Graphs, geometry, and representations for language models and networks of entities

Sammy Khalife

▶ **To cite this version:**

Sammy Khalife. Graphs, geometry, and representations for language models and networks of entities. Discrete Mathematics [cs.DM]. Institut Polytechnique de Paris, 2020. English. NNT : 2020IPPAX055 . tel-03052500

HAL Id: tel-03052500
https://theses.hal.science/tel-03052500

Submitted on 10 Dec 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2020IPPAX055

Thèse de doctorat

Graphes, géométrie et représentations pour le langage et les réseaux d'entités

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Ecole Polytechnique

École doctorale n°626 Institut Polytechnique de Paris (IPP)
Spécialité de doctorat : Mathématiques appliquées, Informatique

SAMMY KHALIFE

Composition du Jury :

Antonio Mucherino Maître de conférences, HDR, Université de Rennes 1, France	Rapporteur
Guillaume Fertin Professeur, Université de Nantes, France	Rapporteur
Michalis Vazirgiannis Professeur, Ecole Polytechnique, France	Directeur de thèse
Nelson Maculan Professeur émérite, Université de Rio de Janeiro	Examinateur
Jean-Marc Steyaert Professeur émérite, Ecole Polytechnique, France	Examinateur
Vasileios Megalooikonomou Professeur, Université de Patras, Grèce	Examinateur
Maximilien Danisch Maître de conférences, Sorbonne Université, France	Examinateur
Catuscia Palamidessi Directrice de recherche, INRIA/Ecole Polytechnique, France	Présidente du Jury

ECOLE POLYTECHNIQUE

LABORATOIRE D'INFORMATIQUE, INSTITUT POLYTECHNIQUE DE PARIS

Graphes, géométrie et représentations pour le langage et les réseaux d'entités

RAPPORT DE THÈSE DE DOCTORAT

SAMMY KHALIFE

Composition du Jury :

Antonio Mucherino Maître de conférences, Université de Rennes 1, France	Rapporteur
Guillaume Fertin Professeur, Université de Nantes, France	Rapporteur
Michalis Vazirgiannis Professeur, Ecole Polytechnique, France	Directeur de thèse
Nelson Maculan Professeur émérite, Université de Rio de Janeiro	Examinateur
Jean-Marc Steyaert Professeur émérite, Ecole Polytechnique, France	Examinateur
Vasileios Megalooikonomou Professeur, Université de Patras, Grèce	Examinateur
Maximilien Danisch Maître de conférences, Sorbonne Université, France	Examinateur
Catuscia Palamidessi Directrice de recherche, INRIA/Ecole Polytechnique, France	Examinatrice

Août 2020

Sammy Khalife: Graphes, géométrie et représentations pour le langage et les réseaux d'entités ©2020

Résumé

Le traitement informatique des objets qui nous entourent, naturels ou créés par l'homme, demande toujours de passer par une phase de traduction en entités traitables par des programmes. Le choix de ces représentations abstraites est toujours crucial pour l'efficacité des traitements et est le terrain d'améliorations constantes. Mais il est un autre aspect émergent : le lien entre l'objet à représenter et "sa" représentation n'est pas forcément bijectif ! Ainsi la nature ambiguë de certaines structures discrètes pose problème pour la modélisation ainsi que le traitement et l'analyse à l'aide d'un programme informatique. Le langage dit "naturel", et sous sa forme en particulier de représentation textuelle, en est un exemple. Le sujet de cette thèse consiste à explorer cette question, que nous étudions à l'aide de méthodes combinatoires et géométriques. Ces méthodes nous permettent de formaliser le problème d'extraction d'information dans des grands réseaux d'entités ainsi que de construire des représentations géométriques utiles pour le traitement du langage naturel.

Dans un premier temps, nous commençons par démontrer des propriétés combinatoires des graphes de séquences intervenant de manière implicite dans les modèles séquentiels. Ces propriétés concernent essentiellement le problème inverse de trouver une séquence représentant un graphe donné. Les algorithmes qui en découlent nous permettent d'effectuer une comparaison expérimentale de différents modèles séquentiels utilisés en modélisation du langage.

Dans un second temps, nous considérons une application pour le problème d'identification d'entités nommées. A la suite d'une revue de solutions récentes, nous proposons une méthode compétitive basée sur la comparaison de structures de graphes de connaissances et moins coûteuse en annotations d'exemples dédiés au problème. Nous établissons également une analyse expérimentale d'influence d'entités à partir de relations capitalistiques. Cette analyse suggère l'élargissement du cadre d'application de l'identification d'entités à des bases de connaissances de natures différentes. Ces solutions sont aujourd'hui utilisées au sein d'une librairie logicielle dans le secteur bancaire.

Ensuite, nous développons une étude géométrique de représentations de mots récemment proposées, au cours de laquelle nous discutons une conjecture

géométrique théoriquement et expérimentalement. Cette étude suggère que les analogies du langage sont difficilement transposables en propriétés géométriques, et nous amène a considérer le paradigme de la géométrie des distances afin de construire de nouvelles représentations.

Enfin, nous proposons une méthodologie basée sur le paradigme de la géométrie des distances afin de construire de nouvelles représentations de mots ou d'entités. Nous proposons des algorithmes de résolution de ce problème à grande échelle, qui nous permettent de construire des représentations interprétables et compétitives en performance pour des tâches extrinsèques. Plus généralement, nous proposons à travers ce paradigme un nouveau cadre et piste d'explorations pour la construction de représentations en apprentissage machine.

Abstract

The automated treatment of familiar objects, either natural or artifacts, always relies on a translation into entities manageable by computer programs. The choice of these abstract representations is always crucial for the efficiency of the treatments and receives the utmost attention from computer scientists and developers. However, another problem rises: the correspondence between the object to be treated and "its" representation is not necessarily one-to-one! Therefore, the ambiguous nature of certain discrete structures is problematic for their modeling as well as their processing and analysis with a program. Natural language, and in particular its textual representation, is an example. The subject of this thesis is to explore this question, which we approach using combinatorial and geometric methods. These methods allow us to address the problem of extracting information from large networks of entities and to construct representations useful for natural language processing.

Firstly, we start by showing combinatorial properties of a family of graphs implicitly involved in sequential models. These properties essentially concern the inverse problem of finding a sequence representing a given graph. The resulting algorithms allow us to carry out an experimental comparison of different sequential models used in language modeling.

Secondly, we consider an application for the problem of identifying named entities. Following a review of recent solutions, we propose a competitive method based on the comparison of knowledge graph structures which is less costly in annotating examples dedicated to the problem. We also establish an experimental analysis of the influence of entities from capital relations. This analysis suggests to broaden the framework for applying the identification of entities to knowledge bases of different natures. These solutions are used today in a software library in the banking sector.

Then, we perform a geometric study of recently proposed representations of words, during which we discuss a geometric conjecture theoretically and experimentally. This study suggests that language analogies are difficult to transpose

into geometric properties, and leads us to consider the paradigm of distance geometry in order to construct new representations.

Finally, we propose a methodology based on the paradigm of distance geometry in order to build new representations of words or entities. We propose algorithms for solving this problem on some large scale instances, which allow us to build interpretable and competitive representations in performance for extrinsic tasks. More generally, we propose through this paradigm a new framework and research leads for the construction of representations in machine learning.

List of Publications

The following publications are included in parts or in an extended version in this thesis:

1. Khalife, S. (2020). Sequence graphs: characterization and counting of admissible elements (Accepted in CTW Graphs and Combinatorial Optimization)

2. Khalife, S. and Vazirgiannis, M. (2019). Scalable graph-based method for individual named entity identification. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 17–25

3. Khalife, S., Read, J., and Vazirgiannis, M. (2019b). Empirical analysis of a global capital-ownership network. In *International Conference on Complex Networks and Their Applications*, pages 656–667. Springer

4. Khalife, S., Liberti, L., and Vazirgiannis, M. (2019a). Geometry and analogies: a study and propagation method for word representations. In *International Conference on Statistical Language and Speech Processing*, pages 100–111. Springer

The following submissions under review are also covered:

5. Khalife, S. and Ponty, Y. (2020). Sequence graphs realizations and ambiguity in language models. HAL open archives (hal-02495333, v2)

6. Khalife, S., Read, J., and Vazirgiannis, M. (2020b). Structure and influence analysis of worldwide capitalistic ownership (Submitted to Journal of Applied Network Science)

7. Khalife, S., Gonçalves, D., and Liberti, L. (2020a). Distance geometry for word embeddings. HAL open archives (hal-02892020, v1)

Furthermore, the following publication was part of my work during my Ph.D. research but are not covered in this thesis:

8. Pallanca, O., Khalife, S., and Read, J. (2018). Detection of sleep spindles in NREM 2 sleep stages: Preliminary study & benchmarking of algorithms. In Zheng, H. J., Callejas, Z., Griol, D., Wang, H., Hu, X., Schmidt, H. H. H. W., Baumbach, J., Dickerson, J., and Zhang, L., editors, *IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2018, Madrid, Spain, December 3-6, 2018*, pages 2652–2655. IEEE Computer Society

Remerciements

Au delà du travail de son auteur, le développement de cette thèse doit autant à l'aide et au support de plusieurs personnes durant de longues années; sa finalisation venant achever cette période d'études fastes en rencontres et recherche scientifique.

Tout d'abord, je tiens exprimer ma sincère gratitude au Professeur Michalis Vazirgiannis pour les conseils et la confiance dans la supervision qu'il m'a accordée pour mener à bien ce sujet de thèse, ainsi que pour les opportunités d'enseignement à l'Ecole Polytechnique mais aussi à l'Université Saint Joseph, à Beyrouth. Ces expériences ont été particulièrement enrichissantes et j'en suis très reconnaissant.

Parmi mes collaborateurs scientifiques, j'aimerais également remercier Jesse Read, Leo Liberti, Douglas Gonçalves et Yann Ponty, pour le travail que nous avons accompli ensemble qui a permis d'explorer des horizons scientifiques passionnants et d'enrichir significativement le contenu de cette thèse. A travers ces rencontres et collaborations, j'ai eu la chance de découvrir et mettre en pratique une variété de méthodes et approches scientifiques. Ces dernières m'ont par ailleurs permis de prendre une peu de hauteur: l'agréable panorama me permet de voir loin, et d'envisager les futurs échanges et travaux avec impatience. Je suis également reconnaissant envers Jean-Marc Steyaert pour le soin qu'il a apporté à la lecture de l'introduction et la conclusion de ce manuscrit.

De plus, je tiens à exprimer ma gratitude aux éminents chercheurs qui ont accepté de faire partie de mon comité de soutenance de thèse de doctorat. En particulier, je tiens à remercier les rapporteurs Antonio Mucherino et Guillaume Fertin qui ont fourni des suggestions et commentaires détaillés qui ont permis de parvenir à une meilleure version de ce texte, ainsi qu'à Nelson Maculan de me faire l'honneur de prendre part au Jury.

Je souhaite également saluer d'autres membres (anciens ou courants) du Laboratoire d'informatique de l'Ecole Polytechnique qui ont contribué à maintenir une agréable atmosphère d'échange et de travail: Kostas Skianis, Stratis Limnos, Giannis Siglidis, Guokan Shang, Changmin Wu, Christos Xypolopoulos, Yang Qiu,

Nikolaos Tziortziotis, Giannis Nikolentsos, Christos Giatsidis, et tant d'autres. Je tiens particulièrement à remercier mon ami et collègue Olivier Pallanca, pour l'aide et soutien réciproque que nous avons maintenu au cours de ces années.

Je remercie les personnes impliquées dans l'organisation de l'ensemble des infrastructures de l'Ecole Polytechnique, notamment ses installations sportives, qui m'ont permis de conserver un équilibre stimulant ces dernières années. Un merci également à l'organisation de son laboratoire d'informatique, à l'INRIA Saclay pour l'accès à certaines ressources, à l'organisation de l'EXED pour les différentes opportunités d'enseignement professionnel, mais aussi à Fabio Roda et Anne Vilnat pour les conseils qu'ils m'ont fourni en début de première année. Je tiens également à exprimer ma gratitude à BNP Paribas et à l'Association nationale de la recherche et de la technologie pour le financement de cette thèse, et en particulier à Geoffrey Scoutheeten pour l'implication et le soutien dont il a fait preuve.

Je profite de cette occasion pour rendre hommage aux enseignants qui ont contribué de manière fondamentale à mon parcours scientifique et professionnel jusqu'ici. Je pense en particulier à Alain Juhel et Jean Voedts, ex-enseignants en mathématiques spéciales du Lycée FAIDHERBE de LILLE - grâce à qui j'ai pu cultiver une passion et un idéal non seulement pour les mathématiques, mais pour la Science en général. Je pense également aux nombreux enseignants-chercheurs de l'Unité de Mathématiques Appliquées (UMA) de l'ENSTA Paristech, et en particulier à Pierre Carpentier, dont les cours qui, au travers d'un article de recherche, m'ont interpellé sur les riches et passionnantes interactions entre mathématiques appliquées et informatique.

Finalement, mes pensées vont à ma famille et mes proches pour leur amour et soutien pendant ces années. Sans eux, je n'aurais probablement pas poursuivi cette voie, ou du moins, pas de cette manière: qu'il en soient ici profondément remerciés.

Sceaux, Printemps 2020
Sammy KHALIFE

Acknowledgments

The development of this thesis was also fueled by the presence, help and support of several people the last years. Its ends concludes a rich journey on both personal and scientific aspects.

First, I would like to thank Professor Michalis Vazirgiannis for his supervision, his advice and the confidence he has placed in me to carry out this thesis subject, as well as the teaching opportunities at the École Polytechnique but also at Université Saint-Joseph, in Beirut. These experiences have been particularly enriching and I am very grateful.

Among my scientific collaborators, I also wish to thank Jesse Read, Leo Liberti, Douglas Gonçalves and Yann Ponty, for the work that we have accomplished together which has made it possible to explore fascinating scientific horizons and to significantly enrich the content of this thesis. The variety of the topics and related scientific questions offered me a very pleasant overview and confidence in future collaborations. I am also thankful to Jean-Marc Steyaert for the care he took in reading the introduction and the conclusion of this manuscript.

Furthermore, I want to express my gratitude to the distinguished researchers who accepted to be part of my PhD thesis defense committee. In particular, I want to thank the reviewers Antonio Mucherino and Guillaume Fertin who provided detailed and insightful comments which helped to reach a better version of this monograph, as well as to Nelson Maculan for doing me the honor of taking part to the Jury.

I would like to greet other colleagues and members (former or current) of the *Laboratoire d'informatique* of Ecole Polytechnique who have contributed to maintaining a pleasant atmosphere of exchange and work: Kostas Skianis, Stratis Limnos, Giannis Siglidis, Guokan Shang, Changmin Wu, Christos Xypolopoulos, Yang Qiu, Nikolaos Tziortziotis, Giannis Nikolentsos, Christos Giatsidis, and many others. I would particularly like to thank my friend and colleague Olivier Pallanca, for the mutual help and support that we have maintained over these years.

I would like to thank the people involved in the organization of all of the Ecole Polytechnique's infrastructure, including its sports facilities, which have enabled me to maintain a stimulating balance in recent years. Thanks also to the organization of its computer laboratory, to INRIA Saclay for resources access, to the organization of EXED for the various professional education opportunities, but also to Fabio Roda and Anne Vilnat for the advice they gave me at the start of the first year. I would also like to express my gratitude to BNP Paribas and the National Research and Technology Association for funding this thesis, and in particular to Geoffrey Scoutheeten for the involvement and support he has shown.

I take this opportunity to pay tribute to the teachers who have contributed fundamentally to my scientific and professional journey so far. I think in particular of Alain Juhel and Jean Voedts, ex-teachers in special mathematics of *Lycée FAIDHERBE* in Lille - thanks to whom I was able to cultivate a passion and an ideal not only for mathematics but for Science in general. I also think to all teacher-researchers of the *Unité de Mathématiques Appliquées (UMA)* at ENSTA Paristech, in particular to Pierre Carpentier, whose course shed light on rich and exciting interactions between applied mathematics and computer science.

Ultimately, my thoughts are with my family, friends and loved ones for their presence and support. Without them, I probably would not have followed this path. Thank you from the bottom of my heart.

Sceaux, Spring 2020
Sammy KHALIFE

Contents

List of Figures

List of Tables

1 Introduction

1.1 A bit of History: Computer science and Language

1.1.1 Formal languages

The concept of language and its study as a mathematical object has a rich scientific history developed in the second part of the 20^{th} century, which formalized the initial idea of the grammarian Panini (around 5th century B.C, for a relatively recent reference, we refer to (Cardona, 1997)). This formalization is based on the founding idea that language must respect structural rules described by a formal grammar. Given a set of symbols, usually containing a start symbol S (and possibly a stop symbol), the language is constructed by applying different production rules recursively to an initial sequence of symbols. Let us consider the following example. Given the vocabulary $\Sigma = \{a, b\}$ and start symbol S, let $\mathbb{N}^*$ be the set of positive natural integers and let us consider the formal rules:

$$\begin{aligned} &1. \quad S \to aSb \\ &2. \quad S \to b \\ &3. \quad a^k S b^k \to a^k b a^k \quad \forall k \in \mathbb{N}^* \end{aligned} \tag{1.1}$$

where the operator $\to$ consists in replacing the left term (called) *nonterminal symbol* within a string, and Σ are the *terminal* symbols, which cannot be input of the

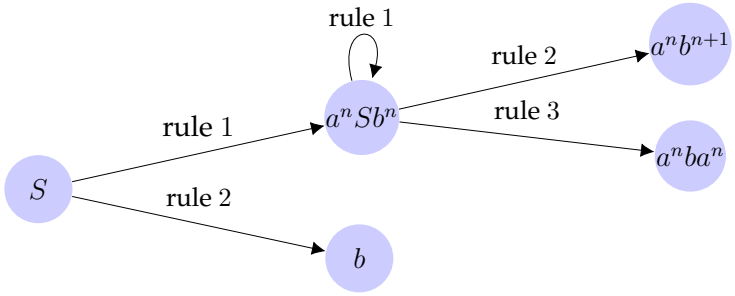


Figure 1.1 – Generation of the formal language defined with the set of rules described in Equations 1.1, where n is an strictly positive integer describing $\mathbb{N}^*$; namely the node $a^n b a^n$ corresponds to the set of strings $\{a^n b a^n \mid n \in \mathbb{N}^*\}$

operator $\rightarrow$. Here a^n is the n-concatenation of a. Then, the construction rules must be applied as follows: starting with S, the choice of rule leads to a path possible strings illustrated in Figure 1.1. By immediate recursion, the language defined by Equation 1.1 is exactly $\{b, a^n b^{n+1}, a^n b a^n \mid n \in \mathbb{N}^*\}$ (by convention, we do not include the strings containing the start symbol S because it is not in the alphabet). The set of production rules contained a formal grammar thereby defines a language, which are all the set of strings composed of symbols in the alphabet reachable from the start symbol and using the formal rules.

Based on this formulation, fertile theoretical results have been established, such as the classification of Chomsky ((Chomsky, 1956), cf. Fig. 1.2), which establishes a hierarchy of families of formal languages based on the characteristics of the grammars defining them or, equivalently, of the automata recognizing them.

For example, let us consider the *left regular* grammars. If A is a nonterminal symbol, and a, b are terminal symbols (hence a, $b \in \Sigma$), and ϵ is the empty string, then the corresponding derivation rules have the following form:

$$\begin{aligned} &1. \quad A \rightarrow a \\ &2. \quad A \rightarrow Ba \\ &3. \quad A \rightarrow \epsilon \end{aligned} \tag{1.2}$$

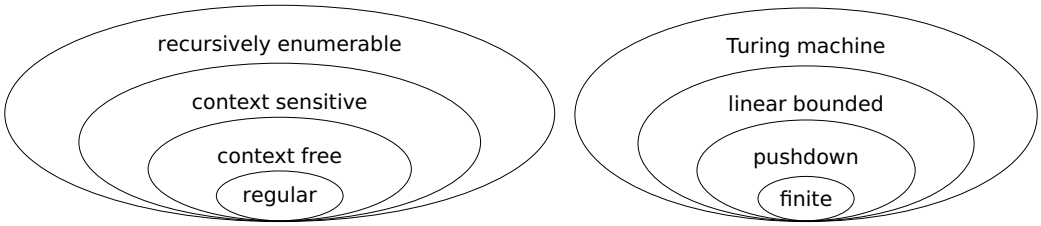
If the rule $A \rightarrow Ba$ is replaced by $A \rightarrow aB$, then the grammar is *right regular*. In the standard terminology, the most simple group of grammars, called *regular* are composed of the *left and right regular grammars*. Besides, a *finite automaton*, which models a certain form of sequential derivations, is defined as $(Q, \Sigma, \delta, q_0, F)$ where:

- Q is a finite set of states
- Σ is a finite set of input symbols (alphabet)
- δ is a transition function: $\delta : Q \times \Sigma \rightarrow Q$
- q_0 is an initial state
- F is a set of accept states

A finite automaton is said to recognize a string s if a sequence of states, $r_0, r_1, \cdots, r_n$ exist in Q such that r_0 is a initial state, $r_n \in F$ and $r_{i+1} = \delta(r_i, a_{i+1})$ Then, it can be proved that finite automaton exactly recognizes all strings generated by *regular* grammars. For instance, the formal language in Figure 1.1 is not *regular* (and actually not even *context-free*) since the derivation rules are not of the form of those in Equations 1.2 (or those of a *right regular* grammar). Thus, it cannot be recognized by a finite automaton.

The other main types of formal grammars are *contex-free*, *contex-sensitive* and *recursively enumerable*; and each of those can also be associated to specific type of automaton (respectively *pushdown*, *linear bounded* and *Turing machine*) as depicted in Figure 1.2. We refer our reader to (Eilenberg, 1974) for the mathematical foundations of Automata theory, and (Hopcroft et al., 2001) for a general introduction and

relations with languages. This classification is a fundamental result of language theory, and has lots of theoretical and practical implications in complexity and learning theories, but also for the design of programming language and compiling systems.



(a) Formal grammar (b) Type of automaton

Figure 1.2 – Chomksy hierarchy and Grammar / Automaton correspondence

Historically, a subfield of combinatorics emerged during the same period, and studies the properties of sequences formed on a finite alphabet X. Axel Thue, in the early 1900s, was one of the first interested in such questions, among which the existence of factorization of words (Thue, 1912). Such ideas were rediscovered later on, for special cases, including the study of square-free sequences, which are not of the form $f\,f$ where f is a pattern built on the alphabet X. This field also led to developments in discrete algebra; related conjectures on the repetitions of patterns were formulated in 1972 by (Dejean, 1972), and proven in the beginning of the 20^{th} century (Rao, 2011; Currie and Rampersad, 2011), with Dejean's Theorem. We refer to (Berstel and Perrin, 2007) for a thorough description of the origins of this field.

More recently, important progress has been achieved both in hardware and software engineering in the second part of the 20^{th} century, allowing to perform operations on very large databases. In this context, massive network infrastructures were developed, along with the indexing of a large amount of web pages, giving birth to the irresistible search engines in the end of the 20^{th} century. However, first attempts to model natural language using simple vector space models go back to the 1970s, namely Index terms (Salton et al., 1975), term frequency inverse document frequency (TF-IDF) (Ramos et al., 2003), and corresponding software solutions SMART (Salton, 1971b), Lucène (Hatcher and Gospodnetic, 2004).

In the late 2000s and beginning of the 2010s, the computing power associated with large data volume collection encouraged the intensive (and in some cases, ill-advised) use of Machine learning techniques to perform tasks of natural language such as text classification or language translation. Despite significant experimental progress, especially concerning information retrieval problems and machine translation, both fundamental understanding of natural language and designing interpretable models in the applications remain largely opened.

1.1.2 Formal Language, Natural Language and Artificial Intelligence

Given the theoretical and practical results concerning formal languages, a legitimate question to consider is whether natural language can be described using Chomksy's formalism. Several attempts of formalization have been developed in the literature. (Kornai, 1985) presented three independent arguments that natural language stringsets can be considered as regular (type 3 in Chomsky classification), if arbitrary long "sequences" can be considered.

However, it seems that the formalization of the grammar of a natural language such as English (i.e not only considering stringsets nor semantics) is not Context-free (Shieber, 1985) (i.e not even type 2 in Chomsky classification). However such statements have been nuanced in (Gross, 1981), due to the fact that counter-examples are relatively rare although significant. Even a context-free grammar has been proposed for French (Salkoff, 1980). Here, we will not develop the formalizations or proofs of such statements, the point being that formalization of natural language is now rather unsuccessful, for instance in the objective to construct machines able to hold a conversation as a human would. We will rather adopt and reformulate a distinction already discussed namely in (Thom, 1970). A total formalization of natural languages seems to be excluded for the following reasons:

- If a simultaneous total formalization of the given language, but also of the meta-language (i.e the semantics) required to express the theory was possible, the paradoxes that prohibit formalization of global arithmetic would appear. The sentence: "The sentence that I write right now is not well formed" would be contradictory.

- If we only impose the formalization of the language but not the semantics (meta-language), any formalization of natural languages contains "self-filling" axioms which increase the length of well-formed expressions. For instance, if "E" is a well-formed expression, the expression "I" say that "E" is also well formed. This leads to consider "well-formed" expressions as long as we want. However, it is accepted, even for skilfull authors with very long sentences (e.g Marcel Proust), that there exists an upper bound on the length of the sentences. In a standard formal system, there are no restrictions as to the possibility of using the axioms as often as you like, which would yield a contradiction. Therefore, any attempt to explain the linguistic form must necessarily have a dynamic aspect which makes counts, given a sentence, for its creation as a process that ensures its grammatical correction. However, all of these processes are subject to psycho-physiologic constraints which limit the number and the relative disposition of axioms. In other terms, axioms

cannot be used infinitely. For this matter, Thom (1970) uses the wonderful expression: "Such axioms would get tired when we use them."

- The notion of "well-formed" expression in a natural language is not absolute: there is a practically continuous graduation of non-grammaticality or semantically unacceptable sentences. It should also be noted that any strict border between non-grammaticality and semantic unacceptability is arbitrary.

We see that the difficulty lies mainly in the semantics, which cannot be decuced trivially from syntax. However, the current formal theories (e.g.generative grammars) have their own mathematical interest, and have undeniable value in terms of a local description of the forms language. They have revealed, in particular, universal aspects of formal mechanisms found among all human languages (Everaert et al., 2015). Other empirical approaches highlighted such universal similarities (del Prado, 2011).

It is also worth to note that this development is related to Alan Turing's original question of artificial intelligence (Turing, 1950) "Can machine think?" If natural language, among other dimensions of human activity, cannot be formalized totally, should we expect automata to perform? In his original paper, an argument going in favor of a positive answer, is that although there are formal limitations to the powers of any particular machine, it has been implicitly considered that: "without any sort of proof, that no such limitations apply to the human intellect". However, the difficulty may lie in the fact that human capacity to introduce new ideas or reasonings is not bound to any type of formal system: what we usually call intuition escapes formal law of logic! Therefore, the design of learning machines obeys rules that our brain is not - at least in appearance - tight to.

1.1.3 "Learning" Machines

Modern artificial intelligence is interested in - among others - algorithms performing empirically well for end-tasks (e.g. image or text classification) with respect to a ground truth. In this regard, a *regressor* is a function $f : X \to Y$, that maps observations $X_1, ..., X_n$ to respective observations $Y_1, ..., Y_n$. A *regressor* is said to be a *classifier* where Y is discrete (usually finite), for which the most simple example is *binary classification*. This situation is commonly referred to as *supervised learning*. Also, regressors are usually defined as a family of functions indexed by a set a parameters. When the set of parameters is finite, the situations is referred to as *parametric* learning, and in the other case, *non-parametric* (which can be misleading). Finally, if empirical observations of Y cannot be used (or are not available) but still exist in the model, the framework is called *unsupervised learning*.

The learning phase consists in computing a function that could map as well as possible the observations from X to their corresponding value in Y. Supposing

that f belongs to a set of functions indexed by a set of parameters, it corresponds to finding the best parameters for this problem. Then, inference for a new point X_{n+1} is computed as $f(X_{n+1})$. There exists a large family of these algorithms, called learning algorithms. The theoretical and empirical studies of this branch of algorithms form the field of Machine Learning.

The Theory of Learnability is concerned with the formalization of learnability for programs in terms of complexity (Valiant, 1984; Schapire, 1990). This formalism has namely led to demonstrate that regular and context free languages (in the sense of Chomsky) cannot be learnt, which means that there is no polynomial time algorithm executable on a Turing machine that would allow to approximate any task performed by an Oracle (e.g. Human), when given examples of inputs and outputs. Does this mean that we should not expect a machine to perform well for natural language (since natural language is not even context-free)? The answer to this question is not clear, due to the very general nature of the original statement. Indeed, a possible weaker statement of this result would be: learning to approximately solve all natural language processing related problems is NP-hard. However, when considered isolated natural language problems, the unknown still remains.

Statistical learning theory is a subfield of machine learning studying the question of learning and prediction from a statistical point of view. In particular, the relation between empirical risk minimization (find an optimal set of parameters to minimize the error rate) and generalization, which is the capacity of a learning algorithm to "correctly" predict or classify new samples. Moreover, several concepts developed in this field, such as the Vapnik–Chervonenkis dimension (Vapnik, 1995), have given tools useful for complexity theory.

In this context, Deep Neural Networks, which can be defined as a certain family of regressors inspired from the humain brain (LeCun et al., 2015), achieved remarkable experimental results in the beginning of the 2010s. They give state-of-the-art performances for several problems, including image classification with thousands of complex classes (Krizhevsky et al., 2012) (with a significant margin), but also speech recognition (Hinton et al., 2012), biomedical applications (Leung et al., 2014). They also provide competitive or state-of-the-art performance in natural language understanding (Sutskever et al., 2014) but with a non significant margin. Despite research on the mathematical framework to analyse their properties (Mallat, 2016), many of these remain unknown, both concerning their numerical analysis (such as guaranteed convergence of optimization algorithms), or approximation theorems, which are necessary to fully understand the mathematics of these networks. Also, little to none of their generating mechanisms is fully understood: this is essentially due to the fact that these approaches are essentially statistic and not causal.

Besides, the question of representation learning, originally refered to as feature learning, is concerned with the problem of automatically constructing a pertinent representation for learning algorithms, which would ideally replace manual feature engineering, or be computed in the same time as the learning procedure.

Several approaches have been proposed for this matter, such as greedy iterative methods (Hinton et al., 2006), auto-encoders (Ballard, 1987), manifold learning; or sparse coding. For more details on the subject, we refer to the survey (Bengio et al., 2013). One of the challenges of representation learning is the difficulty in establishing a clear objective for training. In the case of standard end-tasks, such as classification, the objective is very often to minimize the number of misclassifications on the training dataset. In the case of representation learning, the objective is a priori removed from the ultimate objective, which is to learn a predictor.

Representation learning is also concerned with the curse of dimensionality, an important bottleneck in statistical learning, which implies that the number of samples to reach convergence is exponential with regards to the dimension. Indeed, images, but also speech and text data and are naturally represented high dimensional spaces if taken as such. Theoretical results concerning representations in machine learning are somehow limited; indeed, such representations are rather prefered empirically given an end-task.

1.2 Examples of Related Problems

Among the many artificial intelligence problems, there exist a plethora of natural language processing and information retrieval tasks. This thesis is obviously not intended to address each of these, but some of them are here addressed and/or used as applications. These problems do not have an empty intersection, and many of them are closely related. In some cases, reductions exist from one to another. Also, it appears than "performing well" empirically for some of them is *a priori* more difficult than others. Due to crucial role of semantics in natural language, it is however very difficult to establish a strict hierarchy or separation of problems. However, we think it is wise to group them as follows. This list might is not exhaustive, but we believe it contains central problems in natural language processing:

- Automatic speech recognition (audio), Speech-to-text, Text-to-speech
- Combinatory Categorical Grammar, Constituency or dependency parsing (extract a tree from a sentence that represents its syntactic structure).
- Part-of-speech tagging or named entity recognition.

- Text classification: attribute a label to a sentence or document a label (**adressed in this thesis**)
- Information extraction, knowledge base completion, Coreference resolution, Word sense disambiguation, Entity identification (**adressed in this thesis**)
- Grammatical error correction
- Machine translation, Lexical normalization (translating/transforming a non standard text to a standard register)
- Multi-modal emotion recognition, Sentiment analysis
- Relationship extraction between named entities (e.g. relations between characters in a novel).
- Question answering, Semantic textual similarity, Common sense, Natural language inference: determining whether a "hypothesis" is true (entailment), false (contradiction), or undetermined (neutral) given a "premise".
- Summarization, Simplification (modifying the content and structure of a text in order to make it easier to read and understand)
- Intent Detection and Slot Filling: interpreting user commands/queries by extracting the intent and the relevant slots.
- Taxonomy learning (hierarchically classify concepts from text)

1.3 Overview of the Contributions

As discussed in the previous Section, significant experimental progress has been obtained with models such as deep neural networks for a plethora of applications including natural language processing. However, several practical issues remain such as domain adaptation (i.e the capacity to transfer knowledge from some task or dataset to another), combined with the need for large annotated data volume.

Therefore, we investigate a potential weaknesses of some “standard” sequential models. To this end, we conduct in Chapter 3 a theoretical analysis of the ambiguity for a family of language models, refered to as context-based language models, or co-occurence based models.

However, in Chapter 4, we show that these models allow to design practical and scalable algorithms which can challenge deep-learning on specific tasks, and which would require relatively few annotated samples to reach good experimental performance. This led us to show that based on a adequate formulation of the

problem of entity identification - which consists in mapping text entities to entries in a database similarly to a search engine-, standard graph-based models have better of competitite performance requiring fewer annotated samples.

These context based models have also been used in order to output word or sentence vector representations which have played a major role in artificial intelligence and natural language processing. For some of these representations, a geometrical conjecture connecting word vectors and the notion of semantic analogy was formulated, which we analyze in Chapter 5. We also provide an analysis of a partition function of word vectors.

Finally, these geometric properties lead us to consider other geometric methods to generate word embeddings, and in particular within the framework of distance geometry in Chapter 6.

The next subsections sum up our contributions. Their purpose is to provide the reader with the general idea of each chapter and how they articulate with each other. The background notions and terminology necessary to understand their content will be presented in Chapter 2.

1.3.1 Sequence Graphs and Ambiguity in Language Models: a Combinatorial Study

The intent of several natural language models is to extract the semantic information contained in a sentence or a document. Popular methods to achieve this objective is to consider the co-occurences of words within context of size w (Mikolov et al., 2013a; Pennington et al., 2014; Arora et al., 2016b), which can be represented as a weighted graph (which we refer to as *sequence graph*, or a *graph of words*). However, these representations induce a level of ambiguity, as displayed in Figure 1.3.

(a) No ambiguity ($w = 3$) (b) Ambiguity ($w = 2$)

Figure 1.3 – The two sequence graphs built for the sentence "To be or not to be" using window sizes 3 (a) and 2 respectively (b). In the second case, the information to reconstruct the sentence is insufficient, hence creating ambiguity, since any circular permutation of the words admits the same representation. Chapter 3 explores this question in generality.

In Chapter 3, we are interested in questions related to the level of ambiguity generated with these models. In particular, we define the notion of sequence graph, a combinatorial object encoding their information. Two main theoretical combinatorial properties are investigated, in particular the existence of a sequence

given an arbitrary graph, and the number of possible sequence (i.e sentences or documents) from which a given graph can originate from.

We also show an experimental comparison between these context-based models and recent sequential derived from recurrent neural networks, which by construction do not share this property.

1.3.2 Entity Identification and Analysis of a Knowledge Graph

Despite the form of ambiguity induced by co-occurence based models as discussed in Chapter 3, we will show in Chapter 4 they can be very efficient for information retrieval problems.

The structure of a document can be analysed under the prism of the role of central entities in the text, motivating the question of identifying them automatically. In the context of natural language processing, the definition of an entity is not universal but will be properly defined in Chapter 2 and Chapter 4. In this Section, let us consider that an entity is a real-world object and usually has a physical existence, and usually denoted with a proper name. The problem of Entity discovery consists in detecting these entities and identifying them within a database. For instance, in the sentence:

John Kennedy was the president of the *United States*.

A named entity recognition program should be able to label *John Kennedy* and *United States* as entities (and the remaining words as non-entities), and classify the type of entities within categories such as *Person*, *Place*, or *Organisation*. The process of identification, which we will be interested in Chapter 4 consists in identifying the pertinent corresponding entry in a database. These two problems are interesting in the sense thay they allow to create a structure from a natural language document, and thereby helping to contribute to its automatic treatment.

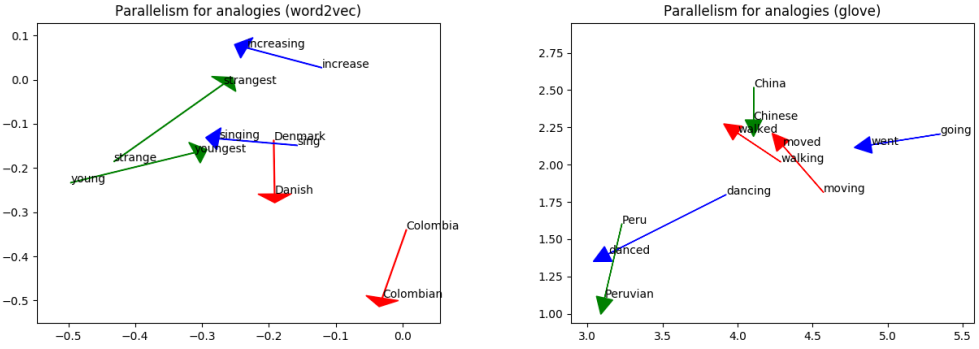
More precisely, in the frame of Chapter 4, this "database" is represented by a *Knowledge base*, or *Knowledge graph*. A *Knowledge graph* is a form of relational database providing supplementary descriptive and semantic information about entities. The semantic information is contained in a graph structure, where a node represents an entity, and an edge represents a semantic relation. Each node of the graph potentially contains metadata, such as text description or classification via an *Ontology*.

Therefore, in Chapter 4, we suppose that the identification of entities within text documents always relies on the existence of background knowledge. We also provide in Section 8.2 of the Appendix an analysis of a knowledge graph constructed from economic and financial data.

1.3.3 Study of a Geometrical Conjecture and Partition Function Property for Word Vectors

Co-occurence based models are also used to generate word or sentence representations in vectorial form. Indeed, the majority of machine learning algorithms take vectorial input, whereas the native structure of text are discrete sequences. Several procedure exist to generate word embeddings, which we present in Chapter 2.

In Chapter 5, we will discuss several properties of theses representations. First, we discuss a geometrical conjecture relating semantic analogies and word vectors.



(a) word2vec (b) glove

Figure 1.4 – Example of a geometric property of word vectors for analogies (the x and y axis represent the coordinates of the word vectors, and each colour correspond to a quadruplet of words implied in an analogy), discussed in Chapter 5.

Besides, we discuss a probabilistic property for word vectors, named the concentration the partition function. This property was presented in previous work in a model for word vectors (Arora et al., 2016a), where the text generation process is driven by a random walk $(c_t \mid 1 \leq t \leq T)$, representing a latent discourse vector. If w_t is the word at step t, c_t the discourse vector, and v_w vector of word w then this model supposes the following conditional model:

$$\mathsf{P}(w_t = w|c_t) \propto \exp(\langle c_t, v_w \rangle)$$

And the partition function Z_c, similarly to statistical physics, is computed as a sum of states $Z_c = \sum_v exp(\langle v, c \rangle)$. Then, under some assumptions, it can be proven that this partition function concentrates around its mean with high probability, and was suggested that it played an important role in the co-occurrence based models since it allows to derive formal relations between pointwise mutual information (PMI), which is pure statistical information based of co-occurrences of words, and

the scalar product of word vectors, which is a geometric quantity. By weakening the assumptions of the generative model, in particular that the words vectors are generated uniformly and independently, and that the discourse vector are close to a sphere of radius $R \leq 2$, we show that a similar concentration phenomenon happens, suggesting this property is not specific to these assumptions.

1.3.4 Efficient Representations with Distance Geometry

Traditional methods to construct word representations are the output of optimization schemes solved by Stochastic Gradient Descent (for a detailed explanation of this notion, cf. Chapter 2) .

In Chapter 6, we propose a new word embedding method based on the Distance Geometry Problem (DGP), whose aim is to find object positions based on a subset of their pairwise distances. Considering the empirical Pointwise Mutual Information (PMI) as an inner product approximation, we discuss two algorithms to obtain approximate solutions of the underlying Euclidean DGP on large instances. The resulting algorithms are considerably faster than state-of-the-art algorithms, with similar performance for classification tasks. The main advantage of our approach for practical use is its significantly lower computational complexity, which allows us to train representations much faster with a negligible quality loss, a useful property for domain specific corpora.

1.3.5 Software

The code implementation of the algorithms and experiments presented in this dissertation are available (links in the corresponding chapters), except the third one for confidentiality reasons:

- *Sequence graph recognition and counting algorithms: Direct approach, dynamic programming and linear integer programming formulations* (Matlab and Python)
- *Scalable graph based method for named entity identification* (Python)
- *Analysis of a economic and financial knowledge graph* (Python).
- *Geometry and Analogies: A Study and Propagation Method for Word Representations* (Python)
- *Distance geometry for word representations (Matlab and Python)*

2

Background Notions and Terminology

In this chapter we present the background material and terminology that will be used throughout the different chapters of this dissertation. Special mention is given to notions in Graph Theory, Combinatorics, Complexity theory. Some basic notions of Analysis and Linear Algebra are also presented.

2.1 Graph Theory and Combinatorics

Graphs, which are well-studied structures, are utilized to model different types of entities and their relationships. Graph-based representations have become ubiquitous in many application domains. For instance, social networks, protein and gene regulatory networks, and textual documents are commonly represented as graphs. Furthermore, in the past years, graph classification has arisen as an important topic in many domains such as in Computational Biology (Schölkopf et al., 2004), in Chemistry (Mahé and Vert, 2009), where one wants to predict the mutagenicity of a chemical compound by comparing its graph representation with other compounds of known functionality. These representations have also been used successfully for document similarity (Nikolentzos et al., 2017a).

Besides, in this dissertation we will be interested in estimating and demonstrating several properties of computational compexity, especially in Chapter 3 and Chapter 6. To this end, in the remaining of this subsection, we present the useful terminology and notations employed from graph theory and complexity theory.

- A *graph* G is a couple (V, E) where V is a discrete set of elements called *nodes*, or *vertices*, and $E \subset \{\{x, y\} \mid (x, y) \in V^2\}$ a set of *edges*. Similarly, a *multigraph* is a couple (V, E) where parallel edges are authorized.

- An *directed graph*, or *digraph*, is a graph where the edges are oriented, i.e $E \subset \{(x, y) \mid (x, y) \in V^2\}$.

- A *strongly connected component* of a digraph is a maximum set of vertices (in terms of inclusion) such that there exists a path between each pair of vertices

in S. A digraph is strongly connected if it has only one strongly connected component (equal to itself). For undirected graphs, such component is simply said to be *connected*.

- A *path* (or a walk) is a finite or infinite sequence of edges which joins a sequence of vertices. A *cricuit* is a path starting and ending at the same vertex. A *trail* is a path without repeated edges. A *cycle* is a trail starting and ending at the same vertex.

- A *Eulerian path* of a graph is a path that visits every of its edges exactly once. A graph is said to be *semi-Eulerian* if it has a Eulerian path, and *Eulerian* if it has a Eulerian cycle.

- A *Hamiltonian path* of graph is a path that visits every of its vertices exactly once. A graph that contains a Hamiltonian cycle is called a *Hamiltonian* graph.

- A *tree* is an undirected graph in which any two vertices are connected by exactly one path, or equivalently a connected acyclic undirected graph.

- A *directed tree* (or *polytree*) is a directed acyclic graph (DAG) whose underlying undirected graph is a tree.

- A *forest* is an acyclic undirected graph, or equivalently a disjoint union of trees. Similarly, a *directed forest* (or *polyforest*) is a directed acyclic graph whose underlying undirected graph is a forest.

- A *vertex order* of a graph with n vertices is a bijective correspondence between its vertices and $\{1, ..., n\}$ (equivalently, it is a permutation of its vertices).

- Let G be a graph, and let H be a *subgraph* of G, i.e., $H \subset G$. By definition a subgraph H is defined to be a *k-core* of G, denoted by G_k, if it is a maximal connected subgraph of G in which all nodes have degree at least k.

- The *degeneracy* of a graph G is defined as the maximum integer k for which graph G contains a non-empty k-core subgraph.

- A node i of a graph G has *core number* k, if it belongs to a k-core of G but not to any of its $(k+1)$-core.

- For $(k, n) \in \mathbb{N}^2$ and $n \geq k$, $\binom{n}{k}$ is the standard *binomial coefficient* given by $\binom{n}{k} = \frac{n!}{k!(n-k)!}$

- $\mathfrak{S}_n$ represents the *set of permutations* of $\{1, ..., n\}$.

- Card S, $\#S$ or $|S|$ represent the *cardinal* of a set S.
- If X and Y are two sets, $X \simeq Y$ means there exists a *bijection* $f : X \longrightarrow Y$.

2.2 Complexity Theory

In several parts of this dissertation, especially in Chapter 3, 4 and 6, we will be interested in the complexity analysis of certain algorithms. In this section we remind some related notions and terminology. For a general introduction and development to the field of Complexity Theory, the reader may refer to Refs. Papadimitriou (2003); Arora and Barak (2009).

- The *computational complexity* of an algorithm is a measure of the number of the number of elementary operations in order to execute the said algorithm. An elementary operation is any one of the arithmetic operations (addition, subtraction, multiplication, division) or a comparison between two numbers, or the execution of a branching instruction (e.g inequality testing).
- $f = O(g)$ (resp. $f = \tilde{O}(g)$) means that f is *upper bounded* by g (resp. upper bounded ignoring logarithmic factors) in the neighborhood considered. For instance, if $f(n)$ representing the complexity of an algorithm depending on the size of the input represented by n, $f(n) = O(\log n)$ means that there exists a constant C such that $f(n) \leq C \log n$
- We make use of standard list of logic symbols. In particular, $\wedge$ and $\vee$ represent respectively the *logical conjunction* and *logical (inclusive) disjunction*.
- A *decision problem* is a mathematical problem whose answer is Boolean.
- A *reduction of a decision problem* is a transformation of its variable inputs and parameters into a new decision problem such that finding the solution of the original problem is equivalent to find the solution of the new problem. In other terms, the function f used for the reduction maps the inputs of L having *True* output, to inputs of L' having *True* output, and *False* inputs of L to *False* inputs of L'. Therefore, if L (resp. L') represents the set of inputs of the original problem (resp. the new problem) having *True* output, then $x \in L \iff f(x) \in L'$. From a complexity theory point of view, different types of reductions can be considered. However, in this dissertation, we will implicitly consider only the most standard types of reductions, often referred to as *Karp* reductions. A Karp reduction uses a polynomial time computable function to reduce one problem to another; which means only a polynomial number of operations are required for the transformation f, with respect to

the size of the input of the initial problem. This fundamental property of Karp reductions implies that they preserve the main classes of complexities defined below.

- Based on the notion of reduction, we remind the standard definitions of P/NP and {NP, #P}-completeness and hardness:
  - A decision problem is in P if it can be solved by a deterministic Turing machine in polynomial time.
  - A decision problem is in NP if a given solution to the problem can be verified to be correct by a deterministic Turing machine in polynomial time.
  - A decision problem is NP-hard if all the problems of NP can be reduced to it after a polynomial reduction.
  - A decision problem is NP-complete if it is NP and NP-hard.
  - #P is the set of the counting problems associated with the decision problems in NP. The definitions of #P-hardness/completeness are defined as in the two previous points replacing NP by #P.

  Finally, an optimisation problem $\min_{x \in X} f(x)$ is said to belong to one of these classes when its following decisional formulation

$$\text{Given } x \in X \quad \exists ?\ y \in X \quad \text{such that} \quad f(y) < f(x)$$

  verifies the class definition.

2.3 Distance Geometry

In the historical context described in Section 1.1, several common tasks in data science rely explicitly or implicitly on distances between entities, and many of the designed algorithms take input in vectorial form. Distance geometry studies the properties of points given some of their relative distances. It has appeared under several forms over centuries but its formalization is relatively recent, in the end of the 19^{th} century. In the general theory, the considered space can be a metric space, or a differential manifold, but in this dissertation we consider the Euclidean Distance Geometry Problem, whose formulation takes place in a vector space of finite dimension, where $||\cdot||$ is a Euclidean norm:

> Distance Geometry Problem (DGP). Given an integer $K > 0$, and a simple undirected graph $G = (V, E)$ whose edges are weighted by a positive function d : $E \to \mathbb{R}_+$, determine whether there is a function x :

$V \to \mathbb{R}^K$ such that:

$$\forall \{u, v\} \in E, \ ||x(u) - x(v)|| = d(\{u, v\}) \tag{2.1}$$

In the 1950s, Blumenthal considered the fundamental problem of distance geometry, which he called the "subset problem" (Blumenthal, 1970) [Ch. IV §36, p.91], i.e. finding necessary and sufficient conditions to decide whether a given matrix is a distance matrix. The first results concerning these problems were obtained by Cayley (Cayley, 1841), and then Menger (Menger, 1931) with sufficient conditions that all $(K + 3) \times (K + 3)$ square submatrices of the given matrix are distance matrices (see (Blumenthal, 1970) [Thm. 38.1]. In the last part of this thesis, we will explore the paradigm of Euclidean distance geometry in order to construct vectorial representations. This idea originated from the observation that distances between objects are often available, whereas machine learning algorithms usually take vectorial representations as inputs. Supplementary notions and results in distance geometry will be adressed in Chapter 6. For a survey on Euclidian distance geometry, we refer to Liberti et al. (2014).

We also define the notion of lateration for a vertex order which will be useful, especially in Chapter 3 and 6 as follows:

(K+1)-Lateration Order. Let K be a integer > 0, and let G be a graph. A vertex order of G is called $(K + 1)$-lateration order if:

1. the first $K + 1$ vertices form a clique;
2. for every vertex $i > K + 1$, vertex i has at least $K + 1$ adjacent predecessors.

Finally, we remind the reader some general notions in global optimization useful for Chapter 5. Let K and K' be stricly positive integers, f and g are functions from $\mathbb{R}^d$ to $\mathbb{R}$ and $\mathbb{R}^K$ to $\mathbb{R}^{K'}$ respectively. We consider an optimization problem formulated as:

$$\begin{aligned} \min_{x \in \mathbb{R}^K} \quad & f(x) \\ \text{s.t.} \quad & g(x) = 0 \end{aligned}$$

Then, the *Karush Kuhn and Tucker conditions* (or KKT conditions (Karush, 2014)) are necessary conditions for its solutions. They are expressed as follows:

$$\begin{aligned} \exists \lambda \in \mathbb{R}^K \qquad \nabla f(x) + \lambda \nabla g(x) = 0 \\ g(x) = 0 \end{aligned}$$

If f is convex and g is affine, these conditions become sufficient.

Furthermore, if $\min$ is replaced by $\max$, KKT conditions for this problem are the same. However, they become sufficient if f is concave and g is affine.

2.4 Analysis and Linear Algebra

For this section let c, d and m be stricly positive integers.

