by average over the entire candidates and references corpora.

These metrics are often debated not to be ideal [Callison-Burch et al., 2006; Graham, 2015;

	e-SNLI				
Model	Accuracy	BLEU	ROUGE-1-F1	ROUGE-2-F1	ROUGE-L
`JointSmpl`	83.03	17.8	42.87	22.15	38.51
`JointAux`	70.27	18.66	43.82	23.21	39.51
`ExplAsGen`	**91.03**	**28.05**	**58.67**	**38.46**	**54.9**
Camburu et al.	81.71	27.58	-	-	

Table 5.1: performance (ROUGE-1/2-F1: ROUGE overlap F1-score for 1/2-grams; ROUGE-L: ROUGE of the longest common subsequence) of our approach's variants on the e-SNLI test set, and those of the ExplainThenPredictAttention model from Camburu et al. [2018].

Wieting et al., 2019], both for Machine Translation, Automatic Summarization, and other types of language generation tasks, as they are somewhat shallow, ignoring aspects such as synonymy or differences in formulations. On the other hand, they are also very commonly used, relatively cheap to compute, and in theory suited to cross-architecture comparisons (unlike perplexity), but further research and experiments with different automated metrics better suited specifically to NLEs generation may be required: for example, METEOR [Banerjee and Lavie, 2005], proposes expending upon the principles of BLEU and ROUGE, allowing to count non-exact matches between tokens, by running different *alignment* (1-to-1 or 1-to-0 mappings between two sequences) modules sequentially, which may match tokens if they share the same lemma or stem, or if they are recorded to be synonyms, for example, in addition to exact matching.

Table 5.1 thus shows a comparison of our three setups with respect to (a) classification accuracy and (b) automatic measures for the evaluation of generated explanations (BLEU and ROUGE) on the e-SNLI dataset. We can see accuracy is good in all cases, with two setups (JointSmpl and ExplAsGen) being better than existing work on the same data. The second one is even 10% higher than the best performing model from Camburu et al. [2018], ExplainThenPredictAttention. Their "best" model is here considered with respect to all all metrics and criteria, including perplexity, BLEU, and subjective explanation quality. Their best performing model on accuracy alone is instead 2% better than ExplainThenPredictAttention, at 83.96%, however, it is significantly worse on all other aspects, with a BLEU of 22.4, and only approximately half as many (34.68% vs. 64.27%) of the first 100 produced explanations judged correct by human evaluators.

ExplAsGen also has a better BLEU and ROUGE scores than the other compared systems, by a very wide margin, though this may be very likely partially explained by addition of the fixed "*<Label> because, ...*" pattern to the explanations. In addition, as mentioned above, BLEU and ROUGE are somewhat non-ideal, especially in the case of generating explanations, as there can be a large overlap of tokens between compared sentences, but subtle and small differences can

change the nature of explanations (negation, word order, etc.). This is in addition particularly true for NLI, which tries to logically relate parts of two different sentences.

Table 5.2 shows a similar comparison of our variants on the CoS-E dataset, alongside the accuracy of the best performing model from Rajani et al. [2019], CAGE-reasoning. Compared to our setups, CAGE-reasoning was trained with automatically generated explanations, produced by a model itself fine-tuned only on the CoS-E human-annotated explanations, without the Question Answering objective. This may explain the lower performance of our variants, as we trained only on the "raw" human-annotated explanations. This may also further validate the results and observations from the authors, the CAGE framework allowing for a sort of "denoising" of the admittedly (by the authors, and our own qualitative judgement) very noisy human explanations. Again, we obtain the best automated text generation metrics with the ExplAsGen variant, which also seems to confirm that the static pattern inserted in the decoder inputs as part of the reformulation of the task as a purely generative one, may at least partially artificially inflate those metrics. performance on the main task still remain quite low, compared to best models Rajani et al. [2019], though ExplAsGen does slightly beat their non-CAGE experiment, where they simply feed the same BERT classifier directly with the noisy CoS-E explanations, similarly to our variants, obtaining an accuracy of 65.5%, compared to our 67.79%. For some reason, the JointAux is unable to learn to solve the main task, despite seemingly being able to produce explanations somewhat successfully. After a subjective qualitative analysis of the task, we chose to focus the rest of our work on the e-SNLI dataset only. Indeed, we thought this task might be perhaps too complicated for our approach: as advertised in its abstract, the CommonsenseQA [Talmor et al., 2019] dataset is overall very challenging, as it requires a vast amount of world-knowledge, due to the conciseness and lack of contextual information present in most of the questions. We found that a lot of instances seem to require employing a strategy of "eliminating" the two most incorrect, or even incoherent answers, rather than selecting the best or most appropriate one, which is often non-trivial even for humans (which may explain the current human accuracy of "only" 88.9%, on the version 1.11 of the dataset). Moreover, we found through qualitative analysis that both the underlying CommonsenseQA questions and sets of answers, as well as the human-annotated explanations added in CoS-E are overall quite noisy: for example, in Table 5.3 we showcase a few such problematic instances, from the first 250 instances of the dev set (out of 950). In many instances, whether in the question, answers, or explanation, we can find typos, ungrammatical structures or ordering of words, ambiguous or even mislabeled answers, and in general just poor text fluency, which may negatively affect models which are not as performant at dealing with such types of noise in addition to the capabilities required to deal with the specified Question Answering task and the generation of accompanying explanations.

While downstream performance on the main classification (or question answering) tasks

	CoS-E				
Model	Accuracy	BLEU	ROUGE-1-F1	ROUGE-2-F1	ROUGE-L
`JointSmpl`	61.9	5.92	18.26	7.51	16.29
`JointAux`	33.16	8.13	20.54	9.82	18.52
`ExplAsGen`	67.79	**17.98**	**40.75**	**25.92**	**38.19**
Rajani et al.	**72.6**	4.1	-	-	

Table 5.2: performance (ROUGE-1/2-F1: ROUGE overlap F1-score for 1/2-grams; ROUGE-L: ROUGE of the longest common subsequence) of our approach's variants on the CoS-E development set (the test set ground-truth answers are not publically available), and those of the CAGE-reasoning model from Rajani et al. [2019].

are relatively easy to analyze, as we feared, the automated text generation metrics were not sufficient to judge the quality of the produced explanations. Therefore, we chose to also manually qualitatively evaluate some of the produced explanations, for the variants trained on e-SNLI, as was done in other similar work.

Question	Answers (correct)	Human Explanation	Type of issues
What island country is ferret popular?	great britain, hutch, own home	great britain having the largest area	*Q:* Fluency, *E:* Relevance
The teens were trying to hide that they get drink, but when they walked in the door their what gave it away?	stagger, vomit, fall down	walk or move unsteadily, as if about to fall.	*Q:* Fluency, *E:* Utility
She feared that she had cancer, but upon discovering truth that she hadn't, what was her attitude toward life?	conclusion, happiness, relief	happiness explains the feeling that she has at the moment	*A:* Ambiguity, *E:* Utility
Bob's only light source was a small bulb. There were four walls, if there was a door he couldn't see it. What was Bob in?	closed room, dard, sky	closed room in the bub	*Q:* Coherence, *A:* Noise, *E:* Noise
What must happen for an animal to and it's offspring to continue livng?	reproducing, eventually die, food consumed	food consumed by anmimal	*Q:* Fluency, *Q:* Noise, *A:* Mislabeled, *E:* Noise
If somebody buys something and gives it to me as a free gift, what is the cost status of the gift?	imprisoned, paid for, expensive	most presents to friends and family will fall below the annual threshold for taxable gifts. in 2016 and 2017, a taxpayer could give up to $14,000 per person per year without being taxed on the gift	*Q:* Fluency, *E:* Utility

Table 5.3: A few examples of noisy or problematic instances from the CoS-E dataset. In the fourth column, we list the location of each problem (Q = Question, A = Answers, E = Explanation) followed by its nature, and then highlight the issue in the instance with the associated color.

5.4 Explanation Quality Evaluation

Since an overall natural language explanation quality is hard to formalize and describe [Miller, 2017], we decided to implement a multi-criterion evaluation scheme which seemed easier to judge than a single encompassing score, and could lead to a better separation of human perceptions and subjectivity of produced explanations. Evaluators were asked to assess the three following criteria:

1. the **fluency** of the produced explanation, that is, how syntactically, grammatically, and semantically (in a limited sense) close to natural human language it is judged to be, without necessarily taking into account the actual "explanatory content". Informally, the criterion was put as: Ignoring the rest of the instance, does this output "sound" or "read" like natural English?
   For example, a low fluency NLI instance might look like this (with the relevant issues highlighted in **bold**):

   PREMISE: "*Two dogs are playing catch in a field.*"
   HYPOTHESIS: "*Two dogs are outside.*"
   LABEL & EXPLANATION: "*Entailment, because **feld outside**.*"

   Here, fluency is poor because a spelling error is present, and a word is missing from the explanation. A higher fluency version might instead look like:

   PREMISE: "*Two dogs are playing catch in a field.*"
   HYPOTHESIS: "*Two dogs are outside.*"
   LABEL & EXPLANATION: "*Entailment, because a field is outside.*"

2. **relevance and coverage**, which encompass the quantity and quality of references in the produced explanation, to the relevant elements of the two input sentences in the E-SNLI instance. Informally, the criterion was put as: Were each relevant elements of the premise and hypothesis mentioned adequately in the explanation?
   For example, a low relevance and coverage NLI instance might look like this (with the relevant issues highlighted in **bold**):

   PREMISE: "*A Land Rover is being driven across a river.*"
   HYPOTHESIS: "*A vehicle is driving through water.*"
   LABEL & EXPLANATION: "*Entailment, because a Land Rover is a vehicle and **the sun is bright**.*"

   Here, the relevance is low because of the last second part of the explanation which does not refer to an element that is present in either the premise or hypothesis, and the coverage is low because a relevant element is missing from the explanation. A higher relevance & coverage version might instead look like:

PREMISE: "*A Land Rover is being driven across a river.*"
HYPOTHESIS: "*A vehicle is driving through water.*"
LABEL & EXPLANATION: "*Entailment, because a Land Rover is a vehicle and a river is flowing water.*"

3. the overall perceived **utility** of the produced explanation. Informally, this was put as: Subjectively, is the explanation useful in understanding why this specific inference label was selected by the model? For example, a low utility NLI instance might look like this:

   PREMISE: "*A man playing an electric guitar on stage.*"
   HYPOTHESIS: "*A man playing banjo on the floor.*"
   LABEL & EXPLANATION: "*Contradiction, because a man playing an electric guitar on stage cannot be playing banjo on the floor.*"

   Here, the utility is low because the explanation is both overly verbose and not specific enough, copying large portions of the input verbatim. A higher utility version might instead look like:

   PREMISE: "*A man playing an electric guitar on stage.*"
   HYPOTHESIS: "*A man playing banjo on the floor.*"
   LABEL & EXPLANATION: "*Contradiction, because a banjo is not an electric guitar and the floor is not a stage.*"

In addition to using three criteria instead of a single subjective rating, contrary to Camburu et al. [2018], and inspired by the manual evaluation protocol on fact-checking explanations found in Atanasova et al. [2020], we also did not associate a score to each criterion. Instead, we asked them to perform pairwise comparisons for a set of anonymized system outputs, generated for a number of instances of the e-SNLI test set (10 instances for a three-way comparison in Table 5.4, and 30 instances for a two-way comparison in Table 5.5, for a total of 30 comparisons per annotator for each criterion), where for each of the three criteria, they had to vote for the output they considered the best in that criterion, with an option for ties (*"Indecision"* in the Tables). This evaluation was performed on the test set, similar to the human evaluation in Camburu et al. [2018].

