Contents

Chapter 1

Introduction

Contents

1.1 Overview

Today, there are more than four billion pages indexed on the Web. Among these data, it is argued that 70% is non-structured, namely text. Indeed, organizations, such as newspapers, universities, etc, are constantly generating freely-accessible, textual content. As a representative example, the entire English version of the Wikipedia encyclopedia has an ever-growing size of more than 16 Gb consisting of text. the community contributes daily to increase an improve the quality of the text (and hopefully the quality of the content). Alongside the development of community-driven creation of content on the Web, we find the effort put forward to design and implement open source data analysis tools. These sophisticated off-the-shelf applications and libraries help us study the data through Data Mining, and more concisely, Natural Language Processing (NLP) techniques.

These last two factors, copious amounts of accessible data and open-source analysis tools coupled to computational prowess yields the opportunities that actually push research efforts in the area. A practical question that follows is how do we extract the useful meaning or knowledge latent within the text. Briefly, to analyze textual data, we need to represent textual units (e.g., words, sentences, paragraphs, documents) by means of features that describe them. Moreover, text may be represented from different points of view. For example, we can characterize words with respect to its neighboring words or to the syntactical relations it appears in, or its role in the structure of a phrase, and so on. Each one of these dimensions distinguishes a given word in different useful ways and more importantly they also complement each other. We argue that in order to thoroughly analyze text, we need a multi-layered model able to take into account these levels of representation in order to find complementary relatedness among text during the analysis.

The large amount of text data plus the need to represent it as globally as possible poses non-trivial challenges for comprehensive textual analysis. Firstly, we need an efficient and straight-forward model to store heterogeneous textual information and the links within it. Secondly, we require efficient algorithms that take advantage from the information stored. Thirdly, we require to leverage the diversity of the representations by finding complementary ways textual units interact among them.

In the literature, language representation models are generally based on two types: the vector space model and the network (or graph) model. The former represents textual units through their features in a vector space. The latter represents the interactions between textual units in a graph-based by means of nodes and edges. In the end though, both representations are intertwined as we can represent a graph as an object-feature matrix through its incidence or adjacency matrices.

Language (or linguistic in this thesis) networks have been studied profoundly. Its theory draws heavily from several research fields (complex networks, graph theory, linguistics, natural language processing). Using the division proposed by [Choudhury 2009a], in the related literature, works follow one of two directions: they analyze the nature of the network from a structural point of view (citations here); or they leverage the network to solve NLP tasks (citations here). The second direction can be further divided into two sub-classes: those that contain homogeneous textual features; and those that comprise heterogeneous descriptors (citations here). In the majority of these works, we note two main shortcomings:

1. the majority of related works that deal with the application of language networks do not take into consideration different types of feature representations. Having a global insight into textual data allows for deeper structural and application analysis.

2. the works that do implement heterogeneous solutions, do not aim to explicitly profit from the complementarity of the features diversity. Finding relatedness between textual units at different levels of representation can be useful to improve the results of NLP applications.

The main topic of this dissertation is language representation through a heterogeneous linguistic network and its applications on NLP. Our objective is twofold: first, to conceive a model able to handle multiple textual-data features extracted from. Second, to use this model to solve NLP tasks efficiently and making explicit use of the features diversity. Furthermore, we aim to leverage open-source corpus to complement in-domain datasets. To achieve our goals, we base our model, and proposed applications, on five main concepts: linguistic representations, the distributional hypothesis, network analysis (and related methods), multimedia fusion techniques, and lastly, machine learning methods, namely supervised and unsupervised approaches. these techniques are used as the tools to fit learning models over our data. Indeed, to accomplish the second goal, we propose solutions to two well-known natural language processing tasks: word sense induction and disambiguation and named entity recognition.

1.2 Contributions

Having cited the disadvantages of current language representations and applications, we set out to fill the gaps with our own propositions. First, in order to have a dataset to test and work with, we parsed and stored the English Wikipedia taking into account largely-ignored syntactic information. Secondly, we fitted our linguistic network to a sample of the parsed Wikipedia corpus. Thirdly, we proposed an algorithm to solve word sense induction and disambiguation by leveraging the network's structure. Finally, we present a method that combines textual features to solve both named entity recognition and again word sense induction and disambiguation. More concisely, we make the following contributions in this dissertation:

- An annotated parsed dump of the English Wikipedia. In this parse we focus onto the syntactical characteristics of words: aside from the classical Part-of-Speech (PoS) tags and dependency parsing relations, we provide the full constituent parse branch for each word in a succinct way.

- Using the information extracted in the previous step, we propose a heterogeneous language network that takes into account lexical and syntactical linguistic information. The proposed model aims to take advantage of the information stocked within our dump and to overcome the limitations of current language networks. We propose a novel hypergraph model able to link together words according to different characteristics.

- An algorithm that leverages the structure of the network in order to solve word sense induction and disambiguation. Our proposed method takes into consideration the "real-world" characteristics of the graph to group similar senses and then assign them to target words. We improve the overall performance compared to other similar graph-based techniques.

- We explore multimedia fusion techniques to complement the features hold by the network. Specifically, we focus on solving named entity recognition and word sense induction and disambiguation by applying feature-combination methods that have already shown their efficiency in the multimedia analysis domain. Our results show that the combination of textual features indeed improves the performance compared to single feature representations and trivial features concatenation. Even more, we experiment with background information from open text sources to further improve the performance of our models. Finally, we also present an analysis on the behavior of features according to different types of senses and entity classes.

- We render public our parsed Wikipedia dump as well as the tools (and its source code) used to perform the parse. The implemented methods are also freely available.

1.3 Structure of the Dissertation

The remainder of this thesis is structured as follows:

Chapter 2 This chapter has a detailed overview about the first four axes of this thesis: how we can represent words from a lexical, syntactic and semantic point of view, then, how we find relatedness among words based on the distributional hypothesis, and how we link both words and features together in a language network to discover their structure and to extract useful knowledge. Afterwards, we detail the machine learning methods used to solve the NLP tasks. Finally, we present the two NLP tasks we solve using our propositions: Word Sense Induction and Disambiguation (WSI/WSD) and Named Entity Recognition (NER).

Chapter 3 We discuss previous work related to the main contribution of this work, linguistic networks. We cover the previous work specific to each task in its corresponding chapter.

Chapter 4 We present and define a novel structure to hold language information: the hypergraph linguistic model. We discuss its characteristics and the intuitions behind its conception: the hypergraph choice,the role of nodes and edges, the type of features stored, the advantages it present in terms of accessing and manipulating the data. In the following chapters, we address the applications that leverage the properties of the structure proposed.

Chapter 5 In this chapter, we present an algorithm that exploits the structure of the network, i.e., the connections between nodes. Namely, we derive senses from a test corpus by finding related words connected to a single main node. We test the linguistic and lexical features and discuss about its qualities. Our results improve on the performance of similar propositions from the literature. We study the combination of features using multimedia fusion techniques in the next chapter.

Chapter 6 We explore the application of multimedia fusion techniques using linguistic features to solve NLP tasks. Briefly, fusion techniques may consist on trivially concatenating feature columns or a combination that aims to transfer relatedness information from one representation to a second one. We experiment with these methods on three datasets for named entity recognition and one dataset for word sense induction and disambiguation. Indeed, we show that using certain configurations of fusion techniques can lead to improvements over single-feature and trivial-concatenation representation matrices. Furthermore, we explore the contribution from each representation kind to each sense and class in each task respectively.

Chapter 7 This chapter describes the process of parsing and storing an English Wikipedia Syntactical Dump. We use open source tools on freely available data to generate a software that is able to extract lexical and syntactic data from a given corpus.We thus create and publicly release a syntactically annotated version of the English Wikipedia containing often-neglected information, stored in a format that facilitates its manipulation.

Chapter 8 We conclude this dissertation and present possible avenues for future work.

Chapter 2

Background

Abstract. *This chapter goes into detail about the five theoretical axes of this dissertation. First, we present language representations and specifically the three different types that interest us: lexical, syntactic and semantic. Secondly, we cover how we can infer relatedness among words based on the neighbors of each word. This intuition is expressed by the distributional hypothesis. Thirdly, we explain how we can join words together through their common features by means of a graph-like structure. This structure is known as a language network. Fourth, we detail the techniques used in the multimedia literature to combine (or fuse) features coming from different sources and improve the performance of a knowledge-discovering system. In our work, instead of combining different multimedia sources, we combine diverse textual representations. Finally, we explain the (supervised and unsupervised) methods we employ to solve the NLP tasks at hand.*

Contents

2.1 Linguistic Features

2.1.1 Lexical Representations

2.1.2 Syntactic Representations

2.1.3 Semantic Representations

2.2 Distributional Hypothesis

A simple and effective method of determining the nature of any textual unit (either words or larger quantities of text), without any other external information (human annotations), is to leverage the distributional hypothesis. Briefly, it states that a word meaning is determined by its own context. The most basic form of co-occurrence is determined by neighbor words that appear before or after a word of interest.

The propositions we present in this thesis are greatly based on the analysis of a word context as a source of valuable information to discover its nature. This context-analysis insight was introduced by [Harris 1954] and states that words that occur in similar contexts tend to have similar meanings. In a more light-hearted phrasing, [Firth 1957] formulated it as "You shall know a word by the company it keeps!".

The distributional hypothesis is popularly exploited in the NLP/Text mining domain by using the vector space model. In this model a word is represented by a vector in which the elements are derived from the occurrences of the word in various contexts, such as windows of words (n-words to the right and to the left), syntactical positions or grammatical dependencies, among other contexts. In our research we chose not to use vectors directly but instead use graphs (and hypergraphs) as an intuitive way to model the context of words in a corpus. Graphs have the capability to naturally encode the meaning and structure of a text by connecting language units (words in our case) through various relationships. Indeed, recent years have seen a rise in natural language methods based on graph-theoretical frameworks [Mihalcea 2011]. Specifically, in this work we represent words as vertices and the edges that link these nodes (words) correspond to various linguistic context properties shared among them.