- $\mathbb{Z}$, $\mathbb{R}$ and $\mathbb{C}$ are respectively the set of *integers*, *reals*, and *complex numbers*. $\mathbb{N}$ is the set of *positive integers*.
- If $\Omega \subset \mathbb{R}^d$, $\mathring{\Omega}$ is a shorthand for the topological *interior* of Ω.
- Let $\Omega \subset \mathbb{R}^d$. If it exists, the *Lebesgue integral* of a function $f : \Omega' \supset \Omega \longrightarrow \mathbb{R}^m$ is referred to as

$$\int_\Omega f(x)dx$$

- A function $\Omega \subset \mathbb{R} \longrightarrow \mathbb{R}$ is said to be *strictly increasing* if and only if $\forall (x, y) \in \Omega^2 \quad x < y \implies f(x) < f(y)$. f is strictly decreasing if and only if $-f$ is strictly increasing.
- A function $\Omega \subset \mathbb{R} \longrightarrow \mathbb{R}$ is said to be *non decreasing* (or increasing) if and only if $\forall (x, y) \in \Omega^2 \quad x \leq y \implies f(x) \leq f(y)$. f is *non increasing* if and only if $-f$ is non decreasing.
- Let $\Omega \subset \mathbb{R}^d$ be a *convex set*, i.e $(x, y) \in \Omega^2 \implies tx + (1-t)y \in \Omega$.
  A function $f : \Omega \subset \mathbb{R}^d \longrightarrow \mathbb{R}$ is *convex* if and only if

$$\forall t \in [0, 1] \quad \forall (x, y) \in \Omega^2 \quad f(tx + (1-t)y) \leq tf(x) + (1-t)f(y)$$

  f is *concave* if and only if $-f$ is convex. Furthermore, f is said to be *logarithmically convex* (resp. concave) if and only if $\log f$ is convex (resp. concave).
- Given a probability space, the *probability* of an event $\mathcal{E}$ is referred to as $\mathbb{P}(\mathcal{E})$, or $\mathsf{P}(\mathcal{E})$. An empirical probability is also referred to as $p(\mathcal{E})$.
- Given a function f, $\mathsf{P}(x) \propto f(x)$ means that the probability of x is equals $f(x)$ up to a normalization constant (independent of x).
- When a random variable X admits a density, $\mathbb{E}[X]$ is the *expectation* of X, and $\mathbb{V}[X]$ its *variance*.
- Let $\mathbb{K}$ be a subset of $\mathbb{R}$ (not necessarily a field, but at least a ring). $\mathcal{M}_{c,d}(\mathbb{K})$ is a shorthand for the $c \times d$ matrices with coefficients in $\mathbb{K}$ and $\mathcal{M}_d(\mathbb{K})$ the square matrices of size $d \times d$. If $M \in \mathcal{M}_{c,d}(\mathbb{K})$, $M^t \in \mathcal{M}_{d,c}(\mathbb{K})$ is the transpose of M, $\mathrm{Sp}(M)$ represents its set of *eigenvalues* (some of these eigenvalues might be in $\mathbb{C}$) and $\mathrm{Tr}(M)$ its *trace*, i.e the sum of its eigenvalues.

- We use other standard notations of linear algebra. In all this dissertation, we will essentially consider linear subspaces of $\mathbb{R}^d$ for $d > 0$ associated with the field of real scalars. In particular,

$$\mathsf{span}\{x_1, \ldots, x_m\}$$

  is a shorthand for the *linear subspace* generated by the vectors $x_1, ..., x_m$.

- Unless stated otherwise, $||.||$ is the euclidian norm on $\mathbb{R}^d$ with the associated scalar product $\langle \cdot, \cdot \rangle$, i.e $||x|| = \sqrt{\langle x, x \rangle} = \sum_{i=1}^{d} x_i^2$. Besides, $||.||_F$ is the Frobenius norm over $\mathcal{M}_d(\mathbb{R})$, with $||M||_F = \sqrt{\langle M, M \rangle_F} = \mathrm{Tr}(M^t M)$.

- O_m is the set of orthogonal matrices, i.e:

$$O_m = \{M \in \mathcal{M}_m(\mathbb{R}) \mid M^t M = I_m\}$$

  where I_m is the identity matrix.

- If $M \in \mathcal{M}_d(\mathbb{R})$ is a symmetric matrix, it admits a decomposition: $M = U\sigma U^t$ where $U \in O_d$. Moreover, if $\mathrm{Sp}(M) \subset \mathbb{R}^+$ (set of positive reals), (i.e. all the coeffcients of σ are positive) then M is said to be *positive semi-definite (PSD)*. In this case, the *square root* of M is defined as $\sqrt{M} = U\sqrt{\sigma}V$ where $\sqrt{\sigma}$ is the diagonal matrix obtained from the square roots elements of σ.

- Let $M \in \mathcal{M}_{c,d}(\mathbb{K})$. A *singular value decomposition* of M is the triplet (U, σ, V) where $(U, V) \in (O_c \times O_d$ and $\sigma \in \mathcal{M}_{c,d}(\mathbb{K})$ is a diagonal matrix. This decomposition always exists, but is not unique in general.

- The *pseudo-inverse* of a matrix M is obtained with a singular value decomposition $M = U\sigma^* V$ with U and V are orthogonal matrices and σ a diagonal matrix. Then the pseudo-inverse of M can be defined as $M^+ = V\sigma^+ U^t$, where σ^+ is the diagonal matrix whose non zero coefficients have been inversed.

- A matrix M is an upper *Hessenberg matrix* if and only if the indices of its non zero-coefficients verify: $i \leq j+1$. M is a lower Hessenberg matrix if and only if M^T is an upper Hessenberg matrix, where M^T is the transposed matrix of M.

2.5 Empirical Evaluation

For several artificial intelligence problems, and in particular for natural language processing, evaluating perfectly the performance of a program for any end-task is an open-research problem. For instance, an algorithm that would evaluate the

"performance" of another learning algorithm for a natural language-related task, such as text summarization, would require to be able to model the semantics of language in order to evaluate pertinence of the output of the algorithm. Therefore solving (in the strong sense) such evaluation can be considered, in some sense, as difficult as the initial problem.

However, in this thesis, the applications we consider are supposed to belong to a subclass of problems known to be well evaluated. We will adopt here standard evaluation measures, for which ground truth annotations have already been provided, by a human or a program:

- *Binary classification:*
  - *True Positive (TP)* – the system correctly predicts a positive class for a positive example, resulting in a correct acceptance.
  - *True Negative (TN)* – the system correctly predicts a negative class for a negative example, resulting in a correct rejection.
  - *False Positive (FP)* – the system wrongly predicts a positive class for a negative example, resulting in a false alarm.
  - *False Negative (FN)* – the system wrongly predicts a negative class for a positive example, resulting in a miss.
  - *Precision (or specificity)*: It is probably the most intuitive metric and corresponds to the fraction of correct predictions restricted to the positive class $P = \frac{TP+TN}{TP+TN+FP+FN}$
  - *Recall (or sensitivity)*: Consider a medical doctor that uses a specialized search engine to retrieve all ill patients – we realize that no miss can be tolerated, which translates into wanting to reduce the number of false negatives with respect to true postives. This leads to the definition of recall: $R = \frac{TP}{TP+FN}$
  - *F1-score*: Depending on the task at hand and the context, priority is given to sole precision or recall. However, in general both are important. However, it is arbitrary to compare two systems where one has a better precision and the other one a better recall. Several combinations are considered, but we will here mainly consider the standard F1-score measure: $F1 = \frac{2P \times R}{P+R}$.
- *Information retrieval:* Similarly to classification, given a query, an algorithm returning a set of items for the query, and a known set of relevant items, the following scores are defined:
  - *Precision@k* = $\frac{\text{Number of items that are relevant within top-}k\text{ results}}{k}$

– $Recall@k = \frac{\text{Number of items that are relevant within top-}k\text{ results}}{\text{Total number of relevant items}}$

These metrics are used for several problems concerned with large databases or networks of entities, e.g. those mentioned in this manuscript:

- Network of entities described with a knowledge graph (database paired with a semantic graph).
- Network of legal entities in the economy. Each entity represents an organization or a person.

2.6 Other Definitions

In this section we provide some general definitions in linguistics and natural language processing useful throughout this dissertation. The other notions used locally will be properly defined in the beginning of each chapter.

2.6.1 Linguistics

- *Language:* A language is a structured system of communication. It can be seen the method of communication between two entities (and thereby non limited to humans).
- *Grammar:* From Greek γραμματική. In linguistics, grammar is the set of rules governing the use of a formal or natural language, in particular the composition of phrases and words.
- *Syntax:* From Greek συν + τάσσω: "to put together". In linguistics, syntax is the study of sentences and their structure which conform to the grammar of the language.
- *Semantics:* Semantics (from Ancient Greek: σημαίνω: to signify) is the study of the meaning of a language, from a linguistic and philosophical point of view. It is concerned with the relationship between signifiers - like words, phrases, signs, and symbols. It also plays an important role in the study of formal programming languages.
- *Predicate:* In linguistics, a predicate is a property of a subject in a sentence. A predicate is therefore an expression that contains a verb. In this thesis, we will consider the traditionnal grammar definition, where the predicate can include more than the verb. In the sentence: He **is in the park**; He is the subject and **is in the park** is the predicate. The grammar notion of predicate has to be distinguished from the one in mathematical logic, where it is commonly

understood to be a Boolean-valued function: $X \to \{0, 1\}$ called a predicate on X.

- *Ontology:* (From Greek ὄν (genitif ὄντος: "being", or more precisely: "that which is") and λόγος): discourse) In computational linguistics and information science, an ontology is composed of a representation, definition of the categories, properties and relations between the concepts, data and entities. In practice, an ontology can be represented by a Knowledge graph which contains the properties of a subject area and how they are related, by defining a set of concepts and categories that represent the subject.

- *Lexicon:* A lexicon (from Greek λεξικόν: dictionary) is the complete set of meaningful units in a language (or a subject); the words, etc., as in a dictionary, but without the definitions. In this thesis, we might also refer to the lexicon as the vocabulary. We suppose the lexison is finite and by doing so, we suppose that there exists $n \in \mathbb{N}^*$ such that $V \simeq \{1, ..., n\}$.

- *Homonymy:* From Greek ὁμοιος: "same" and ὀνομα "name". In linguistics, homonyms, broadly defined, are words sharing the same spelling, regardless of pronunciation) or homophones (words that share the same pronunciation), or both. For example, according to this definition, the words bat (animal) and bat (baseball instrument) are homonyms. The words see (vision) and sea (body of water) are homophones.

- *Polysemy:* Polysemy (from Greek πολύ: "many" and σημαίνω: "to mean") is the capacity for a word (or sentence) to have several meanings. Polysemy is thus distinct from homonymy (or homophony for that matter) which is often an accidental similarity between two words. Such distinction can be established in practice by considering the history of the word to see if the two meanings are historically related.

2.6.2 Natural language processing

We present three methods of word vector generation, which are useful for Chapter 5 and Chapter 6, where additional discussions are provided.

Window size and context. In order to present these methods, few preliminary definitions are necessary, and will also be useful throughout all the dissertation. A document (or a sentence, which can be seen as a short document) is composed of *tokens* whose values are words. In this document, a *window* of size $w \in \mathbb{N}^*$ represents a consecutive sequence of w tokens. For instance, in a document composed of $p \in \mathbb{N}^*$ tokens, a window of size w defines exactly $p - w$ contexts in the document.

word2vec. (Mikolov et al., 2013a) Also referred to as Skip-gram model. The skip-gram model aims to predict the surrounding context words given a center word, using the maximum likelihood principle on the probability:

$$\mathsf{P}(\, w_{t-\omega}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+\omega} \mid w_t).$$

which represents the probabilities of the surrounding context words (for a window size $w+1$, cf. previous definition) given a word at position t. The objective of the skip-gram model is to maximize the empirical log-likelihood:

$$\sum_{t=1}^{T} \sum_{c \in C(t)} \log p(w_c | w_t), \tag{2.2}$$

where $C(t)$ is the set of indices of words surrounding the word w_t. and T is the number of tokens (size of the training set).

The skip-gram model makes the following assumption:

$$\mathsf{P}(w_c | w_t) \propto exp(\langle v_c, v_t \rangle)$$

where v_c (resp v_t) are the word vectors of word w_c (resp. w_t). Therefore, for a chosen context position $c \in C(t)$, using the binary logistic loss, we can consider the following negative log-likelihood to be minimized:

$$\log\left(1 + e^{-\langle v_t, v_c \rangle}\right) + \sum_{n \in N(t,c)} \log\left(1 + e^{\langle v_t, v_n \rangle}\right),$$

where $N(t,c)$ is a set of negative examples sampled from the vocabulary.

By summing over $t = 1, ..., T$, and over $c \in C(t)$, this yields the following objective function:

$$\sum_{t=1}^{T} \left[\sum_{c \in C(t)} \left(\log\left(1 + e^{-\langle v_t, v_c \rangle}\right) + \sum_{n \in N(t,c)} \log\left(1 + e^{\langle v_t, v_n \rangle}\right) \right) \right]. \tag{2.3}$$

Usually, this optimization problem is solved approximately using Stochastic Gradient Descent (Carpentier and Cohen, 2017; Bottou et al., 2018).

fastText. In (Bojanowski et al., 2017) An extension of the skip-gram model (i.e word2vec) is fastText, which takes account the morphology of words: a vector representation is associated to each character s-gram and words are represented as the sum of these vectors.

Glove (Pennington et al., 2014) Given a natural integer w strictly positive, Glove embeddings are constructed with an embedding
$v : \mathcal{V} \to \mathbb{R}^K$, by solving a weighted least-squares regression problem

$$\min_{v,\hat{a},\hat{b}} \quad \sum_i \sum_j f(C_{ij}) \left(\langle v_i, v_j \rangle + \hat{a}_i + \hat{b}_j - \log(C_{ij}) \right)^2, \tag{2.4}$$

where C_{ij} represents the number of co-occurences of word i and j in a window of size w, and $f(C_{ij}) = \min(C_{ij}, 100)^{3/4}$.

2.7 Datasets

In this section, we briefly describe the datasets used in this thesis. The ones used Chapter 3 are synthetic datasets, so generated from our implementation (cf. Section 1.3). The remaining datasets are open-source, except (Dijk, 2018).

- Knowledge bases: DBPedia/Wikipedia 2016 (Lehmann et al., 2015). DBPedia is a very large database, meta data of wikipedia which classifies its webpages into classes by the means of an Ontology.

- Orbis Van Dyuk (Dijk, 2018): Dataset built from *Orbis* is a database composed of about one hundred million entities, developed by a specialized group based in Bureau Van Dijk's Brussels office, aggregating several sources of data. More details are given in Chapter 4, Section 8.2.2.

- Named entity linking: TAC-KBP (Ji et al., 2014): Dataset from the National institute of Standards ant technologies (NIST), within the Text Analysis Conference's Knowledge Base Population (TAC-KBP) track, which aimed to link a given named entity mention from a source document to an existing Knowledge Base (KB).

- AIDA/CoNLL (Yosef et al., 2011): It contains assignments of entities to the mentions of named entities annotated for the original CoNLL 2003 entity recognition task.

- In the experiments of Chapter 5, we use the datasets of analogies (Mikolov et al., 2013a). It contains 19544 question pairs (8,869 semantic and 10,675 syntactic (i.e. morphological) questions) and 14 types of relations (9 morphological and 5 semantic). Each line contains a quadruplet of words composing an analogy, e.g:

  Paris France Beirut Lebanon

- In the experiments of Chapter 5 and 6, we use several Word embeddings datasets: Glove (Pennington et al., 2014), word2vec Mikolov et al. (2013a), and fastText (Bojanowski et al., 2017), where each file represent a different dictionnary between words and their vector representation.

- In the text classification experiments of Chapter 5 and 6, we make use of:

  1. *WebKB*: 4 most frequent categories of webpages from The 4 Universities Data Set.

  2. *Amazon* (Blitzer et al., 2007) is composed of product reviews acquired from Amazon over four different sub-collections.

  3. *Subjectivity* (Pang and Lee, 2004) contains sentences considered as subjective gathered from the websites Rotten Tomatoes and sentences considered as objective gathered from IMDB.

3

Sequence Graphs and Ambiguity in Language Models: A Combinatorial Study

Several sequential and language models rely on an assumption modeling each local contexts as a "bag of words". In this chapter, we study the combinatorial implications of such assumption for the corresponding word or sentences representations. In particular, we present theoretical results concerning the family of sequence graphs, for which realizations yield equivalent representations given this assumption. Several combinatorial problems are presented, depending on three levels of generalisation (window size, graph orientation, and weights), and whether some of these are NP-complete is left opened. Based on these results, we also establish different algorithms, including a dynamic programming formulation, to count and explicit the different realizations of a sequence graph. This allows us to show that the bag of words assumption can induce an important number of sentences to have the same representations, even for relatively short context window sizes.

3.1 Introduction

About the notion of context

To understand the meaning of a word, it is necessary to take into consideration its environment. The very notion of context has been developed theoretically as a linguistic concept in (Gross, 2010), for which the main conclusions are as follows:

- The lexicon cannot be separated from the syntax, from the combinatorics of the words;
- The semantics is not autonomous: it is the result of the combination of lexical elements organized in a certain way.

- We can postulate that there are three independent levels in the linguistic description, that of the lexicon, of the syntax and of the semantics, because we do not see how they could be articulated, if they were independent.
- The existence of polysemy, which is one of the fundamental properties of natural languages, obliges to link these three levels, which can only be done on the basis of the lexicon.
- The use of a word, represented by a simple sentence, consists of a predicate with a determined pattern of arguments. This predicate has properties, different from those it might have in another environment.
- The description of such use implies recognition of its context, that is to say of its appropriate environment.
- It stands to reason that the minimal unit of analysis is not the word level, but rather the sentence level, hence their importance in language models.

This motivates the study of the representations of a context, from a computational point of view, which we address in this chapter.

Context and representations

In the fields of Natural Language Processing (NLP) and Information Retrieval (IR), concise representations of words and textual documents are essential for several tasks, including document classification (Skianis et al., 2018), role labelling Roth and Woodsend (2014), and named entity recognition (Nadeau and Sekine, 2007). In particular, the Bag-Of-Words (BOW) representations (Salton et al., 1975; Ramos et al., 2003) encode a text and/or a sentence as a vector x of weighted occurrences. If D represents the a corpus of documents, and x a document in D, then two standard representations can be defined as follows:

$$(f, t) \in \mathbb{R}^v \times \mathbb{R}^v$$

$$f_i = \text{frequency of word } i \text{ in } x \quad \text{and} \quad t_i = f_i \times \text{idf}_\text{i} = f_i \times \log \frac{|D|}{|\{d_j : i \in d_j\}|}$$

f is the vector of frequencies of the words in the document, and t is composed of these frequencies weighted by the rarity of the term considered, measured with the ratio $|D|/|\{d_j : i \in d_j\}|$. This term can tend to infinity with a large corpus, hence the $\log$ term. Using BOWs, classic tasks such as document similarity computation and indexing can be efficiently computed using sparse linear algebra, leading to their presence at the core of popular software solutions such SMART Salton (1971b) or Lucene Hatcher and Gospodnetic (2004).

(a) No ambiguity ($w = 3$)

(b) Ambiguity ($w = 2$)

Figure 3.1 – Sequence graphs (or *graphs-of-words*) built for the sentence "Linux is not UNIX but Linux" using window sizes 3 (a) and 2 respectively (b). In the second case, the sequence graph is ambiguous, since any circular permutation of the words admits the same representation.

The invariance of the BOW representation to permutations of the words in the document can lead to multiple documents being summarized by the same BOW. Indeed, the number of documents having the same BOW representation grows at factorial rate with the size of the document. Moreover, this high degree of **ambiguity** has practical consequences for fine-grained classification tasks, as empirically shown by Malliaros and Skianis (2015); Skianis et al. (2018) in the context of multi-label classification. For these reasons, more recent works introduced the **Graph of Words** Gibert et al. (2011); Rousseau et al. (2015) and Peng et al. (2018) (GOW), which supplements the contents of BOW with statistics of co-occurrences within a **window** of fixed size w, introduced to mitigate the degree of ambiguity induced by the representation.

Several models such as word2vec Mikolov et al. (2013a), Glove Pennington et al. (2014) also use the same type of information and allow to increase performance for multiple tasks in natural language processing.

While GOW representations are more precise than Bag-of-Words, they still induce some level of ambiguity, *i.e.* a given graph can represent several sequences, as illustrated in Figure 3.1. Our study is thus motivated by a quantification of the level of ambiguity, seen as an algorithmic problem, coupled with an empirical assessment of the consequences of ambiguity in the context of classification. As a first natural step, we also consider the realizability of a given GOW, *i.e.* the existence of a sequence admitting an input GOW as its representation.

After introducing in Section 3.2 the formal definition of a sequence graph, which is the combinatorial abstraction of a Graph-Of-Words, and descriptions of our main problems, we establish in Section 3.4 complexity aspects of deciding the existence and counting sequences in GOWs associated with a window size $w = 2$. Then we consider in Section 3.5 the general case $w \geq 3$, and propose an exponential dynamic programming algorithm to count admissible sequences. Finally, we assess the prevalence of ambiguity within a synthetic dataset, and observe that sequences invariant with respect to the GOW representation do not lead to invariance with respect to LSTM, a popular neural network.

3.2 Definitions and Problem Statement

In the following, let $x = x_1, x_2, ..., x_p$ be a finite sequence of discrete elements among a finite vocabulary X. Without loss of generality, we can suppose that $X = \{1, ..., n\}$, and let $I_p = \{1, ..., p\}$.

Definition 3.1. *$G = (V, E)$ is the graph of the sequence x with window size $w \in \mathbb{N}^*$ if and only if $V = \{x_i \mid i \in I_p\}$, and*

$$(i, j) \in E \iff \exists (k, k') \in I_p^2,\ |k - k'| \leq w - 1,\ x_k = i \text{ and } x_{k'} = j \tag{3.1}$$

For a digraph $G = (V, A)$, Eq. 3.1 is replaced by

$$(i, j) \in A \iff \exists (k, k') \in I_p^2,\ k \leq k' \leq k + w - 1,\ x_k = i \text{ and } x_{k'} = j \tag{3.2}$$

Finally, a weighted sequence digraph G is endowed with the matrix $\Pi(G) = (\pi_{ij})$ such that:

$$\pi_{ij} = \text{Card } \{(k, k') \in I_p^2 \mid k \leq k' \leq k + w - 1,\ x_k = i \text{ and } x_{k'} = j\} \tag{3.3}$$

By convention, a weighted (undirected) sequence graph is endowed with $\Pi = (\pi_{ij})$, $\pi_{ij} = \pi_{ij}^{'} + \pi_{ji}^{'}$ if $i \neq j$ and $\pi_{ij}^{'}$ otherwise, where π' verifies Eq. 3.3.

We say that x is a w-admissible sequence for G if G is the graph of the sequence x. G is referred to as the w-sequence graph of x.

π_{ij} represents the number of co-occurrences of i and j in a window of size w. Hence, the graph of a sequence x is unique for a given w. In the following, we use $G_w(x)$ as a shorthand for the w-sequence graph of x. In the weighted and directed case, it can be obtained with Algorithm 1.

Algorithm 1: Construction of the weighted sequence digraph of a sequence x

Data: Sequence x of length p, window size w, $p \geq w \geq 2$
Result: $(G_w(x), \Pi)$

```
V ← Ø;
d ← number of distinct elements of x;
Initialize Π = (π_{i,j}) to d × d matrix of zeros;
for i = 1 → p − 1 do
    V ← V ∪ {x_i, x_{i+1}} ;
    for j = i + 1 → min(i + w − 1, p) do
        π_{x_i,x_j} ← π_{x_i,x_j} + 1;
Return V, Π
```

If G is not directed, one should replace line 7 of Algorithm 1 by the "symmetrized" update:

$$\begin{aligned} \text{if } \pi_i \neq \pi_j : \quad & \alpha \leftarrow \pi_{x_i,x_j}, \qquad \pi_{x_i,x_j} \leftarrow \alpha + 1, \qquad \pi_{x_j,x_i} \leftarrow \alpha + 1 \\ \text{else} : \quad & \pi_{x_i,x_i} \leftarrow \pi_{x_i,x_i} + 1 \end{aligned} \tag{3.4}$$

The procedure in Algorithm 1 defines a correspondence between the sequence set $X^\star$ into the set of all finite graphs:

Definition 3.2. *Let X^* be the set of finite sequences built on the alphabet X, $\mathcal{G}$ the set of finite graphs and $w \in \mathbb{N}^*$ a window size. Based on the previous definition, let ϕ_w be the function defined as:*

$$\begin{aligned} X^\star &\to \mathcal{G} \\ \phi_w : x &\mapsto G_w(x) \end{aligned} \tag{3.5}$$

Therefore, in this chapter, $G \in \operatorname{Im} \phi_w = \{\phi_w(x) \mid x \in X^\star\}$ exactly means that G is a $w-$sequence graph. We also extend the definition of ϕ_w to the case of weighted sequence-graphs by endowing $G_w(x)$ the weights matrix Π defined in Algorithm 3.1.

For a given w, the two problems we address in this chapter are the characterization (or recognition) of w-sequences graph, and the counting of the number of their w-admissible sequences.

Related work

Sequence graphs encode the information of several co-occurrences based models (Arora et al., 2016a; Mikolov et al., 2013c; Pennington et al., 2014). To the best of our knowledge, the ambiguity and realizability questions addressed in this chapter were never systematically addressed by prior work in computational linguistics. Furthermore, we believe the problems studied in this chapter are interesting from a graph theoretical and algorithmic point of view, and appear to be devoid of reduction to other well-known problems.

However, some similarities exist between our problem and others studied in the Distance Geometry (DG) literature. In distance geometry, the input consists of a set of pairwise distances between points, having unknown positions in a K-dimensional space. The problem then consists in determining (the existence of) a set of positions for the points, satisfying the distance constraints. When the distances are precise, a position is fully characterized from $K + 1$ constraining neighbors, the problem can be solved by finding a sequential order for processing points, such that the assignment of a point is always by at least $K + 1$ among its neighbors (Liberti et al., 2014). This statement shares some level of similarity with our problem since an admissible sequence for a window $w = K + 2$ also represents

a linear ordering of its nodes, in which $w - 1 = K + 1$ of the neighbors have lower value with respect to the order.

The reasons for the insufficiency of linear ordering in DG to solve our realizability problem are twofold. First, each element of the sequence x in the ordering for "traditional" distance geometry (sensors, or atoms in a protein) is associated a unique vertex in the graph. This is not the case we investigate here, since a symbol can be repeated several times in the sequence, but only one vertex is created in the graph. This implies that, in a linear ordering, the vertex associated to the i^{th} element $(i \geq w)$ of x can have strictly less than $w - 1$ distinct neighbors while being a sequence graph. Such situation is is forbidden in DG ordering, even when the loops are authorized (Lavor et al., 2013). Finally, in the majority of cases, the graphs are undirected in distance geometry, although recent but preliminary studies consider the *orientation* as available information (Billinge et al., 2018).

Notations

In the following, we use $\mathcal{M}_d(\mathbb{N})$ as a shorthand for the square $d \times d$ matrices over the set of natural integers, $\mathrm{Tr}(M)$ for the trace of a matrix M, and $\mathrm{Sp}(M)$ for its set of eigenvalues.

Problem 1 (REALIZABLE$_w$)**.**
Parameters: *Window size* w
Input: *Graph* G *(and optional matrix weights* Π*)*
Output: *True if* (G, Π) *is the* w*-sequence graph of some sequence* x*, False otherwise.*

Problem 2 (NUMREALIZATIONS$_w$)**.**
Parameters: *Window size* w
Input: *Graph* G *(and optional matrix weights* Π*)*
Output: *The number of* ***realizations*** *of* G*,* i.e. *preimages of* G *through* ϕ_w *such that* $|\{x \in X^\star \mid \phi_w(x) = G\}|$ *if finite, or* $+\infty$ *otherwise.*

Note that the last problem strictly generalizes the previous one, as REALIZABLE$_w$ can be solved by testing the nullity of the number of suitable realizations computed by NUMREALIZATIONS$_w$.

3.3 Motivations, Summary of the Theoretical Results and Relation with Language Models

As presented in Chapter 1 the difficulty for the design of automatic treatment of natural language lies mainly in the semantics rather than syntax. Many co-occurence based models thus attemps to capture semantic relations between words, "drowned" in the syntax. To do so, some of these models take as unique input the

statistics of the presence of words within contexts of a constant size. For instance, let us consider the traditional pointwise mutual information (PMI), defined as:

$$\mathsf{PMI}(i, j') \triangleq \log \frac{\mathsf{P}(i,j)}{\mathsf{P}(i)\mathsf{P}(j)} \tag{3.6}$$

where $\mathsf{P}(i,j)$ is the probability of finding words i and j in the same window of size w in a document, and $\mathsf{P}(i)$, $\mathsf{P}(j)$ are the marginal probabilities. Then, PMI information is empirically computed from the co-occurence statistics encoded in the weights matrix Π, so that several realizations of the corresponding sequence graph (in the sense of Definition 3.1) would yield the same PMI values.

There exist a variety of other models taking the similar input (Mikolov et al., 2013a; Pennington et al., 2014; Arora et al., 2016a) which will be addressed in details in Chapter 5 and 6. Similarly to PMI, sequence graphs encode the information of their input. Therefore, the output of NUMREALIZATIONS$_w$ is a measure of the level of ambiguity generated by these models when representing a document. REALIZABLE$_w$ is a second theoretical graph problem which is related to NUMREALIZATIONS$_w$, and appeared naturally in our study as a intermediate step towards the resolution of NUMREALIZATIONS$_w$.

The remaining of this chapter is organized as follows. In the next two sections we present the combinatorial results for $w = 2$, and the general case ($w \geq 3$). We remind our reader that the parameter w is fixed so the complexity results only take in account the Graph G and (potential) matrix weights Π.

In particular, Section 3.4, contains full characterizations of REALIZABLE$_2$ and NUMREALIZATIONS$_2$ are presented, in terms of complexity, and algorithms to solve them. The case $w = 2$ is separated from the case $w \geq$ since it corresponds to the simplest case, solved via an Eulerian reduction.

In Section 3.5, several theoretical results are presented. We prove a complexity result of REALIZABLE$_w$, in the undirected and unweighted case. We also present an polynomial time algorithm to solve REALIZABLE$_w$ in this case. Besides, a Linear integer programming formulation of REALIZABLE$_w$ and a dynamic programming formulation NUMREALIZATIONS$_w$ in the general case are discussed. We test these formulations on several instances and show that NUMREALIZATIONS$_w$ can have an output stricly greater than one, even for large window sizes.

In Section 3.6, we present a complexity study of one of the algorithm which allowed to solve NUMREALIZATIONS$_w$, whose average complexity to be much better than worse case scenario. We also provide several instances we solved using this formulation in the Appendix 8.

In Section 3.7 we present a use of these results for the understanding of the level of ambiguity of co-occurence based models. To do so, we elaborate and discuss the results of two experiments. First we illustrate the number of realizations representing a weighted digraph, averaged on 500 sequence graphs computed on

natural language documents of less than 100 words. In a second experiment, we compare the ambiguity of co-occurence based models with a recurrent neural network. To do so, we estimate the normed difference of the representations of the network (weights obtained after training) between two realizations of a given sequence graph, and between a realization and a random sequences generated from the same vocabulary.

Conclusions are given in Section 3.8.

3.4 Complexity Results over 2-Sequence Graphs

In this section, we consider $w = 2$. Algorithm 1 encodes each adjacency in the sequence x as an edge in $G_w(x)$. This characterization enables relatively simple algorithmic treatment, leading to the results summarized in Table 3.1, which we further elaborate in this section. Namely, $\psi(G)$ is a linear transformation of G with respect to its number of vertices and edges (cf. Definition 3.4).

Table 3.1 – Complexity for various instances of our problems $(w = 2)$

	NUMREALIZATIONS$_2$		REALIZABLE$_2$	
Data Instance	Complexity	#Sequences	Complexity	Characterization
Unweighted graph	P	$\{0, +\infty\}$	P	G connected
Weighted graph	#P-hard	$\{0, 1\} \cup 2\mathbb{N}^*$	P	$\psi(G)$ (semi)Eulerian
Unweighted digraph	P	$\{0, 1, +\infty\}$	P	Theorem 3.1
Weighted digraph	P	$\mathbb{N}$	P	$\psi(G)$ (semi)Eulerian

Obviously, the simplest case concerns undirected graphs as stated in:

Proposition 3.1. *Let $G = (V, E)$ be an unweighted and undirected graph with $|V| > 1$. Then, the following assertions are equivalent:*

(i) G is connected

(ii) G has a 2-admissible sequence

(iii) G admits an infinite number of 2-admissible sequences

Proof. If G is connected, a sequence is obtained by visiting all edges, for instance using a list of arbitrary sequences and shortest paths. The other implications are immediate. □

For digraphs, the previous characterization is wrong, even with strong connectivity. A counter example is given in Fig. 3.2a. However, strong connectivity remains a sufficient condition:

Proposition 3.2. *Let $G = (V, A)$ be a unweighted digraph. If G is strongly connected then $G \in \operatorname{Im} \phi_2$. Moreover, a 2-admissible sequence can start or end at any given vertex of G.*

Proof. Straightforward, similarly to (i) $\implies$ (ii) for Proposition 3.1. □

Proposition 3.3. *Let $G = (V, E)$ be an unweighted digraph. If G is Eulerian or semi-Eulerian, then $G \in \operatorname{Im} \phi_2$.*

Proof. If G is Eulerian or semi-Eulerian, there exists a walk going through all its arcs, this walk defines a 2-admissible sequence. □

Again the converse of Proposition 3.3 does not hold as depicted in Fig. 3.2b. First, it is natural to consider the case of directed acyclic graphs (DAGs):

Proposition 3.4. *Let $G = (V, A)$ be a DAG. G is a 2-sequence graph if and only if it is a directed path, i.e G is a directed tree (or polytree) where each node has at most one child and at most one parent. In this case, G has a unique 2-admissible sequence.*

Proof. If G is a directed path, since G is finite, it admits a source node. Therefore a 2-admissible sequence is obtained by simply going through all vertices from the source node. This is obviously the only one.

Conversely, let us suppose G is a DAG and a 2-sequence graph. If G is not a directed path, there are two cases: either there exists a vertex having two out-vertices, or two in-vertices. Let s be a vertex having 2 distinct out-vertices c_1 and c_2. This is not possible since there cannot be a walk going through (s, c_1) and (s, c_2): G would have a circuit otherwise. Finally a vertex v cannot have two in-vertices p_1 and p_2: if a 2-admissible sequence existed, it would have to go through (p_1, v) and (p_2, v), creating a cycle, hence the contradiction. □

Every directed graph G is a DAG of its strongly connected components. In the following, let $R(G)$ be the DAG obtained by contracting the strongly connected components of G.

Proposition 3.5. *Let $G = (V, A)$ be a digraph. If G is a 2-sequence graph then $R(G)$ is a 2-sequence graph.*

Proof. Let G be a 2-sequence graph, and let us suppose that $R(G)$ is not a 2-sequence graph. Since $R(G)$ is a DAG, then using Proposition 3.4, it cannot be a directed path, so $R(G)$ has either a node having two out-vertices or two in-vertices. Let S be a node of $R(G)$ having at least 2 distinct children C_1 and C_2. This means that there exist three distinct corresponding nodes in V, s, v_1 and v_2 such that $(s, v_1) \in E$ and $(s, v_2) \in E$. Since G is a 2-sequence graph, there exists a walk covering (s, v_1) and (s, v_2), such walk would make S, C_1 and C_2 the same node in $R(G)$, hence the contradiction. The case for which a vertex has two parents is dealt with similarly. □

The converse of Proposition 3.5 does not hold as depicted in Fig. 3.2d, which motivates the following definition.

Definition 3.3. *Let G be a digraph, and $R^+(G)$ be the weighted DAG obtained from $R(G)$, such that the weight of an arc (u, v) of $R(G)$ is attributed the number of arcs between the two corresponding strongly connected components of G.*

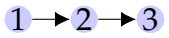
Theorem 3.1. *Let $G = (V, E)$ be an unweighted digraph. G is a 2-sequence graph if and only if $R^+(G)$ is a directed path and its weights are all equal to 1.*

Proof. If G is a 2-sequence graph, $R(G)$ is a 2-sequence graph using Proposition 3.5. Also Proposition 3.4 implies that $R(G)$ and $R^+(G)$ are directed paths. Moreover, if $R^+(G)$ had a weight strictly greater that 1, then there would be strictly more than one arc between two connected components C_1 and C_2. All these edges go in the same direction otherwise $C_1 \cup C_2$ would form a strongly connected component. This is a contradiction since any 2-admissible sequence would have to go from C_1 to C_2 and then come back to C_1 (or conversely) which would would make $C_1 \cup C_2$ a strongly connected component.

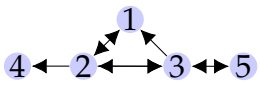
Conversely, let us suppose $R^+(G)$ is a a directed path and its weights are equal to one. First, there exists a walk $x_1, ..., x_p$ covering all arcs of $R^+(G)$ verifying: (i) $\forall i,\ x_i \in V$ or x_i represents a strongly connected component of G, (ii) there is only one arc in G between from x_i to x_{i+1} and (iii) x has no repetition, i.e there is no common node in G between x_i and x_{i+1}. We construct a 2-admissible sequence y for G by means of the following procedure.

Initialisation: If $x_1 \in V$, we simply set $y \leftarrow x_1$. Otherwise, x_1 corresponds to a strongly connected component C_1 of G and we add to y any 2-admissible sequence of C_1.

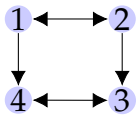
For $i \in \{1, .., p-1\}$:



(a) 1 2 3 is a 2-admissible sequence but G is not strongly connected



(b) 3 5 3 1 2 1 2 3 2 4 is a 2-admissible sequence but the graph is not Eulerian nor semi-Eulerian



(c) G is not a 2-sequence graph...

(d) ... whereas $R(G)$ is.

Figure 3.2 – Counter examples for $w = 2$

- If $(x_i, x_{i+1}) \in E$: we add x_{i+1} to the sequence y.

- If $x_i \in V$ and x_{i+1} is a strongly connected component C_i of G: By assumption, there exists only one arc of G from x_i to a node of C_i, say c_0^i. Since C_i is strongly connected, using Proposition 3.2, C_i has a walk going through all of its arcs and starting in c_0^i, say $c_0^i, ..., c_p^i$. We add $c_0^i, ..., c_p^i$ to y.

- If x_i corresponds to a strongly connected component C_i and $x_{i+1} \in V$: we perform similar operations by stopping on the single node of C_i that has a edge to x_{i+1} (this is possible thanks to Proposition 3.2).

- x_i and x_{i+1} both correspond to strongly connected components C_i and C_{i+1} , there exists only one arc between in E between C_i and C_{i+1}, say $e_i = (v_i, v_{i+1})$. We can complete y by a walk from the last vertex visited which belong to C_i and v_i, and then by a 2-admissible sequence through C_{i+1} starting in v_i and ending in v_{i+1}.

End For

The process stops when $i = p - 1$, and all arcs are visited by the sequence y. □

Therefore, an algorithm to decide if a digraph is a 2-sequence graph is obtained by extracting its strongly connected components (there exist linear time algorithms e.g (Sharir, 1981)), and to count the number of arcs between these.

Corollary 3.1.1. *Let G be an unweighted digraph. The possible numbers of 2-admissible sequences for G is exactly $\{0, 1, +\infty\}$. Moreover, G admits a unique 2-admissible sequence if and only if G is a directed path.*

Proof. Let G a be 2-sequence graph. G verifies the characterization of Theorem 3.1. If $R(G)$ has a node C representing a strongly connected component of G (or a node with a loop), then by adding an arbitrary number of cycles in C to the admissible sequence y, the new sequence is still admissible. Otherwise, if every node of $R(G)$ is in V without self-loops in E, then G is a DAG. Using Proposition 3.4, y is the unique 2-admissible sequence. □

Weighted 2-sequence graphs

The weighted case cannot be treated similarly due to the constraint of Equation 3.3. A counterexample is depicted in Fig. 3.3. Moreover, a weighted graph has a finite number of admissible sequences. This property can seen using Proposition 3.6 below.

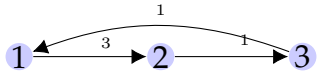


Figure 3.3 – G is strongly connected but is not a 2-sequence graph

Proposition 3.6. *If a graph is a weighted w-sequence graph, all of its admissible sequences have the same length.*

Proof. Let x be a w-admissible sequence for G of length p. If G is a digraph, Algorithm 1 increments $(p-w+1)(w-1)+\frac{(w-1)(w-2)}{2}$ times the total weight, therefore:

$$\sum_{i,j} \pi_{ij} = (p-w+1)(w-1) + \frac{(w-1)(w-2)}{2} \tag{3.7}$$

If $w \geq 2$, this yields: $p = w - 1 - \frac{w-2}{2} + \frac{1}{(w-1)} \sum_{i,j} \pi_{ij}$

Otherwise, if G is undirected, the weights matrix obtained with Algorithm 1 does not yield Eq. 3.7, due to the update of Eq. 3.4. The weights on the diagonal remain the same, but the others are multiplied by 2, hence the formula:

$$\sum_{i,j} \pi_{ij} + \mathrm{Tr}(\Pi) = 2(p-w+1)(w-1) + (w-1)(w-2) \tag{3.8}$$

leading to $p = \frac{1}{2(w-1)}[\sum_{i,j} \pi_{ij} + \mathrm{Tr}(\Pi)]$. □

Corollary 3.1.2. *Let G be a weighted w-sequence digraph, and Π its weights matrix. If w even, then $(w-1) \mid \sum_{i,j} \pi_{ij}$.*

Corollary 3.1.3. *Let G be a w-sequence (unoriented) graph and Π its weights matrix. Then $2(w-1) \mid \sum_{i,j} \pi_{ij} + \mathrm{Tr}(\Pi)$.*

Definition 3.4. *Let $\psi(G)$ be the auxiliary multigraph with the same vertices as $G = (V, E)$ and with π_{ij} edges between $(i, j) \in V^2$.*

Due to the previous study, the characterization of weighted 2-sequence graphs using $\psi(G)$ is immediate. A semi-eulerian graph is a graph that admits a Eulerian walk (instead of cycle for eulerian graphs).

Theorem 3.2. *If G is a weighted graph (directed or not), with $\Pi(G) \in \mathcal{M}_d(\mathbb{N})$, then: $G \in \mathrm{Im}\,\phi_2 \iff \psi(G)$ is connected and semi-eulerian.*

Proof. $G \in \mathrm{Im}\,\phi_2$ means that there is a path going through each edge $(i, j) \in E$ exactly π_{ij} times. This trail corresponds to a semi-eulerian path in $\psi(G)$. □

Counting 2-admissible sequences for weighted graphs

Counting eulerian paths in graphs has already been studied. Here, we use lemma 3.1 to conclude.

Lemma 3.1. *Let $G = (V, E)$ a weighted 2-sequence graph (possibly oriented). Let $\mathcal{E}$ be the set of eulerian paths of $\psi(G)$ and $\mathcal{S}$ be the set of 2-realizations of G. Then*

$$\#\mathcal{E} = (\#\mathcal{S}) \prod_{e \in E} \pi_e! \tag{3.9}$$

Proof. We will first prove this result for digraphs. If $e = (v_1, v_2)$ is an edge of a digraph, we will represent the source and target vertex of e as $e(s)$ and $e(t)$. Let $(e_1, e_2, ..., e_h)$ be a eulerian path of $\psi(G)$ defined as a sequence of its edges. Then $\forall (i, j) \in \{1, ..., h\}^2,\ e_i \neq e_j$ and $\forall i \in \{1, ..., h-1\},\ e_i(t) = e_{i+1}(s)$. Let us consider the transformation:

$$\begin{gathered} \mathcal{E} \longrightarrow \mathcal{S} \\ (e_1, e_2, ..., e_h) \mapsto (e_1(s), e_2(s), ..., e_{p-1}(s), e_h(t)) \end{gathered} \tag{3.10}$$

We have already shown this transformation is surjective: any 2-sequence of G can be obtained with a eulerian path of $\psi(G)$. We will now consider the action of $\mathfrak{S}_h$ on $\mathcal{E}$. For a eulerian path, let us suppose that two edges of $\psi(G)$ have been permuted, say e_1 and e_{i_0} without loss of generality. If the two corresponding sequences are the same:

$$(e_{i_0}(s), e_2(s), ..., e_h(t)) = (e_1(s), e_2(s), ..., e_{i_0}(s), ..., e_h(t))$$

Obviously, $e_{i_0}(s) = e_1(s)$. Also $e_1(t) = e_2(s)$ implies $e_{i_0}(t) = e_1(t)$. This shows that e_{i_0} and e_1 are associated to the same edge in E. Therefore, given a 2-sequence, the choice of a corresponding eulerian path corresponds to the choice of $\sigma = (\tau_1, ..., \tau_{|E|})$ where τ_e is a permutation of $\{1, ..., \pi_e\}$ (or $\emptyset$ if $\pi_e = 0$) representing the visit order of the related edges of $\psi(G)$. Therefore $\#\mathcal{E} = (\#\mathcal{S}) \prod_{e \in E} \pi_e!$

If G is undirected, the proof is still valid, but the operators $e \mapsto e(s)$ and $e \mapsto e(t)$ are now induced by the natural direction of the eulerian path considered. □

Proposition 3.7. *Couting the number of 2-sequences for a weighted graph is $\#P$-complete. However, if G is a weighted digraph, with $\Pi(G) \in \mathcal{M}_d(\mathbb{N})$, then, the number p_2 of 2-admissible sequences is given by:*

$$p_2 = \frac{t(\psi(G))}{\prod_{e \in E} \pi_e!} \prod_{v \in V} \big(\deg_{\psi(G)}(\psi(v)) - 1\big)! \tag{3.11}$$

where t(G) is the number of spanning trees of G. If L is the Laplacian matrix of G, then $t(G)$ is given by:

$$t(G) = \prod_{\substack{\lambda_i \in \mathrm{Sp}(L) \\ \lambda_i \neq 0}} \lambda_i \tag{3.12}$$

Proof. Counting the number of eulerian paths in an undirected graph has been proven to be a $\#P$-complete problem (Brightwell and Winkler, 2005). On the other hand, BEST Theorem (van Aardenne-Ehrenfest and de Bruijn, 2009)) and Matrix tree theorem (Chaiken, 1982)) yield a formula for digraphs. Since $G \mapsto \psi(G)$ is bijective, we use Lemma 3.1 to conclude. □

To use formula 3.11, $\deg_{\psi(G)}(\psi(v))$ can be obtained using the following formula:

$$deg_{\psi(G)}(\psi(v)) = \sum_{u \in V} \pi_{uv} + \sum_{u \in V} \pi_{vu} \tag{3.13}$$

Corollary 3.2.1. *The possible number of realizations of a weighted 2-digraph is exactly* $\mathbb{N}$.

Proof. Two proofs (at least) are possible: one using Equation 3.11. The most elegant consists in considering the cycle of n vertices where each weight is equal to 1 (Figure 3.4). It has exactly n realizations since a 2-admissible sequence can start at any vertex.

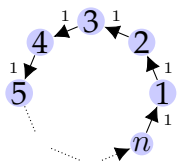


Figure 3.4 – Cycle of length n

□

Proposition 3.8. *The possible number of realizations of a weighted 2-graph (undirected) is exactly* $\{0, 1\} \cup 2\mathbb{N}^*$.

Proof. By convention, a graph with one vertex has one realization. Now, let $G = (V, E)$ be a 2-graph with strictly more than one vertex. Let $(v_1, v_2) \in E$ be an edge such that $v_1 \neq v_2$. In this case, we will show that the number of eulerian paths of $\psi(G)$ is even and conclude using Equation 3.9. Let $\mathcal{E}_1$ (resp. $\mathcal{E}_2$) be the set of eulerian paths of $\psi(G)$ visiting v_1 then v_2 (respectively v_2 then v_1). Let $f : \mathcal{E}_1 \longrightarrow \mathcal{E}_2$ be the transformation reversing a path. f is bijective hence $\#\mathcal{E}_1 = \#\mathcal{E}_2$. Moreover, by definition of a eulerian path, $\mathcal{E}_1 \cap \mathcal{E}_2 = \emptyset$. Therefore $\#\mathcal{E} = 2\,\#\mathcal{E}_1$. □

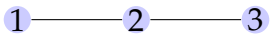
3.5 General Sequence Graphs

The characterization of 3-graphs is not the same for 2-graphs, as shows the counterexample in Fig 3.5a: the depicted graph has no self-edge so there must exist at least

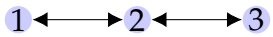
one clique of size 3, which is not the case. Similarly, Fig. 3.5b depicts a counter example for directed graphs: G does not have self-edges, so if it had a 3-admissible sequence, such sequence must be of the form

$$\{1\,2\,3\,1\ldots, 1\,3\,2\,1\ldots, 2\,3\,1\,2\ldots, 3\,2\,1\,3\ldots, 2\,1\,3\,2\ldots\}$$

but then $(3, 1)$ would form an edge.



(a) G is connected



(b) G is strongly connected

Figure 3.5 – Counter-examples for $w = 3$

3.5.1 Direct Approach

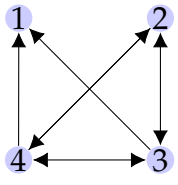
Similarly to the procedure in Sec. 3.4, we will use an auxiliary graph built on G. Let $H(G) = (E, H_E)$ be the new graph obtained with the following procedure. Two edges $e = (v_1, v_2)$, $f = (v_3, v_4)$ of E are connected in $H(G)$ if and only if:

$$v_2 = v_3 \quad \text{and} \quad (v_1, v_4) \in E \tag{3.14}$$

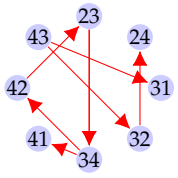
This defines an injective correspondence $E_H \to V^3$: an edge of $H(G)$ can be seen as an unique triplet v_1, v_2, v_3 where $(v_1, v_2), (v_1, v_3)$ and $(v_2, v_3) \in E$. Therefore, by definition, a walk P in $H(G)$ is always of the form:

$$P = (t_1, t_2), \ldots, (t_{p-1}, t_p) \quad \text{s.t} \quad \forall i \in \{1, \ldots, p-1\},\ (t_i, t_{i+1}) \in E \tag{3.15}$$

It is clear that if $H(G)$ is a 2-graph, then G is a 3-graph since there is a walk going through all edges of $H(G)$ (thus visiting every non isolated node and creating all edges of G). However, the converse is not true as depicted in Fig. 3.6.



(a) G



(b) $H(G)$

Figure 3.6 – G is a 3-sequence graph with $3\,4\,2\,3\,4\,1$ as a realization but $H(G)$ is not a 2-sequence graph (since it is not connected).