We first performed this evaluation on the outputs of our three variants, whose results are shown in Table 5.4. In addition to the total votes for each model and criterion (as well as the ties, marked as *"Indecision"*, we also computed the inter-evaluator agreement using Fleiss' Kappa [Fleiss et al., 2003]. We can observe a moderate level of agreement, which, from post-hoc analysis, we found we could mainly attribute to a discrepancy between one judge and the other two, as well as a lot of variations between votes and ties for specific pairs of systems and criteria. For example, fluency was very often judged too similar in each pair of outputs to take a decision, for almost half of the comparisons. We find striking that the best system regarding

Model	Fluency	Relevance & Coverage	Utility
`JointSmpl`	12	22	**26**
`JointAux`	14	**26**	16
`ExplAsGen`	**24**	13	21
Indecision	40	29	27
Fleiss' Kappa	0.44	0.59	0.49

Table 5.4: Total pair-wise comparison votes and inter-evaluator agreement for our three model variants, for each of our three chosen criteria.

Model	Fluency	R&C	Utility
`JointSmpl`	**26**	**43**	**49**
Camburu et al.	23	10	12
Indecision	41	37	29
Fleiss' Kappa	0.17	0.15	0.47

Table 5.5: Total votes and inter-evaluator agreement for our `JointSmpl` model against the `ExplainThenPredictAttention` model from Camburu et al. [2018], for each of our three chosen criterion.

accuracy and BLEU/ROUGE scores on the automated task evaluation (ExplAsGen) seems to be weakest by far on the relevance & coverage criterion, and much lower than JointSmpl on utility, while seemingly being the most fluent. This contributes to show how automated metrics like BLEU or ROUGE can be misleading when judging the quality of explanations. While the other two criteria still have quite a high number of indecisions, fluency appears to be the most subjective criteria overall, as can be seen from the lowest agreement measure as well, though from a more qualitative analysis, we believe this to be because all three models are surprisingly fluent, considering the amount of noise in the training data.

Based on this first evaluation, we thus decided that JointSmpl was a good compromise model, as it has the most overall utility votes, decent relevance & coverage, and a good classification accuracy on the main prediction task. The fact it is the least fluent could also be seen as beneficial, since it is reasonable to expect that low fluency would normally negatively impact the evaluators' votes in its utility, yet it is still the best rated of the three in that regard. We thus performed a second similar human evaluation, comparing the outputs of this model to those of the ExplainThenPredictAttention model from Camburu et al. [2018], against which we previously evaluated our models, on the e-SNLI task. We followed the same protocol, but used different instances from the previous human evaluation, waiting a few weeks between the two, such that evaluators were less able to remember and re-indentify the previously selected model. Results from this second evaluation are shown in Table 5.5. Compared to the previous

results, we can see that inter-evaluator agreement is much lower for fluency and relevance & coverage this time, to the point where we can question whether evaluators could differentiate the two systems, at least on these two criteria, whereas for the utility criterion, the results are much closer to those of the previous evaluation, showing a clear preference in favor of our model compared to the previous work from Camburu et al. [2018]. As a sanity check, we performed a sign test for this evaluation, combining the votes from all three evaluators, for which we obtain the following p-values: 0.7754, 5.55×10^{-6}, 1.97×10^{-6} for fluency, relevance & coverage, and utility, respectively. These seem to confirm that fluency was again the most subjective, and probably the least pertinent criterion overall to discriminate between both our three variants, and JointSmpl against ExplainThenPredictAttention, as most generated explanation seem to be written in well-structured and grammatically correct, if perhaps often a little verbose English (which is not necessarily a drawback).

After these two evaluations, we also did a short qualitative analysis of some of the outputs from all four systems. We note that, all things considered, the quality of generated explanations is surprisingly good, especially when considering the noisiness of the training human-annotated explanations. Indeed, if we look at a few example instances with their respective outputs in Table 5.6, we can see that the different model-generated explanations differ significantly from their human-annotated counterparts, at least in their form. Interestingly, we note that quite often, when a model misclassifies an instance, the explanation it provides can be viewed as an acceptable justification for its label prediction, which can lead one to question whether the underlying NLI task, or at the very least its particular incarnation in the SNLI dataset, is perhaps not a little too subjective with regard to the closed entailment classes which are to be predicted. For example, in sub-Table 5.6a, the JointAux model mispredicts the *neutral* label when the ground-truth is supposedly *entailment*. Looking at the generated explanation, while this is not made explicit by the model itself, one could consider the justification and thus the prediction also acceptable for this particular instance: indeed, while the most likely human extrapolation of the premise is that the mentioned children are playing in some kind of pool and would therefore very likely be exposed directly to water, this is actually not specified in the premise text itself, and one may therefore equally *not* make this assumption. The children could be playing in a shallow puddle while fully clothed in rain-gear, for example, which one may then not necessarily consider as equivalent to being directly wet. A lot of this we believe has to do with how the SNLI dataset was constructed [Bowman et al., 2015]: indeed, the premises in this corpus are originally image captions, and, while the crowd-workers who then produced different hypotheses for each inference label did not have access to these images, simply having the knowledge that they were dealing with descriptions of photographs (which was explicitly specified in the data collection instructions presented to the annotators) will invariably bias humans when mentally "filling the gaps" left out of the caption. To test whether our models may perhaps be able to correct their prediction if presented with additional more

explicit situational information, we tried modifying some instances which some of our models had technically misclassified but still produced an acceptable label and explanation for. In Table 5.7 we show one such example pertaining to the previously mentioned instance from Table 5.6a: as we can observe, specifying that the children are unambiguously playing in a pool while wearing swimsuits, which should indeed entail that they would be directly wet, does not lead the JointAux model to revise its decision, and it persists with roughly the same explanation form and content as with the official instance. While trying other variations, we also find cases where this particular model gives an adequate explanation for the *entailment* label, and yet still predicts *neutral*. This may be a sign that the auxiliary classification module is occasionally misaligned with the explanation generation module, but this is definitely not something that is specific to this model, though we were not able to ascertain if it was a more prevalent phenomenon with this particular variant or not. More work will be required to properly study these aspects, which will most likely require some degree of automation, as manually altering instances and subjectively evaluating the new outputs is not really feasible. Camburu et al. [2020] for example, followed-up on their work on e-SNLI by developing a form of adversarial attacking scheme, to reveal inconsistencies in automatically generated explanations.

Following this, we also took a look at definitely misclassified and/or wrongly explained instances, such as the example shown in Table 5.8, for which we noticed various types of "failure modes": firstly, and as mentioned above, we found quite a number of instances where the predicted label and explanation were not in accord with each other, whether the predicted label was correct or not. These unfortunately were to be expected, as no machine learning architecture currently is able to perform either classification or text generation perfectly on these types of datasets, and there are no reasons to expect this to be different for an architecture which does both of these at once. However, it is even more problematic with this type of explanation-*explanandum* mismatch, as it is fundamentally different from traditional misclassification/misprediction errors: indeed, outside of a model's underperformance, the latter can also often in more or less great parts be explained by *label noise* introduced mainly by human annotators, and for which various techniques have been and are being worked on (see Frenay and Verleysen [2014]; Frénay and Kaban [2014] for general overviews) to deal with this type of issue, mainly through the automated identification of mislabelled instances, using some type of confidence metric [Hovy et al., 2013; Chang et al., 2017; Swayamdipta et al., 2020], followed by generally either reweighing or straight-up removal of those instances from the affected datasets. Unfortunately, with explanation-*explanandum* mismatches, *label noise* cannot be pointed at as a direct culprit: ideally, even if such a model were to predict the incorrect label for a particular instance, it would still be strongly desirable for the produced explanation to help make sense of the wrong decision which was taken. Similarly, if the correct label is predicted, but the explanation for it does not make sense, it reduces the trust one may have in the model's ability to actually explain its reasoning. A second type of "failure mode" which we found to be

qualitatively more pervasive and less trivial to notice at a glance however could be labelled as "miscomprehension errors": if we look at Table 5.8 for example, we can see examples of what we consider to be varying degrees of low-quality produced explanations, which seem to betray that the different models did not manage to properly parse and refer to elements of the input sentences. On this particular instance, both JointAux and ExplAsGen did predict the correct label, however, and in particular for JointAux, the produced explanations feel unsatisfactory as they heavily paraphrase the input premise, without actually focusing specifically on the elements which prove the entailment relation (in this case, that a large American flag is an American flag). On the other hand, JointSmpl and ExplainThenPredictAttention [Camburu et al., 2018] misclassify the instance, and seem to do, according to their produced explanations, because they did not manage to fully (for the former) or accurately (for the latter) parse the entire premise sentence. In both cases, we believe a probable cause of this type of issue is due to the length and number of elements contained to potentially attend to in the input, which may be acting unintentionally as distractors for the different models. In the former two cases in particular, this may also be in parts explained due to the numerous human-annotated explanations in the dataset which follow simplistic templates, as remarked by Camburu et al. [2018], where the premises and hypotheses sentences were included in as is, often verbatim. While the original dataset was filtered for and re-annotated for the latter, we note that a large number of human annotations still seem to follow such templates, only with partial inclusions of the two input sentences in the explanations, or heavy paraphrasing, which may still lead models trained on this dataset to learn this type of behavior.

Table 5.6: Three examples of instances from E-SNLI and their corresponding combined (following the "*<Label> because, <Explanation>*" pattern) outputs from our three model variants, and also from the EXPLAINTHENPREDICTATTENTION [Camburu et al., 2018] model, and the three human-annotated gold explanations for these instances. As can be observed, models often use similar explanation "templates", mostly dependent on the type of inference-relation, with variations in overall relevance and usefulness of explanations and/or accuracy of predicted labels.