2.3 Networks and Linguistic Networks

2.3.1 Network Representation

2.3.2 Types of Language Networks

2.4 Fusion Techniques

2.4.1 Early Fusion

2.4.2 Late Fusion

2.4.3 Cross Fusion

2.5 Supervised and Unsupervised Methods

2.5.1 Logistic Regression

2.5.2 Perceptron and Structured Perceptron

Chapter 3

Related Work

Abstract. *In this chapter, we present an overview of linguistic network's related work. We first categorize them by their objectives and then by the data contained within. Next, we discuss what and how different methods are used with language networks to extract knowledge from their structural properties. Finally, we discuss the limitations of the current propositions and the advantages of the model we introduce in the following chapter.*

Contents

3.1 Types of Linguistic Networks

According to their objectives, we can consider two types of contributions in the linguistic-network literature: on the one hand, there are those approaches that investigate the nature of language via its graph representation, and on the other hand, we find those that propose a practical solution to a given NLP problem [Choudhury 2009b]. In particular, we pay attention to two aspects of a given network-based technique: (1) the characteristics of the linguistic data within the network, and (2), the algorithms used to extract knowledge from it.

In the following paragraphs we introduce the general categories of LNs according to their type of content and relations. We will introduce these categories as well as the approaches that make use of them.

[Mihalcea 2011] defines four types of Language Networks (LN): co-occurrence network, syntactic-dependency network, semantic network and similarity network. Meanwhile, from a deeper linguistic point of view, [Choudhury 2009b] defines broader categories, each having several sub-types. The main difference (in our context) between both definitions lies in the separation of categories. In [Choudhury 2009b], they conflate syntactic-dependency and co-occurrence networks into the same category: word co-occurrence networks. Similarly, they join semantic and similarity networks together and place them inside a broader category of lexical networks. The third family defined concerns phonological networks which is out of the scope of this work. In this work we will explore four categories of linguistic networks:

semantic, lexical co-occurrence, syntactic co-occurrence and heterogeneous networks. The following sections will elucidate what each kind of network represent, we will mention works that employ this kind of networks and also list the main methodology differences that variate from one approach to another.

Semantic Networks A Semantic Network (SN) relates words, or concepts, according to their meaning. The classical example of a SN is the renowned knowledge base Wordnet. This network, which serves also as an ontology, contains sets of synonyms (called synsets) as vertices and semantic relations as their edges. Typical semantic relationships include synonymmy-antnonimy, hypernym-hyponym, holonym-meronym. However, other semantic similarities can be defined. The edges are usually not weighted, although in some cases certain graph similarity measures may be used.

Word sense induction is indeed a task usually solved using semantic networks, specially Wordnet (and to a lesser extent, BabelNet) [Mihalcea 2004, Sinha 2007, Tsatsaronis 2007, Navigli 2007, Agirre 2008, Klapaftis 2008, Agirre 2009, Klapaftis 2010, Silberer 2010, Moro 2014]. Given an input text with a set of ambiguous target words to process, these approaches follow a two-step algorithm:

1. Link target words (usually nouns, skipping stop-words and functional words) with their corresponding sense (or synset in the case of WordNet-like dictionaries) and extract their vertices and edges into a new, smaller, SN.

2. Apply a node ranking technique, usually a random-walk-based method, and select, for each ambiguous word in the input text, its top ranking synset node as the correct sense.

The amount of edges a SN has grows depending of the size of the version of Wordnet used or the level of polysemy of a given word. In order to avoid redundancy or contradiction between linking nodes, [Mihalcea 2004, Navigli 2007] applied pruning techniques to avoid *contamination* while calculating ranking metrics in order to define a pertinent sense. Regarding edge similarity measures, in [Sinha 2007, Tsatsaronis 2007] they test some metrics individually and also combined. They found that the best results are indeed obtained when several metrics are used at the same time.

Other semantic tasks can also be solved using a SN. For example, Entity linking [Moro 2014]. In their work, they leverage the BabelNet LKB to jointly disambiguate and link polysemous words to Wikipedia articles.

Concerning the measure of semantic affinity between two terms, in [Yeh 2009] they quantify word similarity by means of projecting them into a Wikipedia space. First, they represent each word by a vector representing its most pertinent pages, and then they calculate a vectorial similarity measure between them.

In [Matuschek 2013] they propose a technique that aligns SNs, i.e., they link senses from two different networks. This task is called word sense alignment. Several SNs are used (Wordnet, Wikipedia, Wiktionary, etc.) thus nodes can represent synsets, articles, or concepts. The links may depict semantic relations or may be links joining two concepts or pages together. Their approach aims to find the shortest path between nodes of any two given SNs while leveraging already existing links between equal concepts found in both SNs.

Finally, extracting entities from text can also benefit from the use of SNs. The work proposed by [Kivimäki 2013] aims to extract technical skills from a document. Again, using Wikipedia as SN, they first represent each article and the input document in a token vector space model. Next, they find the document top 200 similar pages by calculating the cosine similarity between the document and each page. This serves to extract a Wikipedia subgraph which is used to calculate the most relevant pages for the entry document. Finally, the top pages are filtered by means of selecting those articles that actually represent a skill using a fixed list of skill-related tokens. Once again, the nodes represent Wikipedia articles and the edges the hyperlinks that join them.

The cited methods vary in how they make use of their SN, not so much in the network per se. These differences boil down to three aspects:

- Type of relationship implied by the edges linking the nodes of the network,
- The algorithm used to rank the nodes after the semantic network is built, and
- The weight assigned to each edge.

Lexical Co-occurrence Networks Most co-occurrence based intuitions in NLP have their origin in the distributional hypothesis. An effective way to represent word co-occurrences is by means of a graph structure. Indeed, this kind of graphs are the central column of a Lexical Co-occurrence Network (LCN). In these structures, nodes represent words and edges indicate co-occurrence between them, i.e., two words appear together in the same context. A context can vary from a couple of words (before or after a given word) to a full document, although it is usually defined at sentence level. The edges' weight represent the strength of a link and is generally a frequency based metric that takes into account the number of apparitions of each word independently and together.

To solve a task in a truly unsupervised way, researchers generally use this kind of networks instead of LKBs. It is then natural that word sense disambiguation approaches leverage lexical co-occurrence networks, and in return, the distributional hypothesis, to automatically discover senses for a given target word. That is why WSI methods [Véronis 2004, Klapaftis 2007, Navigli 2010, Klapaftis 2008, Di Marco 2011, Jurgens 2011] are tightly related to LCNs. The cited works use a LCN as described before while other works such as [Navigli 2007, Qian 2014] represent the co-occurrence by means of a hypergraph scheme. In short, a hypergraph structure is a graph generalization where an edge (called hyperedge) can link more than two vertices per edge and thus it is able to provide a more complete description of the interaction between several nodes [Estrada 2005].

In their paper, given an input document with several contexts for each target word, they first group together the contexts via a topic-modeling technique. Thus, each context is assigned a particular group (in this case, a topic). Secondly, a hypergraph is built where the vertices represent contexts and the hyperedges link two nodes together if they share the same topic. Thirdly, the hypergraph is clustered and the words of each context (of each node) are used to build vectorial representations

WSI systems generally perform four steps. Given an input text with a set of target words and their contexts (target words must have several instances throughout the document to cluster them), the steps are the following:

1. Build a LCN, assigning tokens as nodes and establishing edges between them if they co-occur in a given context (usually if they both appear in the same sentence),

2. Determine the weights for each edge according to a frequency metric,

3. Apply a graph clustering algorithm. Each cluster found will represent a sense of the polysemous word, and

4. Match target word instances with the clusters found by leveraging each target word context. Specifically, assign a cluster (a sense) to each instance by looking at the tokens in the context.

As with semantic networks, not only WSD or WSI can be solved with LCNs. Finding semantic associated terms in a corpus is a critical step in several NLP systems. This task is solved in the system proposed by [Liu 2011]. They also use a LCN although instead of a co-occurrence graph, they also employ a co-occurrence hypergraph, where nodes represent words and edges describe co-occurrence at the paragraph level. In this work, they use such structure to find related terms in a given corpus. In order to do it, they mine the hypergraph as in a frequent itemsets problem, where the items are the words from a text. The method consists in first finding similar itemsets by means of measuring similarity between nodes. Once the 2-itemsets are found, they induce a graph from the original hypergraph by drawing edges between nodes that have a similarity superior to an arbitrary threshold. Lastly, in order to find k-itemsets ($k > 2$), the find either complete or connected components in the induced graph.

As with WSD, while the LCNs used are mostly the same among approaches, there are certain moving parts that make up the difference between WSI approaches. The most common differences that can arise are:

- The clustering algorithm to find senses in the LCN graph.

- The technique used to match context words to clusters.

- The weight used in the graph edges.

Syntactic Co-occurrence Networks A Syntactic Co-occurrence Network (SCN) is very similar to a LCN in the sense that both exploit the distributional hypothesis. Nonetheless, SCNs go further by leveraging syntactic information extracted from the text. There are two main types of syntactic information both represented as tree structures: constituency-based parse trees and dependency-based parse trees. Briefly, the former structure splits a phrase into several sub-phrases. In this way we can get a glimpse of the role of each word inside a phrase. The latter tells us about the relationships existing between words in the phrase. SCNs employ, most of the time, dependency trees to create a graph that relates words according to their syntactic relations. In the case of [Hope 2013b], a graph is built using syntactic dependencies. It is used to perform WSI using a very similar approach as those systems using LCNs.

A network representation that is on the border line between being a LCN and a SCN is that of [Bronselaer 2013]. They propose a graph document modelization. In their network, nodes represent words and edges their co-occurrence, as any LCN. Still, their graph resembles a SCN because the edges may represent one of three types of words: either prepositions, conjunctions or verbs. As a result, they need to first extract syntactic information from a document, namely the part-of-speech tags of each word. They find the most relevant words of a given text by ranking the nodes of the graph. The words that best represent a document can be used to summarize it, as they show in their work.

Approaches based on SCN are rarely used in WSD or WSI systems, and therefore they are an interesting reserch avenue to explore.

Heterogeneous Networks Until now we have described different types of networks with single types of nodes and relations. Lately, heterogeneous networks have been defined in order to model multi-typed information in a single structure [Han 2009].

Even though this kind of structure seems to open new avenues of research in the semantic analysis domain, only few explicitly take advantage of them, as is the case of [Saluja 2013]. In their approach, they build a graph that links together features with words, and discover similarity measures that leverage the multi-typed nature of their network.

3.2 Algorithms used in Linguistic Networks

We have discussed until now the different types of networks from a content point of view. In this subsection, we address the details of the algorithms used to solve practical NLP tasks. In this section we will cover the details of four different types of graph algorithms.