In order to determine if $G = (V, E)$ has an admissible sequence in the general case, a procedure is to recursively merge pairs of vertices, maintaining constraints depending on E. These constraints are similar to Eq. 3.14. We adopt the following notations, $u_{i,j} = (u_i, u_j)$ and $u_{1:k} = (u_1, ..., u_k)$. The iterative procedure for $w \geq 3$ is summed up in the following equation. Namely, $\forall k \in \{2, ..., w-2\}$, one has

$$E^{(k)} = \{u_{1:k+1} \in V^{k+1} \mid u_{1:k} \in E^{(k-1)}, u_{2:k+1} \in E^{(k-1)} \wedge (u_1, u_{k+1}) \in E\} \tag{3.16}$$

Definition 3.5. *Let $H^{(k)} = (E^{(k-1)}, E^{(k)})$, it can be defined recursively through:*

$$H^{(0)} = G \qquad\qquad \forall k \in \mathbb{N}^*, \;\; H^{(k)} = f(H^{(k-1)}) \tag{3.17}$$

where f transforms edges into vertices and creates edges between new vertices that verify Eq. 3.16.

The computation of $H^{(p)}$ requires p iterations, and the number of vertices and edges of $H^{(k)}$ can increase during iterations (the complete graph is an example for which these numbers increase exponentially).

Relation with adjoint graphs. At first sight, it might seem that $H^{(1)}$ is similar to an adjoint graph, and $H^{(k)}$ an iterated adjoint graph of G. We remind our reader that an adjoint graph (also called *derived* graph or *line* graph in the literature) is defined as graph having the edges of G as its vertices, with adjacency determined by the adjacency of the edges in G. These graphs are interesting for several reasons:

(i) They admit a characterization in terms of forbidden patterns (Beineke, 1970).

(ii) Some NP-complete problems become solvable in polynomial time when restricted to adjoint graphs, such as the stable set problem (Minty, 1980).

(iii) There exists an iteration value for which the line graph is hamiltonian.

More interestingly, (ii) is also valid for a class of graphs containing the class of adjoint graphs, called the claw-free graphs. A graph is said to be claw-free it it does not have a subgraph corresponding the pattern in Figure 3.7.

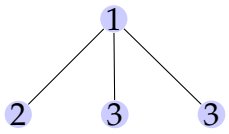


Figure 3.7 – A claw pattern

Unfortunately, for general sequence graphs, $H^{(1)}$ do not have this property. A simple counter example can be obtained considering the sequence:

$$x = 1, 2, 3, 1, 2, 4, 1, 2, 5 \tag{3.18}$$

for which $H^{(1)}(x)$ contains a claw (in the directed and undirected cases).

Definition 3.6. *Let u be a vertex of $H^{(k)}$ for $k \in \mathbb{N}$, $u = (u_1, ..., u_k, u_{k+1})$. The sequence $x = u_1, ..., u_{k+1}$ is the **authentic** sequence of u. We also call an authentic sequence of a walk on $H^{(k)}$:*

$$P = (x_1, ..., x_{k+1}), (x_2, ..., x_{k+2}), ..., (x_v, ..., x_{v+k})$$

*the sequence $x_1, x_2, ..., x_{v+k}$. Finally, P (or x) **generates** G if $\phi_w(x) = G$.*

Proposition 3.9. *Let $x = x_1, ..., x_p$ be a w-admissible sequence of a graph (or digraph) $G = (V, E)$. If $w \leq p$, x, then x is an authentic sequence of a walk of length $p - w + 1$ on $H^{(w-2)}$.*

Proof. Let $x = x_1, ..., x_p$ be a w-admissible sequence of G. In the following, if P is a walk on $H^{(w-2)}$, let $P[i]$ be the i-th element of P, $P[i] \in H^{(w-2)}$: $P[i] = (P[i]_1, ..., P[i]_{w-1})$.

Let us suppose that $w \leq p$ (which we can always do), and let us show the following property by induction on k:

$$\forall k \in \{w-1, ..., p\},\ \exists \text{ walk } P \text{ on } H^{(w-2)} \text{ such that :}$$

$$x_{1:k} = P[1]_1, P[2]_1, ..., P[k-(w-1)]_1, P[k+1-(w-1)]_{1:(w-1)}$$

• Initialization: $k = w - 1$. By construction of $H^{(w-2)}$, $x_{1:w-1}$ is the authentic sequence of the "static walk": $P = P[1] = x_{1:w-1} \in H^{(w-2)}$.
• Induction: let us suppose the property is verified for $k \in \{w-1, ..., p-1\}$, i.e there exists a walk P on $H^{(w-2)}$ such that:

$$x_{1:k} = P[1]_1, P[2]_2, ..., P[k-(w-1)]_1, P[k+1-(w-1)]_{1:(w-1)}$$

Since x is w-admissible, then by definition:

$$\forall i \in \{k+1-(w-1), ..., k\},\ \forall j \in \{i+1, ..., \mathsf{min}\{k+1, i+w-1\}\} :\ (x_i, x_j) \in E$$

Therefore, by definition of $H^{(w-2)}$, $\xi^{k+1} = x_{k+1-(w-1)}, ..., x_{k+1} \in H^{(w-2)}$.

Let $P[k+2-(w-1)] \mathrel{\widehat{=}} \xi^{k+1}$, then

$$P[k+2-(w-1)]_{1:(w-1)} = x_{k+1-(w-1)}, ..., x_{k+1}$$

Besides, from the induction assumption: $\forall i \in \{1, ..., k+1-(w-1)\},\ P[i]_1 = x_i$. This ensures that:

$$x_{1:(k+1)} = P[1]_1, P[2]_1, ..., P[k+1-(w-1)]_1, P[k+2-(w-1)]_{1:(w-1)}$$

which ends the induction and the proof. □

Theorem 3.3. *Let G a graph and $w \in \mathbb{N}^* - \{1, 2\}$. If G is undirected then* REALIZABLE$_w$ *is in P.*

Proof. An algorithm is obtained by going through all the connected components of $H^{(w-2)}$. Let $C_1, ..., C_m$ be the connected components of $H^{(w-2)}$, which can be computed in polynomial time (Hopcroft and Tarjan, 1973). For each $i \in \{1, ..., m\}$ it is possible to construct a path on C_i visiting each edge at least once in polynomial time (for instance iteratively using shortest paths). Let $W_1, ..., W_m$ such walks and $X_1, ..., X_m$ their respective admissible sequences.

Using Proposition 3.9, G is a w-sequence graph if and only if there exists a walk $\tilde{W_{i_0}}$ on some C_{i_0} whose authentic sequence generates exactly the edges of G. However, the authentic sequence X_{i_0} creates more edges than any walk on C_{i_0} by construction.

In conclusion, the assertion:

$$\exists i \in \{1, ..., m\},\ \phi_w(X_i) = G \tag{3.19}$$

is a characterization that G is a w-sequence. This assertion is decidable in polynomial time since for all i, $\phi_w(X_i)$ is computable in polynomial time (cf. Algorithm 1). □

The analogue of the aforementioned procedure for digraphs would consist in enumerating all paths in the DAG $R(H^{(w-2)})$. However, the number of paths can be exponential in this case, even for a sequence graph. This is the case from the sequence graph obtained with Example 3.1.

Example 3.1. *Let $x_{init} \in \mathbb{N}^{22}$ be the sequence*

$$x_{init} = 0, 3, 1, 2, 3, 4, 4, 2, 5, 1, 4, 6, 3, 5, 6, 6, 4, 7, 0, 8, 5, 7 \tag{3.20}$$

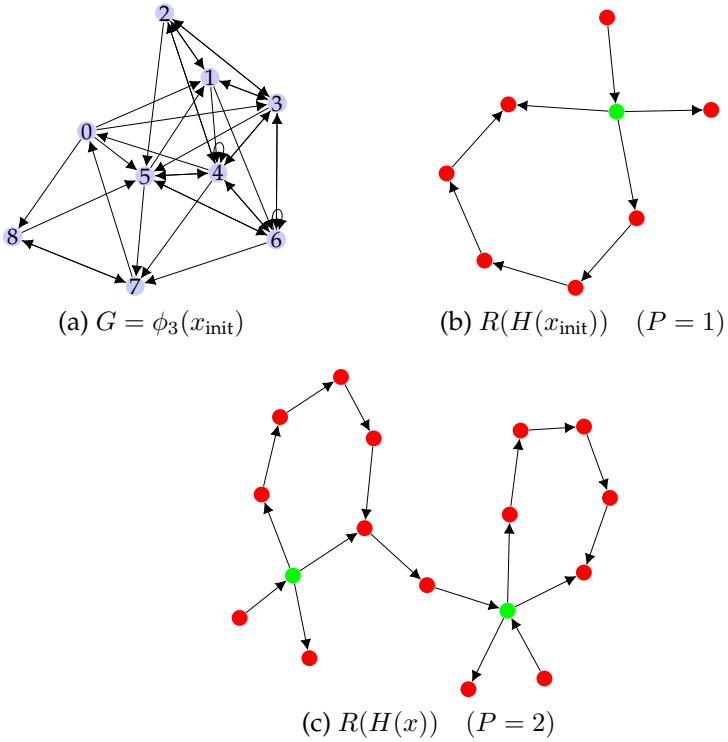
and for $P \in \mathbb{N}^*$, let $x \in \mathbb{N}^{22P}$ be the sequence be defined iteratively by:

$$\forall i \in \{0, ..., P-1\} \qquad x_{22i:22(i+1)} = (1+i)x_{init} \tag{3.21}$$

Then, with the same notations, for $w = 3$, $R(H(x))$ has at least 2^P paths.

Figure 3.8 and 3.9 highlight the exponential increase of the number of paths with respect to P. Therefore, the complexity of the algorithm used to prove Theorem 3.3 can be exponential in the directed case. Obviously, this does not prove anything regarding the complexity class of this problem, which we leave open for future work.

Finally, it should be noted that if C is a (strongly) connected component of $H^{(w-2)}$, there might exist several walks visiting (at least once) every edge. Therefore, the correspondence $C \mapsto X$ between a component and "its" authentic sequence used in the proof of Theorem 3.3 might seem ambiguous. However, the



(a) $G = \phi_3(x_{\text{init}})$ (b) $R(H(x_{\text{init}}))$ $(P = 1)$

(c) $R(H(x))$ $(P = 2)$

Figure 3.8 – Illustration of Example 3.1. 3.8a and 3.8b: Graph G and $R(H)$ associated to the sequence x_{init} are defined with Equations 3.20 and 3.21. Figure 3.8c shows $R(H(x))$ with $P = 2$. Green vertices represent strongly connected components of H, whereas red ones correspond to vertices of H. Each increment of P multiplies the number of paths by (at least) 2.

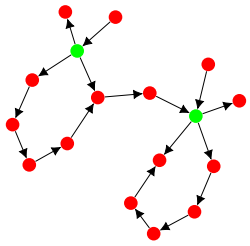
following lemma shows that choice of such representation for the component is not important as long as it visits every edge at least once. Moreover, it is possible to reconstruct all admissible sequences from walks on $R(H^{w-2})$.

Lemma 3.2. *Let x be a walk on $H^{(w-2)}$ whose authentic sequence is w-admissible for G. If x goes through a strongly component $\mathcal{C}$ of $H^{(w-2)}$, adding any supplementary path included in $\mathcal{C}$ to x leaves its authentic sequence w-admissible. Any graph generated by a walk on $H^{(w-2)}$ can be generated by a walk on $R(H^{(w-2)})$.*

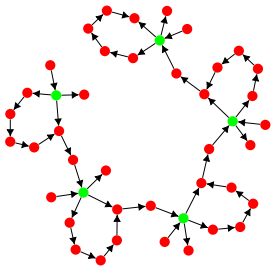
Proof. Let $\mathcal{P} = \mathcal{P}[1], , ..., \mathcal{P}[r]$ a walk on $H^{(w-2)}$ going through a strongly connected component $\mathcal{C}$, with an arbitrary ordering of its vertices: $\mathcal{C} = \{c_1, ..., c_m\}$. This means:

$$\exists (m_0, i_0) \in \{1, ..., m\} \times \{1, ..., r-1\} \quad \mathcal{P}[i_0] = c_{m_0} \quad (c_{m_0}, \mathcal{P}[i_0 + 1]) \in E^{(w-2)}$$

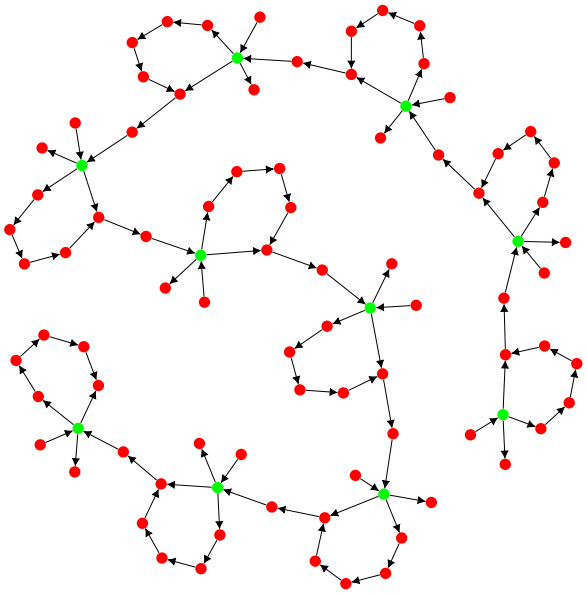
Now, let $\mathcal{P}_{\mathcal{C}} = c_{m_0}, c_{j_1}, ..., c_{j_v}$ be a path in $\mathcal{C}$ with $(c_{j_v}, P[i_0 + 1]) \in E$. Let $\mathcal{Q}$ be the new path: $\mathcal{Q} = P[1], ..., P[i_0], c_{j_1}, ..., c_{j_v}, P[i_0 + 1], ..., P[r]$. The two following properties are immediate:



(a) $P = 2$



(b) $P = 5$



(c) $P = 10$

Figure 3.9 – DAGs $R(H(x)))$ where x is defined with Eq.eqs. (3.20) and (3.21) for $P \in \{2, 5, 10\}$ and $w = 3$. Green vertices represent strongly connected components of H, whereas red ones are vertices of H. $R(H(x))$ has at least 2^P directed paths.

- By construction of $H^{(w-2)}$, the edges (between elements of V) created by any walk on $H^{(w-2)}$ are in E, in particular those generated by $\mathcal{Q}$.
- $\mathcal{Q}$ generates more edges than $\mathcal{P}$.

Therefore, if the authentic sequence of $\mathcal{P}$ is w-admissible, so is the authentic sequence of $\mathcal{Q}$. To prove the second part of the lemma, let us label every node of $R(H^{(w-2)})$ representing a strongly connected component of $H^{(w-2)}$ by any $2-$admissible sequence (one exists thanks to Proposition 3.2). A walk on $H^{(w-2)}$ say $x_1, \ldots, x_p$ can be met by a walk on $R(H^{(w-2)})$ using the following procedure:

For $i \in \{1, ..., p-1\}$:

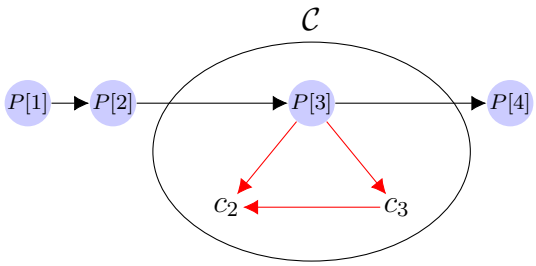


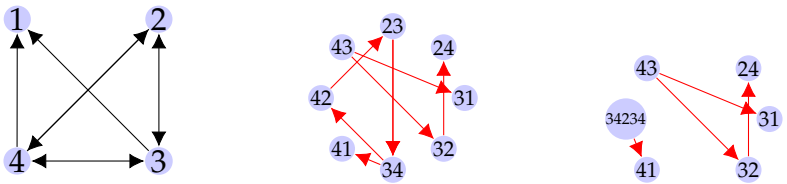
Figure 3.10 – A path (blue nodes) in $H^{(w-2)}$ going through a strongly connected component. Adding any supplementary cycle in $\mathcal{C}$ (e.g red cycle) leaves its authentic sequence admissible.

- if $x_i, x_{i+1} \in E^{(w-2)}$, we keep x_i and x_{i+1}
- if x_i is a vertex of $H^{(w-2)}$, and x_{i+1} is in a strongly connected component of $H^{(w-2)}$ (but a node of $R(H^{(w-2)})$), represented by $c_1, ..., c_{C_i}$, then a path from x_{i+1} to c_1 exists since the component is strongly connected. Let $x_{i+1}, p_1, \ldots, p_m, c_1$ be such path. Then, we keep

  $$x_i, x_{i+1}, p_1, \ldots, p_m, c_1, \ldots, c_{C_i}$$

  Using the aforementioned result, this does not perturb admissibility.
- if x_{i+1} is a vertex of $H^{(w-2)}$ and x_i is in a strongly connected component of $H^{(w-2)}$, we proceed similarly (the roles of x_i and x_{i+1} are swapped).
- if both x_{i+1} and x_i are strongly connected components of $H^{(w-2)}$, we add intermediary nodes to both components similarly.

□



(a) G (b) $H^{(1)}$ (c) $R(H^{(1)})$

Figure 3.11 – Procedure to find a 3-admissible sequence. a) G is a given digraph. b) $H^{(1)}$ is computed following Equation 3.14. c) The walk $34234,\ 41$ in $R(H^{(1)})$, which covers one of its components, generates G. Its authentic sequence is $3\,4\,2\,3\,4\,1$, and therefore is a realization of G.

Algorithm 2: A recognition algorithm for unweighted digraphs

Input: Graph G, window width w
Output: (Boolean, empty set or w-admissible sequence)
Build $H^{(w-2)}$ recursively (e.g with 3.17);
Construct $R_H^w = R(H^{(w-2)})$;
for *source-sink path in* R_H^w **do**
    if *authentic sequence of path is* w*-admissible for* G **then**
        return (True, sequence)
return (False, $\emptyset$);

3.5.2 REALIZABLE$_w$: Linear Integer Programming Formulation

Let $G = (V, E)$ be a graph with integer weights $\pi_{e \in E}$ and n be the number of vertices of G. In this model, we represent a sequence x of length p over the alphabet $\{1, ...n\}$ as a $(0-1)$ matrix $X \in \mathcal{M}_{n,p}(\{0,1\})$ encoding the sequence x:

$$X_{i,j} = \begin{cases} 1 & \text{if } x_j = i \\ 0 & \text{otherwise} \end{cases}$$

It should be noted that the set sequence of sequences over the alphabet $\{1, ...n\}$ is exactly represented by the $(0-1)$ matrices such that

$$\forall j \in \{1, ..., p\} \qquad \sum_{i=1}^{n} X_{i,j} = 1$$

Given a window size w, a unit of $\pi_{e=(v_1,v_2)}$ corresponds to the appearance of two elements v_1, v_2 at a distance $i \in \{1, ..., w-1\}$ in the sequence. Now, let us consider a fixed distance i, and a starting index $j \in \{1, ..., p-i\}$, we we use a intermediate slack variable $y_j^e(i) \in \{0, 1\}$ to model the presence of such appearance using the constraint:

$$X_{v1,j} X_{v_2,j+i} = y_j^e(i) \tag{3.22}$$

Then, the Boolean variable $y_j^e(i)$ is equal to 1 when v_1 is located at position j and v_2 at position $j+i$. We linearize Eq. 3.22 as:

$$\begin{aligned} -X_{v_1,j} \qquad\qquad + y_j^e(i) &\leq 0 \\ - X_{v_2,j+i} + y_j^e(i) &\leq 0 \\ X_{v_1,1} + X_{v_2,j+i} - y_j^e(i) &\leq 1 \end{aligned} \tag{3.23}$$

Each slack variable $y_k^e(i)$ is attributed to an edge e, a relative distance $i \in \{1, ..., w-1\}$ and a starting position $k \in \{1, ..., p-i\}$. Given our constraint formulation, every

slack variable is attributed 3 constraints. For a digraph, the number of possible pair positions for a unit of $\pi_{e=(v_1,v_2)}$ is given by:

$$C = \sum_{i=1}^{w-1}(p-i) = p(w-1) - \frac{w(w-1)}{2} = (w-1)(p-\frac{w}{2}) \tag{3.24}$$

Therefore, in our model, C corresponds to the number of slack variables attributed to the constraints for an edge of the graph.

On the contrary, the absence of an edge $e = (v_1, v_2)$, corresponding to $\pi_e = 0$, can be modeled for a distance $i \in \{1, ..., w-1\}$ and a starting position $j \in \{1, ..., p-i\}$ as:

$$X_{v1,j} + X_{v_2,j+i} \leq 1$$

Then, REALIZABLE$_w$ can be formulated as the following linear integer program:

$$\min_{X \in \{0,1\}^{p \times n}, y \in \{0,1\}^{|E| \times C}} \sum_{e \in E} \sum_{i \in \{1,...,w-1\}} y_1^e(i) + ... + y_{p-i}^e(i)$$

under the constraints

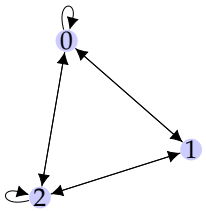
$$\forall j \in \{1, ..., p\} \qquad \sum_{i=1}^{n} X_{i,j} = 1$$

$$\forall e = (v_1, v_2) \in E \\ \forall e' = (v_1', v_2') \notin E \\ \forall i \in \{1, ..., w-1\} \quad \begin{cases} -X_{v_1,1} + y_1^e(i) \leq 0 \\ -X_{v_2,1+i} + y_1^e(i) \leq 0 \\ X_{v_1,1} + X_{v_2,1+i} - y_1^e(i) \leq 1 \qquad X_{v_1',1} + X_{v_2',1+i} \leq 1 \\ \vdots \qquad\qquad\qquad\qquad\qquad \vdots \\ -X_{v_1,p-i} + y_{p-i}^e(i) \leq 0 \quad X_{v_1',p-i} + X_{v_2',p} \leq 1 \\ -X_{v_2,p} + y_{p-i}^e(i) \leq 0 \\ X_{v_1,p-i} + X_{v_2,p} - y_{p-i}^e(i) \leq 1 \end{cases}$$

$$\text{and } \forall e \in E \qquad \sum_{i \in \{1,...,w-1\}} y_1^e(i) + ... + y_{p-i}^e(i) \geq \pi_e$$

If the objective function reaches $\sum_{e \in E} \pi_e$ at its minimum then the output of REALIZABLE$_w(G, \Pi, w)$ is True, and False otherwise. An example of instance solved by this formulation can be seen in Figure 3.12.

However, one of the limitations in this formulation is the high number of constraints combined with the dimension of the decision variables. Indeed, the problem has at least $n^2 \times C$ constraints and at least $p \times n$ decision variables, where $C = (w-1)(p-\frac{w}{2})$ (See details above Equation 3.24).



(a) Graph G

$$\begin{pmatrix} 2 & 3 & 1 \\ 2 & 0 & 2 \\ 2 & 3 & 2 \end{pmatrix}$$

(b) Matrix Π

2 2 1 2 0 0 1 0 2 1

(c) A solution

Figure 3.12 – Example of instance solved using the Linear Integer Programming formulation. The initial graph G and weights matrix Π were constructed from the sequence 2 2 0 1 0 0 1 2 2 1. The output X generated provides another solution, presented in c).

Therefore, with $w = 5$ (window width), $p = 50$ (sequence length), and $n = 20$ (number of vertices), the number of constraints is greater than 76000, and the number of decision variables is greater than 1000. In this case, the problem size make the integer programming formulation intractable.

3.5.3 NUMREALIZATIONS$_w$: Dynamic Programming Formulation

We did not present a way to count admissible sequences in the general case. Although the tractability of our problems (NP-hardness of REALIZABLE$_w$, #P-hardness of NUMREALIZATIONS$_w$) currently remains open for some cases, we present in this subsection a method based on dynamic programming valid for all cases.

The recursion proceeds by extending a partial sequence, initially set to be empty, keeping track of represented edges along the way. Namely, consider $N_w[\Pi, p, \mathbf{u}]$ to be the number of w-admissible sequences of length p for the graph $G = (V, E)$, respecting a weight matrix $\Pi = (\pi_{ij})_{i,j \in V^2}$, preceded by a sequence of nodes $\mathbf{u} := (u_1, \ldots, u_{|\mathbf{u}|}) \in V^\star$ ($V^\star$ is the set of sequences constructed on the alphabet composed of the vertices of G). It can be shown that, for all $\forall p \geq 1$, $\Pi \in \mathbb{N}^{|V^2|}$ and $\mathbf{u} \in V^{\leq w}$, $N_w[\Pi, p, \mathbf{u}]$ obeys the following formula, using the notations of Section 3.5:

$$N_w\left[\Pi, p, \mathbf{u}\right] = \sum_{v \in V} \Phi\left(\Pi'_{(\mathbf{u},v)}, p-1, \mathbf{u}\right) \tag{3.25}$$

where

$$\Phi(\Pi'_{(\mathbf{u},v)}, p-1, \mathbf{u}) = \begin{cases} N_w\left[\Pi'_{(\mathbf{u},v)}, p-1, (u_1, ..., u_{|u|}, v)\right] & \text{if } |\mathbf{u}| < w-1 \\ N_w\left[\Pi'_{(\mathbf{u},v)}, p-1, (u_2, ..., u_{w-1}, v)\right] & \text{if } |\mathbf{u}| = w-1 \end{cases} \tag{3.26}$$

and Π' is an update of the matrix Π following:

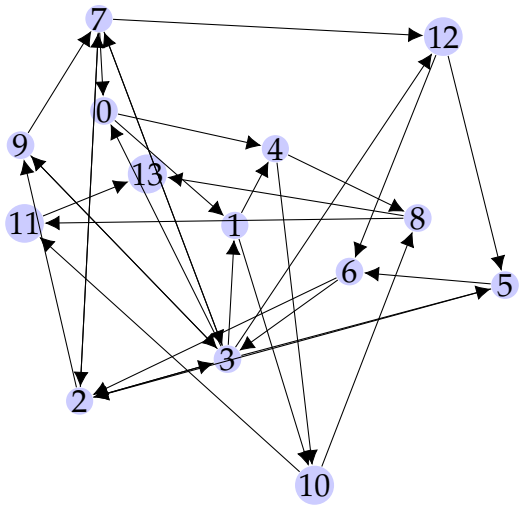
$$\Pi'_{(\mathbf{u},v)} := \left(\, \pi_{ij} - \#\left\{\, k \in \{\, 1, ..., |\mathbf{u}|\} \mid (u_k, v) = (i,j) \,\right\} \,\right)_{(i,j) \in V^2} \tag{3.27}$$

The base case of this recurrence corresponds to $p = 0$, and is initialized as:

$$\forall\ \Pi,\ \ N_w[\Pi, 0, \mathbf{u}] = \begin{cases} 1 & \text{if } \Pi = (0)_{(i,j)\in V^2} \\ 0 & \text{otherwise.} \end{cases} \tag{3.28}$$

The total number of admissible sequences is then found in $N_w[\Pi, p, \varepsilon]$, *i.e.* setting $\mathbf{u}$ to the empty prefix ε, allowing the sequence to start from any node.

A test of this method on an instance is shown in Figure 3.13, for which the dynamic programming formulation with memoization performed relatively fast (about one minute on a laptop with $2,8$ GHz Intel Core i7 4 cores and 16Gb of RAM) . Supplementary exemples are shown in the Appendix8, Section 8.1 for which computing times were at most some minutes in the same configuration.



(a) Sequence graph

$$\begin{pmatrix}
0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}$$

(b) Weights matrix Π

7 3 2 7 3 12 5 6 2 3 9 7 3 0 1 4 10 8 11 13

7 3 12 5 6 2 3 9 7 3 2 7 3 0 1 4 10 8 11 13

7 3 12 5 6 2 3 7 2 3 9 7 3 0 1 4 10 8 11 13

(c) Different realizations

Figure 3.13 – $p = 20$, $w = 3$

3.6 Complexity Study

In this section, we present a detailed analysis of the algorithmic complexity of our dynamic programming formulation, which turned out to be surprinsingly better than the worst case scenario. Indeed, this method allowed us to solve several instances with reasonable sizes such as those presented in Section 8.1.

Linear integer programming

Unlike the dynamic programming formulation, the linear integer programming formulation "complexity" is difficult to establish since the formulation does not appear to have trivial properties (e.g total unimodularity) or reductions to other known problems. In practice, we used Mixed Integer linear programming (MILP) solvers such as Branch and Bound (e.g *intlingprog* in Matlab [R2018a]). They turned out to be efficient for small values of w, p but suffered important limitations as soon as $w \geq 5$ and $p \geq 30$.

Dynamic programming formulation

We will analyse the complexity of our dynamic formulation in details. In particular, we will compare the worst case scenario complexity and the average complexity over a set of matrices Π in the directed case (the undirected case can be analyzed using the same methodology). The recurrence presented in Equations 3.25, 3.26 and 3.27 can be computed in $\mathcal{O}(|V|^w \times \prod_{i,j \in V^2}(\pi_{i,j}+1))$ time using memoization, for p the sequence length. The complexity can be refined by noting that in the directed case:

$$\sum_{i,j \in V^2} \pi_{i,j} \leq w \times p \tag{3.29}$$

Indeed, we remind our reader that in the case of directed graphs (Proposition 3.6):

$$\sum_{i,j} \pi_{ij} = (p-w+1)(w-1) + \frac{(w-1)(w-2)}{2} \tag{3.30}$$

In this case, the product $\prod_{i,j \in V^2}(\pi_{i,j}+1)$ can be bounded more precisely.

3.6.1 Preliminaries

We will start by proving useful lemmas for this section.

Lemma 3.3. *Let m_z, m, p be integers such that $0 < p \leq m$ and $m_z \leq m$. Let R_m be the positive scaled simplex of $(\mathbb{R}^+)^p$:*

$$R_m = \{(x_1, \ldots, x_p) \in (\mathbb{R}^+)^p \qquad \sum_{i=1}^{p} x_i = m\} \tag{3.31}$$

Let S_m and S_{m,m_z} be the sets of integers defined as:

$$S_m = R_m \cap \mathbb{N}^p \tag{3.32}$$

$$S_{m,m_z} = \{x \in S^m \quad \#\{\ i \in \{1,\dots,r\} \mid x_i > 0\ \} = m_z\} \tag{3.33}$$

Let $\phi : \mathbb{R} \longrightarrow \mathbb{R}$ be an affine function, i.e $\phi(x) = \alpha x + \beta$ such that $\alpha > 0$ and $\beta > 0$.

Let f be the function defined by:

$$f(x) \ = \ \prod_{i=1}^{p} \phi(x_i) \tag{3.34}$$

Finally let $\mathcal{L}$ be the line in $\mathbb{R}^p$:

$$\mathcal{L} \ = \ \{\ (x_1,\dots,x_p) \in \mathbb{R}^p \mid \ x_1 = x_2 = \dots = x_p\ \} \tag{3.35}$$

Then:

(i) $\max_{x\in R_m} f(x)$ *is reached for a point in* $\mathcal{L}$

(ii) $\min_{x\in R_m} f(x)$ *is reached for a point of the form* $(0,\dots,0,m)$

(iii) $\max_{x\in S_m} f(x)$ *is reached for a point of the form*

$$(\underbrace{1,1,\dots,1}_{m \text{ terms}},\underbrace{0,0,...,0}_{p-m \text{ terms}})$$

(iv) $\min_{x\in S_{m,m_z}} f(x)$ *is reached for a point of the form*

$$(\underbrace{1,1,\dots,1}_{m_z-1 \text{ terms}}, m-m_z+1, \underbrace{0,0,\dots,0}_{m-m_z \text{ terms}})$$

Proof. This lemma relies on the property of the logarithm:

$$\begin{aligned}&\forall(\epsilon,a,b) \in (\mathbb{R}^{+*})^3 \\ &\qquad b-\epsilon > 0 \implies \log\phi(a) + \log\phi(b) \leq \log\phi(a+\epsilon) + \log\phi(b-\epsilon)\end{aligned} \tag{3.36}$$

Which can be proved by considering the function, for $(a,b) \in (\mathbb{R}^{+*})^2$:

$$g : [0,b[\longrightarrow \mathbb{R}, \quad \epsilon \mapsto \log\phi(a+\epsilon) + \log\phi(b-\epsilon) - \log\phi(a) + \log\phi(b)$$

Since $\phi(x) = \alpha x + \beta$ with $\alpha > 0$ and $\beta > 0$, then for $a > 0$ and $b > 0$:

$$g(\epsilon) = \log(1 + \frac{\alpha\epsilon}{\phi(a)}) - \log(1 - \frac{\alpha\epsilon}{\phi(b)}) > 0$$

where the last inequality holds because α, ϵ and ϕ are strictly positive. Then, g is positive, which proves Equation 3.36.

Now, let us prove (i), (ii), (iii), (iv).

(i) Without loss of generality, let us suppose the maximum is reached for a point $x \in R_m$ such that $x_1 < x_2$. Then, for $0 < \epsilon < x_2$, using inequality 3.36:

$$\log \phi(x_1) + \log \phi(x_2) \ \leq \ \log \phi(x_1 + \epsilon) + \log \phi(x_2 - \epsilon) \tag{3.37}$$

Which yields:

$$\phi(x_1)\phi(x_2) \ \leq \ \phi(x_1 + \epsilon)\phi(x_2 - \epsilon) \tag{3.38}$$

So the value of the function f can be increased by increasing x_1 by ϵ and decreasing x_2 by ϵ (and x still belongs to R_m), until $x_1 = x_2$.

(ii) Similarly, without loss of generality, let us suppose the minimum is reached for a point $x \in R_m$ such that $0 < x_1 \leq x_2$.

Then, we can use a similar reasoning to (i).

Let $\epsilon > 0$ such that $\epsilon < x_1 \leq x_2$ and y, z be defined as $y = x_1 - \epsilon$ and $z = x_2 + \epsilon$. Then $y > 0$ and $0 < \epsilon < z$, and inequality 3.37 becomes:

$$\log \phi(y) + \log \phi(z) \ \leq \ \log \phi(y + \epsilon) + \log \phi(z - \epsilon) \tag{3.39}$$

Which corresponds to:

$$\log \phi(x_1 - \epsilon) + \log \phi(x_2 + \epsilon) \ \leq \ \log \phi(x_1) + \log \phi(x_2)$$

Composing by the exponential yields:

$$\phi(x_1 - \epsilon)\phi(x_2 + \epsilon) \ \leq \ \phi(x_1)\phi(x_2)$$

This shows that the value of f can be decreased by decreasing x_1 by ϵ and increasing x_2 by ϵ (and x still belongs to R_m).

The process can be repeated until $x_1 = 0$.

(iii) To prove (iii), let a and b two integers such that $b - a > 0$. Then, if $b > 2$, let $\epsilon = 1$, then $0 < \epsilon < b$, hence inequality 3.36 becomes:

$$\log \phi(a) + \log \phi(b) \leq \log \phi(a + 1) + \log \phi(b - 1) \tag{3.40}$$

Now, let us suppose, without loss of generality, that the minimum is reached on a point of S_m such that $0 < x_1 \leq x_2 - 1$. If $x_2 = 1$ then $x_1 = 0$ and there is nothing to prove.

Otherwise, $x_2 > 1$, with $\epsilon = 1$ then $\epsilon < x_2$, and using equation 3.40:

$$\log \phi(x_1) + \log \phi(x_2) \ \leq \ \log \phi(x_1 + 1) + \log \phi(x_2 - 1)$$

Therefore we can use the same reasoning as for (i): the value of f can be increased by increasing x_1 of 1 and decreasing x_2 of 1 (and x still belongs to S_m).

(iv) Finally, we can prove (iv) using a similar reasoning as (ii) but with with the constraint $x_1 > 1$ and $x_2 > 1$ and $x_1, x_2 \in \mathbb{N}$.

□

Lemma 3.4. *Let m, r and m_z be three strictly positive integers, such that $m_z \leq \min(m, r)$. Let S_m and S_{m,m_z} the sets be defined as:*

$$S_m = \{(x_1, ..., x_r) \in \mathbb{N}^r \qquad \sum_{i=1}^{r} x_i = m\} \tag{3.41}$$

$$S_{m,m_z} = \{x \in S^m \quad \#\{\ i \in \{1, ..., r\} \mid x_i > 0\ \} = m_z\} \tag{3.42}$$

Then the cardinals of S_m and S_{m,m_z} are given by

$$\#S_m = \binom{m + r - 1}{r - 1} \qquad \#S_{m,m_z} = \binom{r}{m_z}\binom{m - 1}{m_z - 1} \tag{3.43}$$

Proof. We will start by proving the following assertions:

(i) Let $\mathcal{I}_{m_z,m}$ be the set of ***strictly increasing*** functions from $\{1, ..., m_z\}$ to $\{0, ..., m\}$. Then, $\#\mathcal{I}_{m_z,m} = \binom{m+1}{m_z}$. Let $\mathcal{I}^*_{m_z,m}$ be the subset of $\mathcal{I}_{m_z,m}$ composed of functions having no zero. Then, $\#\mathcal{I}^*_{m_z,m} = \binom{m}{m_z}$.

(ii) Let $\mathcal{J}_{r,m}$ be the set of ***non decreasing*** functions from $\{1, ..., r\}$ to $\{0, ..., m\}$. Then $\#\mathcal{J}_{r,m} = \binom{m+r}{r}$.

(iii) $\mathcal{J}_{r,m} \simeq \bigcup_{i=1}^{m} S_i$. If S_i^* is the subset of S_i of strictly positive integers, then $\mathcal{I}^*_{m,m_z} \simeq \bigcup_{i=1}^{m} S_i^*$.

(i) The image of a strictly increasing function from $\{1, ..., m_z\}$ to $\{1, ..., m\}$ is a subset of $\{0, ..., m\}$ of cardinal m_z. Given a subset S of m_z elements of $\{1, ..., m\}$, there exists a unique increasing function from $\{1, ..., m_z\}$ to S. Therefore there is a bijection between strictly increasing functions from $\{1, ..., m_z\}$ to $\{0, ..., m\}$ and the subsets of m_z elements of $\{0, ..., m\}$, whose cardinal is $\binom{m+1}{m_z}$, hence $\#\mathcal{I}_{m_z,m} = \binom{m+1}{m_z}$. Similarly, $\mathcal{I}^*_{m_z,m}$ is in bijection with the subsets of m_z elements of $\{1, ..., m\}$, so $\#\mathcal{I}^*_{m_z,m} = \binom{m}{m_z}$

(ii) Let us consider the function $\xi : \mathcal{J}_{r,m} \longrightarrow \mathcal{I}_{r,r+m-1}$ defined by:

$$\xi(f)[x] = f(x) + x - 1 \tag{3.44}$$

Then it is easy to verify that ξ is a bijection, with $\xi^{-1}(g)[x] = g(x) - x + 1$
Therefore $\mathcal{J}_{r,m} \simeq \mathcal{I}_{r,r+m-1}$ and $\#\mathcal{J}_{r,m} = \#\mathcal{I}_{r,r+m-1} = \binom{m+r}{r}$ using (i).

(iii) To each sequence $(x_1, ..., x_r)$ verifying $x_1 + ... + x_r \leq m$ and $x_1, ..., x_r \in \mathbb{N}$ a corresponds a unique increasing function $(f(x_1) = y_1, ..., f(x_r) = y_r)$ defined by

$$y_k = x_1 + ... + x_k \tag{3.45}$$

Conversely, to an increasing function $(f(x_1) = y_1, ..., f(x_r) = y_r)$ corresponds a unique sequence $(x_1, ..., x_r)$ defined by

$$x_1 = y_1 \quad \text{and} \quad x_k = y_k - y_{k-1} \tag{3.46}$$

verifying $\forall k \in \{1, ..., r\}, x_k \in \mathbb{N}$ and $x_1 + ... + x_r = y_r \leq m$

This proves $\mathcal{J}_{r,m} \simeq \bigcup_{i \leq m} S_i$. Using the same reasoning, $\mathcal{I}^*_{m,m_z} \simeq \bigcup_{i=1}^{m} S^*_i$.

Now, let us prove that $\#S_m = \binom{m+r-1}{r-1}$.

Let $T_m = \bigcup_{i \leq m} S_i$. Since $S_m = T_m \setminus T_{m-1}$ and $T_m \cap T_{m-1} = \emptyset$

$$\begin{aligned} \#S_m &= \#T_m - \#T_{m-1} \\ &= \#\mathcal{J}_{r,m} - \#\mathcal{J}_{r,m-1} \\ &= \binom{m+r}{r} - \binom{m+r-1}{r} \\ \#S_m &= \binom{m+r-1}{r-1} \end{aligned} \tag{3.47}$$

Finally, we will prove that $\#S_{m,m_z} = \binom{r}{m_z}\binom{m-1}{m_z-1}$.

Setting m_z positions $i_1, ..., i_{m_z} \in \{1, ..., r\}$, there are by definition exactly $\#\mathcal{S}^*_{m_z} = \#\mathcal{I}^*_{m_z} - \#\mathcal{I}^*_{m_z-1} = \binom{m}{m} - \binom{m-1}{m_z} = \binom{m-1}{m_z-1}$ corresponding elements of S_{m,m_z}. Considering all the possible positions, finally $\#S_{m,m_z} = \binom{r}{m_z}\binom{m-1}{m_z-1}$. □

Lemma 3.5. *With the same definitions as in Lemma 3.4, for any positive integers m_z, m such that $m_z \leq m$, if $x \in S_{m,m_z}$ then:*

$$2^{m_z-1}(m - m_z + 2) \ \leq \ \prod_{i=1}^{r}(x_i + 1) \ \leq \ (\frac{m}{m_z} + 1)^{m_z} \tag{3.48}$$

Proof. This can be proved directly from Lemma 3.3. □

3.6.2 Main Complexity Results

Proposition 3.10. *Let $\mathcal{W}_C$ be the worst case complexity of our dynamic programming formulation (Section 3.5.3) in the directed case. If w is the window width and p the length of the considered sequences, then:*

$$\mathcal{W}_C \in \mathcal{O}(|V|^w\, 2^{wp}) \tag{3.49}$$

Proof. We already know that $\mathcal{W}_C \in \mathcal{O}(|V|^w \times \prod_{i,j\in V^2}(\pi_{i,j}+1))$.

Let m be defined as:

$$m \triangleq (p-w+1)(w-1) + \frac{(w-1)(w-2)}{2}$$

Then we already mentioned (Proposition 3.6):

$$\sum_{i,j} \pi_{ij} = m$$

Which implies $m \leq wp$, and the worst case complexity can be upper-bounded by estimating

$$M = \max_{\Pi\in S_m} \prod_{i,j\in V^2} (\pi_{i,j}+1)$$

Let $\phi : x \mapsto (x+1)$. Then, ϕ verifies the conditions of Lemma 3.3. Therefore the previous maximum is reached for a point where

$$\pi_{i,j} = 1 \quad \text{or} \quad \pi_{i,j} = 0$$

Hence:

$$M \leq 2^m \leq 2^{wp}$$

□

Thus, despite the apparently extremely high complexity of our algorithm, it is still possible to compute $N_w[\Pi, p, u_{1:w}]$ for "reasonable" values of p and w, even in the worst case scenario.

Now, let us consider an estimation of the average complexity of the dynamic programming formulation. We will consider as a first approximation the randomness of Π and fix the size of the graph instance. Therefore, we are interested in

$$\mathcal{E}_C = \mathbb{E}_\Pi \left[\prod_{i,j\in V^2} (\pi_{i,j}+1) \right] \tag{3.50}$$

where the coefficients $\pi_{i,j}$ are distributed uniformly on the scaled simplex:

$$\sum_{i,j\in V^2} \pi_{i,j} = w\, p \tag{3.51}$$

The final average complexity will be given by:

$$\mathcal{A}_{\mathcal{C}} = V^w \mathcal{E}_{\mathcal{C}} \tag{3.52}$$

We have already proved that $\mathcal{A}_{\mathcal{C}} \leq 2^n$ from the worst case complexity evaluation. The objective of this section is to obtain a tighter upper-bound and show it is significantly lower than the worst case scenario.

Proposition 3.11. *In the conditions of our complexity model where the matrix weights are seen as uniform random variable on the scaled simplex (Eq. 3.51):*

$$\mathcal{L}(wp, n^2) \leq \mathcal{E}_{\mathcal{C}} \leq \mathcal{U}(wp, n^2) \tag{3.53}$$

where

$$\begin{aligned} \mathcal{L}(m,r) &= \frac{1}{\binom{m+r-1}{r-1}} \sum_{m_z=1}^{\min(r,m)} \binom{r}{m_z}\binom{m-1}{m_z-1} 2^{m_z-1}(m-m_z+2) \\ \mathcal{U}(m,r) &= \frac{1}{\binom{m+r-1}{r-1}} \sum_{m_z=1}^{\min(r,m)} \binom{r}{m_z}\binom{m-1}{m_z-1} (\frac{m}{m_z}+1)^{m_z} \end{aligned} \tag{3.54}$$

Proof. Recall that n represents the number of distinct elements of the sequence, p the sequence length and w the window width. Let us consider two strictly positive integers m (representing $w\,p$) and r (representing n^2). Since $n \leq p$, we suppose that $w\sqrt{r} \leq m$. Let S_m be the subset of $\mathbb{N}^r$ defined as:

$$S_m = \{(x_1, ..., x_r) \in \mathbb{N}^r \qquad \sum_{i=1}^{r} x_i = m\} \tag{3.55}$$

The idea of our method is to decompose S_m in a partition where more precise bounds of $E_{\mathcal{C}}$ can be derived. In the following, let m_z be a strictly positive such that $m_z \leq \min(r, m)$ representing a number of non zeros integers, and S_{m,m_z} be the subset of S_m defined as

$$S_{m,m_z} = \{x \in S^m \quad \#\{\ i \in \{1, ..., r\} \mid x_i > 0\ \} = m_z\} \tag{3.56}$$

Then, the S_{m,m_z} define a partition of S_m in the following sense:

$$S_m = \bigcup_{m_z=1}^{r} S_{m,m_z} \qquad \text{and} \qquad \forall m_z \neq m'_z \qquad S_{m,m_z} \cap S_{m,m'_z} = \emptyset \tag{3.57}$$

Using Lemma 3.4:

$$\#S_m = \binom{m+r-1}{r-1} \qquad \#S_{m,m_z} = \binom{r}{m_z}\binom{m-1}{m_z-1} \tag{3.58}$$

Then, if $x_1, ..., x_r$ are sampled uniformly on S_m, we obtain:

$$\begin{aligned}\mathbb{E}\,[\,\prod_{i=1}^{r}(x_i+1)\,] &= \frac{1}{\binom{m+r-1}{r-1}}\sum_{x\in S_m}\prod_{i=1}^{r}(x_i+1)\\ &= \frac{1}{\binom{m+r-1}{r-1}}\sum_{m_z=1}^{p}\sum_{x\in S_{m,m_z}}\prod_{i=1}^{r}(x_i+1)\end{aligned} \tag{3.59}$$

Using Lemma 3.5, $x \in S_{m,m_z}$ implies that

$$2^{m_z-1}(m-m_z+1) \;\leq\; \prod_{i=1}^{r}(x_i+1) \;\leq\; (\frac{m}{m_z}+1)^{m_z} \tag{3.60}$$

Therefore:

$$\begin{aligned}\mathbb{E}\,[\,\prod_{i=1}^{r}(x_i+1)\,] &\leq \frac{1}{\binom{m+r-1}{r-1}}\sum_{m_z=1}^{r}\#S_{m,m_z}(\frac{m}{m_z}+1)^{m_z}\\ &\leq \frac{1}{\binom{m+r-1}{r-1}}\sum_{m_z=1}^{r}\binom{r}{m_z}\binom{m-1}{m_z-1}(\frac{m}{m_z}+1)^{m_z}\end{aligned} \tag{3.61}$$

Let $\mathcal{U}(m,r)$ be defined as the right hand side of the above inequality:

$$\mathcal{U}(m,r) = \frac{1}{\binom{m+r-1}{r-1}}\sum_{m_z=1}^{r}\binom{r}{m_z}\binom{m-1}{m_z-1}(\frac{m}{m_z}+1)^{m_z} \tag{3.62}$$

Similarly,

$$\mathbb{E}\,[\,\prod_{i=1}^{r}(x_i+1)\,] \geq \frac{1}{\binom{m+r-1}{r-1}}\sum_{m_z=1}^{r}\binom{r}{m_z}\binom{m-1}{m_z-1}2^{m_z-1}(m-m_z+2) \tag{3.63}$$

Let $\mathcal{L}(m,r)$ be defined as the right hand side of the above inequality:

$$\mathcal{L}(m,r) = \frac{1}{\binom{m+r-1}{r-1}}\sum_{m_z=1}^{r}\binom{r}{m_z}\binom{m-1}{m_z-1}2^{m_z-1}(m-m_z+2) \tag{3.64}$$

Then

$$\mathcal{L}(m,r) \;\leq\; \mathbb{E}\,[\,\prod_{i=1}^{r}(x_i+1)\,] \;\leq \mathcal{U}(m,r) \tag{3.65}$$

Considering the initial problem parameters:

$$m = w\,p$$
$$r = |V|^2 = n^2$$

We have just shown:

$$\mathcal{L}(wp, n^2) \leq \mathcal{E}_{\mathcal{C}} \leq \mathcal{U}(wp, n^2) \tag{3.66}$$

□

Now, let us make the assumption: $\alpha \in]0, 1]$, which controls the number of repetitions of symbols in the sequence:

$$n = \alpha p \tag{3.67}$$

Then:

$$\mathcal{L}(w\,p, \alpha^2 p^2) \leq \mathcal{E}_{\mathcal{C}} \leq \mathcal{U}(w\,p, \alpha^2 p^2) \tag{3.68}$$

Recall the expression of the average and worst case complexities (respectively noted $\mathcal{A}_{\mathcal{C}}$ and $\mathcal{W}_{\mathcal{C}}$):

$$\mathcal{A}_{\mathcal{C}} = n^w \mathcal{E}_{\mathcal{C}} \qquad \mathcal{W}_{\mathcal{C}} = n^w 2^{w\,p} \tag{3.69}$$

Then it is possible to estimate a lower bound of the ratio between the worst case complexity and the average complexity:

$$\frac{2^{w\,p}}{\mathcal{U}(w\,p, \alpha^2 p^2)} \tag{3.70}$$

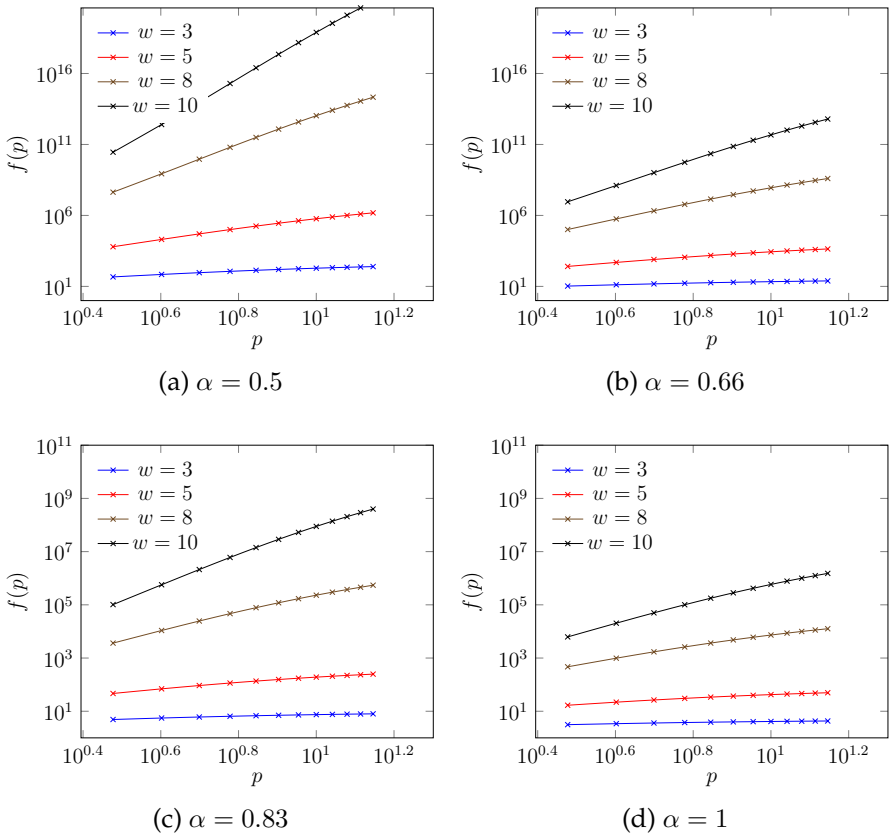
This ratio is displayed on Fig. 3.14 for $p \leq 15$ and several values of α and w. It appears clearly that the average complexity is significantly lower than worst case complexity, which explains the behavior of the algorithm in practice, significantly better than the worst case scenario.