(a) Example of outputs for an *Entailment* instance.

Premise	Four children are playing in some water.
Hypothesis	The children are wet.
Gold label	Entailment
JOINTSMPL	Entailment, because children are playing in some water is same as children are wet.
JOINTAUX	Neutral, because playing in some water does not imply being wet.
EXPLASGEN	Entailment, because the children are playing in water so they must be wet.
Camburu et al.	Entailment, because children playing in water are wet.
Gold Explanation 1	Entailment, because playing in water means you are wet.
Gold Explanation 2	Entailment, because the children became wet as they are playing in water.
Gold Explanation 3	Entailment, because four children are children, and playing in water implies wet.

(b) Example of outputs for a *Contradiction* instance.

Premise	Two dogs are playing catch in a field.
Hypothesis	The two dogs are resting in the field.
Gold label	Contradiction
JOINTSMPL	Contradiction, because playing and resting are not the same.
JOINTAUX	Contradiction, because the dogs cannot be playing catch and resting at the same time.
EXPLASGEN	Contradiction, because the dogs cannot be playing catch and resting simultaneously.
Camburu et al.	Contradiction, because the dogs are either playing catch or resting.
Gold Explanation 1	Contradiction, because if the two dogs are playing, they can not be resting.
Gold Explanation 2	Contradiction, because one cannot be playing with resting simultaneously.
Gold Explanation 3	Contradiction, because " playing " and " resting " contradict one another in that playing shows movement while resting shows non-movement.

(c) Example of outputs for a *Neutral* instance.

Premise	Four young girls playing in the water.
Hypothesis	Four girls are swimming.
Gold label	Neutral
JointSmpl	Neutral, because girls playing in the water are not necessarily swimming.
JointAux	Neutral, because playing in the water does not imply swimming.
ExplAsGen	Entailment, because girls playing in the water are swimming.
Camburu et al.	Neutral, because playing in the water does not imply swimming.
Gold Explanation 1	Neutral, because playing in the water does not mean they are necessarily swimming.
Gold Explanation 2	Neutral, because not all playing ones in water are swimming.
Gold Explanation 3	Neutral, because the girls can be playing in the water without swimming in it.

Table 5.7: Example of a modified e-SNLI instance and the outputs of our models on it. The modified part from the original (see Table 5.6a) are highlighted is **bold**. In spite of the additional information, the JointAux model persists in predicting the *neutral* label, with an explanation that is this time less believable due to the additional situational information given in the modified premise. The other two models however were not perturbed by this particular modification.

Premise	Four children are playing in **a pool in swimsuits.**
Hypothesis	The children are wet.
Gold label	Entailment
JointSmpl	Entailment, because the children are playing in a pool so they must be wet.
JointAux	Neutral, because just because children are playing in a pool in swimsuits does not mean they are wet.
ExplAsGen	Entailment, because children playing in a pool are wet.

Table 5.8: Example of a difficult instance for the various models. Both JointAux and ExplAsGen correctly classify the relation, but provide not very useful explanations (heavily paraphrasing the inputs), whereas JointSmpl and ExplainThenPredictAttention from Camburu et al. [2018] respectively skip over or completely misunderstand an important aspect of the input, and thus have also bad relevance & coverage.

Premise	A man in a hard hat, gray t-shirt and holding a cordless drill saluting in front of a large American flag.
Hypothesis	A man saluting the American Flag.
Gold label	Entailment
JointSmpl	Neutral, because a man in a hard hat, gray t-shirt is a man.
JointAux	Entailment, because man in a hard hat, gray t-shirt and holding a cordless drill saluting in front of a large american flag simply means man saluting the american flag.
ExplAsGen	Entailment, because saluting in front of a large american flag is the same as saluting the american flag.
Camburu et al.	Contradiction, because a man is either holding a power drill or a flag.
Gold Explanation 1	Entailment, because a man saluting in front of a large american flag is saluting the american flag.
Gold Explanation 2	Entailment, because a man is saluting an american flag in both sentences.
Gold Explanation 3	Entailment, because he is saluting the flag because he is saluting in front of a large american flag.

5.5 Conclusion and Perspectives

In this second part, we demonstrated how a Transformer architecture could be fine-tuned to jointly generate predictions, and explanations for those predictions. If this type of approach can be successfully generalized, this would enable such models to communicate a variety of useful information alongside their usual predictions, which would otherwise have to be manually extracted through the use of an appropriate explainability method, which often requires expert-knowledge to be correctly implemented and used. This part was the basis for a paper submitted to the *ACL Rolling Review* initiative in November 2021, and is currently under review.

Overall, as was somewhat initially expected, while the results are relatively impressive, especially considering how the base BART model we used in these experiments was not initially designed or pre-trained for the production of natural language explanations, the main disadvantage of this type of approach is that we rely on a black-box models to hopefully provide us with insights as to their own decision-making abilities. This implies that this process itself may suffer from the same issues associated with black-box models when performing other tasks, that is, lack of intrinsic interpretability. In particular, the **faithfulness** of automatically produced explanations should be put into question, as NLE models should have no reasons to be any more robust to adversarial attacks than any other similar machine learning models, such as shown by Camburu et al. [2020] who designed a relatively simple attack scheme on the NLI hypothesis, which, without changing its overall meaning, causes not just an incorrect flipping of the predicted label, but also causes the Explainer model to produce an incoherent explanation to justify its error. Even if an Explainer model were to be designed and trained to be more robust to these types of attacks, it would not be a proof that its output explanations are faithful: on the contrary, a more "powerful" and robust-seeming model may deceptively learn to appear to be faithful, but may in actuality just produce post-hoc realistic-looking rationalizations in order to fool its evaluators. On the other hand, we argue that this is not exclusive to these models: indeed, even more formal and external explainability methods, such as LIME [Ribeiro et al., 2016] or SHAP [Lundberg and Lee, 2017], may be attacked in such a way as to allow a biased classifier to "hide" its biased nature behind innocuous-seeming explanations produced by these methods [Slack et al., 2020], especially when performed on a "remote" model one does not control [Le Merrer and Trédan, 2020]. Additionally, these more formal methods usually require domain-experts to setup and properly analyze the results, which further complicates the generation of end-user comprehensible access to algorithmic decisions explanations. With the moving trend towards larger and larger architectures, both in and out of NLP domains, as can be seen with the growing popularity models like GPT-3 [Brown et al., 2020], intrinsic interpretability methods, such as Interpretable Word Embeddings, do not seem to be a promising avenue to get useful explanations: indeed, these methods, which already

had complicated applicability in non-intrinsically interpretable models like RNNs, have been made even less effective with deeper Transformer-based architectures, and as the trend towards larger, less easily decomposable models continues, this will most likely not improve. While not necessarily a dead-end, as the general ideas such as sparsity of the learned representations may possibly be adapted to function in more recent architectures, most of the benefits of normally using these intrinsically interpretable representations will be difficult to maintain after so many re-projections through a number of Transformer blocks, without severely modifying the underlying architectures. Similarly, more formal post-hoc explainability methods may have troubles dealing with the inherent complexity associated to deeply non-linear models with growing number of layers, from dozens to hundreds, and growing number of trainable parameters, from millions to billions (and possibly more). As such, it may be that end-to-end black-box production of Natural Language Explanations, especially exploiting the increasing abilities of large architectures to produce higher and higher quality texts automatically, could end up being the only feasible avenue to reach satisfactory explainability in Natural Language Processing, though care should be taken not to compromise so much on explanation desiderata, like faithfulness, that the whole enterprise loses its initial meaning and objectives.

Numerous perspectives are open in this area of research on Natural Language Explanations: first and foremost, better, ideally automatic evaluation tools should be sought, as currently used automated metrics from Machine Translation and Automatic Summarization are not well adapted to this task. Additionally, testing and improving robustness and faithfulness of such Explainer models is paramount if these are ever to be used to inform real end-users: the attack scheme presented in Camburu et al. [2020] and the overall design of the approach in Kumar and Talukdar [2020] (allowing to abductively diagnose how a model *may* have behaved, had some features of an instance changed) may be good starting points as both diagnostic tools as well as potential ways to improve the learning setups for Explainer models. More generally, a study of how well this kind of Transformer-based joint classification-explanation generalizes to Out-of-Distribution or out-of-Domain data is necessary, as high-quality large-scale explanations-annotated datasets are difficult to construct: one possibility could be to try to use multitask learning, possibly combining both various tasks and datasets, but also maybe various *distant supervision* signals, for example, only partially and/or noisy human-annotated data, or even automatically annotated data, taking inspiration from the CAGE framework proposed by Rajani et al. [2019]. To relieve human-annotators from having to produce vast quantities of carefully explained instances, another avenue of research may be into Deep Active Learning techniques (see Ren et al. [2021] for a general review), which aim to empower deep-learning models to actively solicit help on instances which they find difficult, in theory reducing the amount of annotated instances required, as well as adding an interactive component to the training process, which may allow correcting issues as they occur during the learning process instead of after the fact.

General Conclusion

Properly explaining and understanding algorithmic decisions is a challenging task: whether one uses intrinsically interpretable models/components, or post-hoc explainability methods, a variety of issues stand in the way of the various desiderata one may focus on for produced explanations. In particular, models in Natural Language Processing pose their own unique challenges in regard to these aspects: in contrast to more "physical" systems, such as vision or audio, or tabular data, which usually is easy to parse by construction, natural language as a signal is highly abstract and tightly linked to the functioning of the human brain, of which we so far do not have a good grasp of. While it is possible to attempt to inject some intrinsic interpretability into commonly used linguistic representations, such as through the use of Interpretable Word Embeddings, these processes are definitely not perfect, and may not be adapted to the evolutions we are currently seeing in the field, towards larger and more complex pre-trained models. On the other hand, these very evolutions towards ever more opaque models mean there is an increasingly urgent need for explainability, if algorithms are to take part more and more in important decisions. While a number of post-hoc explainability methods currently exist, they are not perfect either, and care must be taken not to put too much faith into any single one method, as each have their quirks and limitations, which often require expert-knowledge to properly take into account. We proposed in a second part here to look at the very extreme of post-hoc explainability methods, in the form of having NLP models themselves produce Natural Language Explanations for their own decisions. While the overall approach poses big challenges, in particular to ensure that the produced explanations are faithful and not deceptive with regard to the models' decision processes, it is also an opportunity to exploit the growing capabilities of modern Transformer-based architectures to generate natural language. More specifically, this may be done in end-to-end setups, where, in addition to the task to be explained, the main costs are only obtaining human-annotated natural explanations, potentially by non-experts, which, while not necessarily easy, may be a good compromise to otherwise needing experts to formalize a specific set of desiderata for explanations, followed by methods to extract them from model decisions.