Edge Weights We begin by describing the metrics used to determine similarity between nodes, usually stored as edge weights. As stated in the previous sections, most of the metrics are frequency based, specially when dealing with LCNs. The main idea of these measures is to assign a weight that decreases as the association frequency of the words increases. Among these measures, the most popular are the Dice coefficient [Navigli 2010, Di Marco 2011, Di Marco 2013], normalized pointwise mutual information [Hope 2013b], and a graph-adapted tf-idf variant [Tsatsaronis 2007] which aims to give importance to frequent edges while also favoring those that are rare.

Edge weights can also be calculated when the vertices of a network do not represent words. Such is the case of [Klapaftis 2010], where nodes represents a target word context (set of tokens around an ambiguous word). This time the Jaccard index is used to quantify similarity between them while considering how many words are shared between a pair of context nodes.

A more sophisticated approach to edge weighting is proposed in [Saluja 2013] where they employ custom-defined functions in order to learn the most appropriate edges' weights for a given set of seed vertices inside a network. The main idea is to enforce *smoothness* (keeping two nodes close if they have related edges) across the network.

As a way to rank edges according to their importance, the ratio of triangles (cycles of length 3), squares (cycles of length 4), and diamonds (graphs with 4 vertices and 5 edges, forming a square with a diagonal) in which an edge participates are calculated [Navigli 2010, Di Marco 2013]. Once the top edges are found, they create a subgraph containing only these edges (and its corresponding vertices).

Finally, instead of applying weights to edges, a case where nodes are weighted can be found in [Kivimäki 2013]. They measure and remove popular nodes in order to avoid their bias during the application of their random walk approach.

Graph Search Usually, in a WSD approach, the first step to follow is to build a graph from a LKB. The goal is to explore the semantic network and find all the senses linked to those found in the context of an ambiguous word. Aside from custom search heuristics applied by certain works [Agirre 2006, Sinha 2007, Agirre 2009], researchers also use well-known graph techniques such as depth-first search [Navigli 2007], breadth-first search [Agirre 2008] and even the Dijsktra algorithm to find the group of closest senses in the network [Matuschek 2013].

Node Connectivity Measures A Connectivity Measure (CM) determines the importance of a node in a graph according to its association with other nodes. In most cases its value ranges from zero to one, where the 0 indicates that the node is of minor importance while 1 suggests a relatively high significance. Nowadays, the most widely used measures are those based on random walks.

A Random Walk (RW) can be simply defined as the traversal of a graph beginning from a given node and randomly jumping to another in the next time step.

PageRank [Brin 1998], the popular random walk based algorithm is used commonly in WSD. The recursive intuition of PageRank is to give importance to a node according to the PageRank value of the nodes that are connected to it. Nonetheless, as a regular random-walk algorithm, in PageRank the probability distribution to change from a node to another is uniform. In such case, the jumps a random walker performs depend solely on the nature of the graph studied. Among the approaches surveyed, those that use the most PageRank are those that solve word sense disambiguation [Mihalcea 2004, Agirre 2006, Navigli 2007, Silberer 2010]. These approaches make a conventional use of PageRank: they apply it and rank nodes to select the most appropriate senses for ambiguous words. Still, there are some improvements over the classical use of PageRank in WSD. Some techniques employ a different version of PageRank called Personalized PageRank (or PageRank with restart [Murphy 2012] or PPR) were a random walker may return to a specific starting node with certain probability rather than jumping to a random node. This formulation allows researchers to assign more weight to certain nodes. For example, in [Agirre 2009] they are able to use the complete Wordnet graph as their SN. They do this by directly adding context words of a polysemous token into Wordnet and then giving a uniform initial distribution to only these nodes. In this way, they force PageRank to give more importance to the context words without the need of extracting a subgraph from the SN. In [Moro 2014] they apply the same technique to obtain a *semantic signature* of a given sense vertex. After applying PPR, they obtain a frequency distribution over all the nodes in the graph. The so-called semantic signature consists in those nodes that were visited more than an arbitrary threshold and that best represent an input sense node.

There are other methods which share the properties of random walk approaches. In [Tsatsaronis 2007, Kivimäki 2013] they apply a method known as spreading activation. This algorithm aims to iteratively diffuse the initial weights of a set of seed nodes across the network. Specifically, once a weighted semantic network is built, they *activate* the nodes representing the context senses, assigning a value of 1, while *disactivating* the rest by setting them to 0. They determine the most pertinent senses to the input nodes by storing, for each of them, the last active sense node with the highest activation value.

Beyond WSD and into the task of determining word similarities, we found the work of [Yeh 2009], where they calculate a semantic similarity between a pair of words while leveraging a Wikipedia SN. For each word, they apply PPR to find the articles that best represent a word. In [Saluja 2013], they also employ PPR to find synonym words given a word-similarity matrix and a new unknown word (also known as out-of-vocabulary word). They link the new word to its corresponding feature nodes and they normalize the similarity matrix to use the weights as probabilities and thus bias the random walk. In [Kivimäki 2013] they use centrality measures to determine the most relevant nodes in a SN and then, in contrast with most approaches, remove them from the graph in order to not bias their graph algorithms.

With regard to other CMs, there are more elementary alternatives to determine the importance of a node. For example, the approaches of [Véronis 2004, Klapaftis 2007, Liu 2011, Bronselaer 2013, Moro 2014] successfully use the degree of a node, or other metric, to determine its importance in a network.

Graph Clustering/Partitioning Graph clustering is defined as the task of grouping the vertices of a graph into clusters while taking into consideration its edge structure [Schaeffer 2007]. As previously mentioned, graph-based word sense induction relies most importantly in the graph clustering step where the actual senses of a word are inferred.

In this subsection we also consider subgraph extracting techniques which are exploited to find separated groups of words and thus induce senses. In this context we found the approaches of [Véronis 2004, Silberer 2010]. These systems make use of both the Minimum and Maximum Spanning Trees algorithms (MinST and MaxST, respectively) as a final step to disambiguate a target word given its context. Meanwhile, both [Liu 2011, Qian 2014] use the Hypergraph Normalized Cut (HCT) approach, a hypergraph clustering method based on minimum cuts, to induce senses.

Most of the reviewed approaches employ state of the art techniques [Klapaftis 2008, Klapaftis 2010, Jurgens 2011, Hope 2013b]. Specifically, they utilize Chinese Whispers (CW) [Biemann 2006], Hierarchical Random Graphs (HRG) [Clauset 2008], Link Clustering (LC) [Ahn 2010], and MaxMax (MM) [Hope 2013a] respectively.

Briefly, CW is a randomized graph-clustering method which is time-linear with respect to the number of edges and does not need a fixed number of clusters as input. It only requires a maximum amount of iterations to perform. HRG, being a hierarchical clustering algorithm, groups words into a binary tree representation, which allows to have more in-detail information about the similarity among words when compared to flat clustering algorithms. Regarding LC, instead of clustering nodes, this procedure groups edges. Thus it can identify contexts related to certain senses, instead of finding groups of words as most approaches do. Finally, MM, is able to assemble words into a fixed cluster (hard clustering) or allow them to be in several groups at the same time (soft clustering). It shares certain characteristics with CW: they are both methods that exploit similarity within the local neighborhood of nodes and both are time-linear. Nonetheless, a key difference is that CW is not deterministic, while MM is, thus MM will find always the same clustering result for the same input graph.

3.3 State of the Art Discussion

We have covered the network attributes of several approaches on semantic related NLP tasks. A summary of these strategies is shown in Table 3.1. In this section we will shortly discuss the reviewed articles from a modelization perspective as well as looking at the evolution of the approaches used to solve the word sense disambiguation and induction tasks.

Regarding WSD approaches, we see that the use of a lexical knowledge base, such as Wordnet, is pervasive in this task. Indeed, new resources, such as BabelNet, solves to some extent the fixed (no new senses are included automatically) nature of this type of resources by leveraging the always evolving knowledge of Wikipedia. Indeed, in the recent years, entity linking has emerged as a related task to WSD. It

Table 3.1: Survey summary table.

Approach	Network Type				Algorithms			
	Semantic	Lexical	Syntactic	Heterogeneous	Edge Wts.	Graph Search	Connectivity Meas.	Graph Clust.
Veronis, 2004 [Véronis 2004]		x					x	x
Mihalcea et al., 2004 [Mihalcea 2004]	x						x	
Agirre et al., 2006 [Agirre 2006]		x				x	x	
Sinha and Mihalcea, 2007 [Sinha 2007]	x					x		
Navigli and Lapata, 2007[Navigli 2007]	x					x	x	
Tsatsaronis et al., 2007 [Tsatsaronis 2007]	x						x	
Klapaftis and Manandhar, 2007 [Klapaftis 2007]		x			x		x	
Klapaftis and Manandhar, 2008 [Klapaftis 2008]		x			x			x
Agirre and Soroa, 2008 [Agirre 2008]	x					x		
Agirre and Soroa, 2009 [Agirre 2009]	x					x	x	
Klapaftis and Manandhar, 2010 [Klapaftis 2010]		x			x			x
Navigli and Crisafulli, 2010 [Navigli 2010]		x			x			
Silberer and Ponzetto, 2010 [Silberer 2010]		x					x	x
Di Marco and Navigli, 2011 [Di Marco 2011]		x			x			
Jurgens, 2011 [Jurgens 2011]								x
Di Marco and Navigli, 2013 [Di Marco 2013]		x			x			
Hope and Keller, 2013 [Hope 2013b]			x		x			x
Moro et al., 2014 [Moro 2014]	x						x	
Qian et al., 2014 [Qian 2014]	x	x						x
Yeh et al., 2009 [Yeh 2009]	x						x	
Liu et al., 2011 [Liu 2011]		x					x	x
Matuschek and Gurevych, 2013 [Matuschek 2013]	x					x		
Bronselaer and Pasi, 2013 [Bronselaer 2013]			x				x	
Kivimäki et al., 2013 [Kivimäki 2013]	x				x		x	
Saluja and Navrátil, 2013 [Saluja 2013]		x		x	x		x	
25	11	12	2	1	9	6	14	8

takes even more advantage from bases that combine both Wordnet and Wikipedia, such as BabelNet. On the other hand, WSI, while being a more flexible approach (language and word-usage independent, does not require human-made bases) for solving WSD, its results are tightly linked to the quality of the clustering algorithm used. With respect to the networks' modelization, we find that few approaches deal with syntactic attributes. We believe that finding semantic similarities can be improved by adding syntactic information not only while using dependency relations but also by leveraging the consituency tree of each word. Moreover, using syntactic data along with semantic and/or lexical co-occurrences takes us into the heterogeneous network domain which has not been addressed in most of the approaches covered. Being able to design new similarity metrics that deal with different types of information opens new avenues of research in the semantic similarity domain. Finally, concerning the algorithms employed, few approaches make direct use of the graph Laplacian representation. New similarities could be defined using the Laplacian as a starting point.