For a fair description of our dynamic programming algorithm, we also have represented in Figure 3.15 lower bounds of the average complexity factor $\mathcal{E}_{\mathcal{C}}$ for the same range of parameters α, w and p, i.e the function:

$$\mathcal{L}(w\,p, \alpha^2 p^2) \tag{3.71}$$

For reasons of visibility, we have only considered the factor $\mathcal{E}_{\mathcal{C}}$, which should normally be multiplied by n^w (cf Eq. 3.69).

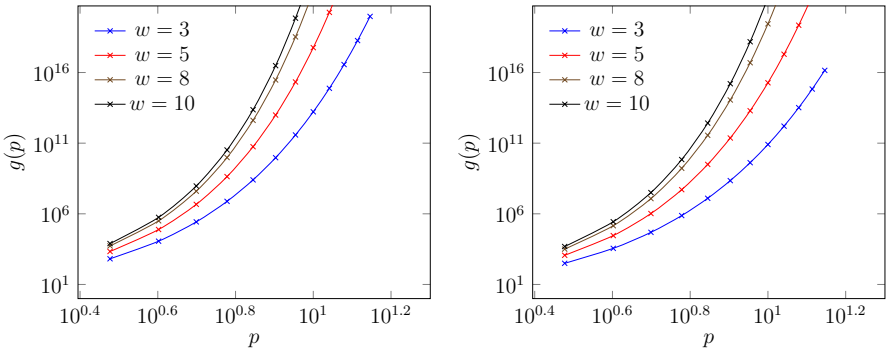
As a conclusion of this complexity study, our dynamic programming formulation behaves significantly better in the average case than in the worst case scenario, but its complexity still suffers exponential complexity. In practice, as soon as $w \geq 10$ and $n \geq 300$, time and memory issues are encountered (the memory required in that case exceeds 32Gb, even with memoization).



(a) $\alpha = 0.5$ (b) $\alpha = 0.66$

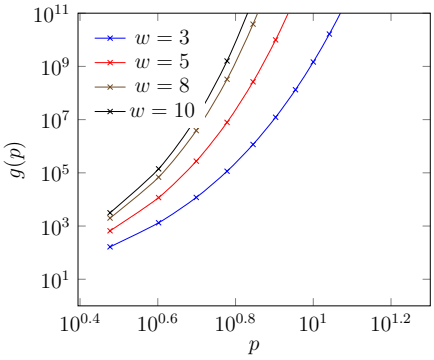
(c) $\alpha = 0.83$ (d) $\alpha = 1$

Figure 3.14 – Average complexity $(\mathcal{A}_\mathcal{C})$ vs. worst case complexity $(\mathcal{W}_\mathcal{C})$ of the dynamic programming formulation. Curves represent a lower bound of the ratio $\frac{\mathcal{W}_\mathcal{C}}{\mathcal{A}_\mathcal{C}}$ corresponding to $f : p \mapsto \frac{2^{wp}}{H(wp,\alpha^2 p^2)}$ for several values of w and α (log − log scale). The average complexity is significantly lower than worst case complexity.

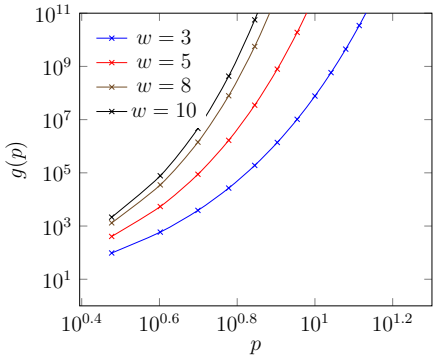


(a) $\alpha = 0.5$

(b) $\alpha = 0.66$



(c) $\alpha = 0.83$



(d) $\alpha = 1$

Figure 3.15 – Lower bound of average complexity $\mathcal{A}_{\mathcal{C}}$ of the dynamic programming formulation, displayed with: $g : p \mapsto \mathcal{L}(w\,p, \alpha^2 p^2)$ for several values of w and α (log – log scale). The average complexity suffers from exponential cost. For reasons of readability, we have only considered the factor $\mathcal{E}_{\mathcal{C}}$, which should normally be multiplied by n^w (cf Eq. 3.69).

3.7 Application to Sequential Models

3.7.1 Number of Equivalent Sequences for Weighted Sequence Digraphs

Since the dynamic programming method in Sec. 3.5.3 is exponential in the worst case, we provide results for relatively short sequences generated from text data (a dump of english Wikipedia, 2016) of 500 documents, each of them having a length $p \in \{50, 100, 150\}$. Each document contained a minimum of $\frac{3}{4}p$ distinct words. For each $w \in [3, 10]$, we estimate the number of admissible sequences yielding the same representations for a set of documents and different window size using the procedure described in Sec. 3.5.3 to compute N_w. It should be noted that N_w is a lower bound of the number of total admissible sequences, since a starting pattern is required (the first w tokens).

Results are reported in the first plot of Figure 3.16. For $w = 2$, the number of sequences (obtained using Proposition 3.7) was significantly larger ($> 10^5$), so not reported in the figure for clarity. As expected, the number of distinct admissible sequences tends to 1 when the window size increases. This suggests that window sizes used in co-occurrence based models should be usually larger or equal to 5. In natural language processing, a frequent configuration is $w = 10$ (Pennington et al., 2014; Sanjeev et al., 2017). However, some examples with different realizations exist, even for $w = 10$ and $p = 50$.

3.7.2 Comparison with a Recurrent Neural Network

The second experiment we consider is to evaluate the difference of the parameters between a sequential model trained on two admissible sequences of a given graph. The sequential model we are considering are a class of recurrent dynamical recurrent models, referred to as long short term memory (LSTM) networks (Hochreiter and Schmidhuber, 1997). These models have attracted new interest due to experimental progress for time series prediction (Greff et al., 2016) and natural language processing (Bartunov et al., 2016; Chen et al., 2016; Vaswani et al., 2017). LSTMs are a class of recurrent neural networks:

$$h_t = f(W_x W_x^{x_t} + W_h h_{t-1} + b) \tag{3.72}$$

where f is the activation function,W_x, W_h and b are parameters, and x_t is the t-th token in the sentence. LSTM is a recurrent neural network architecture designed to capture long-distance dependencies. The set of parameters we compare are the set of parameters (independent of time t) which fully defined the transitions (they do not exactly correspond to W_x, $W_x^{x_t}$ and b but are easily available from the implementations).

Given a window size w, the task we consider is to predict the next element of the sequence given the $w-1$ previous ones. If the sequences were equivalent for the sequential model, the model weights should, if not converge, be similar after training.

To generate pairs of admissible sequences encoding for non trivial graphs (i.e not the complete graph), we use an algorithm based on Lemma 3.2. We generate w-admissible sequences (thousand tokens long), for $w \in \{2, 3\}$, but could not provide other pairs for $w > 3$ due to excessive computational time. We compare the pairs of admissible sequences with a pair of one of the sequences, and a sequence generated randomly uniformly on the same vocabulary. We implemented the LSTM network using the Python library Keras (Chollet et al., 2015), a high-level API running over TensorFlow (Abadi et al., 2015). In order to remove randomness from the training algorithm, we froze the seed generating initial weights (the optimization directions being fixed by the data). We chose $\tanh$ as main activation function, sigmoid for the recurrent activations, with 2 units. The number of units is chosen relatively low in order to obtain a reasonable number of weights (in this case 16).

Two last plots of Fig. 3.16 report the results for $w \in \{2, 3\}$, W_x represents all the weights of the network for a sequence x. For $w = 2$, the norm of the difference between the weights for 2 admissible sequences is lower than with one of the sequence and a random one, but this proximity does not appear to be significant compared with a random sequence. For $w = 3$, the recurrent network has relatively close weights for two admissible sequences when compared with a random one.

3.8 Conclusion

In this preliminary study, we presented some theoretical results and practical algorithms for the family of sequence graphs. The problems REALIZABLE$_w$ and NUMREALIZATIONS$_w$ seem to be devoid of reductions, although the apparent connections between REALIZABLE$_w$ with vertex ordering problem in Distance Geometry (Omer and Mucherino, 2020).

We applied these results to experiments for sequential models, with a focus on natural language modeling. This study can be of use used for several sequential models, such as continuous bag of words (CBOW), skip-grams ((Mikolov et al., 2013a; Goldberg and Levy, 2014; Song et al., 2018)), pointwise mutual information models (Pennington et al., 2014) and generative probabilistic models (Arora et al., 2016a; Sanjeev et al., 2017). These experiments suggest that ambiguity induced by graphical representations are not present with recurrent neural networks, suggesting a semantic difference for the initial considered sequences.

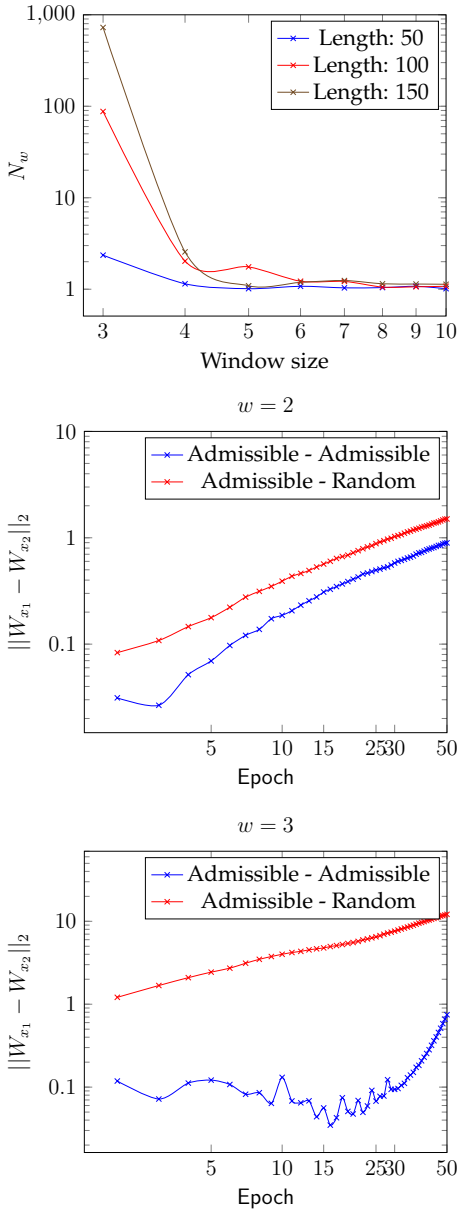


Figure 3.16 – The first plot displays the average lower bound (N_w) on the average number of realizations, computed for 500 sequence graphs, as a function of the window size. Two next plots (first one for $w = 2$ and second for $w = 3$) display the norm of the difference of the weights for two sequences (either two admissible sequences, or an admissible sequence and a random one), i.e $||W_{x_1} - W_{x_2}||_2$ as function of the Epoch, during the training of a LSTM. All plots are in $\log - \log$ scale.

The computational complexity hardness of the considered problems remains open for some instances, and we plan to address their complexity in future work,

for instance by exploiting the structural properties of sequence graphs (*e.g.* existence of forbidden patterns).

4

Entity Identification

In the previous chapter we highlighted the degree of ambiguity induced by graph-based representations of sequences. This degree of ambiguity can be problematic when the problem at hand requires an accurate resolution of the representations used as input of learning algorithms. In this chapter, we are interested in demonstrating that such representations can nevertheless be very effective for an information extraction problem. To do so, we present a problem consisting in identifying named entities from textual data, referred to as named entity identification. We will show that this problem can be solved efficiently using a graph-based approach; namely, the sequence graphs studied in Chapter 3 will be used as representations for named entity identification.

Firstly, the problem of named entity identification (or named entity linking) we are concerned with is defined as follows: we assume a set of queries, named entities, that have to be identified within a knowledge base. This knowledge base is represented by a text database paired with a semantic graph, endowed with a classification of entities (ontology). Besides, we also made an empirical analysis of a knowledge base representing an Ownership network. The motivation of this analysis was to apply the methods studied in this chapter for named entity identification. This knowledge base represents a network of entities, endowed with a simple ontology (individuals and organizations). However, the content of this Knowledge Base, and in particular the text description of the entities, was found to be insufficient we could note use this knowledge graph as a sufficient support for named entity identification.

In sections 4.1 to 4.5, we present state-of-the-art methods in named entity identification, and propose a new method for individual identification requiring few annotated data samples. We demonstrate its scalability and performance over standard datasets, for several ontology configurations.

Our approach is well-motivated for integration in real systems. Indeed, recent deep learning methods, despite their capacity to improve experimental precision, require lots of parameter tuning along with large volume of annotated data.

4.1 Introduction

4.1.1 Basic Concepts and Definitions

The structure of a document can be analysed under the prism of the role of central entities in the text. The purpose of *Named entity discovery* (NED) in information retrieval is two-fold. First, it aims at extracting pre-defined sets of words from text documents: this corresponds to Named entity recognition (NER). These words are representations of *named entities* (such as names, places, locations, ...). Then, these *entity mentions* paired with their context are seen as *queries* to be identified within a database: this corresponds to named entity identification or named entity linking (NEL). NEL is also refered as named entity disambiguation. The interest in NEL has grown recently in several fields: in bioinformatics, to obtain locations of viral sequences from databases Weissenbacher et al. (2015), or to process biomedical litterature Zheng et al. (2015). It also revealed to be useful in recruitment in order to identify employer names in a database Liu et al. (2018).

Firstly, it is important to stress that the subtask of NED, Named entity recognition (NER), is not trivial since we do not have an exhaustive list of the possible spellings of named entities. Moreover their text representation can change (for example, “J. Kennedy" vs. “John Kennedy"). In this chapter we focus on the second task, *Named entity linking (NEL)*. This motivates a more precise definition of a ***Named entity (and Mention/Query)***. An entity is a real-world object and usually has a physical existence. It is denoted with a proper name. In the expression “Named Entity", the word “Named" aims to restrict the possible set of entities to only those for which one or many rigid designators stands for the referent (Nadeau and Sekine, 2007). When a named entity appears in a document, its surface form can also be refered as a *mention*. Finally, a *query* refers to the mention, the context where it appears, and a type of entity considered.

Besides, the classification of these entities leads to the notion of ***Ontology***. In this chapter, an ontology is represented as a tree of entity types. In the following, the variable $\boldsymbol{T}$ represents the total number of nodes of this tree minus one (we do not count the root node since it is uninformative). Originally, entities had a very limited number of types (Nadeau and Sekine, 2007), such as person (*PER*), organization (*ORG*), and localization (*GPE*) (and in that case, $\boldsymbol{T} = 3$). These types play a central role for named entity recognition and identification. An example of ontology is given in Fig. 4.1. More recently, due to the increase in the volume of the Web data, fine-grained classifications are available, with hundreds of entity types. DBPedia[1] (Lehmann et al., 2015) is an example of such fined-grained classification.

The identification of named entities thereby relies on background knowl-

[1] http://wiki.dbpedia.org/services-resources/ontology

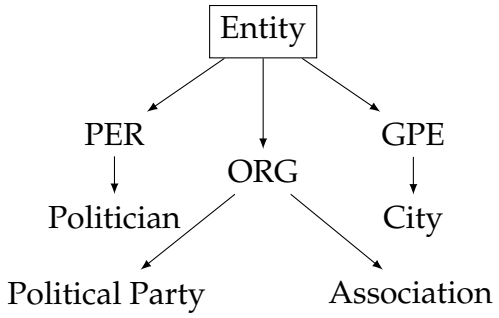


Figure 4.1 – Example of ontology, $\boldsymbol{T} = 7$

edge, usually constructed in a ***Knowledge base/graph:*** A knowledge base is a database providing supplementary descriptive and semantic information about entities. The semantic information is contained in a knowledge graph, where a node represents an entity, and an edge represents a semantic relation. The knowledge graph can be of any kind (directed, weighted, ...). See Figure 4.2 for an example.

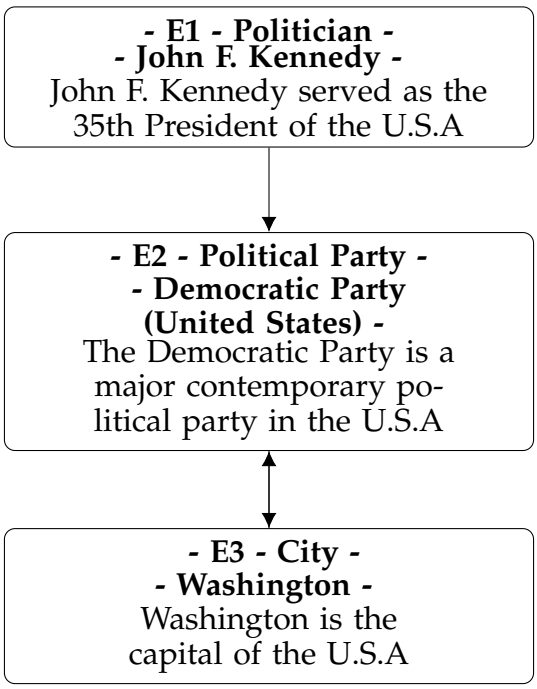


Figure 4.2 – Representation of an *unweighted directed semantic graph* (Wikipedia/NIST TAC-KBP Challenge 2010). An edge between two entities E_1 and E_2 represents a url link from E_1 web page to E_2 web page.

Given a named entity query, the purpose of ***Named entity linking (NEL)*** is to identify the corresponding ground truth entity (*gold entity*) in a database (*knowledge*

base). For a detailed description of a concrete competition in entity linking, we refer to (Ji et al., 2014).

Linking can be done *individually* or *collectively* (***Individual and collective linking***. In the first case, queries are independent. In the collective framework, we consider a set of queries that usually originates from the same document, and for which gold entities (i.e *ground truth* entities) should have some proximity, or coherence. In this chapter, we propose individual linking approach.

4.1.2 Contributions

In this chapter, we provide a brief survey of existing methods for named entity linking. Then, we investigate a method for individual named entity linking. The first step of this method, refered as *entity filtering*, reduces entity candidates to top $\boldsymbol{K}$ entities for one *query*. The second step, refered as *entity identification*, aims at identifying the true entity among the remaining $\boldsymbol{K}$ candidates, based on a new graph-based algorithm. We include an experimental evaluation of our method with several datasets, with an analysis of the impact of parameter $\boldsymbol{K}$, the ontology parameter $\boldsymbol{T}$, and a detailed comparison with existing approaches. The implementation used for experiments are available at our repository[2].. We do not include NIL-detection problem (detect if a query is referring to an entity that is not in the knowledge base, for instance (Ji et al., 2014)).

4.1.3 Another Example of Knowledge Base: Ownership network

The construction of knowledge bases represent an important challenge in the field of Information Retrieval. Their structure allow to address other tasks including Question Answering (QA) or relation extraction (Ji and Grishman, 2011). As mentioned in the Section 4.1.1, we consider DBPedia (Lehmann et al., 2015) in our experiments as the knowledge base for named entity identification.

Besides, we also made an exploratory analysis of another knowledge base representing an Ownership network (cf Illustration in Figure 4.3). The motivation of this analysis was to apply the methods studied in this chapter for named entity identification. This knowledge base represents a network of entities, endowed with a simple ontology (individuals and organizations). However, the content of this Knowledge Base, and in particular the text description of the entities, was found to be insufficient we could note use this knowledge graph as a sufficient support for named entity identification. For the sake of consistency with the subject of this Chapter, we leave in the Section 8.2 of the Appendix the discussion of the methodology and results of this analysis. The presented methodology can be

[2]Link to repository

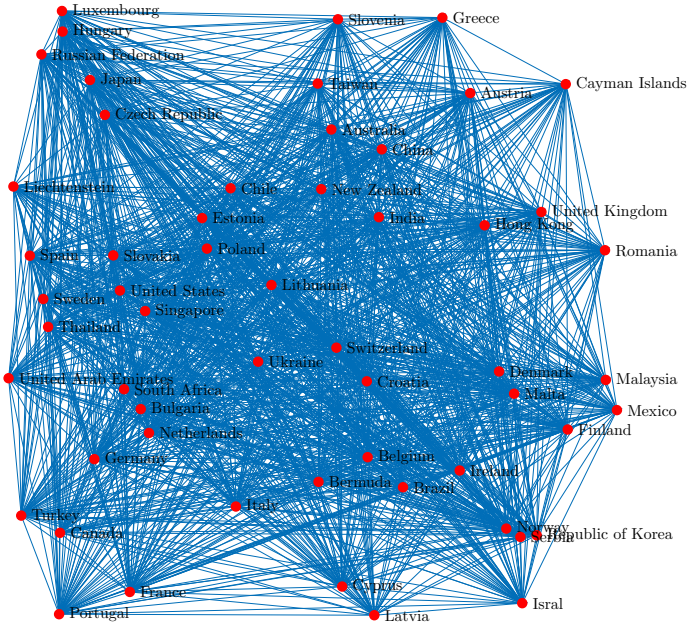


Figure 4.3 – Illustration an ownership network knowledge graph. The analysis is provided in the Appendix, Section 8.2. This subgraph represents a dense community of entities agregated by country (*Orbis* database)

used in the case of pre-processing and understanding of a knowledge base. An extended version of this work is available in (Khalife et al., 2019b).

4.2 Related Work

In the following subsections, we present three families of algorithms for named entity linking. In the remaining of this chapter, we adopt the following notations: $\boldsymbol{E} = \{1, ..., E\} \subset \mathbb{N}$: indexes of entities and $\boldsymbol{Q} = \{1, ..., Q\} \subset \mathbb{N}$: indexes of queries, $\hat{e}_i$: system's output entity index for query index q_i.

4.2.1 Graphs for NEL

Formulation: Given a scoring function measuring a form of similarity between a query and an entity, let $W_{i,j}$ be the corresponding score between the query i and the entity j. For individual disambiguation, one wants to perform independent query-entity attribution. A straightforward formulation is:

$$\hat{e}_i = \arg\max_{j \in E} W_{i,j} \tag{4.1}$$

In this case, the total cost is separable in the variable i, but the score $W_{i,j}$ can use the knowledge graph structure: this is the case in our approach. For the sake of

completeness, we give a description of the collective linking formulation. In this framework the optimization formulation is different: the underlying *gold entities* should respect some arbitrary semantic coherence. The coherence information is represented within a *coherence function* $\psi : \boldsymbol{E}^Q \rightarrow \mathbb{R}$ between the entity candidates. Usually ψ is defined from the knowledge graph structure. For example ψ can be defined using the opposite sign of the shortest-path function on the *knowledge graph*. With these notations, the set of selected entities are formally defined as:

$$\hat{e_1}, ..., \hat{e_Q} = \underset{j_1,..,j_Q \in E^Q}{\arg\max} [(\sum_{l=1}^{Q} W_{l,j_l}) + \psi(j_1, ..., j_Q)] \tag{4.2}$$

Eq. (4.2) can be formulated as a Boolean integer program. Its *NP-hardness* Cucerzan (2007) does not allow to solve the general case for an important number of queries. Ratinov et al. (2011) evaluated local and global approaches to find solutions of an approximation of Eq. (4.2) with Eq. (4.3), given a new coherence function $\tilde{\psi}$, and for each query indexed by l a disambiguation context of entities C_l:

$$\hat{e_1}, ..., \hat{e_Q} = \underset{j_1,..,j_q \in E^Q}{\arg\max} [(\sum_{l=1}^{Q} W_{l,j_l} + \sum_{k \in C_l} \tilde{\psi}(j_l, j_k))] \tag{4.3}$$

The formulation with Eq. (4.3) is halfway between individual and collective linking: it suggests to select a convenient set of disambiguation contexts, and then solving locally for each query. In the same time, it still enforces some coherence among the predicted entities. Collective linking also has other formulations: Han et al. (2011) proposed a collective formulation for entity linking decisions, in which evidence can be reinforced into high-probability decisions.

For individual graph based linking, a rule based on the importance of the entity node in the knowledge graph rule has been studied experimentally Guo et al. (2011).

Other graph-based approaches have been developed. Hoffart et al. (2011), and Alhelbawy and Gaizauskas (2014) proposed to link efficiently a query to its corresponding entity using the weighted undirected bipartite graph (Fig. 4.4). The idea is to extract a dense subgraph in which every query node is connected to exactly one entity, yielding the most likely disambiguation. In general, this combinatorial optimization problem is *NP-hard* with respect to the number of nodes, since they generalize Steiner-tree problem Hoffart et al. (2011). However heuristics to solve this problem have been experimented: Hoffart et al. (2011) and Alhelbawy and Gaizauskas (2014) proposed a discarding algorithm using taboo search and local similarities with polynomial complexity. Adaptations of PageRank algorithm were carried out to provide each entity a popularity score: Usbeck et al. (2014)

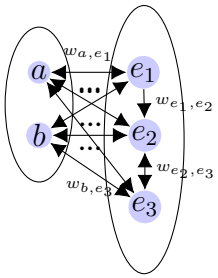


Figure 4.4 – Directed query/entity bipartite weighted graph. Nodes e_1, e_2, e_3 are entities in the knowledge base (same as Fig. 4.2). Nodes a and b are entity queries extracted from text documents, and $w_{x,y}$ is the score between two node x and y.

built a weighted graph of all queries and entities based on local and global similarities, and capitalize on the Hyperlink-Induced Topic Search (HITS) algorithm to produce node authority scores. Then, within similar entities to queries, only entities with high authority will be retained.

4.2.2 Probabilistic Graphical Models

Another interesting idea is to consider named entity queries as random variables and their true entities as hidden states. Unlike character recognition where the hidden state space is constant, with a cardinal of 26 for latin alphabet, the number of possible states S per entity is large (usually $S \geq 10^6$). Since Viterbi algorithm has an $\mathcal{O}(N\,|S|^2)$ complexity, where N is the number of observations, inference is inefficient. To overcome this issue, Alhelbawy and Gaizauskas (2013) considers a reduced set of candidates per query using query text information. Using annotation, an Hidden Markov Model (HMM) is trained on the reduced set of candidates. Inference is made using message passing (Viterbi algorithm) to find the most probable named entity sequence. Another approach using probabilistic graphical model has been provided by Ganea et al. (2016), with a factor graph that uses popularity-based prior.

4.2.3 Embeddings and Deep Architectures

Recent advances in neural networks conception suggested to use word embeddings and convolutional neural networks to solve the named entity linking problem.

Sun et al. (2015) proposed to maximize a corrupted cosine similarity between a query, its annotated gold entity and a false entity. An example of learning representations for entities using a neural architecture is achieved in Yamada et al. (2017), a linking system based on the similarity of average of pre-trained entity

embeddings has been proposed Yamada et al. (2016), with an $O(QE^2)$ complexity. Finally other architectures have been proposed Sil et al. (2018); Raiman and Raiman (2018), the latter using a fine-grained ontology type system and reaching promising results on several datasets.

4.3 Methodology

In this section we present a novel graph-based method for NEL. As a preprocessing step, we propose a new but simple entity filtering method using information retrieval techniques to obtain a limited number of entity candidates. The novelty of our method lies in the subsection 4.3.2 where we present a new graph-based method for final entity identification.

4.3.1 Entity Filtering

To discard wrong entity candidates, we use the three sources of information in the query $\boldsymbol{q} = (\boldsymbol{m}, \boldsymbol{c}, \hat{\boldsymbol{t}})$: the mention name, the information contained in the rest of document, and the entity type. Obviously, the NEL problem becomes easier as the ontology size grows (i.e the larger is $\boldsymbol{T}$), on the contrary of the NER problem. In order to improve existing entity filtering algorithms, we propose a routine based on three main components below. The algorithm is summarized in Algorithm 3).

a* -** preProcess:*** For trivial queries having a mention name equal to an existing entity name and type, we implemented a naive match pre-processing. If a mention has the same name and the same type, its gold entity is labelled as the corresponding entity.

b* -** acronymDetection *&* acronymScore:*** Acronym detection and expansion is a common topic in bioinformatics. We refer to Ehrmann et al. (2013) for a survey of acronym detection methods. We implemented a simple rule-based decision for acronym detection, following Gusfield (1997): a string is tagged as an acronym if there are two or more capital letters, and that distance between two consecutive capital letters is always one. The similarity score for acronym extension is chosen as the length of longest common substring Apostolico and Guerra (1987) between the acronym and capital letters of the target.

c* -** JN *&* contextScore:*** When the named entity mention is not tagged as an acronym, comparison with entity titles is performed by computing N-grams for $N \in \{2, 3, 4\}$, and use Jaccard Index of mention name and entity title. It is refered as JN in algorithm 3. We also mesure similarity between the context of the query and the text description of an entity in the knowledge base. We experimented several techniques: TF-IDF, BM25 , BM25+ based on the probabilistic retrieval framework developed in the 1970s and 1980s (we refer to Robertson et al. (2009) for a recent

description). The experimental results were very similar in terms of recall. We present results obtained with TFIDF (cf. Sec. 4.4).

Algorithm 3: Entity filtering (generation of entity candidates)

Input: Parameter $\boldsymbol{K}$, Query $(\boldsymbol{q} = (\boldsymbol{m}, \boldsymbol{c}, \hat{\boldsymbol{t}}))$, Entities and types $(\boldsymbol{e}_j, \boldsymbol{t}_j)_{1 \leq j \leq E}$,
Output: Top K entities

$\boldsymbol{ds} = [\,]$;
$\boldsymbol{y}_{acr} \leftarrow \mathsf{acronymDetection}(\boldsymbol{m})$
for $j = 1 \rightarrow E$ **do**
  if $t_j == \hat{t}$ **then**
    if $\boldsymbol{y}_{acr} == 1$ **then**
      $s_n = \mathsf{acronymScore}(\boldsymbol{m}, \boldsymbol{e}_j)$;
    else
      $s_n = \mathsf{JN}(\boldsymbol{m}, \boldsymbol{e}_j)$;
    end
    $s_c = \mathsf{tfidfScore}(\boldsymbol{c}, \boldsymbol{e}_j)$;
    $s_t = \frac{1}{2}(s_n + \mathsf{contextScore}(\boldsymbol{c}, \boldsymbol{e}_j))$;
    Sorted insertion by value of $\{j : s_t\}$ in $\boldsymbol{ds}$;
  end
end
Return $\boldsymbol{ds}[: \boldsymbol{K}]$ ($\boldsymbol{K}$ top entities) ;

4.3.2 Graph-based Identification

In this section, we present our graph-based method for *named entity identification*. This graph-based method uses enriched features extraction from the knowledge graph, in order to re-rank top entity candidates.

Feature extraction: Let $\boldsymbol{q}$ and $\boldsymbol{e}$ respectively be a query and an entity. $\boldsymbol{T}$ still represents the number of distinct entity types in the ontology. Let s be a scoring function between a query and an entity. Let $\mathcal{N}_t(\boldsymbol{e})$ the set of entity neighbors of type t (cf. Fig 4.5 for an example). By convention if $\mathcal{N}_t(\boldsymbol{e}) = \emptyset$, then $s(\boldsymbol{q}, \mathcal{N}_t(\boldsymbol{e})) \triangleq 0$. $f(\boldsymbol{q}, \boldsymbol{e})$ is the filtering score obtained with Algorithm 3. We define the features vector associated with the couple $(\boldsymbol{q}, \boldsymbol{e})$, $\boldsymbol{X}^{\boldsymbol{q},\boldsymbol{e}}$ as the scores concatenation:

$$\begin{aligned} (\boldsymbol{X}^{\boldsymbol{q},\boldsymbol{e}})_0 &= f(\boldsymbol{q}, \boldsymbol{e}) \\ \forall t \in \{1, ..., \boldsymbol{T}\},\ (\boldsymbol{X}^{\boldsymbol{q},\boldsymbol{e}})_t &= s(\boldsymbol{q}, \mathcal{N}_t(\boldsymbol{e})) \end{aligned} \tag{4.4}$$

Figure 4.5 – Two homonyms: Cambridge cities. Each color is assigned to a node in the ontology. If t_1 is associated to the entity type *Country*, and t_2 to *Football player*, then
$\mathcal{N}_{t_1}(\text{Cambridge 1}) = \{\text{England}\}$
$\mathcal{N}_{t_2}(\text{Cambridge 1}) = \{\text{Wilf Mannion}\}$
$\mathcal{N}_{t_1}(\text{Cambridge 2}) = \{\text{United States}\}$
$\mathcal{N}_{t_2}(\text{Cambridge 2}) = \emptyset$

The label of a couple $(\boldsymbol{q}, \boldsymbol{e})$ is defined as:

$$Y^{\boldsymbol{q},\boldsymbol{e}} = \begin{cases} 1 & \text{if } \boldsymbol{e} \text{ is the gold entity of } \boldsymbol{q} \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

Supervised NEL: With this formulation, we can train *NEL* standard regressors or classifiers in a supervised learning framework. At inference, the couple $(q, \hat{e})$ maximizing the prediction score yields predicted entity $\hat{e}$. If same scores are returned for different couples, we return one of the candidates (either randomly or by alphabetical order, but this situation did not occur in practice). The feature extraction and inference procedures are summed up in Algorithm 4 and Algorithm 5 respectively.

Algorithm 4: Feature extraction using knowledge graph and ontology

Input: Knowledge Graph $\boldsymbol{G}$, Query $\boldsymbol{q}$, Entity candidate $\boldsymbol{e}$ with initial filtering score s^0, Types $(\boldsymbol{t}_j)_{1 \leq j \leq \boldsymbol{T}}$
Output: Score vectors between query and candidates

$\boldsymbol{X}^{q,e} = [s^0]$;
Get neighbor nodes of $\boldsymbol{e}$;
for $j = 1$ *to* $\boldsymbol{T}$ **do**
  Aggregate text description of neighbors of type t_j;
  Compute score s_{t_j} between $\mathcal{N}_{t_j}(\boldsymbol{e})$ and the query $\boldsymbol{q}$;
  Append s_{t_j} to $X^{q,e}$;
Score vectors $(\boldsymbol{X}^{q,e})_{1 \leq j \leq \boldsymbol{T}+1}$;

Algorithm 5: Named entity identification (Inference)

Input: Knowledge base $\boldsymbol{B}$ and its graph $\boldsymbol{G_B}$, queries $(\boldsymbol{q}_i)_{1\leq i\leq M}$, scoring threshold $\boldsymbol{K}$, trained predictor $\hat{\boldsymbol{F}}$

Output: List of estimated entities

for $i = 1$ *to* M **do**

- Use filtering on query $\boldsymbol{q}_i$ and $\boldsymbol{B}$, return a list of $\boldsymbol{K}$ top ranked entities $(\boldsymbol{e}_h^i)_{1\leq h\leq K}$;
- Use Algorithm 4 using $\boldsymbol{G_B}$, on $\boldsymbol{K}$ entity candidates, return new score vectors;
- Evaluate $\hat{\boldsymbol{F}}$ on each vector score and use maximum a posteriori to infer estimated entity $\hat{\boldsymbol{g}}_i$;

Return $(\hat{\boldsymbol{g}}_i)_{1\leq i\leq M}$ (list of estimated entities) ;

Graph-based scoring functions: In the identification step, features defined from Eq. (4.4) require the choice of a scoring function. First of all, several representations for $\boldsymbol{q}$ and $\boldsymbol{e}$ are possible. In our first experiment, we used the standard TFIDF representation for the supervised learning procedure described previously, and the corresponding scoring function with cosine similarity. This allowed to increase slightly empirical accuracy over entity filtering.

In order to explore a broader class of scoring functions, we consider the graph of words representations introduced in Chapter 3 as sequence graphs. Indeed, bag of words representations can be considered as a special case of graph of words representations, for which edge deleting operations have been applied. Here, we consider that the query context and the entity description are both composed of at least 10 words for GOW to be meaningful. The final step to define a scoring function as in Eq. (4.4), is to compare the two graph structures (one from the query context and the other from the entity description).

Given two graphs G and H, determining if G is isomorphic H allows to measure graph similarities Cordella et al. (2004). However, for several applications, including the topic of this Chapter, isomorphic conditions are too rigid since two documents can be similar without isomorphic GOWs. Also, we are interested in graph similarity measures taking in account structure (word relations) and node attributes (words). For this reason, graph kernels have been popularized as a powerful tool to measure graph similarity in a continuous fashion.

Following the notations of Sec. 4.3.2, and k a graph kernel, we considered the family of scoring functions: $(\boldsymbol{q}, \boldsymbol{e}) \mapsto k(GoW_{\boldsymbol{q}}, GoW_{\mathcal{N}_t(\boldsymbol{e})})$ in our experiments. If $\mathcal{N}_t(\boldsymbol{e})$ contains more than one node, we concatenate their text content and compute a GoW. As mentioned previously, this family of functions contains some of the bag of words scoring functions, such as TFIDF. We obtained better empirical results

using standard graph kernels (cf. next paragraph for examples). It should be noted that we could not use graph kernels for the first step (entity filtering), since the computation time would be too long. On the contrary, the identification step takes as input a limited amount of entity candidates, which makes the computation time reasonable.

Graph of words window, graph kernels & regressors: We selected as a graph-of-word window $w = 4$ (same results were obtained for $w \in \{3, 4, 5, 6\}$), with different graph kernels, including Shortest-path kernel, Weisfeiler-Lehman kernel. The accuracy results for each graph kernel were very close, but higher than with TFIDF scoring (1% to 2% better). In Sec. 4.4, we report results for the pyramid match graph kernel, for its low complexity among standard kernels (See Nikolentzos et al. (2017b)). Finally, we used several standard classifiers: regression trees, support vector machines, and logistic regression. We obtained better results with logistic regression (reported in Table 4.1).

Computational complexity: The average total complexity (filtering and identification) is: $\mathcal{O}(M(E + \boldsymbol{KT}G))$ where G is an upper-bound of the complexity required to compute the graph kernel value between the query GOW and the entity GOW. G is a relatively small constant since the query and entity texts are relatively short (typically less than 200 words), so that a kernel value can be computed in less than a second on a laptop with a $2,8$ GHz Intel Core i7. We report this in Table 4.2, along with some experimental computing times.

4.4 Experimental Setup and Evaluation

The source code of our experiments along with documentation, and datasets samples are available at our repository.

4.4.1 Datasets, Entity Types and Ontology

We remind the description of the datasets used in this Section (we also remind our reader that Section 2.7 contains a description of all the datasets used throughout all the dissertation.).

First, TAC-KBP (Ji et al., 2014) is a dataset from the National institute of Standards ant technologies (NIST), within the Text Analysis Conference's Knowledge Base Population (TAC-KBP) track, which aimed to link a given named entity mention from a source document to an existing Knowledge Base (KB).

Second, AIDA/CoNLL (Yosef et al., 2011) contains assignments of entities to the mentions of named entities annotated for the original CoNLL 2003 entity recognition task.

Each query in theses datasets contains its gold entity id and type. TAC-

Table 4.1 – Comparison with state-of-the art methods for $\boldsymbol{K} = 7$ and $\boldsymbol{T} = 249$. PGMs stands for probabilistic graphical model. /: not available in the reference. ND: not detailed.

Reference	Method	Nil detect.	Train. size	$\boldsymbol{P}@1$ **(Accuracy)** $\pm$ std %		
				TAC09	TAC10	AIDA
61	PGM	No	$\sim 10^6$	/	/	87.39
62	PGM/DL	No	$\sim 10^6$	/	/	92.22
167	DL	No	$\sim 10^6$	82.26	83.92	/
186	DL	No	$\sim 10^6$	/	85.2	93.1
187	DL	No	$\sim 10^6$	/	87.7	94.3
67	DL	Yes	$\sim 10^6$	/	87.2	92.7
162	DL	ND	$\sim 10^6$	/	87.4	93.0
140	DL	ND	$\sim 10^6$	/	90.85	**94.87**
77	Graphs	Yes	$\sim 10^4$	84.89	82.40	/
86	Graphs	No	$\sim 10^4$	/	/	81.91
Ours	Graphs	No	$\sim 10^3, 10^4$	**93.67**$_{\pm 0.06}$	**94.70**$_{\pm 0.05}$	93.56$_{\pm 0.06}$

KBP is composed of a knowledge base containing 818741 entities. TAC09 contains 1675 test queries, and TAC10 1074 for train and 1020 for test. CoNLL/AIDA is composed of 22516 queries for training and 4379 queries for test.

The other methods, mainly deep learning (DL) in Table 4.1 use millions of training examples from Wikipedia's anchor links and corresponding entities. In our method, we did not use this additional training data, but only those provided by the original challenges.

Also, we considered a more recent Knowledge base (Wikipedia 2016 dump with 2880838 entities) since the original Wikipedia 2010 dump is not available anymore. The ontology we considered is available on DBPedia[1]. We must remind that our method does not include fined-grained entity recognition from the queries: we suppose this given as input in the data. For the implementation of graph kernels, we used the GraKeL software library (Siglidis et al., 2018).

4.4.2 Results

We compare our methods with most performing baselines. Table 4.1 sums up our experimental results (averaged $\boldsymbol{P}@1$ is also referred as accuracy (Sun et al., 2015)). We included standard deviation of the accuracy, but could not include p-significance of our method, due to the difficulty to reproduce other baselines experiments (no source code is publicly available, or filtering method is not detailed). Our method yields remarkable accuracy on TAC09 dataset, CoNLL/AIDA

Table 4.2 – Computing times rounded to the minute. $Q = 1000$, $E = 2.8 \times 10^6$, $G \leq 200$, $\boldsymbol{K} = 7$, $\boldsymbol{T} = 249$. Setup 1: Single CPU with 32Gb Ram, 4-cores 2.40GHz. Setup 2: Distributed cluster with variety of 20 CPU processors equivalent to setup 1 (Spark/Hadoop technology)

		Time (mn.)	
Component	**Complexity**	**Setup 1**	**Setup 2**
Filtering	$\mathcal{O}(ME)$	196	15
Identification	$\mathcal{O}(Q\boldsymbol{K}\boldsymbol{T}G)$	153	10

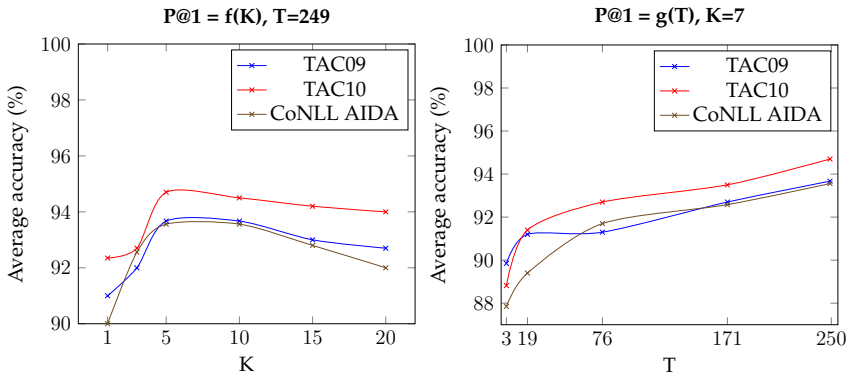


Figure 4.6 – Impact of $\boldsymbol{K}$ and $\boldsymbol{T}$ on average $\boldsymbol{P}$@1.

and TAC10 datasets. It performs better than any existing graph-based methods, outperforms all existing methods on two NIST TAC09 and TAC10, and is competitive with state-of-the arts methods on CoNLL/AIDA. We also report impact of parameter K on average precision $\boldsymbol{P}$@1 (accuracy). Results are in Fig. 4.6. Low values of $\boldsymbol{K}$, corresponding to limited exploration, lead to a decrease accuracy. High values of K yield too many entity candidates and an imbalanced learning problem, resulting in a decrease of accuracy. Results are similar for $5 \leq \boldsymbol{K} \leq 10$, and allow a 2% to 3% improvement over filtering. As expected, the precision is strictly increasing with respect to $\boldsymbol{T}$ but the variation is bounded by 5% for $\boldsymbol{T} \in [3, 249]$.

4.5 Conclusion

In this chapter, we proposed a new methodology concerning the problem of named entity linking. First, we presented an entity filtering algorithm to return entity candidates that improves over trivial association rules. Then, each entity candidate is matched with a new representation built on a subgraph centered on their node. These representations use information contained in the ontology of the knowledge

base. Finally, we used standard supervised learning to identify entities in the top candidates from filtering. We showed experimentally with standard datasets that named entity linking systematically improves over filtering using graph-based identification (for $2 \leq \boldsymbol{K} \leq 10$), up to 3%. Our experiments show that our method is competitive with state-of-the-art, and is stable with respect to $\boldsymbol{K}$ and $\boldsymbol{T}$, has good complexity allowing reasonable experimental computing time. Our linking system is relatively easy to implement, with few hyper-parameters. Last but not least, it does not require lots of data compared with deep learning to reach good experimental performance: only a few thousands of training samples were used to reach these results.

5

Geometry, Words and Representations

Despite the ambiguity induced by graph-based representations studied in Chapter 3, we have shown in Chapter 4 that combined with a simple filtering method, they allow to address a problem of information retrieval, named entity identification. With a pertinent formulation, these representations allow to reach state-of-the-art performance while requiring few annotated samples.

These representations did not incorporate any language-specific method, but were only based on association rules, graphical representations and the existence of a knowledge graph. Therefore, they are not endowed with any description or information of the language; but are empirically adequate for the given problem.

Moreover, the sequence graphs studied in Chapter 3 and used in Chapter 4 can also be seen as combinatorial objects which attempt to extract the semantics between words based on the co-occurrences in the same context. They also were used to compute several vector representations used in machine learning (Mikolov et al., 2013a; Pennington et al., 2014; Arora et al., 2016a,b). In this regard, we will address in this chapter some part of the relation between semantics and geometry of these representations.

Cognitive but non computational approaches have been developed on the subject. For instance, (Gärdenfors, 2014) proposes a theory of semantics that bridges cognitive science and linguistics which encompasses different theories of cognitive processes. He argues that our mind organizes the information involved in communicative acts in a format that can be modeled in geometric or topological terms. Also, (Chilton, 2014) proposes that spatial cognition provides the foundation of linguistic meanings into new dimensions and develops a theoretical framework based on simple geometric principles.

These links between cognitive system and space can also be transposed to some geometric properties of representations of words. Considering that words can be represented by geometric forms, for example word vectors (vector representations are the main input format for machine learning algorithms, these word vectors are referred to as *word embeddings*), it is then possible to imagine any kind of geometric property translating an underlying semantic link.

5.1 Introduction

More precisely, in this chapter, we discuss two main properties. The first one concerns the well-known claim that language analogies yield almost parallel vector differences for a family of word embeddings. On the one hand, we show that this property, while it does hold for a handful of cases, fails to hold in general especially in high dimension, using the best known publicly available word embeddings. On the other hand, we show that this property is not crucial for basic natural language processing tasks such as text classification. We achieve this by a simple algorithm which yields updated word embeddings where this property holds: we show that in these word representations, text classification tasks have about the same performance.

This geometrical property has lead to a generative model (Arora et al., 2016a), of which a corollary attempts to justify this geometric property. We will be interested in a central property of this model (related to a partition function which depends on the word vectors and discourse vectors, defined in Section 5.2.3 and Section 5.3). This property is central for all the theoretical developments for this model, including the relations between statistical indicators such as pointwise mutual information, and the scalar product of word vectors. Mainly, we will prove that this partition function is almost constant for a family of word vectors which are not following this model. This implies in particular that it does not represent a pertinent test for the quality of word vectors, but also as an indicator of compliance with the random walk model proposed by (Arora et al., 2016a).

5.1.1 Context and Motivations

The motivation to build word representations as vectors in a Euclidean space is twofold. First, geometrical representations can possibly enhance our understanding of a language. Second, these representations can be useful for information retrieval on large datasets, for which semantic operations become algebraic operations. First attempts to model natural language using simple vector space models go back to the 1970s, namely index terms (Salton et al., 1975), term frequency inverse document frequency (TF-IDF) (Ramos et al., 2003), corresponding software solutions SMART (Salton, 1971b), and Lucene (Hatcher and Gospodnetic, 2004). In recent work about word representations, it has been emphasized that many analogies such as "*king* is to *man* what *queen* is to *woman*", yielded almost parallel difference vectors in the space of the two most significant coordinates (Mikolov et al., 2013c; Pennington et al., 2014), that is to say (if $d = 2$):

$$\begin{aligned} (u_i \mid 1 \leq i \leq n) \in \mathbb{R}^d \quad & \text{being the word representations} \\ \text{(3,4) is an analogy of (1,2)} \Leftrightarrow \exists \epsilon \in \mathbb{R}^d \ \text{ s.t } \quad & u_2 - u_1 = u_4 - u_3 + \epsilon \\ \text{where} \quad & ||\epsilon|| \ll \min(||u_2 - u_1||, ||u_4 - u_3||) \end{aligned} \tag{5.1}$$

In Equation (5.1) $||x|| \ll ||y||$ means in practice that $||x||$ is much smaller than $||y||$. Equation (5.1) is stricter than just parallelism, but we adopt this version because it corresponds to the version the scientific press has amplified in such a way that now it appears to be part of layman knowledge about word representations (Mikolov et al., 2013b; Vylomova et al., 2015; Bolukbasi et al., 2016). We hope the study contained in this chapter will help clear a misinterpretation.

Recent work leads us to cast word representations into two families: *static representations*, where each word of the language is associated to a unique element (scope of this chapter), and *contextual representations*, where the entity representing each word may depend on the context (we do not consider this case in this chapter).

5.1.2 Contributions

The attention devoted in the literature to Equation (5.1) might have been excessive, based on the following criteria:

- The proportion of analogies leading to the geometric Equation (5.1) is small.
- The classification of analogies based on Equation (5.1) or parallelism does not appear as an easy task.

Furthermore, we present a very simple propagation method in the graph of analogies, enabling our notion of parallelism in Eq (5.1). Our code is available online.[1]. This method allow us to conduct experiments demonstrating that such property does not appear to be important for text classification.

Finally, in Section 5.3, we study a property of a generative model which attempted to provide theoretical foundations of the property of Equation (5.1) . We will prove that this partition function is almost constant for a family of word vectors which are not following this model. This implies in particular that a test related to a partition function cannot be used as a compliance witness for this generative model, and is not a guarantee for pertinent word vectors.