Bibliography

Rob Abbott, Brian Ecker, Pranav Anand, and Marilyn Walker. Internet Argument Corpus 2.0: An SQL schema for Dialogic Social Media and the Corpora to go with it. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 4445–4452, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL https://aclanthology.org/L16-1704.

Cited on page 40.

Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity Checks for Saliency Maps. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/hash/294a8ed24b1ad22ec2e7efea049b8737-Abstract.html.

Cited on page 63.

Naman Agarwal, Brian Bullins, and Elad Hazan. Second-order stochastic optimization for machine learning in linear time. *The Journal of Machine Learning Research*, 18(1):4148–4187, January 2017. ISSN 1532-4435.

Cited on page 72.

Ulrich Aivodji, Hiromi Arai, Olivier Fortineau, Sébastien Gambs, Satoshi Hara, and Alain Tapp. Fairwashing: The risk of rationalization. In *International Conference on Machine Learning*, pages 161–170. PMLR, May 2019. URL https://proceedings.mlr.press/v97/aivodji19a.html.

Cited on pages 13, 15, and 61.

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Cited on page 50.

Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-Level Language Modeling with Deeper Self-Attention. *Proceedings of the AAAI Conference on Artificial*

Intelligence, 33(01):3159–3166, July 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33013159. URL https://ojs.aaai.org/index.php/AAAI/article/view/4182.
Cited on page 24.

Tariq Alhindi, Savvas Petridis, and Smaranda Muresan. Where is Your Evidence: Improving Fact-checking by Justification Modeling. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 85–90, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5513. URL https://aclanthology.org/W18-5513.
Cited on page 81.

David Alvarez-Melis and Tommi S. Jaakkola. Towards robust interpretability with self-explaining neural networks. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pages 7786–7795, Red Hook, NY, USA, December 2018. Curran Associates Inc.
Cited on pages 15, 61, and 63.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. A Latent Variable Model Approach to PMI-based Word Embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399, 2016. doi: 10.1162/tacl_a_00106. URL https://aclanthology.org/Q16-1028.
Cited on pages 26 and 34.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SyK00v5xx.
Cited on page 27.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear Algebraic Structure of Word Senses, with Applications to Polysemy. *Transactions of the Association for Computational Linguistics*, 6(0):483–495, July 2018. ISSN 2307-387X. URL https://transacl.org/ojs/index.php/tacl/article/view/1346.
Cited on page 27.

Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58: 82–115, 2020. ISSN 1566-2535. doi: 10.1016/j.inffus.2019.12.012. URL https://www.sciencedirect.com/science/article/pii/S1566253519308103.
Cited on pages 13, 14, 15, and 16.

Carlos Aspillaga, Andrés Carvallo, and Vladimir Araujo. Stress Test Evaluation of Transformer-based Models in Natural Language Understanding Tasks. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 1882–1894, Marseille, France, May 2020. European Language Resources Association. ISBN 979-10-95546-34-4. URL https://aclanthology.org/2020.lrec-1.232.

Cited on page 12.

Pepa Atanasova, Jakob Grue Simonsen, Christina Lioma, and Isabelle Augenstein. Generating Fact Checking Explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7352–7364, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.656. URL https://www.aclweb.org/anthology/2020.acl-main.656.

Cited on pages 81 and 108.

David Atkinson, Kumar Bhargav Srinivasan, and Chenhao Tan. What Gets Echoed? Understanding the "Pointers" in Explanations of Persuasive Arguments. 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)*, pages 2911–2921, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1289. URL https://aclanthology.org/D19-1289.

Cited on page 76.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv:1607.06450 [cs, stat]*, July 2016. URL http://arxiv.org/abs/1607.06450.

Cited on page 95.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv:1409.0473 [cs, stat]*, May 2016. URL http://arxiv.org/abs/1409.0473.

Cited on page 86.

Satanjeev Banerjee and Alon Lavie. 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*, pages 65–72, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. URL https://aclanthology.org/W05-0909.

Cited on page 103.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. The WaCky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226, September 2009. ISSN 1574-0218. doi: 10.1007/s10579-009-9081-4.

URL https://doi.org/10.1007/s10579-009-9081-4.
Cited on page 34.

Jasmijn Bastings, Wilker Aziz, and Ivan Titov. Interpretable Neural Predictions with Differentiable Binary Variables. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2963–2977, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1284. URL https://www.aclweb.org/anthology/P19-1284.
Cited on page 75.

Emily M. Bender and Alexander Koller. 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*, pages 5185–5198, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.463. URL https://aclanthology.org/2020.acl-main.463.
Cited on page 12.

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? 🦜. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, pages 610–623, New York, NY, USA, March 2021. Association for Computing Machinery. ISBN 978-1-4503-8309-7. doi: 10.1145/3442188.3445922. URL https://doi.org/10.1145/3442188.3445922.
Cited on page 12.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3(null):1137–1155, March 2003. ISSN 1532-4435.
Cited on page 25.

Yoshua Bengio, Holger Schwenk, Jean-Sébastien Senécal, Fréderic Morin, and Jean-Luc Gauvain. Neural Probabilistic Language Models. In Dawn E. Holmes and Lakhmi C. Jain, editors, *Innovations in Machine Learning: Theory and Applications*, Studies in Fuzziness and Soft Computing, pages 137–186. Springer, Berlin, Heidelberg, 2006. ISBN 978-3-540-33486-6. doi: 10.1007/3-540-33486-6_6. URL https://doi.org/10.1007/3-540-33486-6_6.
Cited on page 25.

J. Bergstra, D. Yamins, and D. D. Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*,

ICML'13, pages I–115–I–123, 2013.
Cited on page 50.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*, NIPS'11, pages 2546–2554, 2011. ISBN 978-1-61839-599-3.
Cited on page 50.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. doi: 10.1162/tacl_a_00051. URL https://www.aclweb.org/anthology/Q17-1010.
Cited on pages 28 and 43.

Tom Bourgeade, Philippe Muller, and Tim Van de Cruys. Plongements Interprétables pour la Détection de Biais Cachés (Interpretable Embeddings for Hidden Biases Detection). In *Actes de la 28e Conférence sur le Traitement Automatique des Langues Naturelles. Volume 1 : conférence principale*, pages 64–80, Lille, France, June 2021. ATALA. URL https://aclanthology.org/2021.jeptalnrecital-taln.6.
Cited on page 55.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal, 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL http://aclweb.org/anthology/D15-1075.
Cited on pages 18, 41, 76, 77, and 110.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. *arXiv:2005.14165 [cs]*, July 2020. URL http://arxiv.org/abs/2005.14165.
Cited on pages 11, 12, 25, 74, 93, and 116.

Jamie Callan, Mark Hoy, Changkuk Yoo, and Le Zhao. Clueweb09 data set, 2009.
Cited on page 34.

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the Role of Bleu in Machine Translation Research. In *11th Conference of the European Chapter of the Association*

for Computational Linguistics, pages 249–256, Trento, Italy, April 2006. Association for Computational Linguistics. URL https://aclanthology.org/E06-1032.
Cited on page 102.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. E-SNLI: Natural Language Inference with Natural Language Explanations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/4c7a167bb329bd92580a99ce422d6fa6-Paper.pdf.
Cited on pages 74, 76, 77, 78, 80, 81, 83, 96, 99, 100, 101, 103, 108, 109, 110, 112, 113, and 115.

Oana-Maria Camburu, Brendan Shillingford, Pasquale Minervini, Thomas Lukasiewicz, and Phil Blunsom. Make Up Your Mind! Adversarial Generation of Inconsistent Natural Language Explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4157–4165, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.382. URL https://www.aclweb.org/anthology/2020.acl-main.382.
Cited on pages 111, 116, and 117.

Samuel Carton, Anirudh Rathore, and Chenhao Tan. Evaluating and Characterizing Human Rationales. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9294–9307, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.747. URL https://aclanthology.org/2020.emnlp-main.747.
Cited on page 76.

Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. Active Bias: Training More Accurate Neural Networks by Emphasizing High Variance Samples. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/hash/2f37d10131f2a483a8dd005b3d14b0d9-Abstract.html.
Cited on page 111.

Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L. Boyd-graber, and David M. Blei. Reading Tea Leaves: How Humans Interpret Topic Models. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 288–296. Curran Associates, Inc., 2009. URL http://papers.nips.cc/paper/3700-reading-tea-leaves-how-humans-interpret-topic-models.pdf.
Cited on page 38.

Danqi Chen, Jason Bolton, and Christopher D. Manning. A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany, 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1223. URL http://www.aclweb.org/anthology/P16-1223.

Cited on page 18.

Lei Chen and Chong Min Lee. Predicting Audience's Laughter During Presentations Using Convolutional Neural Network. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 86–90, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-5009. URL https://aclanthology.org/W17-5009.

Cited on page 41.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179.

Cited on page 84.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions. 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)*, pages 2924–2936, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1300. URL https://www.aclweb.org/anthology/N19-1300.

Cited on pages 39 and 51.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1070. URL https://www.aclweb.org/anthology/D17-1070.

Cited on pages 12, 19, 28, 50, 80, and 85.

R. Dennis Cook and Sanford Weisberg. Characterizations of an Empirical Influence Function for Detecting Influential Cases in Regression. *Technometrics*, 22(4):495–508, 1980. ISSN 0040-1706.

doi: 10.2307/1268187. URL https://www.jstor.org/stable/1268187.
Cited on page 71.

Yogesh Dahiya, Partha Talukdar, et al. Discovering response-eliciting factors in social question answering: A reddit inspired study. In *Tenth International AAAI Conference on Web and Social Media*, 2016.
Cited on page 30.

Zeyu Dai and Ruihong Huang. A Regularization Approach for Incorporating Event Knowledge and Coreference Relations into Neural Discourse Parsing. 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)*, pages 2976–2987, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1295. URL https://www.aclweb.org/anthology/D19-1295.
Cited on pages 42 and 51.

Alexander D'Amour, Katherine Heller, Dan Moldovan, Ben Adlam, Babak Alipanahi, Alex Beutel, Christina Chen, Jonathan Deaton, Jacob Eisenstein, Matthew D. Hoffman, Farhad Hormozdiari, Neil Houlsby, Shaobo Hou, Ghassen Jerfel, Alan Karthikesalingam, Mario Lucic, Yian Ma, Cory McLean, Diana Mincu, Akinori Mitani, Andrea Montanari, Zachary Nado, Vivek Natarajan, Christopher Nielson, Thomas F. Osborne, Rajiv Raman, Kim Ramasamy, Rory Sayres, Jessica Schrouff, Martin Seneviratne, Shannon Sequeira, Harini Suresh, Victor Veitch, Max Vladymyrov, Xuezhi Wang, Kellie Webster, Steve Yadlowsky, Taedong Yun, Xiaohua Zhai, and D. Sculley. Underspecification Presents Challenges for Credibility in Modern Machine Learning. *arXiv:2011.03395 [cs, stat]*, November 2020. URL http://arxiv.org/abs/2011.03395.
Cited on page 13.