Taking into account the described opportunities of research, in the following section we propose a hypergraph modelization of a linguistic network that aims to cover some of the limitations stated above.

Chapter 4

Hypergraph Linguistic Model

Abstract. *We present in this chapter our model proposition to stock linguistic data. This structure, based on the concept of hypergraphs, holds heterogeneous textual features, improving on the limitations of common solutions of the state of the art. At the same time, our network allows for fast access to words and their properties from different points of view. Finally, we also discuss our motivations and the scope and limitations of the network.*

Contents

Building upon previous linguistic representations [Klapaftis 2007, Liu 2011, Qian 2014], our model is also based on the use of a hypergraph. Their single most important difference with regular graphs, being able to relate more than two vertices at the same type, allows for a better characterization of interactions within a set of individual elements (in our case, words) [Heintz 2014]. Indeed, our hypergraph modelization initially integrates four types of relations between tokens: sentence co-occurrence, part-of-speech tags, words' constituents data and dependency relations in a single linguistic structure. These relationships were chosen because its is relatively easy to obtain them for high-resource languages. These features can be seen as building blocks for NLP models. Extracting deeper language features would implicate relying even more on general domain systems. In any case, our goal is to arrive to more complex annotations (e.g., named entities) from the selected features and relations.

4.1 Characteristics of our proposition

In our model we group words together according to these features, into a hypergraph schema. Formally, a hypergraph [Berge 1985] is a graph generalization that allows more than two vertices to be linked by a single edge. Let V denote a finite set of objects, and let E (the hyperarcs or hyperedges) be a group of subsets of V such that $V = \cup_{e_j \in E} e_j$. We call $\mathcal{H} = (V, E)$ a hypergraph with the vertex set V and the hyperedge set E. A hyperedge containing just two nodes is a simple graph edge. A hyperedge e is said to be *incident* with a vertex v when $v \in e$.

In our case, the set of tokens in the corpus are the set of nodes V, and the set of hyperedges E represent the relations between nodes according to different

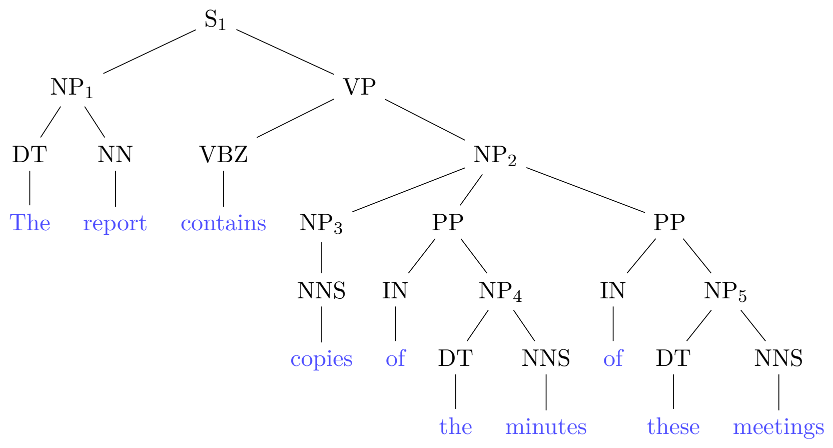


Figure 4.1: Constituency-based tree of the phrase *The report contains copies of the minutes of these meetings.*

linguistic aspects. Each hyperedge may be one of three types: noun phrase[1] constituents (*CONST*), dependency relations (*DEPENDENCY*), or sentence context (*SENTENCE*). We consider that a token v belongs to a hyperedge e_j of type NP or SENTENCE if the token appears in the same noun phrase or in the same sentence. A token v belongs to a hyperedge of type DEPENDENCY if it is the dependent of a certain dependency relation coupled with its corresponding head (or governor). The hypergraph can be represented as a $n \times m$ incidence H matrix with entries $h(i, j) = N(v_i, e_j)$ where $N(v_i, e_j)$ is the number of times $v_i \in e_j$ occurs in the corpus.

We illustrate our hypergraph incidence matrix with the following example phrase: *The report contains copies of the minutes of these meetings.* We tokenize the phrase, keeping all the words, and we lemmatize and parse it to obtain both constituency and dependency trees.

The constituency tree of the example phrase is shown in Figure 4.1. The sentence, as well as each noun phrase (NP) node is identified by a number. We can observe that this phrase is composed by five noun phrases (NP) and one verb phrase. Meanwhile, some of the NPs are formed by other kind of phrases, depending on the grammar production rule used to build each one of them. As is usual in this kind of structures, there is a one to one relation between the number of tokens in the sentence and the number of leaves in the tree.

The dependencies of the example phrase are shown in Table 4.1. They indicate the syntactic relation between the governor of a phrase and dependent. In these relations' examples, the head is the first token to appear followed by the dependent word.

From both of these types of information we can build a hypergraph representation

[1]In this work we consider only noun phrases (NPs). Still, we can easily add other type of phrase chunks.

Table 4.1: Dependency relations of the example phrase.

root(root, contains)	**det**(minutes, the)
det(report, The)	**nmod**(copies, minutes)
nsubj(contains, report)	**case**(meetings, of)
dobj(contains, copies)	**det**(meetings, these)
case(minutes, of)	**nmod**(minutes, meetings)

Table 4.2: Incidence matrix of the example phrase hypergraph modelization.

		CONSTITUENT			**DEPENDENCY**		**SENTENCE**
		NP_1 DT:NN	NP_2 NP:PP:PP	NP_3 NNS	nsubj contains	dobj contains	S_1
NN	report	1			1		1
	contains						1
	copies		1	1		1	1
	minutes		1				1
	meetings		1				1

as stated before. The incidence matrix is illustrated in Table 4.2. For brevity, we only show nouns as well as only the first three noun phrases and the nominal subject (*nsubj*) and direct object (*dobj*) dependency relations. Looking at the table, we can infer that the word *copies* appears in two hyperedges of type CONSTITUENT: first in NP_2, which is built from a NP and two prepositional phrases (PP). Secondly, we see that it is part of NP_3, which indicates a plural noun (NNS).

Regarding the syntactic dependency hyperedges, the word *copies* appear in the *dobj contains* column which indicates that *copies* was the direct object of the verb *contains*.

For the sentence hyperedges, we see that the token *copies* appeared in the same sentence S_1 as the other four noun words.

With this short example we show the intuitive way in which we store three different kinds of relations: lexical co-occurrence, dependency co-occurrence and sentence (at chunk level) co-occurrence.

In the following section we set to solve a natural language processing task using the model described above. Specifically, we address the word sense induction and disambiguation challenges. Both tasks are located on the computational semantics sub-domain of NLP. By making use of the network we want to test the effectiveness of using different types of linguistic features.

Chapter 5

Solving Word Sense Induction and Disambiguation

Abstract. *Word Sense Induction and Disambiguation (WSD/WSI) is an elementary semantic NLP task useful to build larger systems. In this chapter, we leverage the network presented in the previous chapter and propose an algorithm that takes into account the structure of our model. Based on the real-world property of the network, we induce senses by grouping words that share a similar sense. We then assign these senses to target words. We show that our method improves on similar network-based methods.*

Contents

5.1 Introduction

5.2 Related Work

5.3 Approach

5.4 Experiments

5.5 Results

5.6 Discussion and Conclusion

Chapter 6

Fusion Techniques for WSI/WSD and NER

Abstract. *We explore the use of well-known multi-modal fusion techniques to solve two prominent Natural Language Processing tasks. Specifically, we focus on solving Named Entity Recognition and Word Sense Induction and Disambiguation by applying feature-combination methods that have already shown their efficiency in the multi-media analysis domain. We present a series of experiments employing fusion techniques in order to combine textual linguistic features. Our results show that the combination of textual features indeed improves the performance compared to single feature and the trivial feature concatenation. Furthermore, we perform an extensive analysis on the importance of each feature importance with respect to the senses and classes discovered in WSI/WSD and NER, respectively.*

Contents

6.1 Introduction

Named Entity Recognition (NER) and Word Sense Induction and Disambiguation (WSI/ WSD) requires textual features to represent the similarities between words to discern between different words' meanings. NER goal is to automatically discover, within a text, mentions that belong to a well-defined semantic category. The classic task of NER involves detecting entities of type Location, Organization, Person and Miscellaneous. The task is of great importance for more complex NLP systems, e.g, relation extraction, opinion mining. Two common solutions to NER are one of the following: via matching patterns created manually or extracted semi-automatically; or by training a supervised machine learning algorithm with large quantities of annotated text. The latter being the currently more popular solution to this task.

Word Sense Induction and Disambiguation entails two closely related tasks[1]. WSI aims to automatically discover the set of possible senses for a target word given a text corpus containing several occurrences of said target word. Meanwhile, WSD takes a set of possible senses and determines the most appropriate sense for each instance of the target word according to the instance's context. WSI is usually approached as an unsupervised learning task, i.e., a cluster method is applied to the words occurring in the instances of a target word. The groups found are interpreted as the senses of the target word. The WSD task is usually solved with knowledge-based approaches, based on WordNet; or more recently with supervised models which require annotated data.

As stated before, both tasks rely on features extracted from text. Usually, these representations are obtained from the surrounding context of the words in the input corpus. Mainly, two types of representations are used. According to their nature we call these features lexical and syntactical. The first type requires no extra information than that contained already in the analyzed text itself. It consists merely on the tokens surrounding a word, i.e., those tokens that come before and after within a fixed window. The second type, syntactical features, is similar to the lexical representation in that we also consider as features the tokens that appear next to the corpus' words. Nonetheless, it requires a deeper degree of language understanding. In particular, these features are based on part of speech tags, phrase constituents information, and syntactical functionality between words, portrayed by syntactical dependencies. Likewise, specific features, particular to one task are also be employed.