[1] Link to repository

5.2 Related Work

5.2.1 Word Embeddings

In the *static representations* family, after the first vector space models (Index terms, TF-IDF, SMART (Salton, 1971b), Lucene (Hatcher and Gospodnetic, 2004)), Skip-gram and statistical log-bilinear regression models became very popular. The most famous are Glove (Pennington et al., 2014), Word2vec (Mikolov et al., 2013c), and fastText (Bojanowski et al., 2017). Since word embeddings are computed once and for all for a given string, this causes polysemy for fixed embeddings. To overcome this issue, the family of *dynamic representations* has gained in attention very recently due to the increase of deep learning methods. ELMo (Peters et al., 2018), and Bert (Devlin et al., 2018) representations take in account context, letters, and n-grams of each word. We do not address comparison with these methods in this chapter because of the lack of analysis of their geometric properties.

There have been attempts to evaluate the semantic quality of word embeddings (Jastrzebski et al., 2017), namely:

- Semantic similarity (Calculate Spearman correlation between cosine similarity of the model and human rated similarity of word pairs)
- Semantic analogy (Analogy prediction accuracy)
- Text categorisation (Purity measure)

However, in practice, these semantic quality measures are not preferred for applications: the quality of word embeddings is evaluated on very specific tasks, such as text classification or named entity recognition. In addition, recent work (Nissim et al., 2019) has shown that the use of analogies to uncover human biases should be carried out very carefully, in a fair and transparent way. For example (Caliskan et al., 2017) analyzed gender bias from language corpora, but balanced their results by checking against the actual distribution of jobs between genders.

5.2.2 Relation Embeddings for Named Entities

An entity is a real-world object and denoted with a proper name. In the expression "Named Entity", the word "Named" aims at restricting the possible set of entities to only those for which one or many rigid designators stands for the referent. For this reason, named entities have an important role in text information retrieval (Nadeau and Sekine, 2007). An entity relation can be seen as an example of relation we consider for analogies (example: Paris is the capital of France, such as Madrid to Spain). There exist several attempts to model these relations, for example as

translations (Bordes et al., 2013; Wang et al., 2014), or as hyperplanes (Lin et al., 2015).

5.2.3 Word Embeddings, Linear Structures and Pointwise Mutual Information

In this subsection, we will focus on a recent analysis of pointwise mutual information, which aims at providing a piece of explanation of the linear structure for analogies (Arora et al., 2016a, 2018). This work provides a generative model with priors to compute closed form expressions for word statistics. In the following, $f = O(g)$ (resp. $f = \tilde{O}(g)$) means that f is upper bounded by g (resp. upper bounded ignoring logarithmic factors) in the considered neighborhood. The generation of sentences in a given text corpus is made under the following generative assumptions:

- *Assumption 1*: The ensemble of word vectors consists of independent and identically distributed (i.i.d.) samples generated by $v = s\,\hat{v}$, where $\hat{v}$ is drawn from the spherical Gaussian distribution in $\mathbb{R}^d$ and s is an integrable random scalar, always upper bounded by a constant $\kappa \in \mathbb{R}^+$.
- *Assumption 2*: Let d be an stricly positive integer corresponding to the word vectors dimension. The text generation process is driven by a random walk of a vector c_t, i.e. if w_t is the word at step t, there exists a latent discourse vector c_t such that $\mathsf{P}(w_t = w|c_t) \propto \exp(\langle c_t, v_w \rangle)$. Moreover, $\exists \kappa \geq 0$ and $\epsilon_1 \geq 0$ such that $\forall t \geq 0$:

$$\begin{aligned} |s| &\leq \kappa \\ \mathbb{E}_{c_{t+1}}(e^{\kappa\sqrt{d}||c_{t+1}-c_t||_2}) &\leq 1 + \epsilon_1 \end{aligned} \tag{5.2}$$

In the following, $\mathsf{P}(w, w')$ is the probability that two words w and w' occur in a window of size 2 (the result can be generalized to any window size), $\mathsf{P}(w)$ is the marginal probability of w. $\mathsf{PMI}(w, w')$ is the pointwise mutual information between two words w and w'(Church and Hanks, 1990). Under these conditions, we have the following result:

Theorem 5.1. *(Arora et al., 2016a) Let n denote the number of words and d denote the dimension of the representations. If Assumption 1 and Assumption 2 are verified, then the following holds for any words w and w':*

$$\begin{aligned} \mathsf{PMI}(w, w') \triangleq \log \frac{\mathsf{P}(w, w')}{\mathsf{P}(w)\mathsf{P}(w')} &= \frac{\langle v_w, v_{w'} \rangle}{d} \pm O(\epsilon) \\ \textit{with} \quad \epsilon &= \tilde{O}(\frac{1}{\sqrt{n}}) + \tilde{O}(\frac{1}{d}) + O(\epsilon_1) \end{aligned} \tag{5.3}$$

Equation (5.3) shows that we could expect high cosine similarity for pointwise close terms (if ϵ is negligible).

The main aspect we are interested in is the relationship between linear structures and analogies. In (Arora et al., 2016a), the subject is treated with an assumption following (Pennington et al., 2014), stated in Equation (5.4). Let χ be any set of words, and a and b words that are involved in a semantic relation $\mathcal{R}$. Then there exist two scalars $v_{\mathcal{R}}(\chi)$ and $\xi_{ab\mathcal{R}}(\chi)$ such that:

$$\frac{\mathsf{P}(\chi|a)}{\mathsf{P}(\chi|b)} = v_{\mathcal{R}}(\chi)\,\xi_{ab\mathcal{R}}(\chi) \tag{5.4}$$

We failed to fully understand the argument made in (Arora et al., 2016a; Pennington et al., 2014) linking word vectors to differences thereof. However, if we assume Equation (5.4), by Equation (5.3) we obtain the following.

Corollary 5.1.1. *Let V be the $n \times d$ matrix whose rows are the vectors of words in dimension d. Let v_a and v_b be vectors corresponding respectively to words a and b. Assume a and b are involved in a relation $\mathcal{R}$. Let $\log(v_{\mathcal{R}})$ the element-wise log of vector $v_{\mathcal{R}}$. Then there exists a vector $\xi'_{ab\mathcal{R}} \in \mathbb{R}^n$ such that:*

$$V(v_a - v_b) = d\log(v_{\mathcal{R}}) + \xi^{'}_{ab\mathcal{R}} \tag{5.5}$$

Proof. Let x a word, and a, b two words sharing a relation $\mathcal{R}$. On the one hand, taking the $\log$ of Equation (5.4):

$$\log(\frac{\mathsf{P}(x|a)}{\mathsf{P}(x|b)}) = \log(v_{\mathcal{R}}(x)) + \log(\xi_{ab\mathcal{R}}(x)) \tag{5.6}$$

On the other hand, using Equation (5.3), $\exists\, \epsilon_{abx} \in \mathbb{R}$ such that:

$$\begin{aligned}
\log(\frac{\mathsf{P}(x|a)}{\mathsf{P}(x|b)}) &= \log(\frac{\mathsf{P}(x,a)\mathsf{P}(b)}{\mathsf{P}(x,b)\mathsf{P}(a)}) \\
&= \log(\frac{\mathsf{P}(x,a)\mathsf{P}(b)\mathsf{P}(x)}{\mathsf{P}(x,b)\mathsf{P}(a)\mathsf{P}(x)}) \\
&= \mathsf{PMI}(x,a) - \mathsf{PMI}(x,b) \\
\log(\frac{\mathsf{P}(x|a)}{\mathsf{P}(x|b)}) &= \frac{\langle v_x, v_a - v_b\rangle}{d} + \epsilon_{abx}
\end{aligned} \tag{5.7}$$

Combining equations (5.6) and (5.7), for any x:

$$\langle v_x, v_a - v_b\rangle = d\log(v_{\mathcal{R}}(x)) + d(\log(\xi_{ab\mathcal{R}}(x)) - \epsilon_{abx}) \tag{5.8}$$

Let V the matrix whose rows are the word vectors. $V(v_a - v_b)$ is a vector of $\mathbb{R}^n$ whose component associated with word x is exactly $\langle v_x, v_a - v_b \rangle$. Then, let $v_{\mathcal{R}}$ be the element-wise $\log$ of vector $v_{\mathcal{R}}$, and $\xi'_{ab\mathcal{R}}$ the vectors of components $d(\log \xi_{ab\mathcal{R}}(x) - \epsilon_{abx})$. Then, Eq (5.8) is exactly Eq (5.5). □

It is shown in (Arora et al., 2016a) that $||V^+ \xi'_{ab\mathcal{R}}|| \leq ||\xi'_{ab\mathcal{R}}||$, where V^+ is the pseudo-inverse of V (if needed, cf. Definition 2.4). In other words, the "noise" factor ξ' can be reduced. This reduction may not be sufficient if $\xi_{ab\mathcal{R}}$ is too large to start with.

In the next section, we will address in details a fundamental property used for the generative model presented in this subsection, which is one of the sufficient conditions implying 5.1. We will show that it cannot be used as a intrinsic test for the word vectors since it holds as long as the discourse vectors are close to a sphere of radius $R \leq 2$ and some mild assumptions on the distribution of word vectors.

5.3 Discussion about the Concentration of the Partition Function

In this section, we discuss a theoretical property presented in (Arora et al., 2016a), called the concentration of the partition function Z_c, which is defined as

$$Z_c = \sum_v \exp(\langle v, c \rangle) \tag{5.9}$$

where v are the word vectors, and c is a latent discourse vector. We remind our reader that in the generative model discussed in section 5.2.3, the model treats corpus generation as a dynamic process, where the t-th word is produced at step t. The process is driven by a random walk of a discourse vector c. Its coordinates represent the current topic. In this section, we are not interested in the dynamics of this model, but rather by an asymptotic property of the partition function Z_c.

By analogy with statistical physics, this partition function is the sum of probabilities of the particles state given macroscopic parameters such as temperature, over all the particles. More precisely, in our context, the particles considered are words and the states are the appearances of a word given a latent discourse vector (which is the equivalent of the physical temperature). This latent discourse vector represents a context of fixed length (similarly to the context mentioned in Chapter 3). The aim of this section is to study the variations of Z_c with respect to the random variable c. This study is motivated by the use of partition concentration as a theoretical basis to demonstrate the relationships between PMI and scalar product of word vectors (Arora et al., 2016a), which in turn could potentially explain the initial conjecture 5.1.

Following the assumptions of Section 5.2.3, if the word vectors satisfy the Bayesian prior described in the model details, and n is the number of words, then the concentration of partition is stated as follows (Arora et al., 2016a, Lemma 2.1):

$$\mathbb{P}[(1-\epsilon_z)\,Z \;\leq\; Z_c \;\leq\; (1+\epsilon_z)\,Z] \geq 1-\delta \tag{5.10}$$

for some constant Z (independant of c) and $\epsilon_z = \tilde{O}(1/n)$ and $\delta = \exp(-\Omega(\log(n)))$, where again $f = O(g)$ (resp. $f = \tilde{O}(g)$) means that f is upper bounded by g (resp. upper bounded ignoring logarithmic factors) in the considered neighborhood.

We are interested in this property since it is central for the development of all the following theorems and propositions in (Arora et al., 2016a), including the relation between pointwise mutual information and scalar product of word vectors stated in Theorem 5.1.

Furthermore, in the experiments conducted in (Arora et al., 2016a, Section 5.1), the property expressed in Equation 5.10 is evaluated on the word vectors generated by displaying the histogram of the partition function Z_c, which should concentrate around its mean. By doing so, this concentration of partition functions is implicitly considered as a mean to evaluate how well the word vectors follow the generative model. In this section, we will show this property holds by weakening the assumptions, modulo a small constant.

5.3.1 Preliminaries

In this section, we will start by proving three useful lemmas to prove our main result.

Lemma 5.1. *Let $\psi : \mathbb{R} \to [0, +\infty[$ be a twice continuously differentiable strictly convex even function, satisfying the following properties:*

1. $\psi'(0) = 0$
2. $\beta \mapsto \psi'(\beta)/\beta$ *is injective on* $\mathbb{R}^{+*}$.
3. $\forall \beta \neq 0 \quad \psi''(0) - \psi'(\beta)/\beta > 0$
4. $\forall \beta \neq 0 \quad \psi''(\beta) - \psi'(\beta)/\beta < 0$

Then, the optimization problem

$$\begin{aligned} \min \quad & \sum_{i=1}^{d} \psi(x_i) \triangleq \Psi(x) \\ s.t. \quad & \frac{1}{2}\|x\|^2 = \frac{1}{2}R^2 \end{aligned}$$

has the following extreme points:

1. $x^* = \pm Re_k$, *where* e_k *is the* k*-th canonical vector of* $\mathbb{R}^d$ *corresponding to global minimizers;*

2. $x_i^* = \pm\dfrac{R}{\sqrt{d}}$, *for* $i = 1, \ldots, d$, *corresponding to global maximizers.*

Proof. Let us consider the first order optimality conditions. The Lagrange equations are

$$\nabla\Phi(x) + \lambda x = 0$$

or equivalently

$$\forall i \in \{1, \ldots, d\} \qquad \psi'(x_i) + \lambda x_i = 0 \tag{5.11}$$

For the remaining of this proof, let $(x, \lambda) \in \mathbb{R}^d \times \mathbb{R}$ be a fixed vector and scalar verifying Equation (5.11). Such x and λ exists because we are considering a continuous function over a compact set, thus it attains a maximum and a minimum in the feasible set. Notice that $x_i = 0$ solves this equation for any λ. However, we cannot set $x_i = 0$ for every $i \in \{1, \ldots, d\}$, because $x = 0$ is infeasible.

Therefore, there should be components some components of x verifying $x_i \neq 0$. For the non-zero components of x, Equation (5.11) must hold for the same λ. Since the gradient of the constraint does not vanish at any feasible point, the Linear Independence Constraint Qualification (LICQ) holds and hence there exists λ fulfilling Equation (5.11) for some feasible point.

First, we remark that $\lambda \neq 0$. Indeed, if $\lambda = 0$, then from Equation (5.11), $\forall i \;\; f'(x_i) = 0$, but f is convex and $f'(0) = 0$, which implies that $x_i = 0$ for all i, leading to an infeasible point.

Thus, for the non-zero components of x, from Equation (5.11), we obtain

$$x_i \neq 0 \implies \lambda = -\frac{\psi'(x_i)}{x_i} \neq 0$$

But, since $\beta \mapsto \psi'(\beta)/\beta$ is injective on $\mathbb{R}^{+*}$, we conclude that the non-zero components of x must be all equal, i.e $\exists \beta^* > 0 \;\; \text{s.t.} \;\; \forall i \;\; x_i \neq 0 \implies x_i = \beta^*$. From the feasibility of x, we conclude that

$$\beta^* = \pm\frac{R}{\sqrt{\|x\|_0}}$$

where $\|x\|_0$ denotes the number of non-zero entries of x.

Let us now analyze the second order conditions for the feasible points verifying Equation (5.11). Since the objective function is separable, the Hessian of the Lagrangian $\nabla^2_{xx} L(x, \lambda)$ is a diagonal matrix whose diagonal entries are

$$[\nabla^2_{xx} L(x, \lambda)]_{ii} = \psi''(x_i) - \psi'(\beta^*)/\beta^* \quad \text{for} \quad i = 1, \ldots, d.$$

From the properties of f, we can deduce

$$[\nabla^2_{xx} L(x,\lambda)]_{ii} = \begin{cases} \psi''(0) - f'(\beta^*)/\beta^* & \text{if } x_i = 0 \\ \psi''(\beta^*) - \psi'(\beta^*)/\beta^* & \text{otherwise} \end{cases}$$

For the remaining of this proof, for given α and β, let $\delta(\alpha,\beta) \triangleq \psi''(\alpha) - \psi'(\beta)/\beta$. We remind our reader that, by assumption, $\beta \neq 0 \implies \delta(0,\beta) > 0$ and $\delta(\beta,\beta) < 0$.

Therefore, for a given $y \in \mathbb{R}^d$, we have

$$y^T \nabla^2_{xx} L(x,\lambda) y = \delta(0,\beta^*) \sum_{i:x_i=0} y_i^2 + \delta(\beta^*,\beta^*) \sum_{i:x_i\neq 0} y_i^2.$$

If all components of x are non-zero, then we get

$$\forall y \in \mathbb{R}^d \setminus \{0\} \qquad y^T \nabla^2_{xx} L(x,\lambda) y = \delta(\beta^*,\beta^*) \sum_{i:x_i\neq 0} y_i^2 < 0$$

Also, we already proved non zero components of x must be equal; this proves that x verifying $\forall i \in \{1,\dots,d\}, \quad x_i = \pm\dfrac{R}{\sqrt{d}}$ satisfy the second order sufficient conditions for a local maximizer.

Now, let us show that if x has at least one zero component and more than one non-zero components, then x is a saddle-point. Without loss of generality, assume that exactly two entries of x are non-zero, then due to the previous discussion, they must be equal, e.g. $x^T = (0,\dots,0,\beta,\beta)$. The sufficient second order conditions concern the Hessian of the Lagrangian with respect to primal variables, which should be positive definite when restricted on the linear null space of the Jacobian of the constraint inequalities. In this case, this linear space is given by:

$$x^\perp = \{y \in \mathbb{R}^d \ : \ y = (w_1,\dots,w_{d-2},\alpha,-\alpha), w \in \mathbb{R}^{d-2}, \alpha \in \mathbb{R}\},$$

In particular, choosing $y = (w_1,0,\dots,0,\alpha,-\alpha) \in x^\perp$, we obtain

$$y^T \nabla^2_{xx} L(x,\lambda) y = \delta(0,\beta^*) w_1 + 2\delta(\beta^*,\beta^*)\alpha^2$$

Then:

i) $w_1 > 0$ and $\alpha = 0 \implies y^T \nabla^2_{xx} L(x,\lambda) y > 0$

ii) $w_1 = 0$ and $\alpha \neq 0 \implies y^T \nabla^2_{xx} L(x,\lambda) y < 0$

This implies that x is neither a minimizer nor a maximizer.

Finally, if $x = \pm Re_k$, for some canonical vector e_k, we obtain, for every $y \in x^{\perp} \setminus \{0\}$,

$$y^T \nabla^2_{xx} L(x, \lambda) y = \delta(0, \beta^*) \sum_{i:x_i=0} y_i^2 + \delta(\beta^*, \beta^*) \times 0 = \delta(0, \beta^*) \sum_{i:x_i=0} y_i^2 > 0$$

which proves that $x = \pm Re_k$ satisfies the second order sufficient conditions for a local minimizer.

Furthermore, since f is even, and the maximizers (and minimizers) described above only differ by the sign of their entries, we can conclude that all of them are global.

□

We present a similar result for the annulus domain:

Lemma 5.2. *Let η be a strictly positive real and let $\mathbf{1}$ be the vector of ones of appropriate dimension. With the same notations conditions as in Lemma 5.1, replacing $\mathcal{S}_R$ by Ω_η defined by:*

$$\Omega_\eta = \{x \in \mathbb{R}^d \mid R \leq ||x||_2 \leq R + \eta\} \tag{5.12}$$

Then

(i) Ψ reaches its minimum on Ω_η for the vector $x = Re_k$

(ii) Ψ reaches its maximum on Ω_η for the vector $\frac{R+\eta}{\sqrt{d}} \mathbf{1}$

Proof. Both (i) and (ii) can be proved in two steps:

(i) Since ψ is even, we limit the study on the set of positive vectors. We show that the maximum of ψ is reached on the sphere of radius $R + \eta$, and on the sphere of radius R for the minimum. This can be proved by remarking that:

$$x > 0, \quad x \in \mathring{\Omega}_\eta \quad \text{and} \quad R < \lambda||x|| < R + \eta \implies \lambda x \in \Omega_\eta \quad \text{and} \quad \psi(\lambda x) > \psi(x)$$

Which can be deduced by the fact that ψ is strictly convex and $\psi'(0) = 0$, hence ψ is increasing on $\mathbb{R}^+$. This implies that the minimum of Ψ is reached on the sphere of radius R, and its maximum on the sphere of radius $R + \eta$.

(ii) Then, we use Lemma 5.1 to conclude. □

Lemma 5.3. *Consider the function $f\colon \mathbb{R}^d \longrightarrow \mathbb{R}$ defined by:*

$$f(c) \quad = \quad \prod_{i=1}^{d} \begin{cases} \frac{\sinh(Lc_i)}{c_i} & \text{if} \quad c_i \neq 0 \\ L & \text{otherwise} \end{cases} \tag{5.13}$$

Then $\Psi = \log(f)$ verifies the assumptions of Lemma 5.1 and 5.2.

Proof. In order to simplify the expressions, we will consider that $L = 1$ but the general case can be treated similarly. First, let us consider the function

$$\phi : x \mapsto \begin{cases} \frac{\sinh(c_i)}{c_i} & \text{if} \quad c_i \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

And in the following, let $\psi = \log \phi$.

i) First, ψ is twice continuously differentiable. Indeed, ψ is continous on $\mathbb{R}$ and

$$\lim_{x \to 0} \psi'(x) = 0 \tag{5.14}$$

So with the derivation extension theorem, ψ is differentiable in 0 and $\psi'(0) = 0$. We use the same reasoning with ϕ' and show that ψ is twice differentiable on $\mathbb{R}$, $\psi''(0) = \frac{1}{3}$.

ii) ψ strictly logarithmically convex by composition since:
- $\log$ is stricly increasing on $\mathbb{R}^{+*}$
- ϕ is strictly convex on $\mathbb{R}$, this can be seen from its second derivative:

$$\forall x \neq 0 \quad \phi''(x) = \frac{-1 - 2x^2 + \cosh(2x)}{2x^4} > 0$$

which can be deduce from the Taylor series of $\cosh$.

iii) ψ is even since ϕ is. Besides, as proved in i), we have $\psi'(0) = 0$ and $\psi''(0) = \frac{1}{3}$. Furthermore, for $x \neq 0$:

$$\psi''(0) - \frac{\psi'(x)}{x} = \frac{1}{3} - \frac{\psi'(x)}{x} = \frac{1}{3} - \frac{1}{\sinh(x)}\left(\frac{\cosh(x)}{x} - \frac{\sinh(x)}{x^2}\right) \tag{5.15}$$

and

$$\begin{aligned} \psi''(x) - \frac{\psi'(x)}{x} = &-x\frac{\coth(x)}{\sinh(x)}\left(\frac{\cosh(x)}{x} - \frac{\sinh(x)}{x^2}\right) \\ &+\frac{x}{\sinh(x)}\left(-2\frac{\cosh(x)}{x^2} + 2\frac{\sinh(x)}{x^3} + \frac{\sinh(x)}{x}\right) \end{aligned} \tag{5.16}$$

Now, let us prove:

iv) $\forall x \neq 0, \quad \psi''(0) - \frac{\psi'(x)}{x} > 0$

v) $\forall x \neq 0, \quad \psi''(x) - \frac{\psi'(x)}{x} < 0$

vi) $x \mapsto \frac{\psi'(x)}{x}$ is injective on $\mathbb{R}^{+*}$.

After computations, we remind that:

$$\phi'(x) = \frac{x\cosh x - \sinh x}{x^2} \qquad \text{and} \qquad \phi''(x) = \frac{(x^3+2x)\sinh x - 2x^2\cosh x}{x^4}$$

In particular:

$$\begin{aligned}\psi'(x) &= \frac{\phi'(x)}{\phi(x)} \\ &= \frac{x\cosh x - \sinh x}{x\sinh x} \\ &= \frac{\cosh x}{\sinh x} - \frac{1}{x} = \coth x - \frac{1}{x} \\ &= \left(\frac{1}{x} + \frac{x}{3} - \frac{x^3}{45} + \cdots\right) - \frac{1}{x} \\ &= \frac{x}{3} - \frac{x^3}{45} + \cdots\end{aligned}$$

Now, let us consider the function q be defined as:

$$q(x) = \begin{cases} \dfrac{\psi'(x)}{x} & x \neq 0 \\ q(0) = \frac{1}{3} & \text{otherwise} \end{cases}$$

After some algebraic manipulation and Taylor series expansion of $\coth$, we obtain

$$\begin{aligned}\forall x \neq 0 \qquad q(x) &= \frac{-1 + x\coth x}{x^2} \\ &= \frac{-1 + x\left(\frac{1}{x} + \frac{x}{3} + \cdots\right)}{x^2} \\ &= \frac{1}{3} - \frac{x^2}{45} + 2\frac{x^4}{945} + \cdots\end{aligned} \tag{5.17}$$

and

$$\begin{aligned}q'(x) &= \frac{2 - x(\coth x + x\,\mathrm{csch}^2 x)}{x^3} \\ &= \frac{1}{15}(\frac{1}{3} - 1)x + \frac{1}{189}(\frac{1}{5} - 1)2x^3 + \ldots\end{aligned} \tag{5.18}$$

which implies: $\forall x > 0 \quad \frac{\psi'(x)}{x} = q'(x) < 0$.

Besides,

$$q''(x) = \frac{1}{15}(\frac{1}{3} - 1) + \frac{1}{189}(\frac{1}{5} - 1)6x^2 + \ldots \quad < 0$$

hence $q'(0) = 0$ and $\forall x \quad q''(x) < 0$, implying that $q(0) = \frac{1}{3}$ is the global maximum: $\forall x \in \mathbb{R} \quad q(x) \leq 1/3$. This also proves that the function q is injective (since q' is negative and strictly increasing) and Properties v) and vi) are proved.

Moreover,

$$\begin{aligned}\psi''(x) &= \frac{\phi''(x)\phi(x) - \phi'(x)^2}{\phi(x)^2} = \frac{1}{x^2} - \operatorname{csch}^2 x \\ &= \frac{1}{x^2} - \left(\frac{1}{x^2} - \frac{1}{3} + \frac{x^2}{15} - \cdots\right) = \frac{1}{3} - \frac{x^2}{15} + \cdots \end{aligned} \tag{5.19}$$

implying

$$\forall x \neq 0 \quad \psi''(0) - \frac{\psi'(x)}{x} = \frac{1}{3} - q(x) > 0$$

showing Property iii). Finally:

$$\begin{aligned} x \neq 0 \qquad \psi''(x) - \frac{\psi'(x)}{x} &= \frac{2 - x(\coth x + x \operatorname{csch}^2 x)}{x^2} \\ &= \frac{1}{15}(\frac{1}{3} - 1)x^2 + \frac{1}{189}(\frac{1}{5} - 1)2x^4 + \cdots < 0 \end{aligned} \tag{5.20}$$

proving iv). □

5.3.2 Main Inequality

We now present our result concerning the partition function.

Proposition 5.1. *Let n be the number of words, and let us suppose the word vectors are generated independently and uniformly in a centered cube of $\mathbb{R}^d$, and the discourse vectors in domain close to sphere of radius $R \leq 2$. Then, if the discourse vectors are close to a sphere of radius $R \leq 2$ described by Ω_η*

$$\Omega_\eta = \{R \leq ||x||_2 \leq R + \eta\} \tag{5.21}$$

(e.g $\eta \leq 0.25$) then there exists $\gamma \ll 1$ such that $\forall \epsilon > 0$, the following inequality holds with probability $1 - \alpha$:

$$(1 - \epsilon)(1 - \gamma)\,\mathbb{E}[Z_0] \;\leq\; Z_c \;\leq\; (1 + \epsilon)(1 + \gamma)\,\mathbb{E}[Z_0] \tag{5.22}$$

where Z_0 corresponds to a constant discourse vector c_0 and $\alpha \leq exp(-\frac{1}{2}\epsilon^2 n^2)$.

Proof. Let $v, c \in \mathbb{R}^d$ be the word and discourse vectors, respectively, with the following properties:

$$\|v\| \leq \kappa \tag{5.23}$$

$$\mathbb{E}\left[\langle v, c\rangle\right] = 0 \tag{5.24}$$

From (5.23) and Cauchy-Schwarz inequality

$$\langle v, c\rangle \leq |\langle v, c\rangle| \leq \|v\|\|c\| \leq 3\kappa \tag{5.25}$$

where we suppose $||c|| \leq 3$ by assumption. It follows that

$$\exp\langle v, c\rangle \leq \exp 3\kappa \tag{5.26}$$

Since the random vectors v are i.i.d. and by convexity of the exponential, we have from (5.24)

$$\mathbb{E}\left[Z_c\right] = n\mathbb{E}\left[\exp\langle v, c\rangle\right] \geq n \exp \mathbb{E}\left[\langle v, c\rangle\right] = n \exp(0) = n \tag{5.27}$$

Moreover, we are also able to bound the variance of Z_c:

$$\begin{aligned} \text{Var}\left[Z_c\right] &= \sum_v \text{Var}\left[\exp\langle v, c\rangle\right] = n\text{Var}\left[\exp\langle v, c\rangle\right] \\ &\leq n\mathbb{E}\left[\exp 2\langle v, c\rangle\right] \leq n\mathbb{E}\left[\exp(6\kappa)\right] = \exp(6\kappa)n \end{aligned} \tag{5.28}$$

Now let Λ be the constant defined as follows:

$$\Lambda = \exp(6\kappa) \tag{5.29}$$

Let $\epsilon > 0$. Thanks to (5.26) and (5.28), we can apply the Bernstein's inequality to the sum of random variables $Z_c = \sum_v \exp\langle v, c\rangle$, to obtain

$$P[|Z_c - \mathbb{E}\left[Z_c\right]| > \varepsilon n] \leq \exp\left(-\frac{\frac{1}{2}\varepsilon^2 n^2}{n\Lambda + \frac{1}{3}\sqrt{\Lambda}\varepsilon n}\right) \tag{5.30}$$

and from (5.27)

$$P[|Z_c - \mathbb{E}\left[Z_c\right]| > \varepsilon \mathbb{E}\left[Z_c\right]] \leq \exp\left(-\frac{\frac{1}{2}\varepsilon^2 n^2}{n\Lambda + \frac{1}{3}\sqrt{\Lambda}\varepsilon n}\right) \tag{5.31}$$

which shows the concentration of Z_c around $\mathbb{E}\left[Z_c\right]$ for *any* fixed unit norm vector c.

Let us show now that $\mathbb{E}\left[Z_c\right]$ does not vary much with c. To this end, we need additional assumptions about the distribution of v apart from (5.23) and (5.24). We are interested in $\mathbb{E}[Z_c]$, and in particular the amplitude of its variation with respect to c. If the word vectors admit a density function ξ, then:

$$\mathbb{E}_v[exp(\langle v, c\rangle)] = \int_\Omega exp(\langle v, c\rangle)\xi(v)dv \tag{5.32}$$

If the word vectors are independent and identically distributed, it should be noted that:

$$\mathbb{E}_v[Z_c] = n\,\mathbb{E}_v[exp(\langle v, c\rangle)] \tag{5.33}$$

where n is the number of words. Firstly, in order to simplify the calculation, we will consider that v is distributed uniformly on Ω which is the cube of $\mathbb{R}^d$ centered in 0, of side length $2L$. Then, integration using Fubini Theorem yields:

$$\mathbb{E}_v[exp(\langle v, c\rangle)] = 2^d \prod_{i=1}^{d} \frac{\sinh(Lc_i)}{c_i} \tag{5.34}$$

Consider the function f: $\mathbb{R}^d \longrightarrow \mathbb{R}$ defined by:

$$f(c) \quad = \quad \prod_{i=1}^{d} \begin{cases} \frac{\sinh(Lc_i)}{c_i} & \text{if} \quad c_i \neq 0 \\ L & \text{otherwise} \end{cases} \tag{5.35}$$

We will first discuss the variations in the amplitude of f on the sphere $\mathcal{S}_R$ centered in 0 with radius R. The relative amplitude of the variations of f on $\mathcal{S}_R$ is given by:

$$\frac{\max_{c\in\mathcal{S}_R} f(c) - \min_{c\in\mathcal{S}_R} f(c)}{\min_{c\in\mathcal{S}_R} f(c)} = \frac{\max_{c\in\mathcal{S}_R} \mathbb{E}[Z_c] - \min_{c\in\mathcal{S}_R} \mathbb{E}[Z_c]}{\min_{c\in\mathcal{S}_R} \mathbb{E}[Z_c]} \tag{5.36}$$

We first show that $\log f$ verifies the conditions of Lemma 5.1. To do so, in order to simplify the expressions, we will consider that $L = 1$ but the general case can be treated similarly. First,

$$\phi : x \mapsto \begin{cases} \frac{\sinh(c_i)}{c_i} & \text{if} \quad c_i \neq 0 \\ 1 & \text{otherwise} \end{cases}$$

is twice continuously differentiable, and logarithmically convex. Indeed, after computations:

Using Lemmas 5.3 and 5.1, and we can infer the two following properties:

- On the one hand, f reaches its maximum at a point c such that:

$$c_1 = c_2 = \ldots = c_d = \frac{R}{\sqrt{d}} \tag{5.37}$$

And then

$$\max_{c\in S_R} f(c) = [\frac{\sqrt{d}}{R} \sinh(\frac{LR}{\sqrt{d}})]^d \tag{5.38}$$

- On the other hand, the minimum of f is reached for a point where every coordinate has been set to 0 except one (such point exists on the sphere), and therefore, f reaches its minimum on a point c such that

$$\phi(c_1) = \ldots = \phi(c_{d-1}) = L \quad \text{and} \quad \phi(c_d) = \frac{\sinh(LR)}{R} \tag{5.39}$$

Hence,

$$\min_{c\in\mathcal{S}_R} f(c) = L^{d-1}\frac{\sinh(LR)}{R} \tag{5.40}$$

A first interesting result is that the extrema of f do not depend on the dimension if $L = 1$.

It should be noted that the absolute variations of $\mathbb{E}[Z_c] = n\,2^d\,f(c)$ increases exponentially with respect to the dimension d and linearly with respect to the number of words n, the maximum ***relative variation*** of $\mathbb{E}[Z_c]$ in Equation (5.36) is the same as f.

Now, let us observe the behavior of the maximum of f, when the dimension d tends to infinity. The Taylor expansion at order 3 of $\sinh$ in 0 is given by:

$$\sinh(x) = x + \frac{x^3}{6} + o(x^3) \tag{5.41}$$

Therefore, using properties of the exponential:

$$\begin{aligned}\max_{c\in\mathcal{S}_R} f(c) &\underset{d\to+\infty}{=} (\frac{\sqrt{d}}{R})^d\,[\,\frac{LR}{\sqrt{d}} + \frac{1}{6}(\frac{LR}{\sqrt{d}})^3 + o(\frac{LR}{\sqrt{d}})^3\,]^d \\ &= (L + \frac{LR^2}{6d} + o(\frac{1}{d}))^d \\ &= L^d(1 + \frac{R^2}{6d} + o(\frac{1}{d}))^d \\ &\underset{d\to+\infty}{\sim} L^d e^{\frac{R^2}{6}}\end{aligned} \tag{5.42}$$

Then, if $d \gg 1$ (e.g $d \geq 50$):

$$\begin{aligned}\Delta(R) &= \frac{\max_{c\in\mathcal{S}_R} f(c) - \min_{c\in\mathcal{S}_R} f(c)}{\min_{c\in\mathcal{S}_R} f(c)} \\ &\underset{d\to+\infty}{\sim} L\frac{e^{\frac{R^2}{6}}}{\frac{\sinh(LR)}{R}} - 1\end{aligned} \tag{5.43}$$

This ratio does not depend on the dimension, regardless of the radius of the sphere considered. The graph of the function $\Delta : x \mapsto \Delta(R)$ for $L = 1$ is drawn in Figure 5.1. In particular, $||\Delta||_{\infty,[0,2]} \leq 10^{-1}$. In particular, this implies that if $R \leq 2$ (and $L \leq 1$):

$$\frac{\max_{c\in\mathcal{S}_R} \mathbb{E}[Z_c] - \min_{c\in\mathcal{S}_R} \mathbb{E}[Z_c]}{\min_{c\in\mathcal{S}_R} \mathbb{E}[Z_c]} = \Delta(R) \leq 10^{-1} \tag{5.44}$$

Finally, if Ω_η is replaced by the domain defined by

$$R \leq ||x||_2 \leq R + \eta \tag{5.45}$$

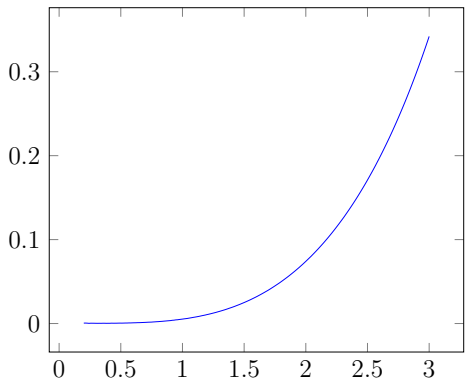
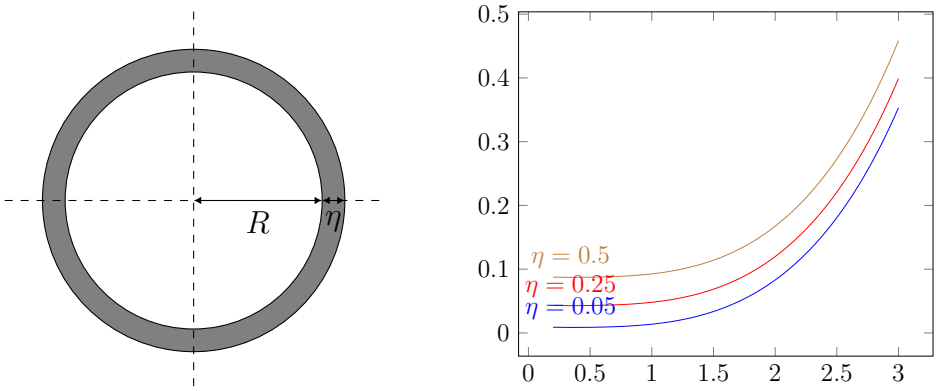


Figure 5.1 – Illustration of the maximum relative variations of $\mathbb{E}[Z_c]$, with the function $\Delta : x \mapsto \frac{e^{\frac{x^2}{6}}}{\frac{\sinh(x)}{x}} - 1$. The x-axis represents the radius considered and the y-axis the value of the maximum relative variation.



(a) Ω_η (b) Δ_η for $\eta \in \{0.05, 0.25, 0.5\}$

Figure 5.2 – Illustration of the maximum relative variations of $\mathbb{E}[Z_c]$ for $L = 1$ on Ω_η. The x-axis represents the radius R considered and the y-axis the value of the maximum relative variation.

Then the extremum of f on Ω_η can be deduced from Lemma 5.2 and are given by

$$\min_{c\,\in\Omega_\eta} f(c) = L^{d-1}\frac{\sinh(LR)}{R} \tag{5.46}$$

$$\max_{c\,\in\Omega_\eta} f(c) = [\frac{\sqrt{d}}{R+\eta}\sinh(\frac{L(R+\eta)}{\sqrt{d}})]^d \tag{5.47}$$

Similarly,

$$\frac{\max_{c\in\Omega_\eta}\mathbb{E}[Z_c] - \min_{c\in\Omega_\eta}\mathbb{E}[Z_c]}{\min_{c\in\Omega_\eta}\mathbb{E}[Z_c]} \underset{d\to+\infty}{\sim} L\frac{e^{\frac{(R+\eta)^2}{6}}}{\frac{\sinh(R)}{R}} - 1 \tag{5.48}$$

Let $\Delta_\eta = \frac{e^{\frac{(R+\eta)^2}{6}}}{\frac{\sinh(R)}{R}} - 1$. Plots of Δ_η for several values of η are given in Fig. 5.2b.

Let Z_0 be a partition function for a constant discourse vector $x_0 \in \mathcal{S}_R$. The two events are equivalent:

$$|\,Z_c - \mathbb{E}[Z_c]\,| > \epsilon\mathbb{E}[Z_c] \iff \Big|\,\frac{Z_c}{\mathbb{E}[Z_0]} - \frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]}\,\Big| > \epsilon\frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]} \tag{5.49}$$

Using the previous study, we know that

$$\Big|\frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]} - 1\Big| \leq ||\Delta||_\infty \tag{5.50}$$

Which implies that

$$\epsilon\frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]} \geq \epsilon(1 - ||\Delta||_\infty) \tag{5.51}$$

From Equation 5.49:

$$|\,Z_c - \mathbb{E}[Z_c]\,| > \epsilon\mathbb{E}[Z_c] \implies \Big|\,\frac{Z_c}{\mathbb{E}[Z_0]} - \frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]}\,\Big| > \epsilon(1 - ||\Delta||_\infty) \tag{5.52}$$

Let $\mathcal{E}$ be the event corresponding to the right hand side. Then:

$$\begin{aligned}\mathbb{P}(\mathcal{E}) &\leq \mathbb{P}(|Z_c - \mathbb{E}[Z_c]| > \epsilon\mathbb{E}[Z_c]) \\ &\leq \alpha\end{aligned} \tag{5.53}$$

where the second line is obtained from Equation 5.31. We recall that ϵ is an arbitrarily small real number, and

$$\alpha = \exp\left(-\frac{\frac{1}{2}\varepsilon^2 n^2}{n\Lambda + \frac{1}{3}\sqrt{\Lambda}\varepsilon n}\right) \tag{5.54}$$

Hence, with (high) probability $1 - \alpha$:

$$\begin{aligned} -\epsilon(1 - ||\Delta||_\infty) + \frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]} \ \leq \ \frac{Z_c}{\mathbb{E}[Z_0]} \ &\leq \ \frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]} + \epsilon(1 - ||\Delta||_\infty) \\ &\leq \ \frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]} + \epsilon(1 + ||\Delta||_\infty) \end{aligned} \tag{5.55}$$

Again, using:

$$1 - ||\Delta||_\infty \leq \frac{\mathbb{E}[Z_c]}{\mathbb{E}[Z_0]} \leq 1 + ||\Delta||_\infty \tag{5.56}$$

We finally have with probability $1 - \alpha$:

$$(1 - \epsilon)(1 - ||\Delta||_\infty)\,\mathbb{E}[Z_0] \ \leq \ Z_c \ \leq \ (1 + \epsilon)(1 + ||\Delta||_\infty)\,\mathbb{E}[Z_0] \tag{5.57}$$

ϵ is arbitrarily small, and we saw that $||\Delta||_\infty \leq 10^{-1}$, for a domain close to a sphere of radius $R \leq 2$. Setting $\gamma = ||\Delta||_\infty$, this concludes the proof. □

Therefore, the condition of the concentration function in (Arora et al., 2016a, Section 5.1) does not appear as a significant quality test for word vectors, since it holds for any context vector close to a sphere a radius $R \leq 2$. Nevertheless, this concentration property is necessary to prove the main theoretical results of (Arora et al., 2016a). These results were advertised as the theoretical foundations for the relation between linear structure of word vectors and semantic analogies. In order to complete our study in this direction, we shall propose in the next sections an empirical analysis of existing embeddings with regard to analogies and parallelism of vector differences.

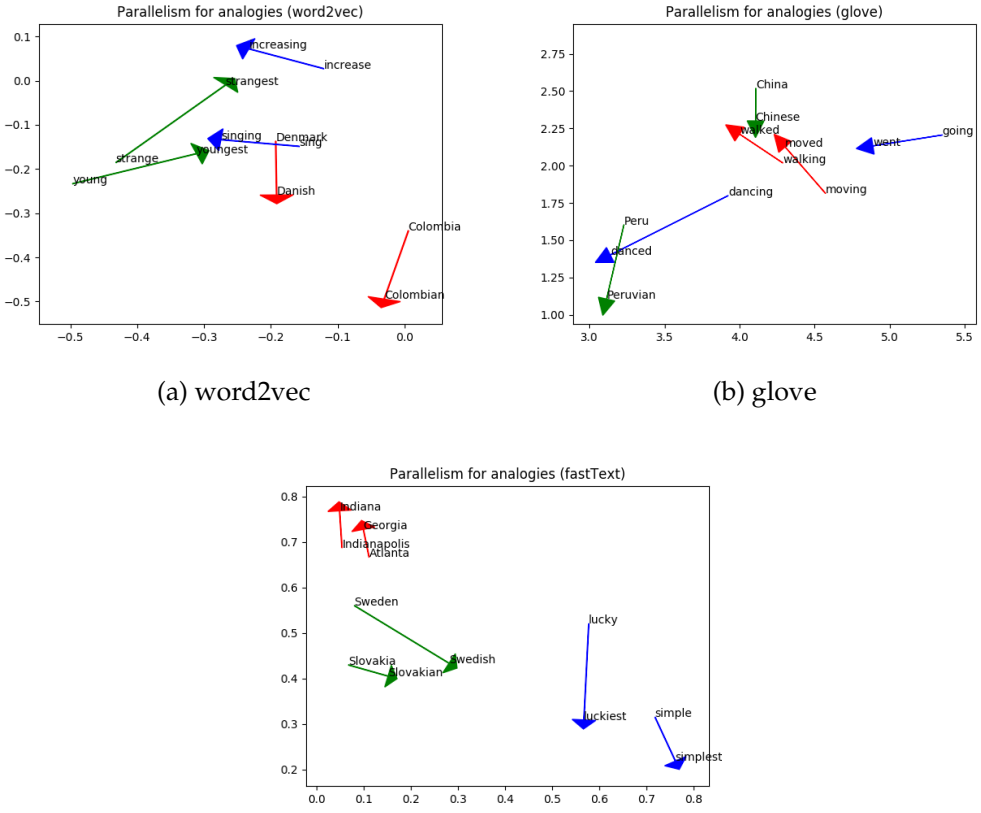
5.4 Experiments with Existing Representations

In this section, we present a list of experiments we ran on the most common used word representations.

5.4.1 Sanity Check

The exact meaning of the statement that analogies are geometrically characterized in word vectors is as follows (Mikolov et al., 2013b; Pennington et al., 2014). For each quadruplet of words involved in an analogy (a, b, c, d), consider the word vector triplet (v_a, v_b, v_c), and the difference vector $x_{ab} = v_b - v_a$. Then we run Principal Component Analysis (PCA, (Jolliffe, 2011)) on the set of word vectors to get representations in $\mathbb{R}^2$. Find the k nearest neighbours of $v_c + x_{ab}$ in the word embedding set (with k small). Finally, examine the k words and choose the most appropriate word d for the analogy $a : b = c : d$.

In order to verify this geometric on some quadruplets, we ran this protocol in many dimensions with a corpus of analogies obtained from (Mikolov et al., 2013a). We display the results which seem to verify property 5.1 obtained in Figure 5.3, which suggest its validity, and motivating a more thorough analysis in this Chapter.



(a) word2vec

(b) glove

(c) fastText

Figure 5.3 – Sanity check run on some analogies obtained from Google analogies (Mikolov et al., 2013a). The x and y axis corresponds to $2D$ dimensional output ot the PCA. Each color represents 4 words implied in an Analogy.

5.4.2 Analogies Protocol

In this subsection we show that the protocol we described in Sect. 5.4.1 for finding analogies does not really work in general. We ran it on 50 word triplets (a, b, c) as input, with $k = 10$ in the k-NN stage, but only obtained 35 correct valid analogies, namely those in Fig. 5.4.

'Athens:Greece=Baghdad:Iraq', 'Ottawa:Canada=Islamabad:Pakistan'
'Ashgabat:Turkmenistan=Athens:Greece', 'Beirut:Lebanon=Bern:Switzerland',
'Bujumbura:Burundi=Conakry:Guinea', 'Doha:Qatar=Hanoi:Vietnam',
'his:her=brothers:sisters', 'easy:easier=simple:simpler',
'low:lower=tight:tighter', 'strong:stronger=bad:worse',
'cold:coldest=low:lowest', 'discover:discovering=enhance:enhancing',
'play:playing=sing:singing', 'think:thinking=implement:implementing',
'Cambodia:Cambodian=Croatia:Croatian', 'Greece:Greek=Italy:Italian',
'flying:flew=jumping:jumped', 'looking:looked=screaming:screamed',
'selling:sold=taking:took', 'thinking:thought=flying:flew',
'child:children=snake:snakes', 'mouse:mice=computer:computers',

Figure 5.4 – Some valid analogies following Protocol 5.4.2 obtained with Google Analogies Dataset. The notation i:j=k:l means that the semantic relation between i and j is similar to the one between k and l.

5.4.3 Turning the Protocol into an Algorithm

The protocol described in Sect. 5.4.2 is termed "protocol" rather than "algorithm" because it involves a human interaction when choosing the appropriate word out of the set of $k = 5$ nearest neighbours to $v_c + (v_b - v_a)$. Since natural language processing tasks usually concern sets of words of higher cardinalities than humans can handle, we are interested in an algorithm for finding analogies rather than a protocol. In this section we present an algorithm which takes the human decision out of the protocol sketched above. Then we show that this algorithm has the same shortcomings as the protocol, as shown in Sect. 5.4.2.

We first remark that the obvious way to turn the protocol of Sect. 5.4.2 into an algorithm is to set $k = 1$ in the k-NN stage, which obviously removes the need for a human choice. If we do this, however, we cannot even complete the famous "king:man=queen:woman" analogy: instead of "woman", we actually get "king" using Glove embeddings.

Following our first definition in Equation (5.1), we instead propose the notion of strong parallelism in Equation (5.58):

$$||v_d - v_c - (v_b - v_a)|| \leq \tau \min(||v_b - v_a||, ||v_d - v_c||) \tag{5.58}$$

where τ is a small scalar. Equation (5.58) is a sufficient condition for quasi-parallelism between $v_d - v_c$ and $v_b - v_a$. The algorithm is very simple: given quadruplets (a, b, c, d) of words, tag the quadruplet as a valid analogy if Equation (5.58) is satisfied. We also generalize the PCA dimensional reduction from 2D to more dimensionalities. We ran this algorithm on a database of quadruplets corresponding to

Table 5.1 – Analogies from Equation (5.58), F1-score

Dimension	word2vec		glove		fastText	
	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.1$	$\tau = 0.2$
2	1.08 %	5.17 %	3.34 %	12.93 %	0.97 %	4.92 %
10	0.00 %	0.00%	0.00 %	0.09 %	0.00 %	0.00%
20	0.00 %	0.00 %	0.00 %	0.00%	0.00 %	0.00%
50	0.00 %	0.00%	0.00 %	0.00%	0.00 %	0.00%
100	0.00 %	0.00 %	0.00 %	0.00%	0.00 %	0.00%
300	0.00 %	0.00 %	0.00 %	0.00%	0.00 %	0.00%

valid analogies (Google analogies, cf. Description 2.7), and obtained the results in Table 5.1. The fact that the results are surprisingly low was one of our initial motivations for this study. The failure of this algorithm indicates that the geometric relation Equation (5.1) for analogies may be more incidental than systematic.

5.4.4 Supervised Classification

The failure of an algorithm for correctly labelling analogies (i.e attributing False or Wrong to a quadruplet of words) based on Equation (5.58) (see Sect. 5.4.3) does not necessarily imply that analogies are not correctly labeled (at least approximately) using other means. In this section we propose a very common supervised learning approach with a simple $k-$NN.