Susanne Dandl, Christoph Molnar, Martin Binder, and Bernd Bischl. Multi-Objective Counterfactual Explanations. In Thomas Bäck, Mike Preuss, André Deutz, Hao Wang, Carola Doerr, Michael Emmerich, and Heike Trautmann, editors, *Parallel Problem Solving from Nature – PPSN XVI*, Lecture Notes in Computer Science, pages 448–469, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58112-1. doi: 10.1007/978-3-030-58112-1_31.
Cited on page 71.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990. ISSN 1097-4571. doi: 10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9.
Cited on page 26.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Cited on pages 11, 12, 56, 74, 88, 89, 92, and 96.

Jonathan Dodge, Q. Vera Liao, Yunfeng Zhang, Rachel K. E. Bellamy, and Casey Dugan. Explaining models: An empirical study of how explanations impact fairness judgment. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*, IUI '19, pages 275–285, New York, NY, USA, March 2019. Association for Computing Machinery. ISBN 978-1-4503-6272-6. doi: 10.1145/3301275.3302310. URL https://doi.org/10.1145/3301275.3302310.

Cited on pages 13 and 69.

Finale Doshi-Velez and Been Kim. Towards A Rigorous Science of Interpretable Machine Learning. *arXiv:1702.08608 [cs, stat]*, March 2017. URL http://arxiv.org/abs/1702.08608.

Cited on pages 14, 15, and 16.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *The Journal of Machine Learning Research*, 12:2121–2159, July 2011. ISSN 1532-4435.

Cited on page 35.

Philipp Dufter and Hinrich Schütze. Analytical Methods for Interpretable Ultradense Word Embeddings. 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)*, pages 1185–1191, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1111. URL https://www.aclweb.org/anthology/D19-1111.

Cited on page 35.

Hicham El Boukkouri, Olivier Ferret, Thomas Lavergne, Hiroshi Noji, Pierre Zweigenbaum, and Jun'ichi Tsujii. CharacterBERT: Reconciling ELMo and BERT for Word-Level Open-Vocabulary Representations From Characters. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6903–6915, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.609. URL https://aclanthology.org/2020.coling-main.609.

Cited on page 24.

Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah A. Smith. Sparse Overcomplete Word Vector Representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1491–1500, Beijing, China, 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1144. URL http://aclweb.org/anthology/P15-1144.

Cited on pages 30, 35, 38, and 47.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can We Beat Google?*, pages 47–54, June 2008.

Cited on page 34.

William Ferreira and Andreas Vlachos. Emergent: A novel data-set for stance classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1168, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1138. URL https://www.aclweb.org/anthology/N16-1138.

Cited on pages 39 and 51.

J. R. Firth. A synopsis of linguistic theory 1930-55. 1952–59:1–32, 1957.

Cited on page 25.

Joseph L. Fleiss, Bruce Levin, and Myunghee Cho Paik. The measurement of interrater agreement. In *Statistical Methods for Rates and Proportions*, chapter 18, pages 598–626. John Wiley & Sons, Ltd, 2003. ISBN 978-0-471-44542-5. doi: 10.1002/0471445428.ch18. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/0471445428.ch18.

Cited on page 108.

Benoît Frénay and Ata Kaban. A Comprehensive Introduction to Label Noise. *Proceedings of the 2014 European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2014)*, 2014.

Cited on page 111.

Benoit Frenay and Michel Verleysen. Classification in the Presence of Label Noise: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 25(5):845–869, May 2014. ISSN 2162-2388. doi: 10.1109/TNNLS.2013.2292894.

Cited on page 111.

Alona Fyshe, Brian Murphy, Partha Talukdar, and Tom Mitchell. Documents and Dependencies: An Exploration of Vector Space Models for Semantic Composition. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages

84–93, Sofia, Bulgaria, August 2013. Association for Computational Linguistics. URL https://aclanthology.org/W13-3510.

Cited on page 30.

Alona Fyshe, Partha P. Talukdar, Brian Murphy, and Tom M. Mitchell. Interpretable Semantic Vectors from a Joint Model of Brain- and Text- Based Meaning. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 489–499, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-1046. URL https://www.aclweb.org/anthology/P14-1046.

Cited on pages 30, 31, and 38.

Alona Fyshe, Leila Wehbe, Partha P. Talukdar, Brian Murphy, and Tom M. Mitchell. A Compositional and Interpretable Semantic Space. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 32–41, Denver, Colorado, 2015. Association for Computational Linguistics. doi: 10.3115/v1/N15-1004. URL http://aclweb.org/anthology/N15-1004.

Cited on page 30.

Philip Gage. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38, February 1994. ISSN 0898-9788.

Cited on page 88.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, November 2020. ISSN 2522-5839. doi: 10.1038/s42256-020-00257-z. URL https://www.nature.com/articles/s42256-020-00257-z.

Cited on pages 13 and 17.

C. Goller and A. Kuchler. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 1, pages 347–352 vol.1, June 1996. doi: 10.1109/ICNN.1996.548916.

Cited on page 40.

Bryce Goodman and Seth Flaxman. European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation". *AI Magazine*, 38(3):50–57, October 2017. ISSN 2371-9621. doi: 10.1609/aimag.v38i3.2741. URL https://ojs.aaai.org/index.php/aimagazine/article/view/2741.

Cited on pages 13 and 74.

J. C. Gower. A General Coefficient of Similarity and Some of Its Properties. *Biometrics*, 27 (4):857–871, 1971. ISSN 0006-341X. doi: 10.2307/2528823. URL https://www.jstor.org/stable/2528823.

Cited on page 71.

Yvette Graham. Re-evaluating Automatic Summarization with BLEU and 192 Shades of ROUGE. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 128–137, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1013. URL https://aclanthology.org/D15-1013.

Cited on page 102.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5):602–610, July 2005. ISSN 0893-6080. doi: 10.1016/j.neunet.2005.06.042. URL https://www.sciencedirect.com/science/article/pii/S0893608005001206.

Cited on page 87.

Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. Recent advances in convolutional neural networks. *Pattern Recognition*, 77(C):354–377, May 2018. ISSN 0031-3203. doi: 10.1016/j.patcog.2017.10.013. URL https://doi.org/10.1016/j.patcog.2017.10.013.

Cited on page 84.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A Survey of Methods for Explaining Black Box Models. *ACM Computing Surveys*, 51(5):93:1–93:42, August 2018. ISSN 0360-0300. doi: 10.1145/3236009. URL https://doi.org/10.1145/3236009.

Cited on pages 14 and 15.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. Annotation Artifacts in Natural Language Inference Data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana, 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-2017. URL http://aclweb.org/anthology/N18-2017.

Cited on pages 19 and 20.

Zellig S. Harris. Distributional Structure. *WORD*, 10(2-3):146–162, August 1954. ISSN 0043-7956. doi: 10.1080/00437956.1954.11659520. URL https://doi.org/10.1080/00437956.1954.11659520.

Cited on pages 25 and 84.

Md Kamrul Hasan, Wasifur Rahman, AmirAli Bagher Zadeh, Jianyuan Zhong, Md Iftekhar Tanveer, Louis-Philippe Morency, and Mohammed (Ehsan) Hoque. UR-FUNNY: A Multimodal Language Dataset for Understanding Humor. 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)*, pages 2046–2056, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1211. URL https://www.aclweb.org/anthology/D19-1211.

Cited on pages 40 and 51.

Peter Hase and Mohit Bansal. Evaluating Explainable AI: Which Algorithmic Explanations Help Users Predict Model Behavior? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5540–5552, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.491. URL https://www.aclweb.org/anthology/2020.acl-main.491.

Cited on page 74.

He He, Sheng Zha, and Haohan Wang. Unlearn Dataset Bias in Natural Language Inference by Fitting the Residual. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 132–142, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-6115. URL https://www.aclweb.org/anthology/D19-6115.

Cited on page 20.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. doi: 10.1109/CVPR.2016.90.

Cited on page 95.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning Distributed Representations of Sentences from Unlabelled Data. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1367–1377, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1162. URL https://aclanthology.org/N16-1162.

Cited on page 85.

Sepp Hochreiter. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 06(02):107–116, April 1998. ISSN 0218-4885. doi: 10.1142/S0218488598000094. URL https://www.worldscientific.com/doi/abs/10.1142/s0218488598000094.

Cited on page 85.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Cited on pages 84 and 85.

J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, April 1982. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.79.8.2554. URL https://www.pnas.org/content/79/8/2554.

Cited on page 84.

Benjamin D. Horne, Dorit Nevo, John O'Donovan, Jin-Hee Cho, and Sibel Adalı. Rating Reliability and Bias in News Articles: Does AI Assistance Help Everyone? *Proceedings of the International AAAI Conference on Web and Social Media*, 13:247–256, July 2019. ISSN 2334-0770. URL https://ojs.aaai.org/index.php/ICWSM/article/view/3226.

Cited on page 74.

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. An Analysis of Natural Language Inference Benchmarks through the Lens of Negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9106–9118, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.732. URL https://www.aclweb.org/anthology/2020.emnlp-main.732.

Cited on pages 12 and 51.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. Learning Whom to Trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL https://aclanthology.org/N13-1132.

Cited on page 111.

P.O. Hoyer. Non-negative sparse coding. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, pages 557–565, September 2002. doi: 10.1109/NNSP.2002.1030067.

Cited on pages 31 and 32.

Miriam Hurtado Bodell, Martin Arvidsson, and M\aans Magnusson. Interpretable Word Embeddings via Informative Priors. 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), pages 6324–6330, Hong Kong, China, November 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D19-1661.
Cited on page 30.

Allan Jabri, Armand Joulin, and Laurens van der Maaten. Revisiting Visual Question Answering Baselines. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 727–739, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46484-8. doi: 10.1007/978-3-319-46484-8_44.
Cited on page 18.

Sarthak Jain and Byron C. Wallace. 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)*, pages 3543–3556, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1357. URL https://www.aclweb.org/anthology/N19-1357.
Cited on page 74.

Harsh Jhamtani and Peter Clark. Learning to Explain: Datasets and Models for Identifying Valid Reasoning Chains in Multihop Question-Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 137–150, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.10. URL https://aclanthology.org/2020.emnlp-main.10.
Cited on page 75.