Most of the approaches in the literature dealing with these tasks use each of these features independently or stacked together, i.e., different feature columns in an input representation space matrix. In the latter case, features are combined without regards to their nature.

The main intuition of the present work is that word similarities may be found at different levels according to the type of features employed. In order to exploit these similarities, we look into multimedia fusion methods. In order to better perform an analysis task, these techniques combine multimodal representations, their corresponding similarities, or the decisions coming from models fitted with these features. In our experiments, we try to mutually complement independent

[1]Even though these tasks are closely related, they are independent from one another. Still, in this chapter we consider them to be a single one.

representations by utilizing said fusion techniques to combine (or fuse) features in the hope of improving the performance of the tasks at hand, specially compared to the use of features independently.

Fusion techniques have previously shown their efficiency, mainly on image-related tasks, where there is a need to model the relation between images and text extracts. Here, in order to apply multimedia fusion techniques, we consider that the textual features come from different modalities. The main contribution of this work is to assess the effectiveness of simple yet untested techniques to combine classical and easy to obtain textual features. As a second contribution, we propose a series of feature combination and recombination to attain better results.

The rest of the chapter is organized as follows: in section 6.2, we go into further details about fusion techniques. We introduce the fusion operators that we use in our experiments in section 6.3. Then, in section 6.4 we show the effectiveness of the presented methods by testing them on NER and WSI/WSD and their respective datasets. Finally, in section 6.6 we present our conclusions and future directions to explore. A more general overview on fusion for multimedia analysis can be found in [Atrey 2010].

6.2 Background and Related Work

We first describe below the fusion techniques we use in our methodology as well as relevant use cases where they have been employed. Then, we focus exclusively on fusion techniques applied to NLP tasks, which is the general domain of the two tasks (NER and WSI/WSD) we focus on.

6.2.1 Multimodal Fusion Techniques

Multimodal fusion is a set of popular techniques used in multimedia analysis tasks. These methods integrate multiple media features, the affinities among these attributes or the decisions obtained from systems trained with said features, to obtain rich insights about the data being used and thus to solve a given analysis task [Atrey 2010]. We note that these techniques come at the price of augmenting the complexity of a given system by increasing or reducing the sparsity of a given feature matrix.

In the multimodal fusion literature we can discern two main common types of techniques: early fusion and late fusion.

6.2.1.1 Early Fusion

This technique is the most widely used fusion method. The principle is simple: we take both modal features and concatenate them into a single representation matrix. More formally, we consider two matrices that represent different modality features each over the same set of individuals. To perform early fusion we concatenate them column-wise, such that we form a new matrix having the same number of lines but increasing the number of columns to the sum of the number of columns of both matrices. The matrices may also be weighted as to control the influence of each modality.

The main advantage of early fusion is that a single unique model is fitted while leveraging the correlations among the concatenated features. The method is also easy to integrate into an analysis system. The main drawback is that we increase the representation space and may make it harder to fit models over it.

6.2.1.2 Late Fusion

In contrast to early fusion, in late fusion the combination of multimodal features are generally performed at the decision level, i.e., using the output of independent models trained each with an unique set of features [Clinchant 2011]. In this setting, decisions produced by each model are combined into a single final result set. The methods used to combine preliminary decisions usually involve one of two types: rule-based (where modalities are combined according to domain-specific knowledge) or linear fusion (e.g., weighting and then adding or multiplying both matrices together). This type of fusion is very close to the so-called ensemble methods in the machine learning literature. Late fusion combines both modalities in the same semantic space. In that sense, we may also combine modalities via an affinity representation instead of final decision sets. In other words, we can combine two modality matrices by means of their respective similarities. A final representation is then usually obtained by adding the weighted similarity matrices.

The advantages of late fusion include the combination of features at the same level of representation (either the fusion of decisions or similarity matrices). Also, given that independent models are trained separately, we can chose which algorithm is more adequate for each type of features.

6.2.1.3 Cross-media Similarity Fusion

A third type of fusion technique, cross-media similarity fusion (or simply cross fusion), introduced in [Ah-Pine 2015, Clinchant 2011], is defined and employed to propagate a single similarity matrix into a second similarity matrix. In their paper, the authors propagated information from textual media towards visual media. In our case, we transfer information among textual features. For example, to perform a cross fusion between lexical and syntactical features, we perform the following steps:

1. Compute the corresponding similarity matrices for each type of feature.
2. Select only the k-nearest neighbor for each word within the lexical similarity matrix. These neighbors are to be used as lexical representatives to enrich the syntactical similarities.
3. Linearly combine both similarity matrices (lexical k-nearest lexical neighbors with the syntactical features) via a matrix product.

Cross fusion aims to bridge the semantic gap between two modalities by using the most similar neighbors as proxies to transfer valuable information from one modality onto another one. Usually, the result of a cross fusion is combined with the previous techniques, early and late fusion. In this work we perform experiment in that sense.

6.2.1.4 Hybrid Fusion

We may leverage the advantages of the previous two types of fusion techniques by combining them once more in a hybrid setting. As described in [Atrey 2010, Yu 2014], the main idea is to simultaneously combine features at the feature level, i.e., early fusion, and at the same semantic space or decision level. Nonetheless, they define a specific type of hybrid fusion. In this chapter, we adopt a looser definition of hybrid fusion. That is, we perform hybrid fusion by leveraging the combination of the fusion strategies described before.

We consider the first three types of fusion techniques (early fusion, late fusion and cross fusion) as the building blocks to the experiments we conduct. While we work with a single modality, i.e., textual data, we consider the different kinds of features extracted from it as distinct modalities. Our intuition being that the semantic similarities among words in these different spaces can be combined in order to exploit the latent complementarity between the lexical and syntactical representations. The fusion should therefore improve the performance of the NLP tasks at hand, NER and WSI/WSD.

Our first goal is to assess the effectiveness of the classic fusion methods and then, as a second goal, to propose new combinations that yield better outcomes in terms of performance than the simpler approaches. The new combinations are found empirically. Nonetheless, as we will show, their effectiveness replicates to different datasets and NLP tasks.

6.3 Applying Fusion Techniques

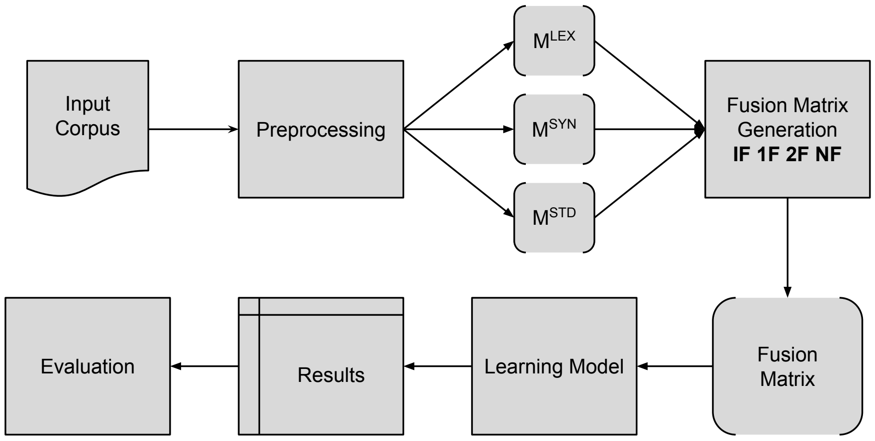
In the present subsection we address the core of the work performed in this chapter. We formally describe the fusion techniques we employ in the next section. Also, we delineate the procedure followed in our experiments.

The experiments we carry on consist in generating fusion matrices that will serve as input to a learning algorithm in order to solve NER and WSI/WSD. These input feature matrices are based upon lexical, syntactical, or other types of representation. The procedure can be sen in Figure 6.1.

6.3.1 Fusion Strategies

We begin by presenting a formal definition of the fusion techniques employed and described in the previous sections. We define (weighted) early fusion, late fusion and cross fusion as follows:

Figure 6.1: Steps followed on our experiments. First the corpus is preprocessed, then features are extracted from the text. A fusion matrix is generated, which in turn is used as input to a learning algorithm. Finally, the system yields its results and to be analyzed.



Early Fusion

$$E(A, B) = \mathbf{hstack}(A, B) \tag{6.1}$$

Weighted Early Fusion

$$wE_\alpha(A, B) = \mathbf{hstack}(\alpha \cdot A, (1 - \alpha) \cdot B) \tag{6.2}$$

Late Fusion

$$L_\beta(A, B) = \beta \cdot A + (1 - \beta) \cdot B \tag{6.3}$$

Cross fusion

$$X_\gamma(A, B) = \mathbf{K}(A, \gamma) \times B \tag{6.4}$$

Parameters A and B are arbitrary input matrices. They may initially represent, for example, the lexical (M^{LEX}) or syntactical based (M^{SYN}) features matrix, or their corresponding similarity matrices, S^{LEX} and S^{SYN}, respectively. In a broader sense, matrices A and B may represent any pair of valid[2] fusion matrices.

In early fusion, $E(A, B)$, the matrices A and B are combined together via a concatenation function **hstack** which joins both of them column-wise. Weighted early fusion represents the same operation as before with an extra parameter: α, which controls the relative importance of each matrix. In the following, we refer to both operations as early fusion. When α is determined, we refer to weighted early fusion.

Regarding late fusion $L_\beta(A, B)$, the β parameter determines again the importance of the matrix A, and consequently also the relevance of matrix B.

[2]Valid in terms of having compatible shapes while computing a matrix sum or multiplication.

In cross fusion $X_\gamma(A, B)$, the $\mathbf{K}(\cdot)$ function keeps the top-γ closest words (columns) to each word (lines) while the rest of the values are set to zero.

Using the previously defined operators, we distinguish four levels of experiments:

1. **Single Features**: in this phase we consider the modalities independently as input to the learning methods. For instance, we may train a model for NER using only the lexical features matrix M^{LEX}.

2. **First Degree Fusion**: we consider the three elementary fusion techniques by themselves (early fusion, late fusion, cross fusion) without any recombination. These experiments, as well as those from the previous level, serve as the baselines we set to surpass in order to show the efficacy of the rest of the fusion approaches. As an example, we may obtain a representation matrix by performing an early fusion between the lexical matrix and the syntactical features matrix: $E(M^{LEX}, M^{SYN})$. In this level we distinguish two types of cross fusion: Cross Early Fusion (XEF) and Cross Late Fusion (XLF). The first one combines a similarity matrix with a feature matrix: $X(S^{LEX}, M^{SYN})$. The second one joins a similarity matrix with a similarity matrix: $X(S^{SYN}, S^{LEX})$.