More precisely, we used a $5-$NN to predict analogies using vector differences, following Equation (5.1). If (a, b, c, d) is an analogy quadruplet, we use the representation:

$$x_{abcd} = (v_b - v_a, v_d - v_c) \tag{5.59}$$

to predict the class of the quadruplet (a, b, c, d) (either no relation or being the capital of, plural, etc). If the angles between the vectors $v_b - v_a$ and $v_d - v_c$ - which is directly related to parallelism - contain important information with respect to analogies, this representation should yield a good classification score. The dataset used is composed of 13 types of analogies, with thousand of examples in total[2]. We considered 1000 pairs of words sharing a relation, with 13 labels (1 to 13, respectively: capital-common-countries and capital-world (merged), currency, city-in-state, family, adjective-to-adverb, opposite, comparative, superlative, present-participle, nationality-adjective, past-tense, plural, plural-verbs), and 1000 pairs of

[2]Link to repository

Table 5.2 – Multi-class F1 score classification of analogies based on Equation 5.59 (5-nearest neighbors)

Dimension	word2vec	glove	fastText
2	62.47 %	69.30 %	68.74 %
10	86.44 %	85.62 %	90.40 %
20	74.74 %	77.45 %	80.57 %
50	55.11 %	61.24 %	55.30 %
100	50.57 %	51.26 %	50.56 %
300	51.12 %	51.72 %	49.98 %

words sharing no relation (label 0). In order to generate different random quadruplets, we ran 500 simulations. Average results are in Table 5.2.

The results in Table 5.2 suggest that the representations obtained from Equation (5.59) allow a good classification of analogies in dimension 10 when Euclidean geometry is used with a $5-$NN. However, in the remaining dimensions, vector differences do not encode enough information with regards to analogies.

5.5 Parallelism for Analogies with Graph Propagation

In this section we present an algorithm which takes an existing word embedding as input, and outputs a modified word embedding for which analogies correspond to a notion of parallelism in vector differences. These new word embeddings will be later used (see Sect. 5.6) to contradict the hypothesis that analogies corresponding to parallel vector differences does not make the word embedding better for common classification tasks.

Let r be a strictly positive integer and let us consider a family of semantic relations $(\mathcal{R}_k | 1 \leq k \leq r)$. For instance, this family can contain the plural or superlative relation. One of the relations $\mathcal{R}_k$ creates the analogy $a : b = c : d$, if and only if: $a\mathcal{R}_k b$ and $c\mathcal{R}_k d$, i.e semantic relations create quadruplets of analogies in the following sense:

$$(a, b, c, d) \text{ is an analogy quadruplet} \iff \exists k,\ a\mathcal{R}_k b \text{ and } d\mathcal{R}_k c \tag{5.60}$$

A sufficient condition for Relation (5.1) to hold for a quadruplet is for each pair a, b in the relation $\mathcal{R}_k$:

$$\exists \mu_k \in \mathbb{R}^d \ , \ a\mathcal{R}_k b \iff v_b = v_a + \mu_k \tag{5.61}$$

Equation (5.61) can be generalized to other functions than summing a constant vector, namely it suffices that

$$\exists f_k : \mathbb{R}^d \longrightarrow \mathbb{R}^d \ , \quad v_a \mathcal{R}_k v_b \iff v_b = f_k(v_a) \tag{5.62}$$

Other choices of f_k might be interesting, but are not considered for the remaining of this Chapter.

In order to generate word vectors satisfying Equation (5.61), we propose a routine using propagation on graphs. The first step consists in building a directed graph of words (V, E) encoding analogies:

$$(i, j) \in E \Leftrightarrow \exists k \ (i \mathcal{R}_k j) \tag{5.63}$$

Therefore, we can label each edge of the type k of analogy involved (namely being the capital of, plural, etc). Then, we use a graph propagation algorithm (Algorithm 6) involving Equation (5.61) relation. We remark that propagation requires initial node representations.

Algorithm 6: Graph propagation for analogies

Input: List of relations, vectors $\mu_1, ..., \mu_r \in \mathbb{R}^d$
Output: New representations

1 Build graph G of analogies (Equation (5.63));
2 Extract connected components $C_1, ..., C_c$ from G;
3 **for** $j = 1 \rightarrow c$ **do**
4 Select source node $s_1 \in C_j$;
5 $v_{s_1} \leftarrow$ Generate initial representation of s_1;
6 $s_2, ..., s_{|C_j|} \leftarrow$ Breadth first search from s_1;
7 **for** $r = 2 \rightarrow |C_j|$ **do**
8 $k \leftarrow$ index of relation between s_r and s_{r+1};
9 $v_{s_{r+1}} = v_{s_r} + \mu_k$;
10 Return $(v_i \mid 1 \leq i \leq |G|)$

Proposition 5.2. *Let G the graph of analogies. If G is a forest, then the representations obtained with Algorithm 6 satisfy Equation* (5.61).

Proof. A forest structure implies the existence of a source node s for each component in G. For each component, every visited node with breadth-first search starting from s has only one parent, so the update defined Line 9 in Algorithm 6 defines a representation that satisfies Equation (5.61) for the current node and its parent. □

However, if G is not a forest, words can have several parents. In this case, if $(parent_1, child)$ is visited before $(parent_2, child)$, our graph propagation method will not respect Equation (5.61) for $(parent_1, child)$. This is the case with homonyms. For example, Peso is the currency for Argentina, but the currency for Mexico too. In practice, we selected $\mu_1, ..., \mu_r$ as a family of independent vectors in $\mathbb{R}^d$. We found better results in our experiments when $\forall i,\ ||\mu_i|| \geq d$. This can be explained by the fact that the vectors of relations needs to be non negligible when compared to difference of the words vectors.

This method allows to construct representations of words which almost perfectly respect the analogies. A priori, this constitutes a strong and useful property for many tasks in language processing. However, the main limitation of this method is its dependence on a set of analogies and therefore cannot be generalized to new ones (the dataset of Google Analogies we consider is not exhaustive). This method nonetheless allows us to discuss the importance of this property in the next section with regard to text classification tasks.

5.6 Experiments with New Embeddings

In this section we present results of the experiments described in Sec. 5.4 with the updated embeddings obtained with the propagation Algorithm 6. We call *X++* the new word embeddings obtained with the propagation algorithm from the word embeddings *X*.

5.6.1 Classification of Analogies

Analogies From "parallelism":

As in Section 5.4.3 using Equation (5.58). Results are in Table 5.3. F1-scores are almost perfect (by design) in all dimensions.

With Supervised Learning:

We conduct similar experiments to those described in Section 5.4.4: 1000 pairs of words sharing a relation with 13 labels (1 to 13), and 1000 pairs of words sharing no relation (label 0). Results are in table 5.4.

5.6.2 Text classification: comparison using k-NN

We used three datasets: one for binary classification (Subjectivity) and two for multi-class classification (WebKB and Amazon). For reasons of time computation

Table 5.3 – Analogies from Equation (5.58) with updated embeddings, F1-score

Dimension	word2vec++		glove++		fastText++	
	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.1$	$\tau = 0.2$	$\tau = 0.1$	$\tau = 0.2$
2	96.80 %	96.50 %	97.92 %	97.15 %	98.15 %	97.61 %
10	98.48%	98.54 %	97.88 %	97.88 %	98.25 %	98.31 %
20	98.12 %	98.18 %	98.14 %	98.43 %	96.56 %	96.56%
50	96.80 %	96.80%	98.28 %	98.36 %	98.17 %	98.17%
100	98.08 %	98.08 %	98.19 %	98.19 %	98.06 %	98.06 %
300	98.41 %	98.41 %	98.40 %	98.40 %	98.30 %	98.30 %

Table 5.4 – Multi-class F1 score on classification of analogies based on Equation 5.59 with updated embeddings (5-nearest neighbors)

Dimension	word2vec++	glove++	fastText++
2	99.73 %	99.44 %	99.31 %
10	99.75 %	99.36 %	99.64 %
20	99.80 %	99.52 %	99.94 %
50	99.56 %	99.63 %	99.49 %
100	99.89 %	99.54 %	99.42 %
300	99.40 %	99.86 %	99.45 %

Table 5.5 – Text classification ($d = 20$), F1-score

	Subject.	WebKB	Amazon
word2vec	81.69 %	71.50 %	65.20 %
word2vec++	81.69 %	71.50 %	65.20 %
glove	81.02 %	71.00 %	63.60 %
glove++	80.38 %	72.00 %	61.00 %
fastText	82.14 %	70.50 %	60.00 %
fastText++	81.57 %	72.00 %	56.40 %

we used a subset of WebKB and Amazon datasets (500 samples). The implementation and datasets are available online[3]. Results are in Table 5.5.

5.7 Conclusion

In this chapter, we discussed some properties of a generative model for words vectors and the well-advertised "geometrical property" of word embeddings w.r.t. analogies. Firstly, we discussed a property of a partition function, and proved that this function is almost constant whenever the discourse vectors are close to a sphere of radius ≤ 2, which suggests that the intrinsic quality of word vectors cannot be tested against this property, since it is only a weak necessary condition of the studied generative model.

Second, by using a corpus of analogies, we showed that the related geometric property does not hold in general, in two or more dimensions. We conclude that the appearance of this geometrical property might be incidental rather than systematic or even likely.

This is somewhat in contrast to the theoretical findings of (Arora et al., 2016a). One possible way to reconcile these two views is that the concentration of measure argument in (Arora et al., 2016a, Lemma 2.1) might yield high errors in vectors spaces having dimension as low as $\mathbb{R}^{300}$. Using very high-dimensional vector spaces might conceivably increase the occurrence of almost parallel differences for analogies. By the phenomenon of *distance resolution* (Beyer et al., 1999), however, algorithms based on finding closest vectors in high dimensions require computations with ever higher precision when the vectors are generated randomly. Moreover, the model of (Arora et al., 2016a) only warrants approximate parallelism. So, even if high dimensional word vectors pairs were almost parallel with high prob-

[3]Link to repository

ability, verifying this property might require considerable computational work related to floating point precision.

By creating word embeddings on which the geometrical property is enforced by design, we also showed empirically that the property appears to be irrelevant w.r.t. the performance of a common information retrieval algorithm (k-NN). So, whether it holds or not, unless one is trying to find analogies by using the property, is probably a moot point. We are obviously grateful to this property for the (considerable, but unscientific) benefit of having attracted some attention of the general public to an important aspect of computational linguistics.

6

Distance Geometry and Representations

The content of the previous chapter focused on two main discussions.

First, we studied a geometrical conjecture for a family of word vectors which stated that quadruplets of words implied in semantic analogies often generate almost parallel lines. We discussed this property in details and show with several experiments that it does not hold in general, in two or more dimensions. Furthermore, we showed that this property does not allow to classify analogies with high precision, suggesting that the appearance of this geometrical property might be rather incidental rather than systematic.

Second, we presented the attempts to provide theoretical foundations of this property. By considering the probabilistic generative model proposed by (Arora et al., 2016a), we discussed an essential property of these foundations, called the concentration of the partition function. This property concerns the concentration of this partition function around its mean (with respect to the discourse vector). This means that with high probability, the partition function remains almost constant when the discourse vector varies. In our study, we proved that if the word vectors are generated uniformly in a centered cube, and the discourse vectors remain close to a sphere of radius ≤ 2 (which is one of the assumption made in (Arora et al., 2016a)), the partition function remains almost constant around its mean, regardless the other asymptotic properties of the discourse vectors, which seemed to play an important role in the aforementioned model.

Therefore, our conclusions were two-fold. First, the assumptions implying geometrical relations for analogies are not verified in practice. Either the conjecture of (Pennington et al., 2014) (stated in Equation 5.4) is false or the assumptions of the generative model are not verified empirically. Second, the concentration of partition functions discussed in Section 5.3 does not seem a good evaluation of the quality of word vectors.

Besides, many methods construct word embeddings by solving an unconstrained optimization problem (either maximizing a likelihood or minimizing a misfit of co-occurrence data) via Stochastic Gradient Descent (Pennington et al., 2014; Mikolov et al., 2013a; Bojanowski et al., 2017; Peters et al., 2018; Devlin et al.,

2018). Traditionally, these optimization formulations arise either from word co-occurrence based models (e.g word2vec, GloVe, fastText discussed in Chapter 5), or encoders combined with a masked language model (e.g BERT (Devlin et al., 2018)). In this chapter we propose word embedding methods based on the Distance Geometry Problem (DGP): find object positions based on a subset of their pairwise distances. Considering the empirical Pointwise Mutual Information (PMI) as an inner product approximation, we discuss two algorithms to obtain approximate solutions of the underlying Euclidean DGP on large instances. The resulting algorithms are considerably faster than state-of-the-art algorithms such as GloVe, fastText or BERT, with similar performance for classification tasks. The main advantage of our approach for practical use is its significantly lower computational complexity, which allows us to train representations much faster with a negligible quality loss, a useful property for domain specific corpora.

6.1 Introduction

Context and Motivations

In recent years, the most successful algorithms for Natural Language Processing (NLP) tasks (e.g. text classification, machine translation, named entity recognition) rely on vector representations of words and sentences constructed with a variety of approaches.First, following the first vector space models based on index terms (TF-IDF, Software: SMART (Salton, 1971a), Lucène (Hatcher and Gospodnetic, 2004)), co-occurrence based models in the early 2010s improved empirical performance for NLP tasks, motivating a geometric conjecture relating analogies on word pairs to proximity between the corresponding word vector differences, as advertised in (Pennington et al., 2014). This conjecture was studied in (Arora et al., 2016a) using a connection between scalar products of word vectors and pointwise mutual information (PMI). Despite the uncertain nature of the assumptions supporting this conjecture, as discussed in (Khalife et al., 2019a), the property connecting scalar product of word vectors and PMI hold with high probability at infinity (i.e as the vocabulary size becomes sufficiently large). These properties popularized several representations obtained with methods such as word2vec, Glove or other similar co-occurrence based models (Mikolov et al., 2013c; Pennington et al., 2014; Arora et al., 2017).

Then, to overcome polysemy caused by *static* word embeddings, the family of *contextual representations* (for which a word can be attributed different vectors, depending on its context) have gained momentum, also due to the increased use of deep learning methods. For instance, ELMo (Peters et al., 2018), and BERT (Devlin et al., 2018) representations are based on bidirectional encoders combined with

masked language models incorporating supplementary information such as position, segment (subword information) and improved significantly the empirical performance for a variety of NLP tasks.

Most of these constructions rely on an unconstrained minimization of a loss function using stochastic gradient descent. In this chapter we propose a new method for constructing word vectors, based on Euclidean Distance Geometry. The fundamental problem of Distance Geometry (DG) consists in identifying point positions from information about a subset of their pairwise distances Liberti et al. (2014). The DG literature provides several tools to address this problems in many situations.

More specifically, we use DG based methods in order to develop faster word vectors construction algorithms. Furthermore, we show empirically that such word vectors behave well on extrinsic tasks Melamud et al. (2016) such as text classification.

This Chapter is organized as follows. Section 6.2 briefly reviews some DG concepts useful to devise our algorithms. Section 6.3 describes the word co-occurrence model and introduce DG methods for word embeddings. These methods are compared to the state-of-the-art in terms of the training model and computational complexity. Section 6.5 shows the performance of the methods on intrinsic and text classification tasks. Conclusions are given in Section 6.6.

6.2 Distance geometry

The Distance Geometry problem (DGP, Liberti et al. (2014)) can be defined as follows:

Problem 3 (Distance Geometry Problem (DGP))**.** *Given an integer $K > 0$ a integer, and a simple undirected graph $G = (V, E)$ whose edges are weighted by a positive function $d : E \to \mathbb{R}_+$, determine whether there is a function $x : V \to \mathbb{R}^K$ such that:*

$$\forall \{u, v\} \in E, \ ||x(u) - x(v)|| = d(\{u, v\}) \tag{6.1}$$

where $\|.\|$ denotes the Euclidean norm, $v_i := v(i) \in \mathbb{R}^K$, for all $i \in \mathcal{V}$, and $d_{ij} := d(\{i, j\})$, for all $\{i, j\} \in E$. Let us denote the number of vertices by $n = |\mathcal{V}|$ and the inner product by $\langle \cdot, \cdot \rangle$. A solution for Equation (6.1) is called *valid realization.*

The DGP is known to be NP-Hard Saxe (1979). The most relevant polynomial time case is that of complete graphs, which correspond to fully defined Distance Matrices $D = (d_{ij}^2)$. We say that D is a Euclidean Distance Matrix (EDM) when

Equation (6.1) admits a solution for some dimension K. In this case we can solve Equation (6.1) or determine its infeasibility by a process similar to Classic Multidimensional Scaling and Principal Components Analysis (PCA) (Vidal et al., 2016), and by Build-p algorithms such as those presented in (Dong and Wu, 2002a, 2003) when the distances are all provided with high accuracy.

Let $\mathbf{1}$ denote the vector of ones of appropriate dimension (the vector of zeros will be refered as 0). For a square matrix Z, $\text{diag}(Z)$ denotes a vector on the diagonal elements of Z. From the relation between inner product and Euclidean norm:

$$\|v_i - v_j\|^2 = -2\langle v_i, v_j\rangle + \|v_i\|^2 + \|v_j\|^2. \tag{6.2}$$

Assuming that $\sum_i v_i = 0$, we can define a linear isomorphism $\mathcal{K}$ from the space of symmetric centered matrices $\mathbb{S}_C = \{Y \in \mathbb{R}^{n\times n} \ : \ Y = Y^\top, \ Y\mathbf{1} = 0\}$ to the space of symmetric null diagonal matrices $\mathbb{S}_H = \{Z \in \mathbb{R}^{n\times n} \ : \ Z = Z^\top, \ \text{diag}(Z) = 0\}$, such that $D = \mathcal{K}(G)$ whenever $D_{ij} = \|v_i - v_j\|^2$ and $G_{ij} = \langle v_i, v_j\rangle$ (Al-Homidan and Wolkowicz, 2005). Such a linear transformation is defined by

$$\mathcal{K}(G) = -2G + \mathbf{1}\text{diag}(G)^\top + \text{diag}(G)\mathbf{1}^\top.$$

Its inverse is given by

$$\mathcal{K}^{-1}(D) = -\frac{1}{2}JDJ \tag{6.3}$$

where $J = I - (1/n)\mathbf{1}\mathbf{1}^\top$ is known as centering matrix.

Due to this one-to-one correspondence, when all pairwise distances are available, the problem of finding $v : \mathcal{V} \to \mathbb{R}^K$ such that $\forall \ \{i,j\}, \ \|v_i - v_j\|^2 = D_{ij}$ is equivalent to finding $v : \mathcal{V} \to \mathbb{R}^K$ such that $\forall \ \{i,j\}, \ \langle v_i, v_j\rangle = G_{ij}$.

A remarkable result in distance geometry is Schoenberg's theorem Schoenberg (1935); Gower (1982); Dokmanic et al. (2015), which states that D is Euclidean if and only if $G = (-1/2)JDJ$ is positive semidefinite (PSD). Moreover, if G is PSD, then it is a genuine Gram matrix (matrix of inner products). Let $r = \text{rank}(G)$. A solution for Equation (6.1) is given by $V = \sqrt{\Lambda_r}Q_r^\top$ (cf. Definition 2.4) where $G = Q\Lambda Q^\top$ is the eigendecomposition of G, Λ_r is a $r \times r$ diagonal matrix with the top r eigenvalues of G, and the columns of Q_r contain the corresponding eigenvectors. If $K \geq r$ (we remind that K is the dimension of the DGP problem 3), V is a solution of Equation (6.1). Else, if $K < r$, we can choose among the PSD matrices X of rank $\leq K$, one that minimizes $\|X - G\|_F$ where X is the matrix to be found. A solution is given by $Q_K\Lambda_K Q_K^\top$, where Λ_K is diagonal with top K eigenvalues of G and $Q_K \in \mathbb{R}^{K\times n}$ contains the corresponding eigenvectors in its columns. Thus, an *approximate* realization for Equation (6.1), in dimension K, is given by $V = \sqrt{\Lambda_K}Q_K^\top$.

Last, but not least, if D is not an EDM (e.g. because some d_{ij} come from a noisy measurement), then G is not PSD. Still, a solution of Problem 3 in the least-squares sense is provided by $V^+ = \sqrt{\Lambda_K^+}Q_K^\top$, where $\Lambda_K^+ = \max(\Lambda_K, 0)$, where the

$\max(\cdot,\cdot)$ is component wise. The approximate realization V^+ is a solution of the following optimization problem:

$$\min_{V\in\mathbb{R}^{K\times n}} \quad \|V^\top V - G\|_F^2 = \sum_i \sum_j (\langle v_i, v_j\rangle - G_{ij})^2 \tag{6.4}$$

where v_i is the i-th column of $V = (v_1\ v_2\ \ldots\ v_n)$.

Since this approach relies on spectral decomposition of a matrix of order n, in general, its complexity is bounded by $O(n^3)$ (Golub and Van Loan, 1996). We refer to (Demmel et al., 2007) for better estimates.

6.3 Methodology

A natural question, which is preliminary to the use of DG methods, is whether there exists a distance between words that measures their "semantic difference". Using DG methods, such a distance, even if only partially defined, would yield a set of word vectors satisfying the property:

(A) Two words are semantically correlated if their corresponding vectors are close.

Here, semantic correlation can be interpreted loosely (e.g synonymy, antonym, or more complicated forms of semantic correlation). However, a function verifying the property (A) may not satisfy the distance axioms. Furthermore, as discussed in Globerson et al. (2007), co-occurrence rates also do not satisfy metric constraints. It is more reasonable to consider the statistical nature of the co-occurrence data Turney and Pantel (2010), and to interpret observed object pairs i and as drawn from a joint distribution that is determined by distances or inner products between vectors of the underlying low-dimensional embedding.

In this work, similarly to Globerson et al. (2007), we consider the following model:

$$p(i,j) \propto p(i)p(j)e^{\langle v_i, v_j\rangle} \tag{6.5}$$

where $p(i,j)$ is the probability of finding words i and j in the same window in a corpus (see Section 6.3.1), $p(i)$, $p(j)$ are the marginal probabilities, and $v_i, v_j \in \mathbb{R}^K$ are the corresponding word vectors. This model allows us to devise an approximation for the inner product based on the PMI. It then makes it possible, using DG methods, to construct the vectors to be assigned to words.

6.3.1 Co-occurrences and PMI estimator

By definition, a corpus is a set of documents. Each document is a sequence of elements called tokens whose values are words. The set of all distinct words in

the corpus is called vocabulary. We consider a window of w consecutive tokens which slides over a document. By convention, windows do not overlap document boundaries. Let W be the number of windows in the corpus and n the vocabulary size. We denote by $B \in \mathbb{R}^{W \times n}$ a matrix whose columns are binary vectors $B^i \in \{0,1\}^W$, for $i = 1, \ldots, n$ with components B^i_ℓ such that:

$$B^i_\ell = \begin{cases} 1 & \text{if word } i \text{ appears in window } \ell \\ 0 & \text{otherwise} \end{cases} \tag{6.6}$$

We define the symmetric $n \times n$ matrix $C = B^\top B$ as the matrix of word-word co-occurrence counts. Notice that $C_{ij} = \langle B^i, B^j \rangle$ is the number of windows in which words i and j co-occur and C_{ii} is the number of windows in which word i appears. Furthermore, the vectors $(B^i)_{1 \leq i \leq n}$ are sparse and can be efficiently computed in both time and memory. Recall that $p(i,j)$ is the probability of words i and j appearing together in a window in a corpus and $p(i) = \sum_j p(i,j)$ and $p(j) = \sum_i p(i,j)$ the marginal sums. Following Levy and Goldberg (2014), we use an information theoretic measure, the pointwise mutual information $\mathsf{PMI}(i,j) = \log(p(i,j)/(p(i)p(j)))$ as a measure of association between words Church and Hanks (1990).

Approximating probabilities by relative frequencies, we have

$$\forall \{i,j\}, \qquad \frac{p(i,j)}{p(i)p(j)} \approx \frac{C_{ij}}{\sum_k C_{kj} \sum_k C_{ik}} \sum_{k,l} C_{kl} =: \rho_{ij} \tag{6.7}$$

where C_{ij} is an entry of the co-occurrence matrix. A natural definition for the empirical PMI matrix is the matrix M whose entries are $M_{ij} = \log \rho_{ij}$. However, notice that entries of M corresponding to zero co-occurrences $C_{ij} = 0$ are not well defined. An alternative, commonly used in NLP Church and Hanks (1990); Levy and Goldberg (2014), is to consider the corrected empirical PMI matrix M^0, where

$$M^0_{ij} = \begin{cases} \log \rho_{ij}, & C_{ij} > 0 \\ 0, & C_{ij} = 0 \end{cases} \tag{6.8}$$

which is, moreover, a sparse matrix. A variant of Equation (6.8), also considered in the literature Levy and Goldberg (2014), is the Positive PMI (PPMI): $M^+ = \max(M, 0)$.

As we shall see in Section 6.3.6, many methods for word embeddings employ $\mathsf{PMI}(i,j)$ as a surrogate model for the inner product $\langle v_i, v_j \rangle$, at least implicitly Levy and Goldberg (2014); Arora et al. (2016a); Hashimoto et al. (2016). Since there is no evident reason for assuming $\forall i < j; \langle v_i, v_j \rangle \geq 0$, and, in view of model in Equation (6.5), we choose Equation (6.8) instead of the PPMI matrix.

6.3.2 Noisy distances and the positive definite assumption

In distance geometry, the resolution algorithms are sensible to noisy distances. The impact of the noise with respect to the quality of the approximated solutions is however difficult to establish precisely in all generality. Some results are available for very small errors in the distances and when the framework is *rigid*, which means that the solution of the given problem cannot be perturbed locally by small transformations others than translations or rotations(Anderson et al., 2010).

The quality of the solutions of algorithms depends a priori on a very high precision measurement. In the application we are interested in this chapter, the distances (or equivalently, the scalar products) are noisy in the sense that they might not exactly correspond to distances between low-dimensional vectors. Consequently, we cannot ensure that the PMI matrices are positive definite. In order to provide an order of magnitude of this approximation, we computed some indicators using the following procedure:

(i) First, we extract submatrices corresponding to some k-cores of the underlying graph of the co-occurence matrix (seen as the adjacency matrix). A k-core (where $k \in \mathbb{N}^*$) represents a densely connected subgraph where the induced degree are at least k (cf definition 2.1). Therefore, the k-cores represent connected subgraphs of words. The reason for considering submatrices are due to the important computational time to compute the eigenvalues of large matrices (the total PMI matrix has $81,653$ rows and columns).

(ii) We compute the set of eigenvalues of the corresponding sub-PMI matrix, on the corpus used for our experiments in Section 6.5.

$$\mathsf{ratio} = \frac{\sum\limits_{\lambda \in \mathrm{Sp}(M), \lambda > 0} \lambda^2}{\sum\limits_{\lambda \in \mathrm{Sp}(M)} \lambda^2}$$

The results are in Figure 6.1. For submatrices of size smaller than 10000, the ratio of positive eigenvalues is greater than 90%, and the evolution suggests a ratio greater about 60% for the complete PMI matrix using linear regression on the linear regime $k \geq 1000$.

6.3.3 Geometric Build-Up Algorithm

In Section 6.2, we saw that if all pairwise squared distances d_{ij}^2 are available, then we can solve Equation (6.1) by computing an eigendecomposition in $O(n^3)$ operations, where n is the number of points. In fact, it is sometimes possible to do better than $O(n^3)$. If the graph $\mathcal{G}$ admits a vertex order such that the first $m \geq K + 1$

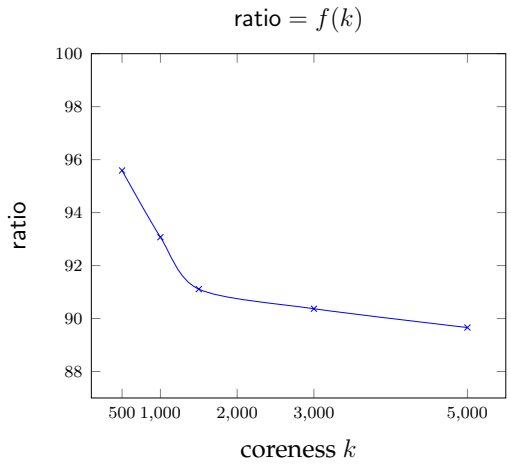


Figure 6.1 – Evolution of the ratio of positive eigenvalues. The x-axis represents the value of the k-core considered. The y-axis represents the ratio (in Euclidean norm) of the positive eigenvalues of the corresponding PMI submatrix.

vertices form a clique and for all other vertices $i > m$, vertex i is adjacent to at least $K + 1$ (with a small language abuse, we say that he has $K + 1$ *adjacent predecessors*), then it is possible to find a solution for the corresponding DGP in linear time (Dong and Wu, 2002b).

For $i > m$, let

$$\delta(i) \subset U(i) = \{j \in \mathcal{V} \, : \, \{j, i\} \in E \, \wedge \, j < i\}$$

be a subset, of size $M \geq K+1$, of the set $U(i)$ of adjacent predecessors of i, with "$<$" being defined by the vertex order. Let us call vertices in $\delta(i)$ by *reference vertices*.

The first m vertices can be embedded first using the process described in Section 6.2, leading to a cost $O(m^3)$. Assuming $|\delta(i)| = K + 1$, for every $i > m$, then, following the vertex order, the position of every other vertex $i = m+1, \ldots, n$ can be found by solving the quadratic system:

$$\forall j \in \delta(i), \quad \|v_j - v_i\|^2 = d_{ji}^2, \tag{6.9}$$

where $\delta(i) = \{j_1, \ldots, j_{K+1}\}$ and $v_{j_1}, \ldots, v_{j_{K+1}}$ are the position vectors of $K + 1$ adjacent predecessors of vertex i. It is not hard to show that when Equation (6.9) admits a solution, it coincides with the one of a $K \times K$ linear system $Ax = b$, where A is nonsingular, provided $v_{j_1}, \ldots, v_{j_{K+1}}$ are affinely independent. See (Dong and Wu, 2002b; Liberti et al., 2014) for further details.

This approach is known in the DG literature as Geometric Build-Up (GBU) (Dong and Wu, 2002b; Wu and Wu, 2007). The total cost of GBU is given by $O(m^3) + (n - m)O(K^3)$, if the involved matrices show no special structure, and therefore is linear with respect to n if m and K are constant.

In the following, we discuss our Distance Geometry methods for word embeddings, which are variants of this idea.

Geometric build-up with inner products. If all pairwise distances between vertices in $\delta(i) \cup \{i\} = \{j_1, \ldots, j_M, i\}$, with $M \geq K + 1$, are known, then, in view of Equation (6.2), the system (6.9) is equivalent to

$$\forall j \in \delta(i), \quad \langle v_j, v_i \rangle = G_{ij}, \tag{6.10}$$

where G_{ij} denotes an entry of the Gram matrix G which, if not directly available, can be computed from Equation (6.3) applied to a submatrix of D containing the squared distances corresponding to the subset of vertices $\delta(i) \cup \{i\}$.

Notice that Equation (6.10) is again a linear system of the form $Ax = b$, but now, with $A^\top = (v_{i_1} \ \ldots \ v_{i_M}) \in \mathbb{R}^{K \times M}$ and $b = (G_{j_1,i}, \ldots, G_{j_M,i})^\top$.

Fixed references, lower cost. Let us suppose that the reference vertices for every vertex $i > m \geq K + 1$ are fixed as $\delta(i) = \{1, \ldots, K + 1, \ldots, m\}$. In this case, the linear system that needs to be solved for each $i > m$ is

$$\langle v_j, v_i \rangle = G_{ij}, \quad j = 1, \ldots, m$$

and, although the right hand side vector $b = (G_{1,i}, \ldots, G_{m,i})^\top$ changes in function of i, the coefficient matrix $A^\top = (v_1 \ \ldots \ v_m)$ is the same for every $i > m$. Thus, concerning the solution of linear systems $Av_i = b_i$, for $i = m + 1, \ldots, n$, we can factor the matrix $A \in \mathbb{R}^{m \times K}$ *only once* and exploit its factorization to actually solve triangular systems of order K for each $i > m$.

Recall that the least-squares solution of an overdetermined, full-rank system of linear equations $Ax = b$, i.e., x that minimizes $\|Ax - b\|^2$, is given by the solution of the triangular system $Rx = Q^\top b$, where $A = QR$, with $R \in \mathbb{R}^{K \times K}$ and $Q \in \mathbb{R}^{m \times K}$ is the "economy size" QR decomposition of A.

This scheme leads to a cost of $O(m^3 + (n-m)K^2)$, where the first submatrix of G of order m is embedded using spectral decomposition (see Section 6.2), followed by QR decomposition of $A = (v_1 \ \ldots \ v_m)^\top$, and the positions of the remaining $n-m$ vertices is found by solving the triangular systems $Rx = Q^\top b_i$, for $i = m+1, \ldots, n$.

6.3.4 Vertex Ordering Problem

The method presented in Section 6.3.3 assumes that vertices of $\mathcal{V}$ can be ordered in a particular way. We remind our reader the definition of a lateration order:

Definition 6.1. *Let K be an integer > 0 and let G be a graph. A vertex order of G is called $(K + 1)$-lateration order*[1]*. if:*

[1]a term that comes from trilateration which is used to find positions in dimension $K = 2$.(Cassioli et al., 2015)

1. *the first $K+1$ vertices form a clique;*

2. *for every vertex $i > K+1$, vertex i has at least $K+1$ adjacent predecessors.*

These orders can be found in polynomial time by a greedy algorithm (Lavor et al., 2012a) which tries to grow a vertex order from a given initial $(K+1)$-clique.

Usually DGP graphs arising from many problems in practice are quite sparse and commonly the set of reference vertices $\delta(i)$ changes for each $i > m \geq K+1$.

However, when the underlying graph $\mathcal{G}$ is complete, *any* vertex order is in fact a $(K+1)$-lateration order. Fortunately this is our case because we know all the entries of the co-occurrence (or empirical PMI) matrix, from which we obtain the adjacency information, and then $\mathcal{G}$ is complete. We recall our reader that sparsity of the PMI matrix does not imply distances are missing: indeed, a 0 coefficient in the matrix does correspond to some the information that the corresponding pair of words do not appear together in a common context. Thus, we are able to apply GBU with fixed references discussed in the previous subsection, where $\delta(i) = \{1, \ldots, m\}$, for every $i > m \geq K+1$.

Therefore, in the GBU-based methods discussed in this chapter, the proposed vertex (word) orderings are aimed simply to improve the quality of the word embeddings. These orderings will determine which entries of the Gram matrix G are taken into account in the GBU method (here, we make the approximation that the PMI matrix is indeed a Gram matrix). Defining a specific vertex order whereas any vertex would allow to obtain solution has also been explored, namely in the case of Multidimensional Scaling (Gramacho et al., 2016). Thus, in terms of a loss function, the vertex order determines the weight of the terms $(\langle v_i, v_j\rangle - G_{i,j})^2$ in the objective: 1 for edges $\{i,j\}$ used in the sequential build-up process and 0 for the others:

$$\min_{V \in \mathbb{R}^{K\times n}} \quad \sum_{i=1}^{m}\sum_{j=i+1}^{m} (\langle v_i, v_j\rangle - G_{ij})^2 + \sum_{i=m+1}^{n}\sum_{j\in\delta(i)} (\langle v_i, v_j\rangle - G_{ij})^2, \tag{6.11}$$

where $m \geq K+1$ is the size of the initial clique. We remark that, when $|E| = n(n-1)/2$ and $\delta(i) = U(i)$, problem (6.11) is equivalent to (6.4).

6.3.5 Divide and Conquer Algorithm

Let G be a Gram matrix of size $n \times n$. The divide and conquer method consists in the two following steps:

• **Divide:** consider submatrices G_i (for $i \leq P$) of G, each having size $n_i \times n_i$, such that the following conditions hold. (i) Each G_i is centered along the diagonal; (ii) G_i and G_{i+1} have at least $K+1$ points in common indexed by I_i; (iii) $\sum_{i=1}^{P-1}(n_i - |I_i|) +$

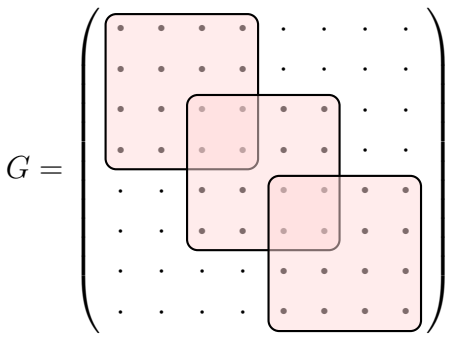


Figure 6.2 – Divide step ($|I_1| = |I_2| = 2$): the order of sub-instances is from top-left to bottom right .

$n_P = n$. The division scheme is displayed in Fig. 6.2. Each submatrix defines a DGP sub-instance. Each such sub-instance is solved with a method such as matrix factorization or GBU to realize the corresponding points.

• **Conquer (merge):** after obtaining the solution of each sub-instance, we have to combine the partial realizations consistently in order to obtain a realization of the whole graph. This operation is carried out sequentially as follows. The solution of the first sub-instance is saved. Then, for an instance $i+1$, for $i \geq 1$, the number I of common points between sub-instances i and $i+1$ must be at least $K+1$ in order to define unique translations and rotations for the common points to be aligned. Let X_{i+1} be the current solution obtained in the divide step, $V_i \in \mathbb{R}^{K \times n_i}$ be aligned vectors obtained at the previous step i, and let $A(:, j)$ denote the j-th column of a matrix A. Then X_{i+1} can be aligned by using Procrustes analysis Schönemann (1966): the best alignment rotation $\hat{Q}_i$ and translation $\hat{T}_i$ are

$$\hat{Q}_i, \hat{T}_i = \underset{Q \in O_K, T \in \mathbb{R}^d}{\operatorname{argmin}} \sum_{k=1}^{I} \|V_i(:, n_i - I + k) - (QX_{i+1}(:, k) + T)\|_2^2. \tag{6.12}$$

Where O_K is the set of orthogonal matrices, i.e:

$$O_K = \{M \in \mathcal{M}_K(\mathbb{R}) \mid M^t M = I_K\}$$

and I_K is the identity matrix. Then, positions V_{i+1} are updated following:

$$V_{i+1} = \hat{Q}_i V_i + \hat{T}_i \, \mathbf{1}^\top \tag{6.13}$$

This method is somehow similar to the one presented (Gonçalves, 2017). However, in this framwork, the author makes the assumptions that we do not know all pairwise distances (or inner products). In order to "complete" some

submatrices, it is necessary to compute missing distances from previously computed positions, which contributes to the error propagation. Also, our method does not contain a pruning part where position candidates are discarded, because the method presented in (Gonçalves, 2017) is intended for problems a with K-lateration order, instead of a (K+1)-lateration where the position are determined without ambiguity (assuming non-degenary of the K+1 references).

It should be noted that PMI-eigs exactly coincides with standard multidimensional scaling (Torgerson, 1952; Cox and Cox, 2008).

6.3.6 Relationship with Other PMI-based Methods

In this section, we put the Distance Geometry methods discussed in previous sections in perspective with other methods for word embeddings. To assess similarities and differences between them, we analyze their underlying optimization problems.

We recall that the empirical PMI matrix M^0, defined by equations (6.7) and (6.8), is used as an approximation of the Gram matrix G (Levy and Goldberg, 2014; Arora et al., 2016a).

PMI-eigs. Word embeddings obtained from the spectral decomposition of the empirical PMI matrix M^0 have been used before in NLP literature (Levy and Goldberg, 2014). In view of (6.4), these word vectors are obtained from

$$\min_{V \in \mathbb{R}^{K \times n}} \quad \|V^\top V - M^0\|_F^2 = \sum_i \sum_j (\langle v_i, v_j \rangle - M^0_{ij})^2, \tag{6.14}$$

where $||.||_F$ is the Frobenius norm. A solution of 6.14 can be constructed from the top K eigenpairs of M^0 as discussed in Section 6.2.

Due to the sparsity of $M^0 \in \mathbb{R}^{n \times n}$, such eigenpairs are usually computed by an Implicitly Restarted Lanczos method (IRLM) (Sorensen, 1992) as implemented in the Matlab routine `eigs`. Its cost per iteration is given by

$$O(q(\gamma n) + (6K + 9)qn + 4q^2 n + 2K^2 n + (K + q)^3)$$

where q is the number of shifts and γ is the average number of nonzero elements of rows of M^0. See (Lehoucq et al., 1998) for more details. If we denote $\tilde{m} = K + q$, then the above cost is $O((\tilde{m} - K)\gamma n + nK^2 + \tilde{m}^3)$.

GBU. By considering the GBU method of Section 6.3.3 with m fixed references, where $\delta(i) = \{1, \ldots, m\}$, for all $i > m$, we aim to solve the optimization problem (6.11), with M^0_{ij} in place of G_{ij}. In this case, the objective function is the same as (6.14), but the sum is not over all pairs $\{i, j\}$, but only those implied by the vertex order. For GBU, this vertex order considers first the m most frequent words in the corpus.

Divide and conquer. Again, the underlying objective function is a variation of (6.14) but summing over the pairs $\{i, j\}$ covered by the sub-matrices $M_1^0, \ldots, M_P^0$ in the divide step. Concerning the vertex order, in DC we have ordered the vertices by decreasing coreness. The coreness of a node is defined as the maximum k such that it belongs to a k-core and not to a $k+1$-core, where a k-core is the maximum subgraph such that each vertex has an induced degree at least k. Computing k-cores can be done in linear time (Batagelj and Zaversnik, 2003).

Notice that the above methods try to fit the inner products $\langle v_i, v_j \rangle$ to the empirical PMI M_{ij}^0. In (Arora et al., 2016a), the relation between the inner product $\langle v_i, v_j \rangle$ and the pointwise mutual information $\mathsf{PMI}(i, j)$ is studied based on a generative model, whereas in (Hashimoto et al., 2016), the authors suggest that when the corpus size tends to infinity, for a sufficiently context window of size w, $\forall, i, j$, there exist a_i and b_i, such that

$$\|v_i - v_j\|^2 \approx -\log(C_{ij}) + a_i + b_j.$$

where C_{ij} is an entry of the co-occurrence matrix. Both models claim to be consistent with matrix factorization methods (Levy and Goldberg, 2014) and others based on regression (Pennington et al., 2014) under certain assumptions.

Glove. In (Pennington et al., 2014), the goal is to find an embedding $v : \mathcal{V} \to \mathbb{R}^K$, by solving a weighted least-squares regression problem

$$\min_{v, \hat{a}, \hat{b}} \quad \sum_i \sum_j f(C_{ij}) \Big(\langle v_i, v_j \rangle + \hat{a}_i + \hat{b}_j - \log(C_{ij}) \Big)^2, \tag{6.15}$$

where $f(C_{ij}) = \min(C_{ij}, 100)^{3/4}$. Therefore, if $\hat{a}$ and $\hat{b}$ were known, one could see (6.15) as a variant of (6.14) weighted by $f(C_{ij})$, by using $G_{ij} \approx \log(C_{ij}) - \hat{a}_i - \hat{b}_j$. Furthermore, for $\hat{a}_i = \log(C_i/\sqrt{S})$ and $\hat{b}_j = \log(C_j/\sqrt{S})$, where $C_i = \sum_j C_{ij}$ and $S = \sum_i \sum_j C_{ij}$, we have $G_{ij} \approx M_{ij}^0 = \log \rho_{ij}$, for $C_{ij} > 0$, i.e., our empirical approximation for $\mathsf{PMI}(i, j)$.

Whenever $C_{ij} = 0$, the corresponding term does not appear in (6.15), but in (6.11), it may contribute to the objective function if either the pair $\{j, i\}$ is in the initial clique or $j \in \delta(i)$.

fastText. In (Bojanowski et al., 2017), an extension of the skip-gram with negative sampling (SGNS) (Mikolov et al., 2013a) is proposed. This extension takes into account the morphology of words: a vector representation is associated to each character s-gram and words are represented as the sum of these vectors.

It was shown in (Levy and Goldberg, 2014) that, under certain assumptions, SGNS corresponds to a matrix factorization problem whose objective is to factor

the word-context PMI matrix (a shifted PMI matrix) via Singular Value Decomposition (SVD, cf. Definition 2.4): $M_{ij}^{SGNS} = \langle W_i, \hat{C}_j \rangle = \langle v_{w_i}, v_{c_j} \rangle = PMI(w_i, c_j) - \log k$, where rows of W and $\hat{C}$ are word and context vectors such that $M = W\hat{C}^\top$, and k is the number of negative samples in SGNS. Since we are considering a simpler co-occurrence based model, our context vectors are in fact the word vectors and we have $\hat{C} = W$ and $M = WW^\top$ such that $M_{ij}^{SGNS} = \langle v_i, v_j \rangle = PMI(w_i, w_j) - \log k$.

Therefore, the above methods are somehow associated, at least implicitly, to a weighted factorization problem where the matrix to be factored is some variant of the PMI matrix.

6.4 Complexity Analysis

In this section, we present a complexity analysis of the different algorithms we compared to construct the word representations. For the sake of completeness, we will present in the next subsection a short description of Arnoldi-Lanczos algorithm used for the PMI matrix decomposition. This presentation is motivated by the lower experimental computing times observed for our Geometric build (GB) and Divide and Conquer (DC) than with PMI-eigs methods. The final complexity comparison and can be found in Section 6.4.2. We also provide experimental running times in Section 6.5.

6.4.1 PMI Eigendecomposition: Arnoldi-Lanczos Algorithm

In this subsection we present an algorithm to obtain the m largest eigenvalues and eigenvectors of a symmetric square matrix A of size $n \times n$, and which is particulary interesting when $m \ll n$. This is the case for the PMI matrix decomposition since the number of words is significantly greater than the desired dimension for the representation.

Given a symmetric square matrix A the Arnoldi-Lanczos process constructs a matrix Q whose columns are orthonormal vectors, such that

$$A = QTQ'$$

where T is symmetric tridiagonal. To this end, the process depends on an initial vector v, and when the algorithm stops, the size of Q is $n \times m$, in which case T has size $m \times m$. The columns of Q form an orthonormal basis for the Krylov subpsace (Watkins, 2007), which is by definition:

$$\mathcal{K}_m(A, v) = \mathsf{span}\{v, Av, A^2v, \ldots, A^{m-1}v\}.$$

This can be thought as the QR decomposition (Gram-Schmidt process) of the matrix

$$K_m = (v \; Av \; \ldots \; A^{m-1}v)$$

i.e., $K_m = Q_m R_m$, with $Q_m \in \mathbb{R}^{n \times m}$. Then, concerning the eigenproblem, the eigenpairs of

$$T_m = Q_m^T A Q_m,$$

will be used to obtain eigenpairs of A.

For now, let us suppose we have built an orthonormal basis $q_1, \ldots, q_j$ of K_j, with $q_1 = \frac{v}{||v||}$. Then, the Gram-Schmidt orthogonalization update of $(A^j v \,|\, 1 \leq j \leq m)$ in order to build a new vector q_{j+1} gives

$$\begin{aligned} r_j &= A^j v - \sum_{i=1}^{j} (q_i^T A^j v) q_i \\ q_{j+1} &= \frac{r_j}{||r_j||} \end{aligned} \tag{6.16}$$

It is important to notice that since $\mathcal{K}_{j+1}(A, v) = \text{span}\{v, Av, \ldots, A^j v\}$, then

$$\mathcal{K}_{j+1}(A, v) = \text{span}\{q_1, Aq_1, \ldots, A^j q_1\}$$

Since $Aq_1 = \alpha q_1 + \beta q_2$, with $\beta \neq 0$, we obtain

$$\mathcal{K}_{j+1}(A, v) = \text{span}\{q_1, q_2, \ldots, A^{j-1} q_2\}$$

and by immediate induction:

$$\mathcal{K}_{j+1}(A, v) = \text{span}\{q_1, q_2, \ldots, q_j, Aq_j\}$$

The main idea to simplify the procedure is to orthogonalize Aq_j at step j instead of $A^j v$ using the Gram-Schmidt scheme:

$$\begin{aligned} r_j &= Aq_j - \sum_{i=1}^{j} (q_i^T A q_j) q_i \\ q_{j+1} &= \frac{r_j}{\|r_j\|} \end{aligned} \tag{6.17}$$

This process may continue until we find $\|r_k\| = 0$, which means that $\text{span}\{q_1, \ldots, q_k\}$ is an invariant subspace of A. Grouping the equations 6.17 for $j = 1, \ldots, k$, we obtain

$$AQ_k = Q_k H_k, \tag{6.18}$$

where $Q_k = (q_1 \ \ldots q_k)$ and H_k is the $k \times k$ matrix defined as follows:

$$(H_k)_{ij} = \begin{cases} q_i^T A q_j & \text{if } j \geq i \\ \|r_i\| & \text{if } i = j+1 \text{ and } i \in \{1, \ldots, k-1\} \\ 0 & \text{otherwise} \end{cases} \tag{6.19}$$

If we apply the process to a symmetric matrix A, then from (6.18)

$$Q_k^T A Q_k = H_k$$

H_k is symmetric because A is. Since H_k is upper Hessenberg (i.e the indices of its non zero-coefficients verify $i \leq j+1$, cf. Definition 2.4), we conclude that $H_k = T_k$ is a symmetric tridiagonal matrix:

$$H_k = \begin{pmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \alpha_2 & \beta_2 & \\ & \ddots & \ddots & \beta_{k-1} \\ & & \beta_{k-1} & \alpha_k \\ & & & \\ & & & \beta_k \end{pmatrix}$$

resulting in an important simplification of (6.17) into the following Arnoldi-Lanczos process: q_{j+1} can be obtained from the three term recurrence:

$$\forall j \in \{1, \ldots, k-1\} \quad A q_j = \beta_{j-1} q_{j-1} + \alpha_j q_j + \beta_j q_{j+1} \tag{6.20}$$

where $\forall j \in \{1, \ldots, m-1\}$:

$$\alpha_j = q_j^T A q_j \tag{6.21}$$

$$r_j = (A - \alpha_j I) q_j - \beta_{j-1} q_{j-1} \tag{6.22}$$

$$\beta_j = \|r_j\|, \quad q_{j+1} = \frac{r_j}{\beta_j} \tag{6.23}$$

We can reformulate equations (6.20)–(6.23) in matrix form:

$$A Q_j = Q_j T_j + \beta_j q_{j+1} e_j^T, \tag{6.24}$$

where e_j is the j-th canonical vector of $\mathbb{R}^j$. The most expensive operation in the iteration defined by (6.21)–(6.23) is the matrix-vector product $A q_j$.