Robin Jia and Percy Liang. Adversarial Examples for Evaluating Reading Comprehension Systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1215. URL https://aclanthology.org/D17-1215.
Cited on pages 12 and 18.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is BERT Really Robust? A Strong Baseline for Natural Language Attack on Text Classification and Entailment. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8018–8025, April 2020. ISSN 2374-3468. doi: 10.1609/aaai.v34i05.6311. URL https://ojs.aaai.org/index.php/AAAI/article/view/6311.
Cited on page 12.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification. In *Proceedings of the 15th Conference of the European Chapter*

of the Association for Computational Linguistics: Volume 2, Short Papers, pages 427–431, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/E17-2068.

Cited on pages 43 and 50.

Lukasz Kaiser, Aidan N. Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. One Model To Learn Them All. *arXiv:1706.05137 [cs, stat]*, June 2017. URL http://arxiv.org/abs/1706.05137.

Cited on page 82.

Nora Kassner and Hinrich Schütze. Negated and Misprimed Probes for Pretrained Language Models: Birds Can Talk, But Cannot Fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.698. URL https://www.aclweb.org/anthology/2020.acl-main.698.

Cited on pages 12 and 51.

Emilie Kaufmann and Shivaram Kalyanakrishnan. Information complexity in bandit subset selection. In *Conference on Learning Theory*, pages 228–251. PMLR, 2013.

Cited on page 66.

Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna Wallach, and Jennifer Wortman Vaughan. Interpreting Interpretability: Understanding Data Scientists' Use of Interpretability Tools for Machine Learning. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, pages 1–14, New York, NY, USA, April 2020. Association for Computing Machinery. ISBN 978-1-4503-6708-0. doi: 10.1145/3313831.3376219. URL https://doi.org/10.1145/3313831.3376219.

Cited on page 69.

Divyansh Kaushik and Zachary C. Lipton. How Much Reading Does Reading Comprehension Require? A Critical Investigation of Popular Benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1546. URL https://aclanthology.org/D18-1546.

Cited on page 18.

Maxime Kayser, Oana-Maria Camburu, Leonard Salewski, Cornelius Emde, Virginie Do, Zeynep Akata, and Thomas Lukasiewicz. E-ViL: A Dataset and Benchmark for Natural Language Explanations in Vision-Language Tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1244–1254, October 2021.

Cited on page 77.

Tushar Khot, Peter Clark, Michal Guerquin, Peter Jansen, and Ashish Sabharwal. QASC: A Dataset for Question Answering via Sentence Composition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8082–8090, April 2020. ISSN 2374-3468. doi: 10.1609/aaai.v34i05.6319. URL https://ojs.aaai.org/index.php/AAAI/article/view/6319.

Cited on page 75.

Najoung Kim, Song Feng, Chulaka Gunasekara, and Luis Lastras. Implicit Discourse Relation Classification: We Need to Talk about Evaluation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5404–5414, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.480. URL https://aclanthology.org/2020.acl-main.480.

Cited on page 42.

Ryan Kiros, Yukun Zhu, Russ R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-Thought Vectors. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://papers.nips.cc/paper/2015/hash/f442d33fa06832082290ad8544a8da27-Abstract.html.

Cited on pages 28 and 85.

Dan Klein and Christopher D. Manning. Accurate Unlexicalized Parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan, July 2003. Association for Computational Linguistics. doi: 10.3115/1075096.1075150. URL https://aclanthology.org/P03-1054.

Cited on page 40.

Tassilo Klein and Moin Nabi. Attention Is (not) All You Need for Commonsense Reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4831–4836, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1477. URL https://aclanthology.org/P19-1477.

Cited on page 74.

Thomas Kober, Julie Weeds, John Wilkie, Jeremy Reffin, and David Weir. One Representation per Word - Does it make Sense for Composition? In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and Their Applications*, pages 79–90, Valencia, Spain, April 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-1910. URL https://www.aclweb.org/anthology/W17-1910.

Cited on pages 27 and 43.

Pang Wei Koh and Percy Liang. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney,*

NSW, Australia, 6-11 August 2017, pages 1885–1894, 2017. URL http://proceedings.mlr.press/v70/koh17a.html.
Cited on page 71.

I. Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. Problems with Shapley-value-based explanations as feature importance measures. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5491–5500. PMLR, November 2020. URL https://proceedings.mlr.press/v119/kumar20e.html.
Cited on page 69.

Sawan Kumar and Partha Talukdar. NILE : Natural Language Inference with Faithful Natural Language Explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8730–8742, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.771. URL https://www.aclweb.org/anthology/2020.acl-main.771.
Cited on pages 81 and 117.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7:452–466, March 2019. doi: 10.1162/tacl_a_00276. URL https://aclanthology.org/Q19-1026.
Cited on page 39.

Erwan Le Merrer and Gilles Trédan. Remote explainability faces the bouncer problem. *Nature Machine Intelligence*, 2(9):529–539, September 2020. ISSN 2522-5839. doi: 10.1038/s42256-020-0216-z. URL https://www.nature.com/articles/s42256-020-0216-z.
Cited on page 116.

Yann LeCun, Bernhard Boser, John Denker, Donnie Henderson, R. Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten Digit Recognition with a Back-Propagation Network. In *Advances in Neural Information Processing Systems*, volume 2. Morgan-Kaufmann, 1990. URL https://proceedings.neurips.cc/paper/1989/hash/53c3bce66e43be4f209556518c2fcb54-Abstract.html.
Cited on page 83.

Yann LeCun, Fu Jie Huang, and Léon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, pages 97–104, USA, June 2004. IEEE Computer Society.
Cited on page 83.

Yann LeCun, Koray Kavukcuoglu, and Clement Farabet. Convolutional networks and applications in vision. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 253–256, May 2010. doi: 10.1109/ISCAS.2010.5537907.
Cited on page 83.

D. D. Lee and H. S. Seung. Unsupervised learning by convex and conic coding. In *Proceedings of the 9th International Conference on Neural Information Processing Systems*, NIPS'96, pages 515–521, Cambridge, MA, USA, December 1996. MIT Press.
Cited on page 32.

Daniel D. Lee and H. Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, October 1999. ISSN 0028-0836, 1476-4687. doi: 10.1038/44565. URL http://www.nature.com/articles/44565.
Cited on pages 30, 31, and 32.

Daniel D. Lee and H. Sebastian Seung. Algorithms for Non-negative Matrix Factorization. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, 2001. URL http://papers.nips.cc/paper/1861-algorithms-for-non-negative-matrix-factorization.pdf.
Cited on pages 31, 32, 33, and 46.

Jason Lee, Kyunghyun Cho, and Thomas Hofmann. Fully Character-Level Neural Machine Translation without Explicit Segmentation. *Transactions of the Association for Computational Linguistics*, 5:365–378, 2017. doi: 10.1162/tacl_a_00067. URL https://aclanthology.org/Q17-1026.
Cited on page 24.

Omer Levy and Yoav Goldberg. Neural Word Embedding as Implicit Matrix Factorization. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5477-neural-word-embedding-as-implicit-matrix-factorization.pdf.
Cited on page 28.

Omer Levy, Yoav Goldberg, and Ido Dagan. Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225, 2015. doi: 10.1162/tacl_a_00134. URL https://aclanthology.org/Q15-1016.
Cited on pages 26 and 28.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 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*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://www.aclweb.org/anthology/2020.acl-main.703.
Cited on pages 91, 94, 96, 98, 99, and 100.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. BERT-ATTACK: Adversarial Attack Against BERT Using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.500. URL https://aclanthology.org/2020.emnlp-main.500.
Cited on page 12.

Dongyun Liang, Weiran Xu, and Yinge Zhao. Combining Word-Level and Character-Level Representations for Relation Classification of Informal Text. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 43–47, Vancouver, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-2606. URL https://aclanthology.org/W17-2606.
Cited on page 24.

Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013.
Cited on pages 101 and 102.

Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program Induction by Rationale Generation: Learning to Solve and Explain Algebraic Word Problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1015. URL https://aclanthology.org/P17-1015.
Cited on page 76.

Stan Lipovetsky and Michael Conklin. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, 17(4):319–330, 2001. ISSN 1526-4025. doi: 10.1002/asmb.446. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/asmb.446.
Cited on page 69.

Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and Correcting for Label Shift with Black Box Predictors. In *International Conference on Machine Learning*, pages 3122–3130. PMLR, July 2018. URL http://proceedings.mlr.press/v80/lipton18a.

html.
Cited on page 13.

Zachary C. Lipton. The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, June 2018. ISSN 1542-7730. doi: 10.1145/3236386.3241340. URL https://doi.org/10.1145/3236386.3241340.
Cited on pages 14 and 15.

W. A. Little. The existence of persistent states in the brain. *Mathematical Biosciences*, 19(1):101–120, February 1974. ISSN 0025-5564. doi: 10.1016/0025-5564(74)90031-5. URL https://www.sciencedirect.com/science/article/pii/0025556474900315.
Cited on page 84.

Hui Liu, Qingyu Yin, and William Yang Wang. Towards Explainable NLP: A Generative Explanation Framework for Text Classification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5570–5581, Florence, Italy, July 2019a. Association for Computational Linguistics. doi: 10.18653/v1/P19-1560. URL https://www.aclweb.org/anthology/P19-1560.
Cited on page 81.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. Multi-Task Deep Neural Networks for Natural Language Understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy, July 2019b. Association for Computational Linguistics. doi: 10.18653/v1/P19-1441. URL https://www.aclweb.org/anthology/P19-1441.
Cited on page 51.

Yang Liu and Mirella Lapata. Text Summarization with Pretrained Encoders. 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)*, pages 3730–3740, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1387. URL https://aclanthology.org/D19-1387.
Cited on page 83.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*, July 2019c. URL http://arxiv.org/abs/1907.11692.
Cited on pages 81 and 98.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual Denoising Pre-training for Neural Machine Translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020. doi: 10.1162/tacl_a_00343. URL https://aclanthology.org/2020.tacl-1.47.
Cited on page 96.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. *arXiv:1711.05101 [cs, math]*, January 2019. URL http://arxiv.org/abs/1711.05101.
Cited on page 100.

Scott M Lundberg and Su-In Lee. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf.
Cited on pages 68, 74, and 116.