3. **Second Degree Fusion**: we recombine the outputs of the previous two levels with the elementary techniques. This procedure then yields a recombination of "second-degree" among fusion methods. We introduce the four types of second degree fusions in the following list. Each one is illustrated with an example:

   (a) Cross Late Early Fusion (XLEF): $X(X(S^{STD}, S^{SYN}), M^{STD})$
   (b) Cross Early Early Fusion (XEEF: $X(S^{STD}, X(S^{STD}, S^{SYN}))$
   (c) Early Cross Early Fusion (EXEF): $E(M^{STD}, X(S^{LEX}, M^{STD}))$
   (d) Late Cross Early Fusion (LXEF): $L(M^{STD}, X(S^{STD}, M^{STD}))$

4. **N-Degree Fusion**: in this last level we follow a similar approach to the previous level by combining the output of the second-degree fusion level multiple times (more than two times, or $N > 2u$) with other second-degree fusion outputs. Again, in this level we test the following two fusion operations:

   (a) Early Late Cross Early Fusion (ELXEF): $E(M^{STD}, L(M^{STD}, X(S^{STD}, M^{STD})))$
   (b) Early ELXEF (EELXEF): $E(M^{LEX}, E(E(M^{STD}, L(M^{STD}, X(S^{STD}, M^{STD}))),$ $L(M^{LEX}, X(S^{SYN}, M^{LEX}))))$

6.3.2 Feature Matrices

In the previous subsection we presented the fusion operators used in our experiments. Below we detail the three types of features used to describe the words of the corpus tested.

6.3.2.1 Lexical Matrix (LEX)

For each token in the corpus, we use a lexical window of two words to the left and two words to the right, plus the token itself. Specifically, for a target word w, its lexical context is $(w_{-2}, w_{-2}, w, w_{+1}, w_{+2})$. This type of context features is typical for most systems studying the surroundings of a word, i.e., using a distributional approach [Levy 2014].

6.3.2.2 Syntactical Matrix (SYN)

Based on the syntactic features used in [Levy 2014, Panchenko 2017], we derive contexts based on the syntactic relations a word participates in, as well as including the part of speech (PoS) of the arguments of these relations. Formally, for a word w with modifiers $m_1, \ldots, m_k$ and their corresponding PoS tags $p_1^m, \ldots, p_k^m$; a head h and its corresponding PoS tag p_h, we consider the context features $(m_1, p_{m_1}, lbl_1), \ldots,$ $(m_k, p_{m_k}, lbl_k), (h, p_h, lbl\_inv_h)$. In this case, lbl and lbl_{inv} indicate the label of the dependency relation and its inverse, correspondingly. Using syntactic dependencies as features should yield more specific similarities, closer to synonymy, instead of the broader topical similarity found through lexical contexts.

6.3.2.3 NER Standard Features Matrix (STD)

The features used for NER are based roughly on the same as those used in [Daume 2006, Balasuriya 2009]. The feature set consists of: the word itself, whether the word begins with capital letter, prefix and suffix up to three characters (within a window of two words to the left and two words to the right), and the PoS tag of the current word. These features are considered to be standard in the literature. We note that the matrix generated with these features is exclusively used in the experiments regarding NER.

6.3.3 Learning Methods

We use supervised and unsupervised learning methods for NER and WSI/WSD respectively. On the one hand, for NER, as supervised algorithm, we use an averaged structured perceptron [Collins 2002, Daume 2006] to determine the tags of the named entities. We considered Logistic Regression and linear SVM. We chose the perceptron because of its performance and the lower training time.

On the other hand, for WSD/WSI, specifically for the induction part, we applied spectral clustering, as in [Goyal 2014], on the input matrices in order to automatically discover senses (a cluster is considered a sense). Regarding disambiguation, we trivially assign senses to the target word instances according to the number of common words in each cluster and the context words of the target word. In other words, for each test instance of a target word, we select the cluster (sense) with the maximum number of shared words with the current instance context.

6.4 Experiments and Evaluation

We experiment with four levels of fusion: Single Features (SF), First-degree Fusion (1F), Second-degree Fusion (2F) and N-degree Fusion (NF). The representation matrices for NER come from lexical context features M^{LEX}, syntactical context features M^{SYN} or standard features M^{STD}. On the other hand, experiments on WSI/WSD exclusively employ matrices M^{LEX} and M^{SYN}.

Our first goal is to compare the efficiency of the basic multimedia fusion techniques applied to single-modality multi-feature NLP tasks, namely NER and WSI/WSD. A second goal is to empirically determine a fusion combination setting able to leverage the complementarity of our features.

To this end, we evaluate the aforementioned 4 fusion levels. We note that the fusion combinations in the third and fourth level (2F and NF) are proposed based on the results obtained in the previous levels. In other words, in order to reduce the number of experiments, we restrict our tests to the best performing configurations. This is due to the large number of possible combinations (an argument to a fusion operation may be any valid output of a second fusion operation).

6.4.1 Named Entity Recognition

6.4.1.1 Pre-processing

As is usual when preprocessing text before performing named entity recognition, [Ratinov 2009], we normalize tokens that include numbers. For example, the token 1980 becomes *DDDD* and 212-325-4751 becomes *DDD*-*DDD*-*DDDD* . This allows a degree of abstraction to tokens that contain years, phone numbers, etc. We do not normalize punctuation marks.

6.4.1.2 Features

The linguistic information we use are extracted with the Stanford's CoreNLP parser [Manning 2014]. Again, the features used for these experiments on NER are those described before: lexical, syntactic and standard features, i.e., M^{LEX}, M^{SYN}, and M^{STD}, respectively.

6.4.1.3 Test Datasets

We work with three corpus coming from different domains:

(1) CoNLL-2003 (CONLL): This dataset was used in the language-independent named entity recognition CoNLL-2003 shared task [Sang 2003]. It contains selected news-wire articles from the Reuters Corpus. Each article is annotated manually. It is divided in three parts: training (*train*) and two testing sets (*testa* and *testb*). The training part contains 219,554 lines, while the test sets contain 55,044 and 50,350 lines, respectively. The task was evaluated on the *testb* file, as in the original task.

Table 6.1: NER F-measure results using the Single Features over the three datasets. These values serve as a first set of baselines.

A	Single Features		
	CONLL	**WNER**	**WGLD**
M^{STD}	77.41	77.50	59.66
M^{LEX}	69.40	69.17	52.34
M^{SYN}	32.95	28.47	25.49

(2) WikiNER (WNER): A more recent dataset [Balasuriya 2009] of selected English Wikipedia articles, all of them annotated automatically with the author's semi-supervised method. In total, it contains 3,656,439 words.

(3) Wikigold (WGLD): Also a corpus of Wikipedia articles, from the same authors of the previous corpus. Nonetheless, this was annotated manually. This dataset is the smaller, with 41,011 words. We used this corpus to validate human-tagged Wikipedia text. These three datasets are tagged with the same four types of entities: Location, Organization, Person and Miscellaneous.

6.4.1.4 Evaluation Measures

We evaluate our NER models following the standard CoNLL-2003 evaluation script. Given the amount of experiments we carried on, and the size constraints, we report exclusively the total F-measure for the four types of entities (Location, Organization, Person, Miscellaneous). WNER and WGLD datasets are evaluated on a 5-fold cross validation.

6.4.1.5 Results

We present in this subsection the results obtained in the named entity recognition task, while employing the 4 levels of fusion proposed in the previous section.

In contrast to other related fusion works [Ah-Pine 2015, Clinchant 2011, Gialampoukidis 2016], we do not focus our analysis on the impact of the parameters of the fusion operators. Instead, we focus our analysis on the effect of the type of linguistic data being used and how, by transferring information from one feature type to another, they can be experimentally recombined to generate more complete representations.

Regarding the fusion operators' parameters, we empirically found the best configuration for β, from late fusion $L_\beta(A, B) = \beta \cdot A + (1 - \beta) \cdot B$, is $\beta = 0.5$. This implies that an equal combination is the best linear fusion for two different types of features.

In respect of the γ parameter, used in cross fusion $X_\gamma(A, B) = \mathbf{K}(A, \gamma) \times B$, we set $\gamma = 5$. This indicates that just few high quality similarities attain better results than utilizing a larger quantity of lower quality similarities.

Single Features Looking at Table 6.1, we see that the best independent features, in terms of F-measure come from the standard representation matrix M^{STD}. This is not surprising as these features, simple as they may be, have been used and proved extensively in the NER community. On the other hand, M^{LEX} performs relatively well, considering it only includes information contained in the dataset itself. Nevertheless, this representation that this kind of lexical context features are the foundation of most word embedding techniques used nowadays. While we expected better results from the syntactical features M^{SYN}, as they are able to provide not only general word similarity, but also functional, getting close to synonymy-level [Levy 2014], we believe that the relatively small size of the datasets do not provide enough information to generalize

First Degree Fusion In Table 6.2 we present the First Degree fusion level. The best performance is obtained by trivially concatenating the representation matrices. This baseline proved to be the toughest result to beat. Late fusion does not perform well in this setting, still, we see further on that by linearly combining weighted representation matrices, we can add information to an already strong representation. Finally, regarding the cross fusion techniques, cross early and late fusion, we see that they depend directly on the information contained in the similarity matrices. We note that, as is the case on single features, the combinations with matrix S^{STD} yield almost always the best results. While these fusion techniques by themselves may not offer the best results, we see below that by recombining them with other types of fusion we can improve the general performance of a representation.

Second Degree Fusion The second degree fusion techniques presented in Table 6.3 show that the recombination of cross fusion techniques gets us closer to the early fusion baseline. With the exception of cross late early fusion, the rest of the recombination schemes yield interesting results. First, in cross early fusion, the best results, for the most part, are obtained while using the S^{LEX} matrix combined with the output of $E(M^{LEX}, M^{STD})$, which is still far from the baseline values. Concerning, EXEF, we get already close to surpass the baselines with the M^{STD} matrix, with the exception of the CONLL dataset. In LXEF, even though the cross fusion $X(S^{SYN}, M^{LEX})$ is not the best performing, we found experimentally that by combining it with M^{LEX} through a late fusion, it gets a strong complementary representation. Our intuition in this case was to complement M^{LEX} with itself but enriched with the S^{SYN} information. In the N-degree fusion results we discover that indeed this propagation of information helps us beat the baselines we set before.