Finally, if $\|r_k\| = \beta_k = 0$ for $k \leq m$, we have $A Q_k = Q_k T_k$, or equivalently

$$Q_k^T A Q_k = T_k$$

If s_i is an eigenvector of T_k, with corresponding eigenvalue θ_i, then

$$A Q_m s_i = Q_m T_m s_i = Q_m(\theta_i s_i) = \theta_i (Q_m s_i),$$

so that $x_i = Q_m s_i$ is an eigenvector of A corresponding to θ_i. Furthermore, at any iteration j, from (6.24), we can compute eigenpairs $(\theta_i^{(j)}, s_i^{(j)})$ of T_j, and define

$$x_i^{(j)} = Q_j s_i^{(j)}.$$

The pair $(\theta_i^{(j)}, x_i^{(j)})$ is considered as an approximation for an actual eigenpair of A. The quality of this approximation is measured by the residual

$$\begin{aligned} \beta_j |e_j^T s_i^{(j)}| &= \|AQ_j s_i^{(j)} - Q_j T_j s_i^{(j)}\| \\ &= \|AQ_j s_i^{(j)} - \theta_i^{(j)} Q_j s_i^{(j)}\| = \|Ax_i^{(j)} - \theta_i^{(j)} x_i^{(j)}\| \end{aligned}$$

Arnoldi-Lanczos "convergence rate"

Since $m \leq n$ for a $n \times n$ matrix A, it does not make sense to think $\{\theta_i^{(m)}\}_m$ as an infinite sequence. But it is still valid to question how fast $\theta_i^{(m)}$ approximates an eigenvalue λ_i of A. In order to answer this question, our reader can refer to (Saad, 1980):

Theorem 6.1 (Saad's inequality: Theorem 2 in (Saad, 1980))**.** *Let $\lambda_1 > \lambda_2 > \cdots > \lambda_k$ be the k largest eigenvalues of the matrix A, u_i their corresponding eigenvectors, $\lambda_n = \inf\{\lambda_i\}$ and let m denote the number of Arnoldi-Lanczos iterations. If v is such that $v^T u_i \neq 0$, for $i = 1, \ldots, k$, then*

$$0 \leq \lambda_i - \theta_i^{(m)} \leq (\lambda_i - \lambda_n) \left(\frac{L_i^{(m)} \tan \theta(u_i, v)}{\mathcal{T}_{m-i}(\gamma_i)} \right)^2, \quad i = 1, \ldots, k,$$

where

$$\gamma_i = 1 + 2\frac{\lambda_i - \lambda_{i+1}}{\lambda_{i+1} - \lambda_n},$$

$$L_i^{(m)} = \begin{cases} \prod_{j=1}^{i-1} \dfrac{\theta_j^{(m)} - \lambda_n}{\theta_j^{(m)} - \lambda_i}, & \text{if } i \neq 1 \\ 1, & \text{otherwise} \end{cases}$$

$\theta(u_i, v)$ is the angle between u_i and v and $\mathcal{T}_{m-i}(.)$ is the Chebyshev polynomial of the first kind and degree $m - i$.

We observe that the rate of convergence depends on the separation between λ_k and λ_{k+1} and the width of the spectrum $\lambda_{k+1} - \lambda_n$. In fact, in (Saad, 1980) it is shown that each $\theta_i^{(m)}$ will converge to λ_i, for $1 \leq i \leq k$, with convergence rate bounded by

$$\tau_i = \left(\gamma_i + \sqrt{\gamma_i^2 - 1} \right)^2$$

Furthermore, according to (Bekas et al., 2008), for $v = q_1$,

$$\tan\theta(u_i, \mathcal{K}_m(A, q_1)) \leq \frac{L_i}{\mathcal{T}_{m-i}(\gamma_i)} \tan\theta(u_i, q_1)$$

where

$$\theta(u_i, \mathcal{K}_m(A, q_1)) = \arcsin \frac{\|(I - Q_m Q_m^T)u_i\|}{\|Q_m Q_m^T u_i\|}$$

Restarted Arnoldi-Lanczos Algorithms

It should be noted that if the initial vector $q_1 = v$ has no components in the direction of an eigenvector u_i of A, i.e. $\langle u_i, v\rangle$=0, then if $\forall x \in \mathcal{K}_m(A, v)$:

$$\begin{aligned}
\langle x, u_i\rangle &= \langle \sum_{i=1}^{m} \lambda_i A^i v, u_i\rangle \\
&= \sum_{i=1}^{m} \lambda_i \langle v, A^i u_i\rangle \\
&= \sum_{i=1}^{m} \lambda_i \langle v, u_i\rangle \\
&= 0
\end{aligned}$$

regardless the number m of steps. Restarting techniques exploit this idea and try to progressively build starting vectors $q_1^{(\ell)}$ that at each iteration ℓ, will have larger components in the direction of desired eigenvectors.

At each iteration ℓ of a restarted Arnoldi-Lanczos methods, an Arnoldi-Lanczos decomposition

$$AQ_m^{(\ell)} = Q_m^{(\ell)} T_m^{(\ell)} + \beta_m^{(\ell)} q_{m+1}^{(\ell)} e_m^T$$

of size m is obtained from the ℓ-th initial vector $q_1^{(\ell)}$. In explicit restarting, the next initial vector $q_1^{(\ell+1)}$ is obtained by

$$q_1^{(\ell+1)} = \psi_\ell(A) q_1^{(\ell)},$$

where $\psi_\ell(\lambda)$ is called filter polynomial.

In implicit restarting, a sequence of p shifted QR iterations are applied to the tridiagonal matrix $T_m^{(\ell)}$ resulting in a new Lanczos factorization of size $k < m = k + p$:

$$AQ_k^+ = Q_k^+ T_k^+ + F^+, \tag{6.25}$$

which is the one that would be obtained with initial vector $q_1^+ = \psi_\ell(A) q_1^{(\ell)}$, where

$$\psi_\ell(\lambda) = \frac{1}{\tau} \prod_{i=1}^{p} \left(\lambda - \mu_i^{(\ell)}\right),$$

$\{\mu_1^{(\ell)}, \ldots, \mu_p^{(\ell)}\}$ is the set of shifts and τ a normalization factor. When p unwanted eigenvalues of $T_m^{(\ell)}$ are taken as shifts, the strategy is called restarting with exact shifts. For more details, we refer to (Bekas et al., 2008).

Decomposition (6.25) can be expanded to a new Arnoldi-Lanczos decomposition with maximum length m by applying p Arnoldi-Lanczos steps, arriving at

$$AQ_m^{(\ell+1)} = Q_m^{(\ell+1)} T_m^{(\ell+1)} + \beta_m^{(\ell+1)} q_{m+1} e_m^T.$$

These ideas lead to the Implicitly Restarted Lanczos Method (IRLM). According to (Lehoucq et al., 1998), for fixed values of k and p, the cost of one iteration of IRLM is

- implicit restart: $2n(k^2 + kp) + O((k+p)^3)$;
- p matrix-vector products Av;
- basic Lanczos steps: $9pn$;
- orthogonalization corrections: $4(kp + p^2)n$,

resulting in a total cost of

$$p(\delta n) + (6k + 9)pn + 4p^2 n + 2k^2 n + O((k+p)^3),$$

where δ is the average number of nonzero entries of rows of A.

6.4.2 Complexity Comparison of GB and DC with Other co-occurrence Based Models

Table 6.1 – Computational complexity for several word vectors realization

Method	Complexity
PMI-eigs	$\mathcal{O}((nK^2 + \tilde{m}^3) n_{\text{iter}})$
GBU (fixed references)	$\mathcal{O}(m^3 + (n-m)K^2)$
Divide and conquer (DC)	$\mathcal{O}(\sum_{i=1}^{P} (n_i K^2 + \tilde{m}_i^3) n_{iter}^{\prime(i)} + (K+1)^3 (P-1))$
Glove	$\mathcal{O}(p^{0.8} n_{\text{epochs}} K)$

The complexity of the co-occurrence based methods are summed up in Table 6.1. Let $n = |\mathcal{V}|$ be the size of the vocabulary and K the dimension. Computation of the top K eigenpairs of the PMI matrix requires $\mathcal{O}((nK^2 + \tilde{m}^3) n_{\text{iter}})$, where n_{iter} is the number of Implicitly Restarted Lanczos method (IRLM) iterations, which depends on the spectrum of the PMI matrix, and $\tilde{m} = K + q$, where q is the number of shifts, discussed in Section 6.3.6.

For the divide and conquer method, $n_{\text{iter}}^{\prime(i)}$ represents the number of iterations to solve each sub-instance i using IRLM. Also we consider exactly $K+1$ anchors (also for our experiments), i.e $|I_1| = ... = |I_{P-1}| = K+1$, where P is the number of sub-instance. Finally, it should be noted that usually $n'_{\text{iter}} \ll n_{\text{iter}}$ hence PMI-eigs complexity is not necessarily lower.

The number of shifts q and q_i are internally set in the implementation of `eigs` in Matlab and it is difficult to estimate, even though it is likely that $K \leq q \ll n$ and $K \leq q_i \ll n_i$. Besides, the number of iterations $n'_{\text{iter}}, n_{\text{iter}}$ depends on the distribution of the eigenvalues of the corresponding matrices. For these reasons, we do not know how to compare their theoretical complexity. However, we will compare their empirical running times in Section 6.5.

The complexity study for Glove is detailed in (Pennington et al., 2014, Section 3.2) and indicates $\mathcal{O}(p^{0.8} n_{\text{epochs}})$ where p is the number of tokens in the corpus, and n_{epochs} the number of epochs of the stochastic gradient descent (for a description of Stochastic gradient descent, our reader may refer to (Carpentier and Cohen, 2017; Bottou et al., 2018)). For a fair comparison with other methods, this complexity also depends linearly on the dimension.

For the first three methods of Table 6.1, the dimension seems to be a drawback, but their complexities are good in practice. For example, for a corpus composed of $p = 2.66 \times 10^8$ tokens (corpus described in Section 6.5.1), $n_{\text{epochs}} = 15$ (standard corpus size and parameters), containing about $n = 10^5$ different words, dimension $K = 50$, with $m = 200$ references, then:

$$\begin{aligned} \mathcal{C}_{\mathsf{Glove}} &= p^{0.8} n_{\text{epochs}}\, K && \approx 4.13 \times 10^9 \\ \mathcal{C}_{\mathsf{GBU}} &= m^3 + (n-m)K^2 && \approx 2.50 \times 10^8 \end{aligned} \tag{6.26}$$

These numbers are consistent with running times in our experiments. It should be noted that these complexity numbers do not take in account pre-processing of the corpus, which is in all cases $\mathcal{O}(p)$: this corresponds to one pass through the corpus in order to construct the vocabulary, and possibly ignore low-frequency terms.

6.5 Experiments

6.5.1 Description of The Experiments

To construct our word vectors, we used a corpus of 10^6 documents from Wikipedia 2016 [2], which we cleaned using standard pre-processing methods in NLP (stop words and punctuation removal). is composed of $266,561,061$ tokens and $81,653$

[2] Wikipedia is a standard dataset for the constructions of representations, or the training of neural networks in Natural Language processing, cf. 123; 138.

words. We used a window size of $w = 10$, a standard value used in several word embedding methods (Mikolov et al., 2013a; Pennington et al., 2014; Arora et al., 2016a). We provide two experimental evaluations of our word vectors. First, an intrinsic evaluation of word vectors using QVEC (Tsvetkov et al., 2015), which has a good correlation with performance of the vectors with semantic evaluation tasks. QVEC is an efficient intrinsic evaluation measure of the quality of word vectors based on alignment of features extracted from lexical ressources. These evaluations are reported in Table 6.2. Second, we evaluate the quality of these representations over three text classification tasks compared with other word embeddings. Our implementation includes 3 datasets: WebKb (Multiclass), Subjectivity (Binary) and Amazon (Binary). Results are reported in Table 6.3. We compare with a baseline of random word vectors whose components are drawn from a standard Gaussian. For our classification experiments, we use an implementation of a Convolutional neural network (Kim, 2014) using Tensorflow library (Abadi et al., 2015) version 1.12[3]. We also compare with bidirectional encoders (BERT) (Devlin et al., 2018).

We provide some practical computing times, in the line of our complexity study. The embeddings and corresponding times for PMI-eigs, GBU and DC were generated using Matlab [v.2018.b], running on a CPU of 2 cores Intel(R) Core i5 1.8Ghz with 8 Gb of Ram. Glove and fastText were compiled in the same machine using GCC Apple LLVM version 10.0.0 (clang-1000.10.44.4).

Table 6.4 reports the times for obtaining the word vectors (left) only, for dimensions $K = 50, 100, 200$, and the total time including the CNN training for $K = 200$ (right); parsing time is not included. Besides, for PMI-eigs, DC and GBU, we added the time for computing the matrix M^0 from co-occurrence counts ($\approx 120s$) and for DC and GBU we also consider the time for computing the corresponding vertex order ($\approx 90s$ and $\approx 30s$, respec.). The parameters used in GBU were $M = m = 4K$ and the ones of DC were $n_1 = 20000$ and $n_{p\geq 2} = 800$. For Glove and fastText we kept the default parameters from the official code.

6.5.2 Discussion

Table 6.4 shows that our DG word vectors, when combined with a convolutional neural network lead to F1 scores close to those of BERT in three text classification tasks, but demanding about half to a quarter of the computing time, depending on the dataset.

Moreover, from Table 6.4 we observe that the training times for the DG based methods are remarkably smaller than those of standard word vectors construction (and also than the training time of bidirectional encoders such as BERT). They also improve the computational time with respect to the spectral decomposition

[3]We noticed a significant drop of performance when using version 2.1, the reasons remain unknown.

Table 6.2 – Intrinsic evaluation (QVEC). First and second best are in bold and underline, respectively.

Representation	$K = 50$	$K = 100$	$K = 200$
Random	9.59	14.89	21.82
GBU ($m = M = 4K$)	22.72	30.01	37.41
GBU (PPMI)	22.78	30.35	38.61
DC ($n_1 = 20000, n_{i\geq 2} = 800$)	26.83	34.21	41.42
PMI-eigs	**29.20**	**36.55**	**43.74**
Glove	<u>28.22</u>	35.43	42.06
fastText	28.17	<u>36.06</u>	<u>43.51</u>

Table 6.3 – F1 score - Text classification. First and second best are in bold and underline, respectively.

	$K = 50$			$K = 100$			$K = 200$		
Representation	Subject	WebKB	Amazon	Subject	WebKB	Amazon	Subject	WebKB	Amazon
Random	77.07	90.01	74.96	80.09	91.12	74.26	81.34	91.84	76.33
GBU ($m = M = 4K$)	87.67	92.33	79.65	<u>88.21</u>	93.04	80.58	88.05	**93.68**	81.39
DC ($n_1 = 20000, n_{i\geq 2} = 800$)	<u>87.87</u>	91.42	<u>81.97</u>	87.86	92.03	80.61	<u>88.28</u>	92.65	<u>82.49</u>
PMI-eigs	87.85	91.67	79.23	88.17	92.19	80.0	88.10	92.52	81.86
Glove	86.34	**92.99**	79.35	87.96	<u>93.07</u>	79.75	87.62	93.24	79.76
fastText	87.65	<u>92.84</u>	78.67	87.71	**93.37**	<u>80.64</u>	88.02	<u>93.57</u>	81.57
BERT + fine tuning	**91.19**	91.56	**84.28**	**91.19**	91.56	**84.28**	**91.19**	91.56	**84.28**

of the PMI matrix. The price to be paid for these extremely fast word vectors, whose performance in text classification is close to the state-of-the-art (Table 6.3), is possibly a slightly inferior performance in intrinsic tasks (Table 6.2).

6.6 Conclusion

We proposed a formulation based on the Distance Geometry (DG) problem to generate word vectors. The resulting Geometric Build-Up and Divide and Conquer algorithms are considerably faster than state-of-the-art algorithms, as Glove and fastText.

The word vectors obtained by DG methods have performance close to state-of-the-art in our intrinsic evaluation (QVEC). Also, combined with a convolutional neural network, DG word vectors lead to F1 scores close to those of BERT in three text classification tasks, but demanding about half to a quarter of the computing time, depending on the dataset. The superior speed of our proposed methodology goes towards the aim of rapidly generating word embeddings from given corpora relating to specific applications.

Table 6.4 – Computing times (Left: Word Vectors, Right: Total). First and second best are in bold and underline, respectively. NA: BERT embeddings are trained only for the end-task (right table).

	Dimensions		
Representations	50	100	200
Glove	44m	1h07m	2h07m
fastText	26m	30m	48m
PMI-Eigs	<u>284s</u>	412s	836s
DC ($n_1 = 20000$, $n_{i\geq 2} = 800$)	307s	<u>348s</u>	<u>452s</u>
GBU ($m = M = 4K$)	**159s**	**168s**	**188s**
BERT	NA	NA	NA

	Datasets		
Represent. + Classif.	Subject	Amazon	WEBKB
Glove + CNN	7815s	8640s	9120s
fastText + CNN	3075s	3900s	4380s
PMI-Eigs + CNN	1031s	1856s	2336s
DC + CNN	<u>602s</u>	<u>1368s</u>	<u>1952s</u>
GBU + CNN	**383s**	**1208s**	**1688s**
BERT + fine tuning	663s	3260s	4647s

Indeed, contextual word embeddings (ELMo, BERT) have significantly improved performance for many NLP tasks recently. However, these models have been minimally explored on specialty corpora, such as clinical text, as reported in (Alsentzer et al., 2019). In such cases, there is no publicly-available pre-trained BERT models. A first possibility is to retrain contextualized word embeddings on the new corpus, which can turn out to be time consuming. In this context, representations which can be trained much faster with a negligible quality loss can turn out to be useful. In particular, we showed in this chapter that DG based (non contextual) word embeddings have this property, and are also competitive on general NLP classification tasks. These preliminary results motivate us to investigate the construction of contextualized representations with DG by capitalizing on the time gain.

We believe there are several ways to extend this work. The first one is to enrich the information in order to construct context-aware and out of training samples representations, which would help to address other NLP problems. A potential way to address their construction would be to consider "multichannel" distance geometry, stacking different distances (or a corresponding scalar product) in different channels, and solve the different instances in order to obtain a family of realizations. An example of channel could correspond to subword information (subword similarities), or information from other networks (e.g dictionaries, reference networks (Niwa and Nitta, 1994)). Also, a natural continuation is to replace the Build-Up algorithm with other Distance Geometry methods, such as the ones presented in (Lavor et al., 2012b; Omer and Mucherino, 2020).

Finally, the employed DG algorithms to obtain word vectors scale well with the parameters of the problem (vocabulary size and dimension). Therefore, Distance Geometry seems a promising paradigm for representation learning for other applications in machine learning (e.g graph classification).

7 Concluding Remarks

As discussed in the introduction, the formal modeling of natural language, along with its learnability (Thom, 1970; Valiant, 1984) are doomed to failure. In this thesis, we proposed theoretical and empirical analyzes of efficient representations for the treatment of some information extraction and natural language processing problems.

7.1 Summary of Contributions

First, graph-based representations of sequences, which are the native data structure of written language, pose ambiguity problems, raising theoretical questions at the interface of graph theory and pattern recognition. In Chapter 3, we have developed a theoretical study and presented algorithms solving a recognition and counting problem, which allowed to show that an important degree of ambiguity can arise from co-occurence based models.

However, these graph-based representations allow to deal efficiently with certain fine-grained information retrieval problems, with state-of-the art performances, as shown in Chapter 4 with the problem of entity identification. Through the analysis of a knowledge base of economic and financial nature, we also suggested its application in a broader framework.

Then, in Chapter 5, we discussed a geometric approach, based on a generative statistical model yielding certain geometric properties modulo a noise level. Mainly, these properties are based on the relation between the scalar product of vector representations of words and their local mutual information. Relations with analogies can be deduced, given an additional conjecture. Although some of these properties are mathematically valid, it remains that language does not seem to verify them in practice, which can be due to a high noise level, as these properties hold with high probability at infinity: yet we do not know the convergence speeds of these models.

Finally, in Chapter 6, we proposed a new method for the generation of representations useful for learning algorithms. These representations are based on the paradigm of the geometry of distances which, from a set of distances, realizes the points satisfying these constraints. We have presented theoretically valid res-

olution algorithms when the distances are exact. However, the existence of a Euclidean distance respecting the geometric axioms seems unrealistic. Thus, the distances that we consider are pseudo-distances implicitly obtained with an approximation of a scalar product. Nonetheless, the solutions we obtain are at the same time competitive with state-of-the-art performance for text classification, while being able to be calculated with better complexity.

7.2 Perspectives and Future Work

The first continuation of this work concerns the characterization of sequence graphs, as well as complexity properties of the recognition and counting problems (at a first stage P or NP-hardness). In addition to its theoretical interest, it could also lead to other interesting applications. Due to the three levels of generalizations (orientation, weights, and window size), some cases remain open. A possible direction is the investigation of forbidden patterns in these structures.

Second, we noticed that there is significant room for improvement of entity identification algorithms. Indeed, the use of other methods for the filtering part are preferable, for example with finite state transducers (Hetherington, 2004), which allow to compute association rules very efficiently.

Then, we think it can be of interest to study the speed of convergence or noise levels in the probabilistic equalities of the generative models studied in Chapter 5, which could explain the differences between theoretical and empirical relations.

Distance geometry can also be an interesting new method for exploring other problems in machine learning and artificial intelligence. Concerning natural language, we have seen that it allows to construct performant representations efficiently. One could also enrich the distances used with several distances and construct the corresponding representations in "parallel" as described in the conclusion of Chapter 6. However, it remains difficult to construct Euclidean distances respecting both geometric axioms and the semantic relationships. It may be worth to investigate other types of geometry, for instance differential manifolds for word and sentence representations. In this setting, distances would correspond to geodesics. Another possibility would be to consider geometric shapes (e.g solides, or polytopes).

Finally, it also appeared during this thesis that both graph-based or geometric models seem strongly disconnected from linguistic structures and notions. It seems illusory to think that these models will be able to analyze a text as a human would this way, especially for literature. For this reason, it is desirable to integrate pure linguistic notions into the models. For this regard we thought of the following possibilites:

- Use the referencing of construction types. This has namely be studied for

French in (Gross, 1982).

- Classification of the types of sentences: subject - verb - object sentences using the algebraic description of singularities of dynamical systems (Thom, 1970).

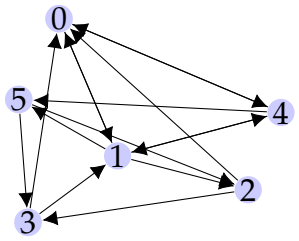
8 Appendix

8.1 Chapter 3: Supplementary Figures and Experiments

In this section we provide supplementary illustrations related to this chapter.
In particular, are depicted:

- Different instances of sequence graphs having several realizations are depicted. These realizations have been computed using our dynamic programming formulation.
- Supplementary illustrations of Example 3.1 and the exponential increase of paths in the corresponding directed acyclic graph $R(H)$.

8.1.1 $p = 10$, $w = 3$



(a) Sequence graph

$$\begin{pmatrix} 0 & 2 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 2 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}$$

(b) Weights matrix Π

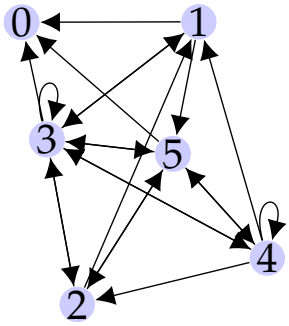
4 1 5 2 3 0 1 4 0 1

4 1 0 4 1 5 2 3 0 1

(c) Different Realizations ($w = 3$)

Figure 8.1 – $p = 10$, $w = 3$

8.1.2 $p = 10$, $w = 4$



(a) Sequence graph

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 2 \\ 0 & 1 & 1 & 2 & 1 & 1 \\ 1 & 0 & 1 & 2 & 1 & 0 \end{pmatrix}$$

(b) Weights Matrix Π

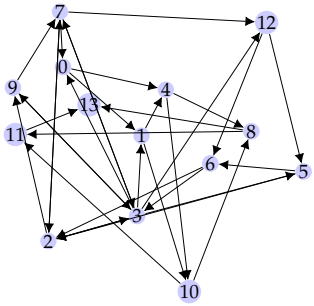
4 3 5 4 2 3 1 5 3 0

4 3 5 4 3 2 1 5 3 0

(c) Different Realizations ($w = 3$)

Figure 8.2 – $p = 10$, $w = 4$

8.1.3 $p = 20$, $w = 3$



(a) Sequence graph

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 2 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

(b) Weights matrix Π

7 3 2 7 3 12 5 6 2 3 9 7 3 0 1 4 10 8 11 13

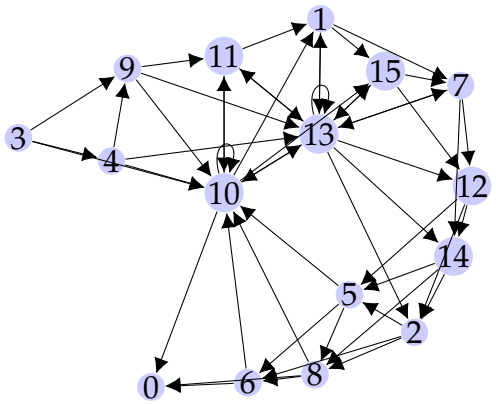
7 3 12 5 6 2 3 9 7 3 2 7 3 0 1 4 10 8 11 13

7 3 12 5 6 2 3 7 2 3 9 7 3 0 1 4 10 8 11 13

(c) Different realizations

Figure 8.3 – $p = 20$, $w = 3$

8.1.4 $p = 20$, $w = 4$



(a) Sequence graph

$$\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\
1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 2 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0
\end{pmatrix}$$

(b) Weights matrix Π

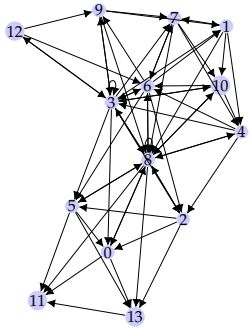
3 4 9 10 13 11 10 1 13 15 7 13 12 14 2 5 8 6 10 0

3 4 9 10 13 11 10 13 1 15 7 13 12 14 2 5 8 6 10 0

(c) Different realizations

Figure 8.4 – $p = 20$, $w = 4$

8.1.5 $p = 20$, $w = 5$



(a) Sequence graph

$$\begin{pmatrix}
0&0&0&0&0&0&0&0&1&0&0&1&0&1\\
0&0&0&0&1&0&0&1&1&0&1&0&0&0\\
1&0&0&0&0&1&0&0&1&0&0&0&0&1\\
1&2&1&2&0&1&2&1&2&2&1&0&1&0\\
0&0&1&1&0&0&1&0&1&0&0&0&0&0\\
1&0&0&0&0&0&0&0&1&0&0&1&0&1\\
0&1&1&2&0&1&0&1&1&1&0&0&0&0\\
0&0&0&0&1&0&1&0&1&0&1&0&0&0\\
1&0&1&1&1&1&1&0&1&0&1&1&0&1\\
0&1&0&1&0&0&0&1&1&0&0&0&0&0\\
0&0&0&1&1&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&2&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0
\end{pmatrix}$$

(b) Weights matrix Π

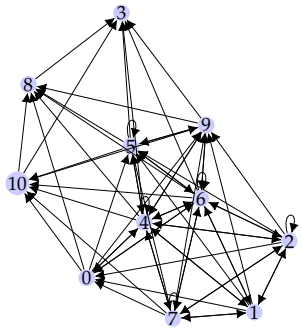
3 12 3 6 9 3 1 7 8 10 4 6 8 3 2 5 0 8 13 11

3 12 3 6 9 3 1 7 8 10 4 6 3 8 2 5 0 8 13 11

(c) Different realizations

Figure 8.5 – $p = 20$, $w = 5$

8.1.6 $p = 20$, $w = 10$



(a) Sequence graph

$$\begin{pmatrix}
0&0&0&0&2&2&2&0&1&1&1\\
1&0&1&0&2&1&2&1&0&1&0\\
2&1&1&0&3&2&5&3&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0\\
2&1&3&2&3&3&7&3&2&2&2\\
0&0&0&2&2&1&2&0&2&1&2\\
2&1&1&2&6&5&4&3&2&2&2\\
2&1&1&0&4&3&4&1&0&1&1\\
0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&1&1&1&1&0&1&0&1\\
0&0&0&1&0&0&0&0&1&0&0
\end{pmatrix}$$

(b) Weights matrix Π

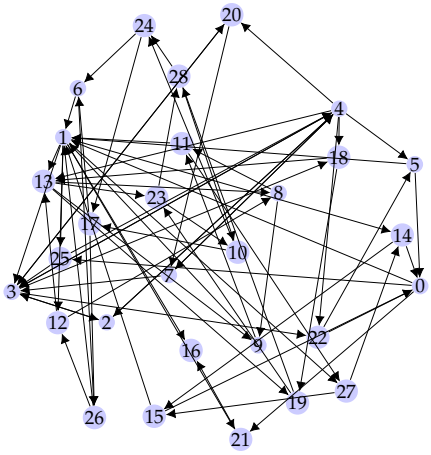
4 2 6 7 1 4 2 6 7 0 4 5 6 9 6 5 4 10 8 3

4 2 6 7 1 4 2 6 7 0 5 4 6 9 6 4 5 10 8 3

(c) Different realizations

Figure 8.6 – $p = 20$, $w = 10$

8.1.7 $p = 40$, $w = 3$



(a) Sequence graph $p = 40$

$$\begin{pmatrix}
0&2&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&1&1&1&0&1&0&0&0&0&1&0&0&0&1&0&1&0\\
0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&4&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&1&0&0&0&0&0&0\\
0&0&1&1&0&1&0&1&0&0&0&0&0&1&0&0&0&0&1&0&1&0&1&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0
\end{pmatrix}$$

(b) Weights matrix Π

3 4 2 3 4 18 13 19 23 28 10 24 17 6 26 1 12 13 8 9 11 1 27 14 15 0 1 25 3 4 20 7 3 4 22 5 0 1 21 16

3 4 2 3 4 20 7 3 4 18 13 19 23 28 10 24 17 6 26 1 12 13 8 9 11 1 27 14 15 0 1 25 3 4 22 5 0 1 21 16

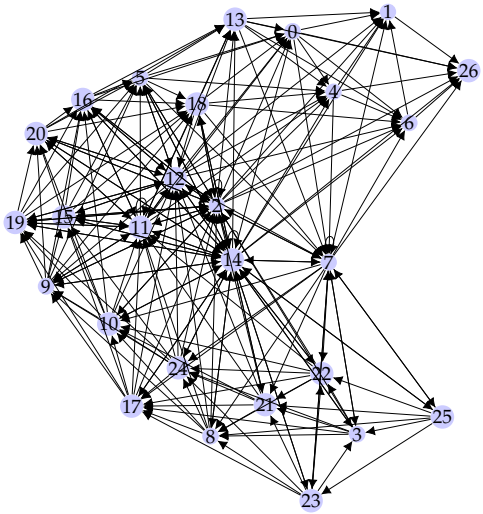
3 4 2 3 4 22 5 0 1 25 3 4 20 7 3 4 18 13 19 23 28 10 24 17 6 26 1 12 13 8 9 11 1 27 14 15 0 1 21 16

3 4 2 3 4 22 5 0 1 25 3 4 20 7 3 4 18 13 19 23 28 10 24 17 6 26 1 12 13 8 9 11 1 27 14 15 0 1 21 16

(c) Different realizations

Figure 8.7 – $p = 40$, $w = 3$

8.1.8 $p = 40$, $w = 10$



(a) Sequence graph $p = 40$

$$\begin{pmatrix}
0&1&0&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
1&1&2&0&1&1&1&1&0&1&1&2&2&1&4&1&1&0&1&1&2&0&0&0&0&0&1\\
0&0&1&0&0&0&0&1&1&0&0&0&0&0&2&0&0&1&0&0&0&1&1&0&1&0&0\\
0&1&0&0&0&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
1&0&1&0&1&0&0&1&0&0&0&0&1&1&2&0&0&0&1&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
0&1&1&2&0&0&1&3&2&0&1&0&0&0&6&0&0&1&0&0&0&3&3&2&2&1&1\\
0&0&1&0&0&0&0&0&0&1&1&1&1&0&2&0&0&1&0&0&0&0&0&0&1&0&0\\
0&0&1&0&0&1&0&0&0&0&0&1&0&0&1&1&1&0&1&1&1&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&1&0&1&1&0&1&1&1&0&0&1&1&0&0&0&0&0&0\\
1&0&2&0&1&1&0&0&0&1&0&1&2&1&3&1&1&0&1&1&1&0&0&0&0&0&0\\
1&1&2&0&1&1&1&1&0&1&0&1&0&1&3&1&1&0&0&1&1&0&0&0&0&0&0\\
1&1&0&0&1&0&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1\\
2&2&5&2&2&1&2&5&2&2&2&3&3&2&7&2&1&2&1&1&1&2&2&2&2&1&2\\
0&0&1&0&0&1&0&0&0&0&0&1&1&0&1&0&1&0&1&1&1&0&0&0&0&0&0\\
1&0&1&0&0&1&0&0&0&0&0&1&1&1&2&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&2&0&0&0&0&0&0&1&1&1&1&0&1&1&0&0&0&1&0&0&0&0&0&0&0\\
1&0&1&0&1&0&1&1&0&0&0&0&1&1&2&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&1&0&0&0&0&0&1&1&0&2&0&1&0&1&0&1&0&0&0&0&0&0\\
0&0&1&0&0&1&0&0&0&0&0&1&1&1&2&0&1&0&1&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&1&0&1&1&1&0&2&0&0&1&0&0&0&0&0&0&1&0&0\\
0&0&1&0&0&0&0&0&1&0&1&1&0&0&2&0&0&1&0&0&0&1&0&0&1&0&0\\
0&0&0&1&0&0&0&1&1&0&0&0&0&0&2&0&0&1&0&0&0&1&1&0&1&0&0\\
0&0&1&0&0&0&0&0&0&0&1&1&1&1&0&1&1&0&1&0&1&0&0&0&0&0&0\\
0&0&0&1&0&0&0&2&1&0&0&0&0&0&2&0&0&0&0&0&0&1&1&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{pmatrix}$$

(b) Weights matrix II

7 14 25 7 14 23 3 7 22 21 8 14 24 17 14 2 10 11 12 9 15 19 2 14 20 16 11 5 18 12 14 2 14 13 0 4 7 6 1 26

7 14 25 7 14 23 3 7 22 21 8 14 24 14 17 2 10 11 12 9 15 19 2 14 20 16 11 5 18 12 14 2 14 13 0 4 7 6 1 26

(c) Different realizations

Figure 8.8 – $p = 40$, $w = 10$

8.2 Chapter 4: Analysis of a Knowledge Graph

We saw in the previous subsections that information retrieval problems such as entity identification depend on the existence of prior information, represented in structures such as a knowledge graph. In this subsection, we present an additional example of knowledge graph of entities, and discuss a methodology to analyze it. We suggest this network can be also treated as in the previous subsections, for named entity identification or more general tasks. We present a summary of the methodology and the results. We believe they can be used in the case of preprocessing and understanding of a knowledge base. An extended version of this Chapter is available in (Khalife et al., 2019b).

8.2.1 Introduction

Relationships between legal entities can be represented as a weighted directed graph. We model the global capital ownership network and establish a methodology for an empirical analysis of its structure and influence of its entities. To do so, we employ a variety of metrics from graph analytics and algorithms from the area of influence maximization. We show that our analysis aligns with macro-economic information, such as the presence of tax heavens in dense subgraphs of countries, and countries whose private capital is principally owned by others (taking France as a case study). Moreover, our results also offer novel intuitions and metrics in this area by highlighting the existence of strong communities of capitalistic property. Finally, we discuss and develop influence maximization methods as a means to evaluate the impact of entities in this context.

A *legal entity* is a juridic term that designates an individual, company, or organization that has legal rights and obligations. In the standard terminology, legal entities are usually divided into individuals and corporations (e.g companies). The economic entity principle, stating that financial transactions must be assigned to a specific business, is considered as one of the fundamental principles of accounting: an entity must have separate accounting records, except for its subsidiaries. This principle implies that the nature and clarity of the information describing legal entities is important for compliance standards. Several types of interactions can exist between these entities, for instance payments or capitalistic property, which define several type of networks. Some such networks have already been studied, namely the interbank market Boss et al. (2004) and interbank payment flows Soramäki et al. (2007). A study on systemic risk in the interbank network, where payment interactions are treated as a complex network was described in Lenzu and Tedeschi (2012). Besides, other work attempted to describe the actual topologies observed in the financial system Inaoka et al. (2004).

8.2.2 Ownership Network

Description and Preprocessing

Orbis is a database composed of about one hundred million entities, developed by a specialized group based in Bureau Van Dijk's Brussels office, aggregating several sources of data. A detailed description is given in Dijk (2018). The database also specifies indirect ownership through a path of auxiliary entities, but this is not strictly capital ownership, and therefore we do not take into account these relationships in our study.

The location of an entity is usually a country (or else a region such as Hawaï (US), La Réunion (Fr.)) that defines where the legal entity has its headquarters. Therefore, a subsidiary and a parent company can have different locations. See Beddi and Mayrhofer (2010) for in-depth insights into the role of location in headquarters and subsidiaries relationships. The sector of a legal entity is a description of its activity.

The database had some inconsistencies and missing values. In some cases entities (and edges) appear multiple times. We merged identical edges into a single edge by taking the maximum weight. About 30% of weights are missing. According to the documentation of *Orbis*, these links are indirect, i.e they represent ownership through other entities and, since we wish to measure direct ownership, we simply removed these edges for our analysis.

Motivation and Justification of the Methods Used in the Analysis

The motivations to analyse the knowledge graph under the prism of centrality measures and entity influence is two-fold.

First, the detection of influential communities of entities has a natural interpretation in economic and financial terms.

Second, the existence of low time complexity algorithms available for this purpose. Indeed, the considered network being very large, any non (quasi-)linear time complexity algorithm would result in computational time issue. Besides, several possibilities exist in order to model the influence of nodes, for instance with the influence interdiction problem (Omer and Mucherino, 2020). However to the best of our knowledge, other formulations do not have linear time algorithm to obtain good approximate solutions efficiently. On the contrary, centrality measures (k-cores) and some influence maximization algorithms, which are relatively straightforward methods, have linear time complexity have proven to be very efficient in the detection of influential communities.

Preliminary definitions

In this section we present supplementary notions and definitions used for the analysis of *Orbis* network.

Aggregation by attributes

The ownership graph also contain entity attributes (location (country or region) and a description of the activity (sector), cf Sec. 8.2.2 for a precise description). Here, we present a method to analyze the graph of entities by attributes. Let $\mathcal{A}$ be the set of values for a given attribute of the entities. For $(a, b) \in \mathcal{A}^2$, let G_a (resp. G_b) be the set of entities having attribute a (resp. b). We define a new graph $G_{\mathcal{A}} = (\mathcal{A}, E_{\mathcal{A}})$ between attribute values in the following way:

$$w_{ab} = \frac{1}{|G_b|} \sum_{j \in G_b} \sum_{i \in G_a \cap \mathcal{N}^-(j)} w_{ij} \tag{8.1}$$

In other words, $G_{\mathcal{A}}$ provides a kind of "meta-graph" based on the pairwise relationship between the values of $\mathcal{A}$. For example, a graph where $\mathcal{A} = \{a, b\}$ defines two countries, will be a graph of two nodes, inheriting the connectivity in the form of an aggregation w_{ab} on up to two directed edges $E_{\mathcal{A}}$. This definition of Equation (8.1) insures the following mathematical conveniences, as also in Equation (8.5):
$\forall (a, b) \in \mathcal{A}^2$

$$0 \leq w_{ab} \tag{8.2}$$

$$w_{ab} \leq \frac{1}{|G_b|} \sum_{j \in G_b} \sum_{i \in G_a \cap \mathcal{N}^-(j)} w_{ij}$$

$$w_{ab} \leq 1 \tag{8.3}$$

and

$$\begin{aligned} \sum_{a \in \mathcal{A}} w_{ab} &= \frac{1}{|G_b|} \sum_{j \in G_b} \sum_{a \in \mathcal{A}} \sum_{i \in G_a \cap \mathcal{N}^-(j)} w_{ij} \\ &= \frac{1}{|G_b|} \sum_{j \in G_b} \sum_{i \in \mathcal{N}^-(j)} w_{ij} \\ &\leq \frac{1}{|G_b|} \sum_{j \in G_b} 1 \end{aligned}$$

$$\sum_{a \in \mathcal{A}} w_{ab} \leq 1 \tag{8.4}$$

Moreover, the choice of Equation (8.1) is a good candidate for the influence of a attribute a over an attribute b (e.g influence of Germany over France), since it corresponds to a quantity representing the percentage or total capital owned by a over b. Therefore, The meta-graph $G_{\mathcal{A}}$ allows us to analyze interactions between attributes (i.e countries or sectors). However the limitation of this analysis lies in the assumption that entities have the same capital: we will come back to this limitation in Section 8.2.3.

Rooted influence graph

For each entity, we define a subgraph, the rooted influence graph (RIG), similarly to the rooted citation graph (RCG) in collaboration analysis Giatsidis et al. (2019). The RIG of a vertex i is the subgraph of G induced by the set of vertices that contain i and all the vertices which can be reached by a directed path. That is, $j \in \mathsf{RIG(i)}$ if and only if there is a directed path from vertex i to vertex j. The resulting directed acyclic graph (DAG) contains all the entites that are directly or implicitly influenced by the entity i.

Based on this definition, it is natural to consider the following quantities in the RIG to measure influence of an entity:

- a) out-degree
- b) average degree of the nodes in the RIG
- c) core influence (i.e core number of the considered entity of the undirected RIG)

Influence maximization

In network and graph theory, influence maximization (IM) is the problem of maximizing influence with regards to seed nodes using a diffusion model. It has been extensively studied recently due to its potential commercial value. An example of application of influence maximization is viral marketing Domingos and Richardson (2001), where an organization wants to spread the adoption of a product from selected adopters. Influence maximization is also the corner stone in other important applications such as network monitoring, rumor control, and social recommendation.

Model and Methodology

In the following, the capital ownership network is represented as a capitalistic graph $G = (V, E)$. A vertex $i \in V$ represents a legal entity, an oriented edge e_{ij} with weight w_{ij} represents capitalistic property: an edge from i to j means that i

owns capital of j in proportion w_{ij}. $\mathcal{N}_i^+$ and $\mathcal{N}_i^-$ are respectively the out-neighbors and in-neighbors of vertex i (i.e., set of nodes connected to and from vertex i). $|I|$ is the cardinal of a finite set I. For mathematical convenience, we impose the following conventions:

$$\begin{aligned} \forall e \in E,\ w_e \in [0,1] \\ \forall i \in V,\ \sum_{j \in \mathcal{N}_i^-} w_{ji} \leq 1 \end{aligned} \tag{8.5}$$

In Sec. 8.2.2, we describe how to ensure Eq. (8.5). Following standard juridic terms, there are two types of entities in the graph:

- Natural person (or physical person): individual human being
- Juridical person: incorporated organizations including corporations, government agencies; or non-governmental organizations. A legal person is composed of natural persons, but has a distinct juridic identity.

We suppose that there exist no inner link between the subgraph of natural persons, eventhough a natural person can have influence over others (we suppose this link is not explicit so it is not considered as an edge in the graph). Therefore, it is possible to model this ownership graph as a graph in Fig 8.9.

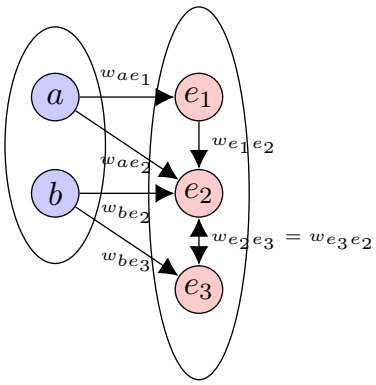


Figure 8.9 – Weighted digraph: natural persons (blue) and organizations (red)

We are interested in studying the influence of legal entities, so the distinction between natural person and organizations raises an important discussion for the evaluation of such influence. How do we use the information from the subgraph of natural persons? A first approximation is to consider that the links between persons and organizations are negligible in terms of global influence. In this case, we are left with the study of the subgraph of the organizations. A second way to operate is to aggregate information from deleted edges between natural persons and

organizations, for example by creating additional links between organizations, or adding node features. In view of the instance at our disposal, and in particular the fact that the data is poor with regards to personal attributes, we decided to consider the first situation, that is to say to keep only the links between organizations.

Unlike market payment graphs which are very dynamic, the capitalistic graph evolves across a relatively long time scale and significant changes do not occur frequently (except rare events, such as the onset of an economic crisis). Therefore, we consider the capitalistic graph constant for our analysis: we will not analyze its dynamic aspect, although this may be considered in future work.

The data is provided by Bureau Van Dijk Dijk (2018), and lists all the physical and legal entities. Each of these entities has corresponding metadata: Name, location (country and continent), and description. They also have a weighted capital property over a list of other entities. The original data format is in a non-relational form (i.e line format): software development was necessary to parse and load data into a graph.For reasons of confidentiality, the data has been anonymized.

Location-sector analysis of the *Orbis* network has been proposed to identify and locate important entities Nakamoto et al. (2019a) and, in a 2015 snapshot, to analyze the location and sector of conduit firms likely to be used for treaty shopping Nakamoto et al. (2019b). Here, we consider a different model and a more recent version of the network (2018). We also consider an entity and its subsidiaries as separate, making the assumption that the influence of an entity is also spread through its subsidiaries.

Degree Distribution and Connected Components

We used the igraph software implementation Csardi and Nepusz (2006) to manage the graph structure emanating from this database. After pre-processing, the graph of organizations is composed of $81,576,517$ nodes and $6,818,574$ edges. Following our discussion in the first section, we consider the subgraph of juridical entities (organizations) for our entity influence analysis. The subgraph of non-isolated organizations is composed of $6,518,718$ nodes and $6,818,574$ edges, with $1,429,853$ components; additional numbers are in Table 8.1. An important metric in graph mining is the *density*, representing the average connectivity (i.e. abundance of edges) of the graph. The density of a directed graph is a real number in $[0,1]$, maximized for cliques and minimized for a graph of isolated nodes. For the graph at hand the density is:

$$D = \frac{|E|}{|V|(|V|-1)} = 1.571 \times 10^{-7}$$

which means that the graph is very sparse. The degree distribution is in Fig. 8.10 and suggests important sparsity. We can see that the degrees vary importantly in

Table 8.1 – Capitalistic ownership graph instance (*Orbis*). (1) All edges, (2) Edges with unknown weights removed

Subgraph	Nodes		Edges	
	(1)	(2)	(1)	(2)
Organizations	$81,576,517$	$81,576,517$	$6,818,574$	$4,242,843$
Non isolated organizations	$6,518,718$	$4,789,294$	$6,818,574$	$4,242,843$
Total (natural persons and organizations)	$105,426,819$	$105,426,819$	$81,111,480$	$49,777,255$

the graph. More specifically there are few entities with up to more that a million capitalistic relations whereas the vast majority have less that one hundred. Also it is interesting to note that generally the outward edges per node supersede the incoming ones (i.e. entities acquire more than they are acquired).

The distributions of entities by sector and country are displayed in Tables 8.2 and 8.3. We see that the distribution is not uniform and is not correlated with the sizes of each country. There are relatively few entities in the United States (US) compared to some European or South American countries. This is because the data source has little knowledge of US entities. One possibility for refining the study in relation to this country would be to supplement *Orbis* with data from other sources (but this is outside the scope of this work).

The subgraph of organizations contains a very large component (i.e. a set of entities where each pair of nodes is connected via a path) of $1,442,704$ nodes and $1,816,874$ edges. The other components are very small, smaller than one thousand nodes. The component-size distribution is in Fig. 8.11, which shows that there are many small components (some hundreds of nodes) and lots of isolated nodes. Two drawings of smaller components (less than a thousand nodes) are depicted in Fig 8.12. We do not include a fine-grained classification of the structure of these small components. However, in the next subsections we provide the analytics on the largest component of juridical entities of $1,442,704$ nodes.

Degeneracy and Centrality Measures

The maximum k-core (defined as the k-core such that C_k is not empty and k is maximal) of the graph allows to find an approximation of the densest part of the graph in linear time Batagelj and Zaversnik (2003).

On the one hand, in the original graph of entities restricted to organizations, the large component of $1,442,704$ million nodes has a degeneracy value equal to 18 composed of 45 entities. C_{18} is a very dense community of entities, where each of them is being owned (resp. owns) the capital of (resp. by) at least 18 other entities in total. Recall that the other components are much smaller (i.e few hundreds of entities) and sparse, so that degeneracy in these components is not very

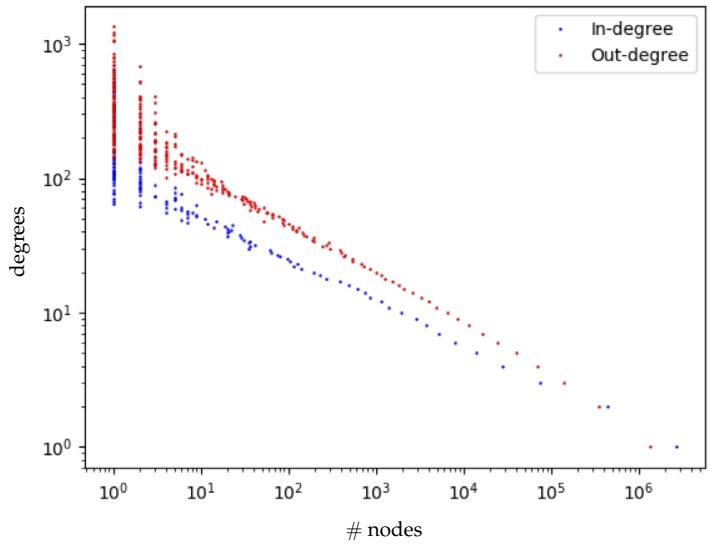


Figure 8.10 – Degree distribution of organizations (*Orbis*)

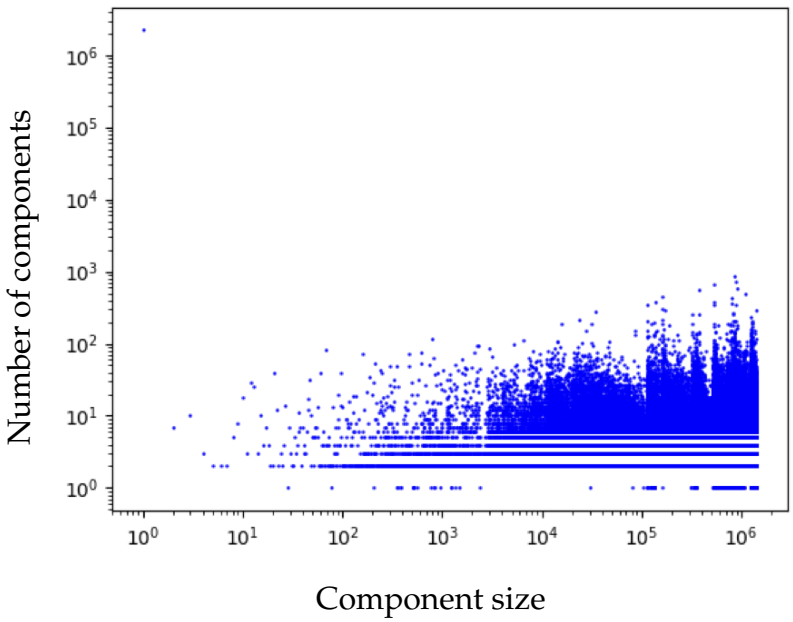


Figure 8.11 – Distribution of the sizes of connected components of juridical organizations (*Orbis*) . It contains one large component ($\sim$ 2 millions nodes), and thousands of small components (from ten and a thousand entities)

informative for data mining.