Scott M. Lundberg, Gabriel G. Erion, and Su-In Lee. Consistent Individualized Feature Attribution for Tree Ensembles. *arXiv:1802.03888 [cs, stat]*, March 2019. URL http://arxiv.org/abs/1802.03888.
Cited on page 69.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL https://aclanthology.org/D15-1166.
Cited on page 86.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning Word Vectors for Sentiment Analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P11-1015.
Cited on page 40.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online Learning for Matrix Factorization and Sparse Coding. *The Journal of Machine Learning Research*, 11:19–60, March 2010. ISSN 1532-4435.
Cited on page 34.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1993. URL https://aclanthology.org/J93-2004.
Cited on page 41.

James Martens. Deep learning via Hessian-free optimization. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pages 735–742, Madison, WI, USA, June 2010. Omnipress. ISBN 978-1-60558-907-7.
Cited on page 72.

R. Thomas McCoy, Junghyun Min, and Tal Linzen. BERTs of a feather do not generalize together: Large variability in generalization across models with similar test set performance. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 217–227, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.blackboxnlp-1.21. URL https://aclanthology.org/2020.blackboxnlp-1.21.
Cited on page 13.

Tom McCoy, Ellie Pavlick, and Tal Linzen. Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1334. URL https://www.aclweb.org/anthology/P19-1334.
Cited on page 19.

Ken McRae, George S. Cree, Mark S. Seidenberg, and Chris Mcnorgan. Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, 37(4):547–559, November 2005. ISSN 1554-3528. doi: 10.3758/BF03192726. URL https://doi.org/10.3758/BF03192726.
Cited on page 31.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013a. URL http://arxiv.org/abs/1301.3781.
Cited on pages 27, 47, and 85.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013b. URL http:

//papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf.
Cited on pages 27, 47, and 85.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 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*, pages 746–751, Atlanta, Georgia, June 2013c. Association for Computational Linguistics. URL https://aclanthology.org/N13-1090.
Cited on page 26.

Tim Miller. Explanation in Artificial Intelligence: Insights from the Social Sciences. *arXiv:1706.07269 [cs]*, June 2017. URL http://arxiv.org/abs/1706.07269.
Cited on pages 69 and 107.

Andriy Mnih and Geoffrey E Hinton. A Scalable Hierarchical Distributed Language Model. In *Advances in Neural Information Processing Systems*, volume 21. Curran Associates, Inc., 2009. URL https://proceedings.neurips.cc/paper/2008/hash/1e056d2b0ebd5c878c550da6ac5d3724-Abstract.html.
Cited on page 27.

Christoph Molnar. *Interpretable Machine Learning. A Guide for Making Black Box Models Explainable*. 2019. URL https://christophm.github.io/interpretable-ml-book/.
Cited on pages 14, 15, 16, and 61.

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, February 2018. ISSN 1051-2004. doi: 10.1016/j.dsp.2017.10.011. URL https://www.sciencedirect.com/science/article/pii/S1051200417302385.
Cited on pages 56 and 63.

Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-Wise Relevance Propagation: An Overview. In Wojciech Samek, Grégoire Montavon, Andrea Vedaldi, Lars Kai Hansen, and Klaus-Robert Müller, editors, *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 193–209. Springer International Publishing, Cham, 2019. ISBN 978-3-030-28954-6. doi: 10.1007/978-3-030-28954-6_10. URL https://doi.org/10.1007/978-3-030-28954-6_10.
Cited on pages 56 and 63.

Frederic Morin and Yoshua Bengio. Hierarchical Probabilistic Neural Network Language Model. In *International Workshop on Artificial Intelligence and Statistics*, pages 246–252. PMLR,

January 2005. URL http://proceedings.mlr.press/r5/morin05a.html.
Cited on page 27.

W. James Murdoch, Peter J. Liu, and Bin Yu. Beyond Word Importance: Contextual Decomposition to Extract Interactions from LSTMs. In *International Conference on Learning Representations*, February 2018. URL https://openreview.net/forum?id=rkRwGg-0Z.
Cited on page 63.

Brian Murphy, Partha Talukdar, and Tom Mitchell. Learning Effective and Interpretable Semantic Models using Non-Negative Sparse Embedding. In *Proceedings of COLING 2012*, pages 1933–1950, Mumbai, India, December 2012. The COLING 2012 Organizing Committee. URL https://www.aclweb.org/anthology/C12-1118.
Cited on pages 30, 31, 34, 38, and 47.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. Stress Test Evaluation for Natural Language Inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/C18-1198.
Cited on pages 12 and 51.

J. A. Nelder and R. W. M. Wedderburn. Generalized Linear Models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384, 1972. ISSN 0035-9238. doi: 10.2307/2344614. URL https://www.jstor.org/stable/2344614.
Cited on page 43.

Shereen Oraby, Vrindavan Harrison, Lena Reed, Ernesto Hernandez, Ellen Riloff, and Marilyn Walker. Creating and Characterizing a Diverse Corpus of Sarcasm in Dialogue. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 31–41, Los Angeles, September 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-3604. URL https://www.aclweb.org/anthology/W16-3604.
Cited on pages 40 and 51.

Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech and Language Processing*, 24(4):694–707, April 2016. ISSN 2329-9290. doi: 10.1109/TASLP.2016.2520371. URL https://doi.org/10.1109/TASLP.2016.2520371.
Cited on page 85.

Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, October 2010. ISSN 1558-2191. doi:

10.1109/TKDE.2009.191.

Cited on page 89.

Bo Pang and Lillian Lee. Seeing Stars: Exploiting Class Relationships for Sentiment Categorization with Respect to Rating Scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics. doi: 10.3115/1219840.1219855. URL https://aclanthology.org/P05-1015.

Cited on page 40.

Abhishek Panigrahi, Harsha Vardhan Simhadri, and Chiranjib Bhattacharyya. Word2Sense: Sparse Interpretable Word Embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5692–5705, Florence, Italy, 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1570. URL https://www.aclweb.org/anthology/P19-1570.

Cited on pages 34, 47, and 49.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040.

Cited on page 101.

Sungjoon Park, JinYeong Bak, and Alice Oh. Rotated Word Vector Representations and their Interpretability. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 401–411. Association for Computational Linguistics, 2017. doi: 10.18653/v1/d17-1041. URL https://doi.org/10.18653/v1/d17-1041.

Cited on page 35.

Barak A. Pearlmutter. Fast exact multiplication by the Hessian. *Neural Computation*, 6(1): 147–160, January 1994. ISSN 0899-7667. doi: 10.1162/neco.1994.6.1.147. URL https://doi.org/10.1162/neco.1994.6.1.147.

Cited on page 72.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://www.aclweb.

org/anthology/D14-1162.
Cited on pages 28 and 47.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. Regularizing Neural Networks by Penalizing Confident Output Distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HyhbYrGYe.
Cited on page 100.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 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)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202.
Cited on pages 12, 28, and 56.

Adam Poliak. A survey on Recognizing Textual Entailment as an NLP Evaluation. In *Proceedings of the First Workshop on Evaluation and Comparison of NLP Systems*, pages 92–109, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.eval4nlp-1.10. URL https://aclanthology.org/2020.eval4nlp-1.10.
Cited on page 41.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. Hypothesis Only Baselines in Natural Language Inference. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 180–191, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/S18-2023. URL https://www.aclweb.org/anthology/S18-2023.
Cited on pages 18, 19, and 20.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, May 2008. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2008/pdf/754_paper.pdf.
Cited on page 41.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. Learning to Deceive with Attention-Based Explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4782–4793, Online, July

2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.432. URL https://www.aclweb.org/anthology/2020.acl-main.432.
Cited on page 74.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. page 12, 2018.
Cited on pages 11, 12, 25, 78, 89, 90, 93, and 96.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. *OpenAI blog*, 1(8):24, 2019.
Cited on pages 11, 12, 25, and 93.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. Explain Yourself! Leveraging Language Models for Commonsense Reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4932–4942, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1487. URL https://www.aclweb.org/anthology/P19-1487.
Cited on pages 76, 77, 78, 79, 80, 81, 83, 104, 105, and 117.

Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B. Gupta, Xiaojiang Chen, and Xin Wang. A Survey of Deep Active Learning. *ACM Computing Surveys*, 54 (9):180:1–180:40, October 2021. ISSN 0360-0300. doi: 10.1145/3472291. URL https://doi.org/10.1145/3472291.
Cited on page 117.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 1135–1144, San Francisco, California, USA, 2016. ACM Press. ISBN 978-1-4503-4232-2. doi: 10.1145/2939672.2939778. URL http://dl.acm.org/citation.cfm?doid=2939672.2939778.
Cited on pages 63, 65, 74, and 116.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Anchors: High-Precision Model-Agnostic Explanations. In *Thirty-Second AAAI Conference on Artificial Intelligence*, April 2018a. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16982.
Cited on page 64.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Semantically Equivalent Adversarial Rules for Debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia, July 2018b. Association for Computational Linguistics. doi: 10.18653/v1/P18-1079. URL

https://aclanthology.org/P18-1079.
Cited on page 12.

Ellen Riloff. Automatically generating extraction patterns from untagged text. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence - Volume 2*, AAAI'96, pages 1044–1049, Portland, Oregon, August 1996. AAAI Press. ISBN 978-0-262-51091-2.
Cited on page 40.

Marko Robnik-Šikonja and Marko Bohanec. Perturbation-Based Explanations of Prediction Models. In Jianlong Zhou and Fang Chen, editors, *Human and Machine Learning: Visible, Explainable, Trustworthy and Transparent*, Human–Computer Interaction Series, pages 159–175. Springer International Publishing, Cham, 2018. ISBN 978-3-319-90403-0. doi: 10.1007/978-3-319-90403-0_9. URL https://doi.org/10.1007/978-3-319-90403-0_9.
Cited on page 15.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A Primer in BERTology: What We Know About How BERT Works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020. doi: 10.1162/tacl_a_00349. URL https://aclanthology.org/2020.tacl-1.54.
Cited on page 74.

Sascha Rothe, Sebastian Ebert, and Hinrich Schütze. Ultradense Word Embeddings by Orthogonal Transformation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 767–777, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1091. URL https://aclanthology.org/N16-1091.
Cited on page 35.

Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, May 2019. ISSN 2522-5839. doi: 10.1038/s42256-019-0048-x. URL https://www.nature.com/articles/s42256-019-0048-x.
Cited on pages 14, 15, 61, and 63.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. ISSN 1476-4687. doi: 10.1038/323533a0. URL https://www.nature.com/articles/323533a0.
Cited on page 84.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv:1910.01108 [cs]*, February 2020.

URL http://arxiv.org/abs/1910.01108.
Cited on page 99.

Maarten Sap, Saadia Gabriel, Lianhui Qin, Dan Jurafsky, Noah A. Smith, and Yejin Choi. Social Bias Frames: Reasoning about Social and Power Implications of Language. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5477–5490, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.486. URL https://aclanthology.org/2020.acl-main.486.
Cited on page 76.