N-degree Fusion Finally, the last set of experiments are shown in Table 6.4. Using a recombination of fusion techniques, a so-called hybrid approach, we finally beat the baselines (single features and early fusion) for each dataset. We note that the best configuration made use of a weighted early fusion with $\alpha = 0.95$. This indicates that the single feature matrix, M^{LEX} is enriched a small amount by the

Table 6.2: NER F-measure results using first degree fusion (1F). B is either indicated on the table or specified as follows. Looking at EF, $\hat{b}_{EF} = E(M^{SYN}, M^{STD})$. In XEF, b^*_{XEF} takes the matrix from the set $\{M^{LEX}, M^{STD}\}$ which yields the best performing result. In XLF, $\hat{b}^*_{XLF}$ corresponds to the best performing matrix in $\{S^{LEX}, S^{SYN}\}$. These configurations serve as the main set of baseline results.

A	B		**Early Fusion**	
		CONLL	**WNER**	**WGLD**
M^{LEX}	M^{SYN}	72.01	70.59	59.38
M^{LEX}	M^{STD}	78.13	79.78	61.96
M^{SYN}	M^{STD}	77.70	78.10	60.93
M^{LEX}	$\hat{b}_{EF}$	**78.90**	**80.04**	**63.20**
			Late Fusion	
		CONLL	**WNER**	**WGLD**
S^{LEX}	S^{SYN}	**61.65**	58.79	44.29
S^{LEX}	S^{STD}	55.64	**67.70**	48.00
S^{SYN}	S^{STD}	50.21	58.41	**49.81**
			Cross Early Fusion	
		CONLL	**WNER**	**WGLD**
S^{LEX}	M^{STD}	49.90	**70.27**	**62.69**
S^{SYN}	M^{STD}	47.27	51.38	48.53
S^{STD}	b^*_{XEF}	**52.89**	62.21	50.15
			Cross Late Fusion	
		CONLL	**WNER**	**WGLD**
S^{LEX}	S^{STD}	27.75	**59.12**	38.35
S^{SYN}	b^*_{XLF}	36.87	40.92	39.62
S^{STD}	b^*_{XLF}	**41.89**	52.03	**39.92**

fusion recombination, which is enough to improve the results of said baselines. In CONLL, the early fusion (see Table 6.2) baseline being 78.13, we reached 78.69, the lowest improvement of the three datasets. Regarding the Wikipedia corpus, in WNER, we went from 79.78 to 81.75; and in WGLD, from 61.96 to 67.29, the largest improvement of all.

In the next section we transfer the knowledge gained in this task to a new one, word sense induction and disambiguation.

Table 6.3: NER F-measure results using second degree fusion (2F). In XLEF, a^* corresponds to the best performing matrix in the set $\{X(S^{STD}, S^{LEX}), X(S^{LEX}, S^{STD}), X(S^{STD}, S^{SYN})\}$. For XEEF, $\hat{b}_{XEEF} = E(M^{LEX}, M^{STD})$. In EXEF, b^*_{EXEF} takes the best performing matrix from $\{X(S^{SYN}, M^{LEX}), X(S^{LEX}, M^{LEX}), X(S^{LEX}, M^{STD}), X(S^{SYN}, M^{LEX}), X(S^{SYN}, M^{STD})\}$. Finally, in LXEF, $\hat{b}_{LXEF}$ takes the best possible matrix from $\{X(S^{LEX}, M^{STD}), X(S^{SYN}, M^{STD}), X(S^{SYN}, M^{LEX})\}$.

A	B	**Cross Late Early Fusion**		
		CONLL	**WNER**	**WGLD**
$\hat{a}$	M^{STD}	37.69	59.44	**41.71**
$\hat{a}$	M^{LEX}	**38.31**	**58.73**	41.56
$\hat{a}$	M^{SYN}	29.31	52.06	34.91
		Cross Early Early Fusion		
		CONLL	**WNER**	**WGLD**
S^{STD}	$\hat{b}_{XEEF}$	**54.34**	**64.20**	39.59
S^{LEX}	$\hat{b}_{XEEF}$	49.71	71.84	**45.14**
S^{SYN}	$\hat{b}_{XEEF}$	47.54	53.77	43.32
		Early Cross Early Fusion		
		CONLL	**WNER**	**WGLD**
M^{STD}	b^*_{EXEF}	49.58	**77.32**	**61.69**
M^{LEX}	b^*_{EXEF}	49.79	66.22	53.54
M^{SYN}	b^*_{EXEF}	**51.53**	70.94	53.70
		Late Cross Early Fusion		
		CONLL	**WNER**	**WGLD**
M^{STD}	$\hat{b}_{LXEF}$	54.82	**75.70**	**54.73**
M^{LEX}	$\hat{b}_{LXEF}$	**56.53**	62.27	52.39

6.4.2 Word Sense Induction and Disambiguation

Having learned the best fusion configuration from the previous task, in this experiments we set to test if the improvements achieved can be transfered into another NLP task, namely Word Sensed Induction and Disambiguation (WSI/WSD).

6.4.2.1 Pre-processing

We simply remove stopwords and tokens with less than three letters.

Table 6.4: F-measure results using N-degree fusion (NF). In ELXEF, $\hat{b}_{ELXEF} = L(M^{LEX}, X(S^{SYN}, M^{LEX}))$. For EELXEF, $\hat{b}_{EELXEF} = E(E(M^{STD}, L(M^{LEX}, X(S^{SYN}, M^{LEX}))), L(M^{LEX}, X(S^{STD}, M^{LEX}))$ for CONLL and $\hat{b}_{EELXEF} = E(E(M^{STD}, L(M^{STD}, X(S^{SYN}, M^{STD}))), L(M^{LEX}, X(S^{SYN}, M^{LEX})))$ for WNER and WGLD. The best result is obtained in EELXEF when $\alpha = 0.95$.

A	B	**Early Late Cross Early Fusion**		
		CONLL	**WNER**	**WGLD**
M^{STD}	$\hat{b}_{ELXEF}$	67.16	79.45	62.37
		Early Early Late Cross Early Fusion		
		CONLL	**WNER**	**WGLD**
M^{LEX}	$\hat{b}_{EELXEF}$	65.01	78.02	62.34
$M^{LEX}_{\alpha=0.95}$	$\hat{b}_{EELXEF}$	**79.67**	**81.79**	**67.05**
EF Baseline		78.90	80.04	63.20

6.4.2.2 Features

We use the same set of features from the previous task, with the exception of the standard NER features, that is, those represented by M^{STD}, as they are specifically designed to tackle NER.

6.4.2.3 Test Dataset

The WSI/WSD model is tested on the dataset of the SEMEVAL-2007 WSID task [Agirre 2007]. The task was based on a set of 100 target words (65 nouns and 35 verbs), each word having a set of instances, which are specific contexts where the word appear. Senses are induced from these contexts and applied to each one of the instances.

6.4.2.4 Evaluation Measures

Being an unsupervised task, the evaluation metrics of WSI/WSD are debated in terms of quality [de Cruys 2011]. We consider supervised recall and unsupervised F-measure, as in the competition original paper [Agirre 2007]. The first one maps the output of a system to the true senses of the target words' instances and the second one measures the quality of the correspondence between the automatically found clusters and the senses. We consider that the number of senses found by the system is also a rather good indicator of performance: the best competition baseline assigns the most frequent sense to each target word (this baseline is called MFS),

Table 6.5: Supervised Recall and Unsupervised F-measure for the Semeval 2007 corpus. We also display the average number of clusters found by each fusion configuration.

Method	**Recall (%)**			**FM (%)**			**# cl**
	all	noun	verb	all	noun	verb	
Single Features							
M^{LEX}	79.20	82.10	75.80	72.70	76.90	67.90	4.13
M^{SYN}	79.10	81.60	76.20	69.30	69.40	69.20	4.47
Early Fusion							
$E(M^{LEX}, M^{SYN})$	78.70	81.11	76.10	74.00	76.66	71.11	4.46
Cross Early Fusion							
$X(S^{LEX}, M^{LEX})$	79.20	82.30	75.70	76.20	79.60	72.50	3.63
$X(S^{LEX}, M^{SYN})$	78.30	80.90	75.30	74.60	75.10	73.90	3.08
$X(S^{SYN}, M^{LEX})$	78.60	80.90	76.10	78.90	80.70	76.90	1.08
$X(S^{SYN}, M^{SYN})$	78.90	81.40	76.10	73.70	77.70	70.00	2.72
Cross Late Fusion							
$X(S^{SYN}, S^{LEX})$	78.70	80.90	76.20	78.90	80.80	76.80	1.01
$X(S^{LEX}, S^{SYN})$	78.80	80.90	76.06	78.70	80.50	76.80	1.33
Cross Late Early Fusion							
$X(X(S^{LEX}, S^{SYN}), M^{LEX})$	78.40	80.40	76.10	70.00	68.70	71.40	3.11
$X(X(S^{LEX}, S^{SYN}), M^{SYN})$	78.90	81.80	75.60	75.20	77.40	72.80	3.16
Early Cross Early Fusion							
$E(M^{LEX}, X(S^{LEX}, M^{LEX}))$	79.20	82.40	75.70	76.00	79.50	72.10	3.57
$E(M^{SYN}, X(S^{LEX}, M^{LEX}))$	78.30	80.50	75.80	75.20	75.40	75.00	1.95
Late Cross Early Fusion							
$L(M^{SYN}, X(S^{LEX}, M^{SYN}))$	78.60	81.10	75.80	67.80	71.40	63.80	4.22
$L(M^{LEX}, X(S^{LEX}, M^{LEX}))$	79.50	82.80	75.70	76.09	79.10	72.70	3.96
Early Late Cross Early Fusion							
$E(M^{LEX}, L(M^{SYN}, X(S^{LEX}, M^{SYN})))$	78.50	81.40	75.40	74.20	78.20	69.80	4.26
$E(M^{LEX}, L(M^{LEX}, X(S^{LEX}, M^{LEX})))$	79.50	82.70	75.90	75.80	78.50	72.70	3.99

thus this baseline system would have an average of 1 sense (cluster) per word. A system that goes near this average may be indeed not resolving the task efficiently but finding the MFS trivial solution. Consequently, to show that we do not fall in the MFS solution, we display in our results the average number of clusters.