On the other hand, the distribution of countries and sectors in $\{C_k \mid k \geq 9\}$ (992 entities) are displayed in Tables 8.4 and 8.5. These tables show that countries

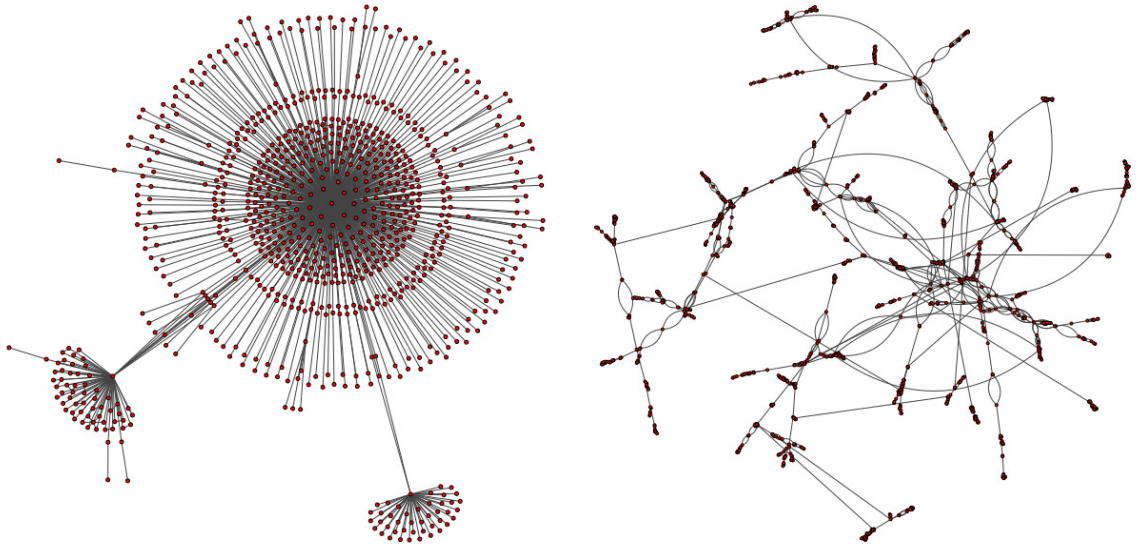
Table 8.2 – Top 20 most frequent sectors in *Orbis*

Location	Number (total)
Unknown	9691495
Restaurants and mobile food service activities	2478398
Construction of residential and non-residential buildings	1584314
Hairdressing and other beauty treatment	1567390
Retail sale of clothing in specialized stores	1515681
Rental and operating of own or leased real estate	1437334
Retail sale in non-specialized stores with food, beverages or tobacco predominating	1351262
Freight transport by road	1207457
Other retail sale of new goods in specialized stores	1190241
Business and other management consultancy activities	1168437
Non-specialized wholesale trade	1088766
Maintenance and repair of motor vehicles	1064392
Other business support service activities n.e.c.	995030
Activities of other membership organizations n.e.c.	932252
Activities of holding companies	930527
Other specialized construction activities n.e.c.	809568
Advertising agencies	793219
Other building completion and finishing	763069
Engineering activities and related technical consultancy	753509
Retail sale of hardware, paints and glass in specialized stores	715492

Table 8.3 – Top 20 most frequent countries in *Orbis*

Country	Number (total)
Brazil	19550646
China	9865149
Italy	4985420
United Kingdom	4030892
France	3740322
Russian Federation	3258544
Germany	2631038
Australia	2554195
Netherlands	2498228
Colombia	1924520
Czech Republic	1803264
Poland	1607002
Sweden	1594381
Japan	1489379
Spain	1163873
India	1139334
Mexico	965197
Bulgaria	898522
Taiwan	863927
Romania	853540

and sectors are not uniform but shared between a limited amount of countries (Singapore, Australia, Germany, Ukraine, and France), and sectors (Activities of holding companies, Buying and selling of own real estate, Other monetary inter-



(a) Connected component 1 (b) Connected component 2

Figure 8.12 – 2 examples of small components (*Orbis*)

mediation, Rental and operating of own or leased real estate, Activities of head offices). These countries and sectors have a more intense interaction for capitalistic property.

Table 8.4 – 20 most frequent countries in the 10 top-k cores (992 entities from *Orbis*)

Countries	Number
Singapore	440
Australia	331
Germany	256
Ukraine	125
France	69
Malaysia	52
New Zealand	52
Bermuda(GB)	33
India	32
Chile	26
Italy	19
Thailand	15
Hong Kong	10
United Kingdom	6
Cayman Islands(GB)	5
China	2
Ireland	2
Netherlands	2
Israel	1
Japan	1
Luxembourg	1
British virgin islands	1

Table 8.5 – Most frequent sectors in top k-cores (992 entities from *Orbis*)

Sector	Number
Activities of holding companies	314
Buying and selling of own real estate	84
Other monetary intermediation	64
Rental and operating of own or leased real estate	53
Activities of head offices	46
Other activities auxiliary to financial services, except insurance and pension funding	42
Trusts, funds and similar financial entities	37
Unknown	36
Activities of other membership organisations n.e.c.	36
Publishing of directories and mailing lists	35
Other financial service activities, except insurance and pension funding n.e.c.	32
Sea and coastal freight water transport	25
Real estate agencies	16
Business and other management consultancy activities	16
Precious metals production	15
Other business support service activities n.e.c.	14
Other credit granting	13
Mining of coal and lignite	12
Publishing of journals and periodicals	11
Mining of other non-ferrous metal ores	11
Security and commodity contracts brokerage	11
Fund management activities	10
Other professional, scientific and technical activities n.e.c.	10
Retail sale in non-specialised stores with food, beverages or tobacco predominating	10

Aggregation by Attribute

In order to provide an example of location analysis of such network, we consider the case of France. We computed the aggregated sites whose capital is most possessed by French entities (top 20 countries ranked by edge weight). Conversely, we computed the top 20 aggregate locations that own capital of French entities. We reported results in Fig. 8.13. The entities most held by the French entities are those located in their former colonies (i.e. Algeria, Togo, Congo, Côte d'Ivoire, Benin, Chad, Gabon, Senegal, Cameroon, Comoros, Madagascar, ...). Conversely, the French entities are mostly owned by those located in economically strong countries, such as Germany, China, the United States, Japan, the United Kingdom, the United Arab Emirates, as well as countries geographically close or sharing important historial lies (Belgium, Italy, Morocco, ...). We also reported top-5 neighbors sorted by weight in descending order (tables 8.13d and 8.13c). Top in weights are much lower that the top out weights, this means that French entities have a tendency to own capital of entities in other countries rather to have capital owned at

the international level.

Fig. 8.14 depicts the densest k-core ($k = 44$) on the meta-graph of countries, and represents the most connected subgraph of countries of capitalistic ownership. We observe an important proportion of countries in Europe (France, Germany, United Kingdom, Italy, Spain, Portugal, ...). Other economically strong countries such as United States, China, United Arab Emirates and Republic of Korea are also represented. These two categories have a dense collaboration in terms of capital ownership. Two important aspects about Fig. 8.14:

- Some "tax havens" are present (Cayman Islands, Malta, Cyprus, ...) despite their relative low number of entities.
- The United States of America are in the densest k-core of the meta-graph of countries, eventhough they are under-represented in the dataset (less than 850000 entities, cf. Table 8.4).

Fig. 8.15 depicts the densest k-core ($k = 218$) on the meta-graph of sectors (limited to 15 sectors with highest degree for visibility). It reveals a dense collaboration between several economic sectors (Financial activities, real estate, engineering and technical consultancy, ...).

Influence Analysis

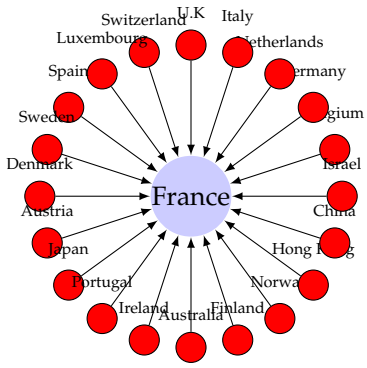
Recall that, in the context of this analysis, influence is *the capacity to have an effect on the development or decisions of an entity*. In this subsection we present two different types of methods to measure it. Then, we compare the results obtained using the same diffusion model.

Coreness and Rooted Influence Graph

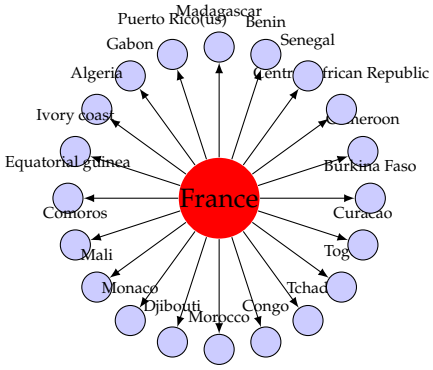
We used two coreness-based methods to measure influence indirectly. First, sorting nodes based on their coreness number in the graph. Second, we compute the coreness of each node in its RIG, and sort them by decreasing value. Then, we keep the top-k nodes and consider the distribution of attributes within these. The results with $k = 10000$ are in Table 8.16 (left side) for location analysis and in Table 8.6 for sector analysis.

Influence Maximization:

The idea is to use the influence maximization (IM) paradigm in order to measure entity influence in the capitalistic graph. Here, we ran the simulations using influence maximization with martingales, with $\epsilon = 0.1$ and a seed set of $10\,000$ nodes. The output is a seed set of 10000 nodes which is an approximation of the IM solution whose quality will be discussed in Sec. 8.2.3. We display the distributions



(a) France as target



(b) France as source

Country to France	Weight in %
Belgium	0.88
Germany	0.77
Netherlands	0.51
Italy	0.50
United Kingdom	0.50

(c) Top 5 corresponding meta-graph weights (France as target)

France to Country	Weight in %
Cameroon	39.07
Central African republic	33.33
Senegal	33.12
Benin	26.09
Madagascar	25.21

(d) Top 5 Corresponding meta-graph weights (France as source)

Figure 8.13 – France as source and target of capital ownership (*Orbis*)

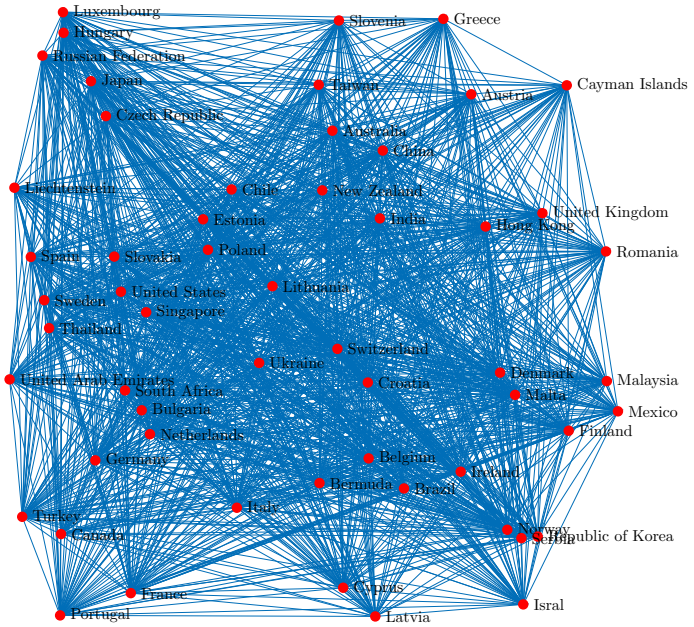


Figure 8.14 – Densest k-core on the meta-graph of countries ($k = 44$, *Orbis* database)

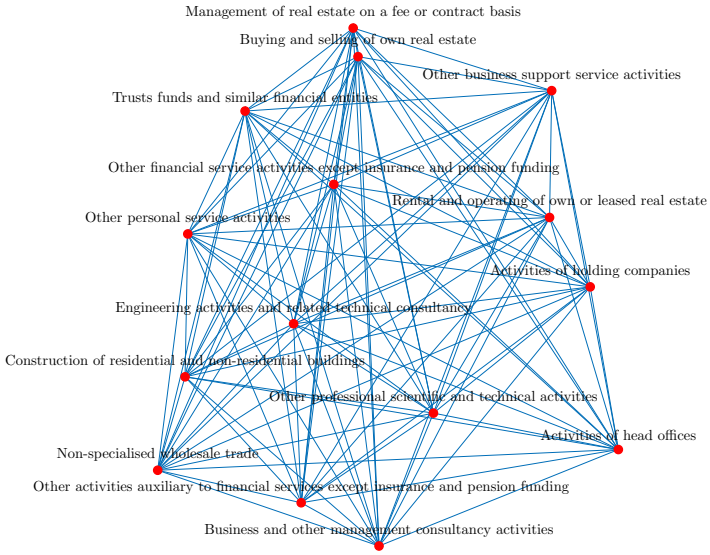


Figure 8.15 – Densest k-core on the meta-graph of sectors ($k = 218$, limited to 15 highest degree sectors for visibility, *Orbis* database)

within this seed set in the right part of Table 8.16 for location analysis, and Table 8.7 for sector analysis.

8.2.3 Discussion

The network of capital ownership data of organizations revealed a large component, that we analyzed according to the different axes listed in the previous section. This allows us to define the communities of countries or sectors that are influencing the economy of capital ownership. In particular we looked at France as a sanity check, and to provide insight of the functionality of *Orbis* instance Dijk (2018). We are measuring implicit entity influence thereof. We highlight the possible limitation of the approximation of different sized entities as having the same capital, since we aggregated the weights without taking in account the capital value. That is to say, the size of capital of entities is not explicitly modeled.

The different influence measurements (RIG and IMM) gave significant differences in the distributions of attributes. In order to estimate the quality of their respective solutions, we compared the number of infected nodes using our diffusion model (independent cascades), with entities obtained in the top k-cores of the graph. We define the ratio τ as the ratio of the seed set size over the number of vertices of the graph. For practical software reasons (NDlib library Rossetti et al.

Figure 8.16 – Top 20 influential countries using coreness on the RIG ((a), left) and influence maximization on the IM ((b), right) on the total network of entities (*Orbis*)

(a) RIG

Country	Number
Italy	1085
Germany	995
Ukraine	880
France	866
India	600
Japan	555
Australia	425
Spain	396
United Kingdom	334
Russian Federation	328
Norway	298
Portugal	277
Austria	258
China	190
Taiwan	190
Belgium	187
Netherlands	178
Thailand	163
Singapore	151
Malaysia	124

(b) IM

Country	Number
Germany	1322
China	1161
Australia	1010
France	642
Italy	628
United Kingdom	530
Austria	381
Norway	337
Spain	328
Ukraine	296
Japan	255
Netherlands	220
Belgium	206
Russian Federation	169
Sweden	163
Singapore	137
New Zealand	115
Portugal	115
Poland	108
India	98

(2018)), we considered larger seed sets ($\tau \in \{5\%, 10\%, 15\%, 20\%\}$ of the considered graph where $\tau = 5\%$ corresponds to 75136 nodes). The results are shown in Table 8.8. For all seed set sizes, we have obtained better results using RIG coreness than with IMM. Influence based on coreness on the initial graph yields better solutions for $\tau \in \{5\%, 10\%, 15\%\}$ seed set sizes, and IMM performs slightly better than top-k coreness for $\tau = 20\%$. This is an interesting discovery. In spite of good theoretical guarantees, the IMM algorithm of influence maximization suffers on some graph instances and seed set sizes found in practice (such as this one). This is also in accordance with experiments of Giatsidis et al. (2019) suggesting that coreness can yield excellent influence measurement in our context of interest of this work. We suggest that for large and relatively sparse network, influence measurement with coreness should be considered.

8.2.4 Conclusion

Our analysis based on centrality measures and influence maximization is consistent with economic and historical facts such as the existence of tax havens, and private capital ownership over its former colonies (we specifically studied the case of France). This therefore allowed us to reveal important locations and sectors communities in the economy, and potentially target most influential entities. Initial seeds using k-cores yield a better influence score for several seed set sizes ranging

Table 8.6 – Top 20 influential sectors using coreness in the rooted influence graph (*Orbis* database)

Sector	Number of entities
Activities of holding companies	1079
Other monetary intermediation	659
Activities of head offices	636
Rental and operating of own or leased real estate	344
Other financial service activities, except insurance and pension funding n.e.c.	312
Other activities auxiliary to financial services, except insurance and pension funding	249
Business and other management consultancy activities	244
Unknown	243
Fund management activities	193
Other business support service activities n.e.c.	185
Trusts, funds and similar financial entities	175
Non-specialized wholesale trade	168
Construction of residential and non-residential buildings	137
Development of building projects	132
Buying and selling of own real estate	114
Retail sale in non-specialized stores with food, beverages or tobacco predominating	107
Security and commodity contracts brokerage	100
Production of electricity	97
Administration of financial markets	85
Insurance, reinsurance and pension funding, except compulsory social security	84
Life insurance	81
Non-life insurance	77

from 5% to 20% using the independent cascade diffusion model. Results suggest that several means should be considered (depending on the size of the seed set) to measure influence in large networks with similar degree distribution.

We outlined results that should be treated with caution, due to the sparsity of data, particularly respective of some countries, for example – in this dataset – the United States. Nevertheless, the overall results provide important insight and information on the influence of the entities and are for the most part consistent with the current global economic situation. Of course, further enrichment of the capital ownership dataset with additional entities and capital information will likely provide an even more precise quantitative measure of influence. We could also consider studying the evolution of this graph over time (monthly or yearly), but this is out of scope of the current study and we leave that line of research for future work.

Table 8.7 – Top 20 influential sectors using influence maximization (*Orbis* database)

Sector	Number of entities
Unknown	1181
Activities of holding companies	520
Rental and operating of own or leased real estate	354
Other business support service activities n.e.c.	298
Activities of head offices	291
Buying and selling of own real estate	214
Business and other management consultancy activities	212
Other activities auxiliary to financial services, except insurance and pension funding	180
Development of building projects	155
Production of electricity	153
Construction of residential and non-residential buildings	133
Real estate agencies	132
Trusts, funds and similar financial entities	130
Engineering activities and related technical consultancy	119
Non-specialised wholesale trade	107
Management of real estate on a fee or contract basis	102
Computer programming activities	100
Other financial service activities, except insurance and pension funding n.e.c.	95
Other professional, scientific and technical activities n.e.c.	89
Other transportation support activities	85

Table 8.8 – Comparison with the IC model for several seed set sizes (50 simulations)

Method	$\boldsymbol{\sigma(S) \pm\ std}$			
	$\boldsymbol{\tau = 5\%}$	$\boldsymbol{\tau = 10\%}$	$\boldsymbol{\tau = 15\%}$	$\boldsymbol{\tau = 20\%}$
Coreness	285250.8 ± 1304.8	351887.4 ± 928.8	418440.6 ± 755.8	474402.2 ± 454.3
IMM	161485.4 ± 1211.2	298874.9 ± 774.3	419304.6 ± 1055.5	521251.7 ± 1033.4
RIG coreness	610138.7 ± 549.8	701871.5 ± 706.7	799244.1 ± 661.1	903330.4 ± 369.1

8.3 Chapter 6: Other distances?

As mentioned in Chapter 6, there are several possibilities for defining distances between random variables, we discuss some of them in the next two new subsections. Unfortunately, these possibilities are inadequate because of excessive computational cost; moreover, preliminary experiments showed that they were far from inducing valid Euclidean distance on $\mathbb{R}^K$ for $K \in \{50, 100, 150\}$.

8.3.1 Distances on Random Variables

Given two discrete random variables X and Y, with the same support $\mathcal{X}$, on a probability space (Ω, A, P), with joint probability distribution $\rho(x, y)$, we have that

$$\mathbb{E}[|X-Y|^p]^{1/p} = (\sum_{(x,y)\in\mathcal{X}\times\mathcal{X}} |x-y|^p \rho(x,y))^{1/p} \tag{8.6}$$

defines a distance $d : (X,Y) \mapsto d(X,Y)$ on the set of corresponding random variables (Deza and Deza, 2009).

Law of large numbers: If $(x_1,y_1),...,(x_n,y_n)$ are i.i.d observations for the random variable (x,y), then the $\hat{e}(X,Y)$ is an estimator of the distance $\mathbb{E}[|X-Y|^p]$:

$$\hat{e}(X,Y) = \frac{1}{n}\sum_{i=1}^{n} |x_i - y_i|^p \tag{8.7}$$

By continuity $\hat{d}(X,Y)^p = \hat{e}(X,Y)^{1/p}$ is an estimator of $\mathbb{E}[|X-Y|^p]^{1/p}$.

Let X (resp. Y) be the binary random variable equal to 1 if the word x (resp.y) appears in a window of size w in a corpus and $\mathcal{X} = \{0,1\}$. Then,

$$\begin{aligned}
\mathbb{E}[|X-Y|^p] &= \sum_{(x,y)\in\mathcal{X}\times\mathcal{X}} |x-y|^p \rho(x,y) && (8.8)\\
&= |0-0|^p\rho(0,0) + |1-0|^p\rho(1,0) + |0-1|^p\rho(0,1) + |1-1|^p\rho(1,1)\\
&= \rho(1,0) + \rho(0,1) && (8.9)
\end{aligned}$$

Then $\hat{d}(X,Y)^p$ is an estimator of distance between random variables associated to word x and y.

For $p=2$, $d(X,Y)$ has a scalar product, which is

$$q(X,Y) = \langle X,Y\rangle = \sum_{(x,y)\in\mathcal{X}\times\mathcal{X}} xy\,\rho(x,y) \tag{8.10}$$

An estimator of $q(X,Y)$ is given by

$$\hat{q}(X,Y) = \frac{1}{n}\sum_{i=1}^{n} x_i y_i \tag{8.11}$$

Which in our case corresponds to the number of co-occurrences between a word x and y in a window of size w in a corpus, divided by n which is the total number of windows considered.

8.3.2 A distance candidate: variation of information

Let $H(X)$ be the entropy of random variable X:

$$H(X) = -\sum_{x\in\mathcal{X}} \rho(x)\log\rho(x) \tag{8.12}$$

$I(X, Y)$ be the mutual information between random variables X and Y defined as:

$$I(X, Y) = \sum_{(x,y)\in\mathcal{X}\times\mathcal{X}} \rho(x, y) \log \frac{\rho(x, y)}{\rho(x)\rho(y)} \tag{8.13}$$

It is worth noting that

$$I(X, Y) = D_{KL}(\rho(x, y)||\rho(x)\rho(y)),$$

where $D_{KL}(.||.)$ denotes the Kullback-Leibler divergence.

And the variation of information $V(X, Y)$ such that:

$$V(X, Y) = H(X) + H(Y) - 2I(X, Y) \tag{8.14}$$

Then, $V : (X, Y) \mapsto V(X, Y)$ defines a metric on the set of discrete random variables.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al. (2015). Tensorflow: Large-scale machine learning on heterogeneous systems, 2015. *Software available from tensorflow. org*, 1(2).

Al-Homidan, S. and Wolkowicz, H. (2005). Approximate and exact completion problems for Euclidean distance matrices using semidefinite programming. *Linear Algebra and its Applications*, 406:109–141.

Alhelbawy, A. and Gaizauskas, R. (2013). Named entity disambiguation using hmms. In *Proceedings of the 2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 03*, pages 159–162. IEEE Computer Society.

Alhelbawy, A. and Gaizauskas, R. J. (2014). Graph ranking for collective named entity disambiguation. In *ACL (2)*, pages 75–80.

Alsentzer, E., Murphy, J., Boag, W., Weng, W.-H., Jindi, D., Naumann, T., and McDermott, M. (2019). Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Anderson, B. D., Shames, I., Mao, G., and Fidan, B. (2010). Formal theory of noisy sensor network localization. *SIAM Journal on Discrete Mathematics*, 24(2):684–698.

Apostolico, A. and Guerra, C. (1987). The longest common subsequence problem revisited. *Algorithmica*, 2(1):315–336.

Arora, S. and Barak, B. (2009). *Computational complexity: a modern approach*. Cambridge University Press.

Arora, S., Li, Y., Liang, Y., Ma, T., and Risteski, A. (2016a). A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399.

Arora, S., Li, Y., Liang, Y., Ma, T., and Risteski, A. (2018). Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association of Computational Linguistics*, 6:483–495.

Arora, S., Liang, Y., and Ma, T. (2016b). A simple but tough-to-beat baseline for sentence embeddings.

Arora, S., Liang, Y., and Ma, T. (2017). 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*.

Ballard, D. H. (1987). Modular learning in neural networks. In *AAAI*, pages 279–284.

Bartunov, S., Kondrashkin, D., Osokin, A., and Vetrov, D. (2016). Breaking sticks and ambiguities with adaptive skip-gram. In *artificial intelligence and statistics*, pages 130–138.

Batagelj, V. and Zaversnik, M. (2003). An o (m) algorithm for cores decomposition of networks. *arXiv preprint cs/0310049*.

Beddi, H. and Mayrhofer, U. (2010). The role of location in headquarters-subsidiaries relationships: An analysis of french multinationals in emerging markets. In *36th Annual EIBA (European International Business Academy) Conference*, pages 26–p.

Beineke, L. W. (1970). Characterizations of derived graphs. *Journal of Combinatorial theory*, 9(2):129–135.

Bekas, C., Kokiopoulou, E., and Saad, Y. (2008). Computation of large invariant subspaces using polynomial filtered lanczos iterations with applications in density functional theory. *SIAM Journal on Matrix Analysis and Applications*, 30(1):397–418.

Bengio, Y., Courville, A., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828.

Berstel, J. and Perrin, D. (2007). The origins of combinatorics on words. *European Journal of Combinatorics*, 28(3):996–1022.

Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U. (1999). When is "nearest neighbor" meaningful? In *International conference on database theory*, pages 217–235. Springer.

Billinge, S. J., Duxbury, P. M., Gonçalves, D. S., Lavor, C., and Mucherino, A. (2018). Recent results on assigned and unassigned distance geometry with applications to protein molecules and nanostructures. *Annals of Operations Research*, 271(1):161–203.

Blitzer, J., Dredze, M., and Pereira, F. (2007). Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447.

Blumenthal, L. M. (1970). Theory and applications of distance geometry.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Bolukbasi, T., Chang, K.-W., Zou, J. Y., Saligrama, V., and Kalai, A. T. (2016). Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*, pages 4349–4357.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Boss, M., Elsinger, H., Summer, M., and Thurner 4, S. (2004). Network topology of the interbank market. *Quantitative finance*, 4(6):677–684.

Bottou, L., Curtis, F. E., and Nocedal, J. (2018). Optimization methods for large-scale machine learning. *SIAM Review*, 60:223–311.

Brightwell, G. R. and Winkler, P. (2005). Counting eulerian circuits is# p-complete. In *ALENEX/ANALCO*, pages 259–262. Citeseer.

Caliskan, A., Bryson, J. J., and Narayanan, A. (2017). Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Cardona, G. (1997). *Pāṇini: A survey of research*. Motilal Banarsidass Publ.

Carpentier, P. and Cohen, G. (2017). Vue d'ensemble de la méthode du gradient stochastique. In *Décomposition-coordination en optimisation déterministe et stochastique*, pages 167–193. Springer.

Cassioli, A., Günlük, O., Lavor, C., and Liberti, L. (2015). Discretization vertex orders in distance geometry. *Discrete Applied Mathematics*, 197:27 – 41. Distance Geometry and Applications.

Cayley, A. (1841). A theorem in the geometry of position. *Cambridge Mathematical Journal*, 2:267–271.

Chaiken, S. (1982). A combinatorial proof of the all minors matrix tree theorem. *SIAM Journal on Algebraic Discrete Methods*, 3(3):319–329.

Chen, Q., Zhu, X., Ling, Z., Wei, S., Jiang, H., and Inkpen, D. (2016). Enhanced lstm for natural language inference. *arXiv preprint arXiv:1609.06038*.

Chilton, P. (2014). *Language, space and mind: The conceptual geometry of linguistic meaning*. Cambridge University Press.

Chollet, F. et al. (2015). Keras. `https://keras.io`.

Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124.

Church, K. W. and Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

Cordella, L. P., Foggia, P., Sansone, C., and Vento, M. (2004). A (sub) graph isomorphism algorithm for matching large graphs. *IEEE transactions on pattern analysis and machine intelligence*, 26(10):1367–1372.

Cox, M. A. and Cox, T. F. (2008). Multidimensional scaling. In *Handbook of data visualization*, pages 315–347. Springer.

Csardi, G. and Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal*, Complex Systems:1695.

Cucerzan, S. (2007). Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of the 2007 EMNLP-CoNLL*, pages 708–716.

Currie, J. and Rampersad, N. (2011). A proof of Dejean's conjecture. *Mathematics of Computation*, 80(274):1063–1070.

Dejean, F. (1972). Sur un théoreme de Thue. *Journal of Combinatorial Theory, Series A*, 13(1):90–99.

del Prado, F. M. (2011). The universal" shape" of human languages: spectral analysis beyond speech. *Nature Precedings*, pages 1–1.

Demmel, J., Dumitriu, I., and Holtz, O. (2007). Fast linear algebra is stable. *Numerische Mathematik*, 108:59–91.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Deza, M. M. and Deza, E. (2009). Encyclopedia of distances. In *Encyclopedia of distances*, pages 1–583. Springer.

Dijk, B. V. (2018). Source: Orbis, bureau van dijk.

Dokmanic, I., Parhizkar, R., Ranieri, J., and Vetterli, M. (2015). Euclidean distance matrices: Essential theory, algorithms, and applications. *Signal Processing Magazine, IEEE*, 32(6):12–30.

Domingos, P. and Richardson, M. (2001). Mining the network value of customers. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 57–66. ACM.

Dong, Q. and Wu, Z. (2002a). A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. *Journal of Global Optimization*, 22(1-4):365–375.

Dong, Q. and Wu, Z. (2002b). A linear-time algorithm for solving the molecular distance geometry problem with exact inter-atomic distances. *Journal of Global Optimization*, 22:365–375.

Dong, Q. and Wu, Z. (2003). A geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. *Journal of Global Optimization*, 26(3):321–333.

Ehrmann, M., Della Rocca, L., Steinberger, R., and Tanev, H. (2013). Acronym recognition and processing in 22 languages. *arXiv preprint arXiv:1309.6185*.

Eilenberg, S. (1974). *Automata, languages, and machines*. Academic press.

Everaert, M. B., Huybregts, M. A., Chomsky, N., Berwick, R. C., and Bolhuis, J. J. (2015). Structures, not strings: linguistics as part of the cognitive sciences. *Trends in cognitive sciences*, 19(12):729–743.

Ganea, O.-E., Ganea, M., Lucchi, A., Eickhoff, C., and Hofmann, T. (2016). Probabilistic bag-of-hyperlinks model for entity linking. In *Proceedings of the 25th International Conference on World Wide Web*, pages 927–938. International World Wide Web Conferences Steering Committee.

Ganea, O.-E. and Hofmann, T. (2017). Deep joint entity disambiguation with local neural attention. *arXiv preprint arXiv:1704.04920*.

Gärdenfors, P. (2014). *The geometry of meaning: Semantics based on conceptual spaces*. MIT Press.

Giatsidis, C., Nikolentzos, G., Zhang, C., Tang, J., and Vazirgiannis, M. (2019). Rooted citation graphs density metrics for research papers influence evaluation. *Journal of Informetrics*, 13(2):757–768.

Gibert, J., Valveny, E., and Bunke, H. (2011). Dimensionality reduction for graph of words embedding. In *International Workshop on Graph-Based Representations in Pattern Recognition*, pages 22–31. Springer.

Globerson, A., Chechik, G., Pereira, F., and Tishby, N. (2007). Euclidean embedding of co-occurrence data. *Journal of Machine Learning Research*, 8:2265–2295.

Globerson, A., Lazic, N., Chakrabarti, S., Subramanya, A., Ringaard, M., and Pereira, F. (2016). Collective entity resolution with multi-focal attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 621–631.

Goldberg, Y. and Levy, O. (2014). word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722*.

Golub, G. H. and Van Loan, C. F. (1996). *Matrix Computations (3rd Ed.)*. Johns Hopkins University Press, USA.

Gonçalves, D. S. (2017). A least-squares approach for discretizable distance geometry problems with inexact distances. *Optimization Letters*, pages 1–15.

Gower, J. C. (1982). Euclidean distance geometry. *Math. Sci.*, 7(1):1–14.

Gramacho, W., Mucherino, A., Lin, J.-H., and Lavor, C. (2016). A new approach to the discretization of multidimensional scaling. In *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pages 591–599. IEEE.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2016). Lstm: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232.

Gross, G. (2010). Sur la notion de contexte. *Meta: journal des traducteurs/Meta: Translators' Journal*, 55(1):187–197.

Gross, M. (1981). Les bases empiriques de la notion de prédicat sémantique. *Langages*, (63):7–52.

Gross, M. (1982). Une classification des phrases figées du français. *Revue québécoise de linguistique*, 11(2):151–185.

Guo, Y., Che, W., Liu, T., and Li, S. (2011). A graph-based method for entity linking. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1010–1018.

Gusfield, D. (1997). *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge University Press.

Han, X., Sun, L., and Zhao, J. (2011). Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

Hashimoto, T. B., Alvarez-Melis, D., and Jaakkola, T. S. (2016). Word embeddings as metric recovery in semantic spaces. *Transactions of the Association for Computational Linguistics*, 4:273–286.

Hatcher, E. and Gospodnetic, O. (2004). *Lucene in action*. Manning Publications.

Hetherington, L. (2004). The mit finite-state transducer toolkit for speech and language processing. In *Eighth International Conference on Spoken Language Processing*.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., and Weikum, G. (2011). Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

Hopcroft, J. and Tarjan, R. (1973). Algorithm 447: efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378.

Hopcroft, J. E., Motwani, R., and Ullman, J. D. (2001). Introduction to automata theory, languages, and computation. *Acm Sigact News*, 32(1):60–65.

Inaoka, H., Ninomiya, T., Taniguchi, K., Shimizu, T., Takayasu, H., et al. (2004). Fractal network derived from banking transaction–an analysis of network structures formed by financial institutions. *Bank Jpn Work Pap*, 4.

Jastrzebski, S., Leśniak, D., and Czarnecki, W. M. (2017). How to evaluate word embeddings? on importance of data efficiency and simple supervised tasks. *arXiv preprint arXiv:1702.02170*.

Ji, H. and Grishman, R. (2011). Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 1148–1158.

Ji, H., Nothman, J., Hachey, B., et al. (2014). Overview of tac-kbp2014 entity discovery and linking tasks. In *Proc. Text Analysis Conference (TAC2014)*, pages 1333–1339.

Jolliffe, I. (2011). *Principal Component Analysis*, pages 1094–1096. Springer Berlin Heidelberg, Berlin, Heidelberg.

Karush, W. (2014). Minima of functions of several variables with inequalities as side conditions. In *Traces and Emergence of Nonlinear Programming*, pages 217–245. Springer.

Khalife, S. (2020). Sequence graphs: characterization and counting of admissible elements (Accepted in CTW Graphs and Combinatorial Optimization).

Khalife, S., Gonçalves, D., and Liberti, L. (2020a). Distance geometry for word embeddings. HAL open archives (hal-02892020, v1).

Khalife, S., Liberti, L., and Vazirgiannis, M. (2019a). Geometry and analogies: a study and propagation method for word representations. In *International Conference on Statistical Language and Speech Processing*, pages 100–111. Springer.

Khalife, S. and Ponty, Y. (2020). Sequence graphs realizations and ambiguity in language models. HAL open archives (hal-02495333, v2).

Khalife, S., Read, J., and Vazirgiannis, M. (2019b). Empirical analysis of a global capital-ownership network. In *International Conference on Complex Networks and Their Applications*, pages 656–667. Springer.

Khalife, S., Read, J., and Vazirgiannis, M. (2020b). Structure and influence analysis of worldwide capitalistic ownership (Submitted to Journal of Applied Network Science).

Khalife, S. and Vazirgiannis, M. (2019). Scalable graph-based method for individual named entity identification. In *Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13)*, pages 17–25.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

Kornai, A. (1985). Natural languages and the Chomsky hierarchy. In *Proceedings of the second conference on European chapter of the Association for Computational Linguistics*, pages 1–7. Association for Computational Linguistics.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Lavor, C., Lee, J., Lee-St. John, A., Liberti, L., Mucherino, A., and Sviridenko, M. (2012a). Discretization orders for distance geometry problems. *Optimization Letters*, 6(4):783–796.

Lavor, C., Liberti, L., Maculan, N., and Mucherino, A. (2012b). The discretizable molecular distance geometry problem. *Computational Optimization and Applications*, 52(1):115–146.

Lavor, C., Liberti, L., and Mucherino, A. (2013). The interval branch-and-prune algorithm for the discretizable molecular distance geometry problem with inexact distances. *Journal of Global Optimization*, 56(3):855–871.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., Van Kleef, P., Auer, S., et al. (2015). Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Lehoucq, R., Sorensen, D., and Yang, C. (1998). *ARPACK Users' Guide*. SIAM, Philadelphia, PA.

Lenzu, S. and Tedeschi, G. (2012). Systemic risk on different interbank network topologies. *Physica A: Statistical Mechanics and its Applications*, 391(18):4331–4341.

Leung, M. K., Xiong, H. Y., Lee, L. J., and Frey, B. J. (2014). Deep learning of the tissue-regulated splicing code. *Bioinformatics*, 30(12):i121–i129.

Levy, O. and Goldberg, Y. (2014). Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185.

Liberti, L., Lavor, C., Maculan, N., and Mucherino, A. (2014). Euclidean distance geometry and applications. *Siam Review*, 56(1):3–69.

Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. (2015). Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, volume 15, pages 2181–2187.

Liu, Q., Chao, J., Mahoney, T., Chern, A., Min, C., Javed, F., and Jijkoun, V. (2018). Lessons learned from developing and deploying a large-scale employer name normalization system for online recruitment. In *Proceedings of the 24th ACM SIGKDD*, pages 556–565. ACM.

Mahé, P. and Vert, J.-P. (2009). Graph kernels based on tree patterns for molecules. *Machine learning*, 75(1):3–35.

Mallat, S. (2016). Understanding deep convolutional networks. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150203.

Malliaros, F. D. and Skianis, K. (2015). Graph-based term weighting for text categorization. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1473–1479.

Melamud, O., McClosky, D., Patwardhan, S., and Bansal, M. (2016). The role of context types and dimensionality in learning word embeddings. In *Proceedings of NAACL-HLT*. ACL.

Menger, K. (1931). New foundation of Euclidean geometry. *American Journal of Mathematics*, 53(4):721–745.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mikolov, T., Yih, W.-T., and Zweig, G. (2013c). 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.

Minty, G. J. (1980). On maximal independent sets of vertices in claw-free graphs. *Journal of Combinatorial Theory, Series B*, 28(3):284–304.

Nadeau, D. and Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Nakamoto, T., Chakraborty, A., and Ikeda, Y. (2019a). Identification of key companies for international profit shifting in the global ownership network. *Applied Network Science*, 4(1):58.

Nakamoto, T., Rouhban, O., and Ikeda, Y. (2019b). Location-sector analysis of international profit shifting on a multilayer ownership-tax network. *Evolutionary and Institutional Economics Review*, pages 1–23.

Nikolentzos, G., Meladianos, P., Rousseau, F., Stavrakas, Y., and Vazirgiannis, M. (2017a). Shortest-path graph kernels for document similarity. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1890–1900.

Nikolentzos, G., Meladianos, P., and Vazirgiannis, M. (2017b). Matching Node Embeddings for Graph Similarity. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2429–2435.

Nissim, M., van Noord, R., and van der Goot, R. (2019). Fair is better than sensational: Man is to doctor as woman is to doctor. *arXiv preprint arXiv:1905.09866*.

Niwa, Y. and Nitta, Y. (1994). Co-occurrence vectors from corpora vs. distance vectors from dictionaries. In *Proceedings of the 15th conference on Computational linguistics-Volume 1*, pages 304–309. Association for Computational Linguistics.

Omer, J. and Mucherino, A. (2020). Referenced Vertex Ordering Problem: Theory, Applications and Solution Methods. HAL open archives.

Pallanca, O., Khalife, S., and Read, J. (2018). Detection of sleep spindles in NREM 2 sleep stages: Preliminary study & benchmarking of algorithms. In Zheng, H. J., Callejas, Z., Griol, D., Wang, H., Hu, X., Schmidt, H. H. H. W., Baumbach, J., Dickerson, J., and Zhang, L., editors, *IEEE International Conference on Bioinformatics and Biomedicine, BIBM 2018, Madrid, Spain, December 3-6, 2018*, pages 2652–2655. IEEE Computer Society.

Pang, B. and Lee, L. (2004). A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *arXiv preprint cs/0409058*.

Papadimitriou, C. H. (2003). *Computational complexity*. John Wiley and Sons Ltd.

Peng, H., Li, J., He, Y., Liu, Y., Bao, M., Wang, L., Song, Y., and Yang, Q. (2018). Large-scale hierarchical text classification with recursively regularized deep

graph-cnn. In *Proceedings of the 2018 World Wide Web Conference*, pages 1063–1072.

Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Raiman, J. and Raiman, O. (2018). Deeptype: Multilingual entity linking by neural type system evolution. *arXiv preprint arXiv:1802.01021*.

Ramos, J. et al. (2003). Using tf-idf to determine word relevance in document queries. In *Proceedings of the first instructional conference on machine learning*, volume 242, pages 133–142. Piscataway, NJ.

Rao, M. (2011). Last cases of dejean's conjecture. *Theoretical Computer Science*, 412(27):3010–3018.

Ratinov, L., Roth, D., Downey, D., and Anderson, M. (2011). Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384. Association for Computational Linguistics.

Robertson, S., Zaragoza, H., et al. (2009). The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Rossetti, G., Milli, L., Rinzivillo, S., Sîrbu, A., Pedreschi, D., and Giannotti, F. (2018). Ndlib: a python library to model and analyze diffusion processes over complex networks. *International Journal of Data Science and Analytics*, 5(1):61–79.

Roth, M. and Woodsend, K. (2014). Composition of word representations improves semantic role labelling. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 407–413.

Rousseau, F., Kiagias, E., and Vazirgiannis, M. (2015). Text categorization as a graph classification problem. 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 1702–1712.

Saad, Y. (1980). On the rates of convergence of the Lanczos and the Block-Lanczos methods. *SIAM Journal on Numerical Analysis*, 17(5):687–706.

Salkoff, M. (1980). A context-free grammar of french. In *COLING 1980 Volume 1: The 8th International Conference on Computational Linguistics*.

Salton, G. (1971a). *The SMART Retrieval System—Experiments in Automatic Document Processing*. Prentice-Hall, Inc., USA.

Salton, G. (1971b). The smart system. *Retrieval Results and Future Plans*.

Salton, G., Wong, A., and Yang, C.-S. (1975). A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Sanjeev, A., Yingyu, L., and Tengyu, M. (2017). A simple but tough-to-beat baseline for sentence embeddings. *Proceedings of ICLR*.

Saxe, J. B. (1979). Embeddability of weighted graphs in k-space is strongly NP-hard. In *Proceedings of* 17^{th} *Allerton Conference in Communications, Control and Computing*, pages 480–489, Monticello, IL.

Schapire, R. E. (1990). The strength of weak learnability. *Machine learning*, 5(2):197–227.

Schoenberg, I. J. (1935). Remarks to Maurice Frechet's article: Sur la definition axiomatique d'une classe d'espaces vectoriels distancies applicables vectoriellement sur l'espace de Hilbert. *Ann. Math.*, 36:724–732.

Schönemann, P. H. (1966). A generalized solution of the orthogonal Procrustes problem. *Psychometrika*, 31(1):1–10.

Schölkopf, B., Tsuda, K., and Vert, J.-P. (2004). *Kernel methods in computational biology*. MIT Press, Cambridge, Mass.

Sharir, M. (1981). A strong-connectivity algorithm and its applications in data flow analysis. *Computers & Mathematics with Applications*, 7(1):67–72.

Shieber, S. M. (1985). Evidence against the context-freeness of natural language. In *Philosophy, Language, and Artificial Intelligence*, pages 79–89. Springer.

Siglidis, G., Nikolentzos, G., Limnios, S., Giatsidis, C., Skianis, K., and Vazirgiannis, M. (2018). Grakel: A graph kernel library in python. *arXiv preprint arXiv:1806.02193*.

Sil, A., Kundu, G., Florian, R., and Hamza, W. (2018). Neural cross-lingual entity linking. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Skianis, K., Malliaros, F., and Vazirgiannis, M. (2018). Fusing document, collection and label graph-based representations with word embeddings for text classification. In *Proceedings of the Twelfth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-12)*, pages 49–58.

Song, Y., Shi, S., Li, J., and Zhang, H. (2018). Directional skip-gram: Explicitly distinguishing left and right context for word embeddings. 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 175–180.

Soramäki, K., Bech, M. L., Arnold, J., Glass, R. J., and Beyeler, W. E. (2007). The topology of interbank payment flows. *Physica A: Statistical Mechanics and its Applications*, 379(1):317–333.

Sorensen, D. C. (1992). Implicit application of polynomial filters in a k-step arnoldi method. *SIAM Journal on Matrix Analysis and Applications*, 13(1):357–385.

Sun, Y., Lin, L., Tang, D., Yang, N., Ji, Z., and Wang, X. (2015). Modeling mention, context and entity with neural networks for entity disambiguation. In *IJCAI*, pages 1333–1339.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Thom, R. (1970). Topologie et linguistique. pages 226–247.

Thue, A. (1912). *Über die gegenseitige Lage gleicher Teile gewisser Zeichenreihen, von Axel Thue ...* J. Dybwad.

Torgerson, W. S. (1952). Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419.

Tsvetkov, Y., Faruqui, M., Ling, W., Lample, G., and Dyer, C. (2015). Evaluation of word vector representations by subspace alignment. In *Proc. of EMNLP*.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59(236):433–460.

Turney, P. D. and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Usbeck, R., Ngomo, A.-C. N., Röder, M., Gerber, D., Coelho, S. A., Auer, S., and Both, A. (2014). Agdistis-graph-based disambiguation of named entities using linked data. In *International semantic web conference*, pages 457–471. Springer.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142.

van Aardenne-Ehrenfest, T. and de Bruijn, N. (2009). Circuits and trees in oriented linear graphs. In *Classic papers in combinatorics*, pages 149–163. Springer.

Vapnik, V. N. (1995). *The Nature of Statistical Learning Theory*. Springer-Verlag, Berlin, Heidelberg.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Vidal, R., Ma, Y., and Sastry, S. (2016). *Generalized Principal Component Analysis*. Springer, New York.

Vylomova, E., Rimell, L., Cohn, T., and Baldwin, T. (2015). Take and took, gaggle and goose, book and read: Evaluating the utility of vector differences for lexical relation learning. *arXiv preprint arXiv:1509.01692*.

Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014). Knowledge graph embedding by translating on hyperplanes. In *AAAI*, volume 14, pages 1112–1119.

Watkins, D. S. (2007). *The matrix eigenvalue problem: GR and Krylov subspace methods*, volume 101. Siam.

Weissenbacher, D., Tahsin, T., Beard, R., Figaro, M., Rivera, R., Scotch, M., and Gonzalez, G. (2015). Knowledge-driven geospatial location resolution for phylogeographic models of virus migration. *Bioinformatics*, 31(12):i348–i356.

Wu, D. and Wu, Z. (2007). An updated geometric build-up algorithm for solving the molecular distance geometry problem with sparse distance data. *Journal of Global Optimization*, 37:661–673.

Yamada, I., Shindo, H., Takeda, H., and Takefuji, Y. (2016). Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259.

Yamada, I., Shindo, H., Takeda, H., and Takefuji, Y. (2017). Learning distributed representations of texts and entities from knowledge base. *Transactions of the Association for Computational Linguistics*, 5:397–411.

Yosef, M. A., Hoffart, J., Bordino, I., Spaniol, M., and Weikum, G. (2011). Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proceedings of the VLDB Endowment*, 4(12):1450–1453.

Zheng, J. G., Howsmon, D., Zhang, B., Hahn, J., McGuinness, D., Hendler, J., and Ji, H. (2015). Entity linking for biomedical literature. *BMC medical informatics and decision making*, 15(1):S4.

Index

INSTITUT POLYTECHNIQUE DE PARIS

Titre : Graphes, géométrie et représentations pour le langage et les réseaux d'entités

Mots clés : Analyse combinatoire, Théorie des graphes, Géométrie des distances, Théorie de la complexité, Modèles de language naturel, Réseaux

Résumé : La nature ambiguë de certaines structures discrètes pose problème pour la modélisation ainsi que le traitement et l'analyse à l'aide d'un programme informatique. Le langage dit "naturel", et sous sa forme en particulier de représentation textuelle, en est un exemple. Le sujet de cette thèse consiste à explorer cette question, que nous étudions à l'aide de méthodes combinatoires et géométriques.
Dans un premier temps, nous commençons par démontrer des propriétés combinatoires d'une famille de graphes intervenant de manière implicite dans les modèles séquentiels. Ces propriétés concernent essentiellement le problème inverse de trouver une séquence représentant un graphe donné, et nous permettent d'effectuer une comparaison expérimentale de différents modèles séquentiels utilisés en modélisation du langage.
Dans un second temps, nous considérons une application pour le problème d'identification d'entités nommées. A la suite d'une revue de solutions récentes, nous proposons une méthode compétitive basée sur la comparaison de structures de graphes de connaissances et moins coûteuse en annotations d'exemples. Nous établissons également une analyse expérimentale d'influence d'entités à partir de relations capitalistiques, suggérant l'élargissement du cadre d'application de l'identification d'entités à des bases de connaissances de natures différentes.
Ensuite, nous développons une étude géométrique de représentations de mots récemment proposées, au cours de laquelle nous discutons une conjecture géométrique théoriquement et expérimentalement. Cette étude suggère que les analogies du langage sont difficilement transposables en propriétés géométriques.
Enfin, nous proposons une méthodologie basée sur le paradigme de la géométrie des distances afin de construire de nouvelles représentations de mots ou d'entités. Nous proposons des algorithmes de résolution de ce problème à grande échelle, qui nous permettent de construire des représentations interprétables et compétitives en performance pour des tâches extrinsèques. Plus généralement, nous proposons à travers ce paradigme un nouveau cadre et piste d'explorations pour la construction de représentations en apprentissage machine.

Title : Graphs, Geometry and Representations for Language Models and Networks of Entities

Keywords : Combinatorics, Graph theory, Distance geometry, Computational complexity, Natural language models, Networks

Abstract : The ambiguous nature of certain discrete structures is problematic for their modeling as well as their processing and analysis with a program. Natural language, and in particular its textual representation, is an example. The subject of this thesis is to explore this question, which we approach using combinatorial and geometric methods
Firstly, we start by showing combinatorial properties of a family of graphs involved in sequential models. These properties essentially concern the inverse problem of finding a sequence representing a given graph. The resulting algorithms allow us to carry out an experimental comparison of different sequential models used in language modeling.
Secondly, we consider an application for the problem of identifying named entities. Following a review of recent solutions, we propose a competitive method based on the comparison of knowledge graph structures which is less costly in annotating examples dedicated to the problem. We also establish an experimental analysis of the influence of entities from capital relations. This analysis suggests to broaden the framework for applying the identification of entities to knowledge bases of different natures.
Then, we perform a geometric study of recently proposed representations of words, during which we discuss a geometric conjecture theoretically and experimentally. This study suggests that language analogies are difficult to transpose into geometric properties, and leads us to consider the paradigm of distance geometry in order to construct new representations.
Finally, we propose a methodology based on the paradigm of distance geometry in order to build new representations of words or entities. We propose algorithms for solving this problem on some large scale instances, which allow us to build interpretable and competitive representations in performance for extrinsic tasks. More generally, we propose through this paradigm a new framework and research leads for the construction of representations in machine learning.

Institut Polytechnique de Paris
91120 Palaiseau, France

INSTITUT POLYTECHNIQUE DE PARIS