Bernhard Schölkopf, Dominik Janzing, Jonas Peters, Eleni Sgouritsa, Kun Zhang, and Joris Mooij. On causal and anticausal learning. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pages 459–466, Madison, WI, USA, June 2012. Omnipress. ISBN 978-1-4503-1285-1. URL https://icml.cc/2012/papers/625.pdf.
Cited on page 13.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL https://aclanthology.org/P16-1162.
Cited on pages 25 and 88.

Sofia Serrano and Noah A. Smith. Is Attention Interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2931–2951, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1282. URL https://aclanthology.org/P19-1282.
Cited on page 74.

Yuanlong Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. Generating High-Quality and Informative Conversation Responses with Sequence-to-Sequence Models. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2210–2219, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1235. URL https://aclanthology.org/D17-1235.
Cited on page 91.

L. S. Shapley. 17. A Value for n-Person Games. In *17. A Value for n-Person Games*, pages 307–318. Princeton University Press, 1953. ISBN 978-1-4008-8197-0. doi: 10.1515/9781400881970-018. URL https://www.degruyter.com/document/doi/10.

1515/9781400881970-018/html.
Cited on page 67.

Shubham Sharma, Jette Henderson, and Joydeep Ghosh. CERTIFAI: A Common Framework to Provide Explanations and Analyse the Fairness and Robustness of Black-box Models. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, pages 166–172, New York, NY, USA, February 2020. Association for Computing Machinery. ISBN 978-1-4503-7110-0. doi: 10.1145/3375627.3375812. URL https://doi.org/10.1145/3375627.3375812.
Cited on pages 13 and 69.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3145–3153, Sydney, NSW, Australia, August 2017. JMLR.org.
Cited on pages 62 and 63.

Craig Silverman. Lies, Damn Lies and Viral Content. 2015. doi: 10.7916/D8Q81RHH. URL https://doi.org/10.7916/D8Q81RHH.
Cited on page 39.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In *In Workshop at International Conference on Learning Representations*, 2014.
Cited on page 62.

Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial Attacks on Post hoc Explanation Methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, pages 180–186, New York, NY, USA, February 2020. Association for Computing Machinery. ISBN 978-1-4503-7110-0. doi: 10.1145/3375627.3375830. URL https://doi.org/10.1145/3375627.3375830.
Cited on pages 64, 69, and 116.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. SmoothGrad: Removing noise by adding noise. *arXiv:1706.03825 [cs, stat]*, June 2017. URL http://arxiv.org/abs/1706.03825.
Cited on page 62.

Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11,

pages 129–136, Madison, WI, USA, June 2011. Omnipress. ISBN 978-1-4503-0619-5.
Cited on page 40.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic Compositionality through Recursive Matrix-Vector Spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211, Jeju Island, Korea, July 2012. Association for Computational Linguistics. URL https://aclanthology.org/D12-1110.
Cited on page 40.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1170.
Cited on pages 40 and 76.

Robyn Speer, Joshua Chin, and Catherine Havasi. ConceptNet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, pages 4444–4451, San Francisco, California, USA, February 2017. AAAI Press.
Cited on page 78.

Julia Strout, Ye Zhang, and Raymond Mooney. Do Human Rationales Improve Machine Explanations? In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 56–62, Florence, Italy, August 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W19-4807.
Cited on pages 55 and 75.

Erik Štrumbelj and Igor Kononenko. Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems*, 41(3):647–665, December 2014. ISSN 0219-1377. doi: 10.1007/s10115-013-0679-x. URL https://doi.org/10.1007/s10115-013-0679-x.
Cited on page 67.

Anant Subramanian, Danish Pruthi, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Eduard H. Hovy. SPINE: SParse Interpretable Neural Embeddings. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th Innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans,*

Louisiana, USA, February 2-7, 2018, pages 4921–4928. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17433.
Cited on pages 30, 36, 38, 47, 48, and 49.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haifeng Wang. ERNIE 3.0: Large-scale Knowledge Enhanced Pre-training for Language Understanding and Generation. *arXiv:2107.02137 [cs]*, July 2021. URL http://arxiv.org/abs/2107.02137.
Cited on page 11.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3319–3328, Sydney, NSW, Australia, August 2017. JMLR.org.
Cited on page 62.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to Sequence Learning with Neural Networks. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html.
Cited on page 84.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A. Smith, and Yejin Choi. Dataset Cartography: Mapping and Diagnosing Datasets with Training Dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.746. URL https://aclanthology.org/2020.emnlp-main.746.
Cited on pages 21 and 111.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. 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)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421.
Cited on pages 77, 78, 83, and 104.

Wilson L. Taylor. "Cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433, 1953.
Cited on page 89.

Trias Thireou and Martin Reczko. Bidirectional Long Short-Term Memory Networks for Predicting the Subcellular Localization of Eukaryotic Proteins. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 4(3):441–446, July 2007. ISSN 1557-9964. doi: 10.1109/tcbb.2007.1015.
Cited on page 87.

Antonio Torralba and Alexei A. Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528, Colorado Springs, CO, USA, June 2011. IEEE. ISBN 978-1-4577-0394-2. doi: 10.1109/CVPR.2011.5995347. URL http://ieeexplore.ieee.org/document/5995347/.
Cited on pages 12 and 17.

Valentin Trifonov, Octavian-Eugen Ganea, Anna Potapenko, and Thomas Hofmann. Learning and Evaluating Sparse Interpretable Sentence Embeddings. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 200–210, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5422. URL https://www.aclweb.org/anthology/W18-5422.
Cited on page 30.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6000–6010, Red Hook, NY, USA, December 2017. Curran Associates Inc. ISBN 978-1-5108-6096-4.
Cited on pages 11, 74, 82, 83, 86, 88, 91, 94, and 96.

Sahil Verma, John Dickerson, and Keegan Hines. Counterfactual Explanations for Machine Learning: A Review. *arXiv:2010.10596 [cs, stat]*, October 2020. URL http://arxiv.org/abs/2010.10596.
Cited on page 69.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 12388–12401. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/92650b2e92217715fe312e6fa7b90d82-Paper.pdf.
Cited on page 12.

Ashwin Vijayakumar, Michael Cogswell, Ramprasaath Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search for improved description of complex scenes, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/

AAAI18/paper/view/17329.
Cited on page 91.

Sandra Wachter, Brent Mittelstadt, and Luciano Floridi. Why a Right to Explanation of Automated Decision-Making Does Not Exist in the General Data Protection Regulation. *International Data Privacy Law*, 7(2):76–99, May 2017. ISSN 2044-3994. doi: 10.1093/idpl/ipx005. URL https://doi.org/10.1093/idpl/ipx005.
Cited on page 13.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual Explanations Without Opening the Black Box: Automated Decisions and the GDPR. *Harvard journal of law & technology*, 31:841–887, April 2018. doi: 10.2139/ssrn.3063289.
Cited on page 70.

Marilyn Walker, Jean Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. A Corpus for Research on Deliberation and Debate. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 812–817, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/1078_Paper.pdf.
Cited on page 40.

Eric Wallace, Jens Tuyls, Junlin Wang, Sanjay Subramanian, Matt Gardner, and Sameer Singh. AllenNLP Interpret: A Framework for Explaining Predictions of NLP Models. 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): System Demonstrations*, pages 7–12, Hong Kong, China, November 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D19-3002.
Cited on page 100.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446.
Cited on pages 11, 19, 90, and 91.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://papers.nips.cc/paper/

2019/hash/4496bf24afe7fab6f046bf4923da8de6-Abstract.html.
Cited on page 11.

Shuohang Wang and Jing Jiang. Learning Natural Language Inference with LSTM. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1442–1451, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1170. URL https://aclanthology.org/N16-1170.
Cited on page 86.

Sarah Wiegreffe and Ana Marasović. Teach Me to Explain: A Review of Datasets for Explainable NLP. *arXiv:2102.12060 [cs]*, February 2021. URL http://arxiv.org/abs/2102.12060.
Cited on page 75.

Sarah Wiegreffe and Yuval Pinter. 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)*, pages 11–20, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1002. URL https://www.aclweb.org/anthology/D19-1002.
Cited on page 74.

John Wieting, Taylor Berg-Kirkpatrick, Kevin Gimpel, and Graham Neubig. Beyond BLEU:Training Neural Machine Translation with Semantic Similarity. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4344–4355, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1427. URL https://aclanthology.org/P19-1427.
Cited on page 103.

Adina Williams, Nikita Nangia, and Samuel Bowman. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL https://www.aclweb.org/anthology/N18-1101.
Cited on pages 18 and 41.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain

Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-demos.6. URL https://aclanthology.org/2020.emnlp-demos.6.
Cited on pages 98 and 99.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv:1609.08144 [cs]*, October 2016. URL http://arxiv.org/abs/1609.08144.
Cited on page 88.

Lin Xiao. Dual Averaging Method for Regularized Stochastic Learning and Online Optimization. In *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009. URL https://papers.nips.cc/paper/2009/hash/7cce53cf90577442771720a370c3c723-Abstract.html.
Cited on page 35.

Hu Xu, Bing Liu, Lei Shu, and Philip Yu. BERT Post-Training for Review Reading Comprehension and Aspect-based Sentiment Analysis. 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)*, pages 2324–2335, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1242. URL https://aclanthology.org/N19-1242.
Cited on page 83.

Hongyu Yang, Cynthia Rudin, and Margo Seltzer. Scalable Bayesian rule lists. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 3921–3930, Sydney, NSW, Australia, August 2017. JMLR.org.
Cited on page 43.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 5753–5763. Curran Associates, Inc., 2019. URL http://papers.nips.cc/paper/8812-xlnet-

generalized-autoregressive-pretraining-for-language-understanding.pdf.
Cited on page 51.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014. doi: 10.1162/tacl_a_00166. URL https://aclanthology.org/Q14-1006.
Cited on page 41.

Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Yin and Yang: Balancing and Answering Binary Visual Questions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5014–5022, 2016. URL https://openaccess.thecvf.com/content_cvpr_2016/html/Zhang_Yin_and_Yang_CVPR_2016_paper.html.
Cited on page 18.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level Convolutional Networks for Text Classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://papers.nips.cc/paper/2015/hash/250cf8b51c773f3f8dc8b4be867a9a02-Abstract.html.
Cited on page 24.

Yuan Zhang, Jason Baldridge, and Luheng He. PAWS: Paraphrase Adversaries from Word Scrambling. 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)*, pages 1298–1308, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1131. URL https://aclanthology.org/N19-1131.
Cited on page 18.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 19–27, December 2015. doi: 10.1109/ICCV.2015.11.
Cited on page 90.