6.4.2.5 Results

Word sense induction and disambiguation results are found in Table 6.5. Again, we aim to surpass the baseline of the single features and early fusion. We experimentally set $\beta = 0.90$ and $\gamma = 50$. In this task, in late fusion, when the first matrix is deemed more relevant than the second one, the performance is higher. This may be due to the fact that, in this task, the feature matrices rows contain types (that is, each line represent an unique word), and thus they are more dense, which may entail more noisy data. By reducing the relevance of the second matrix in late fusion, we are effectively attenuating the less important information. Regarding $\gamma = 50$, again due to the denser characteristic of the matrices, there is a larger quantity of true similar words that are useful to project information into another matrix, through cross fusion.

The WSI/WSD results are shown in Table 6.5. In the following paragraph, we will discuss these result all at once. Due to the page limit constraint, we omit certain configurations that do not yield interesting results either by converging to the MFS solution (1 sense found per target word) or because the performance shown by those configurations is not interesting.

Regarding Single Features, M^{LEX} comes on top of M^{SYN} again. Nonetheless, M^{SYN} is much closer in terms of performance, and as expected, it is actually higher with regards to verbs.

On the 1F level, we see that the early fusion technique in this task does not surpass the independent features representation. Our intuition is that the similarities of both matrices seem to be correlated. In cross early fusion, the best result is obtained by $X(S^{LEX}, M^{LEX})$, regarding the unsupervised F-measure. This configuration already beats our baselines, improving both noun and verb results on the unsupervised evaluation, improving the supervised recall of nouns, and staying on the same level considering all words. Also, it produces more senses than the MSF average number of senses (1 sense per target word), which is good but not indicative of results correctness. Regarding cross late fusion, given the average number of clusters produced, it seems that both results converge towards the MFS, therefore we do not consider these results.

Beginning with the fusion recombinations, in level 2F, both cross late early fusions yield average results. In cross early cross early fusion, the early fusion of M^{LEX} with $X(S^{LEX}, M^{LEX})$ yields very similar results than $X(S^{LEX}, M^{LEX})$. The next natural step is to test this fusion via a linear combination, with a late fusion. The result obtained confirmed the intuition of enriching a single feature matrix with another weighted-down matrix to improve the performance. Indeed, we consider that $L(M^{LEX}, X(S^{LEX}, M^{LEX}))$ gets the best results in terms of all-words supervised recall and the second best all-words unsupervised F-measure (we do not consider solutions that are too close to the MFS baseline).

We test the same configurations as in NER, within the NF level, to try and improve our results. Nonetheless, in general, they do not overcome the best result found previously. In general, we found that the recombination fusion techniques work in terms of improving the performance of the tasks addressed.

In the following, we discuss how each type of feature affects the performance of individual classes in NER or the senses discovered in WSI/WSD.

6.5 Feature Importance Analysis

6.5.1 WSI/WSD

6.5.2 NER

6.6 Conclusion and Future Work

We presented a comparative study of multimedia fusion techniques applied to two NLP tasks: Named Entity Recognition and Word Sense Induction and Disambiguation. We also proposed new fusion recombinations in order to complement the information contained in the single representation matrices. In order to accomplish this goal, we built upon basic fusion techniques such as early and late fusion, as well as cross media fusion to transfer quality information from one set of features to another.

We found that by taking a strong feature, in our case lexical context, M^{LEX}, and enriching it with the output of rather complex fusion combinations, we can improve the performance of the tasks addressed. The enrichment has to give more relevance to the strong feature matrix, by selecting the right parameters.

While there is an improvement, we do note that fusion techniques augment the complexity of the tasks at hand by enlarging the feature space or making it more dense.

In that sense, more intelligent ways of finding the most appropriate fusion must be researched. This is indeed one of our future work paths: determining an optimal fusion path from single features to a N-degree fusion recombination. Coupled with this, the automatic determination of the parameters is still ongoing research in the multimedia fusion community. Consequently, we believe that efficiently determining both parameters and fusion combinations is the general domain of our future work. Another route we would like to explore is testing these techniques on other tasks and with datasets from different domains, in order to assert its effectiveness.

Chapter 7

Wikipedia Syntactic Dump

Abstract. *In order to have a working dataset, we first built a software that process and parse any input corpus. We describe its properties, its inputs, the information extracted, as well as the output generated by the software.*

Contents

CHAPTER 8

Conclusions

Bibliography

[Agirre 2006] Eneko Agirre, David Martínez, Oier López de Lacalle and Aitor Soroa. *Two Graph-based Algorithms for State-of-the-art WSD.* In Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP '06, pages 585–593, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics. (Cited on pages 14, 15 and 17.)

[Agirre 2007] Eneko Agirre and Aitor Soroa. *Semeval-2007 Task 02: Evaluating Word Sense Induction and Discrimination Systems.* In Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval '07, pages 7–12, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. (Cited on page 36.)

[Agirre 2008] Eneko Agirre and Aitor Soroa. *Using the Multilingual Central Repository for Graph-Based Word Sense Disambiguation.* In Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), Marrakech, Morocco, may 2008. European Language Resources Association (ELRA). (Cited on pages 10, 14 and 17.)

[Agirre 2009] Eneko Agirre and Aitor Soroa. *Personalizing PageRank for Word Sense Disambiguation.* In Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, EACL '09, pages 33–41, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. (Cited on pages 10, 14, 15 and 17.)

[Ah-Pine 2015] Julien Ah-Pine, Gabriela Csurka and Stéphane Clinchant. *Unsupervised Visual and Textual Information Fusion in CBMIR Using Graph-Based Methods.* ACM Trans. Inf. Syst., vol. 33, no. 2, pages 9:1–9:31, 2015. (Cited on pages 26 and 32.)

[Ahn 2010] Yong-Yeol Ahn, James P Bagrow and Sune Lehmann. *Link communities reveal multiscale complexity in networks.* Nature, vol. 466, no. 7307, pages 761–764, 2010. (Cited on page 16.)

[Atrey 2010] Pradeep K. Atrey, M. Anwar Hossain, Abdulmotaleb El-Saddik and Mohan S. Kankanhalli. *Multimodal fusion for multimedia analysis: a survey.* Multimedia Syst., vol. 16, no. 6, pages 345–379, 2010. (Cited on pages 25 and 27.)

[Balasuriya 2009] Dominic Balasuriya, Nicky Ringland, Joel Nothman, Tara Murphy and James R. Curran. *Named Entity Recognition in Wikipedia.* In Proceedings of the 2009 Workshop on The People's Web Meets NLP: Collaboratively Constructed Semantic Resources, People's Web '09, pages 10–18, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. (Cited on pages 30 and 32.)

[Berge 1985] C. Berge. Graphs and hypergraphs. Elsevier, Oxford, UK, UK, 1985. (Cited on page 19.)

[Biemann 2006] Chris Biemann. *Chinese whispers: an efficient graph clustering algorithm and its application to natural language processing problems.* In Proceedings of the first workshop on graph based methods for natural language processing, pages 73–80. Association for Computational Linguistics, 2006. (Cited on page 16.)

[Brin 1998] Sergey Brin and Lawrence Page. *The Anatomy of a Large-scale Hypertextual Web Search Engine.* Comput. Netw. ISDN Syst., vol. 30, no. 1-7, pages 107–117, April 1998. (Cited on page 14.)

[Bronselaer 2013] Antoon Bronselaer and Gabriella Pasi. *An approach to graph-based analysis of textual documents.* In Proceedings of the 8th conference of the European Society for Fuzzy Logic and Technology, EUSFLAT-13, Milano, Italy, September 11-13, 2013, 2013. (Cited on pages 13, 15 and 17.)

[Choudhury 2009a] Monojit Choudhury and Animesh Mukherjee. *The Structure and Dynamics of Linguistic Networks.* In Niloy Ganguly, Andreas Deutsch and Animesh Mukherjee, editors, Dynamics On and Of Complex Networks, Modeling and Simulation in Science, Engineering and Technology, pages 145–166. Birkh?user Boston, 2009. (Cited on page 2.)

[Choudhury 2009b] Monojit Choudhury and Animesh Mukherjee. *The Structure and Dynamics of Linguistic Networks.* In Niloy Ganguly, Andreas Deutsch and Animesh Mukherjee, editors, Dynamics On and Of Complex Networks, Modeling and Simulation in Science, Engineering and Technology, pages 145–166. Birkhäuser Boston, 2009. (Cited on page 9.)

[Clauset 2008] Aaron Clauset, Cristopher Moore and Mark EJ Newman. *Hierarchical structure and the prediction of missing links in networks.* Nature, vol. 453, no. 7191, pages 98–101, 2008. (Cited on page 16.)

[Clinchant 2011] Stéphane Clinchant, Julien Ah-Pine and Gabriela Csurka. *Semantic combination of textual and visual information in multimedia retrieval.* In ICMR, page 44. ACM, 2011. (Cited on pages 26 and 32.)

[Collins 2002] Michael Collins. *Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms.* In Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10, EMNLP '02, pages 1–8, Stroudsburg, PA, USA, 2002. Association for Computational Linguistics. (Cited on page 30.)

[Daume 2006] Harold Charles Daume III. *Practical Structured Learning Techniques for Natural Language Processing.* PhD thesis, Los Angeles, CA, USA, 2006. AAI3337548. (Cited on page 30.)

[de Cruys 2011] Tim Van de Cruys and Marianna Apidianaki. *Latent Semantic Word Sense Induction and Disambiguation.* In ACL, pages 1476–1485. The Association for Computer Linguistics, 2011. (Cited on page 36.)

[Di Marco 2011] Antonio Di Marco and Roberto Navigli. *Clustering Web Search Results with Maximum Spanning Trees.* In Proceedings of the 12th International Conference on Artificial Intelligence Around Man and Beyond, AI*IA'11, pages 201–212, Berlin, Heidelberg, 2011. Springer-Verlag. (Cited on pages 11, 14 and 17.)

[Di Marco 2013] Antonio Di Marco and Roberto Navigli. *Clustering and diversifying web search results with graph-based word sense induction.* Computational Linguistics, vol. 39, no. 3, pages 709–754, 2013. (Cited on pages 14 and 17.)

[Estrada 2005] Ernesto Estrada and Juan A Rodriguez-Velazquez. *Complex networks as hypergraphs.* arXiv preprint physics/0505137, 2005. (Cited on page 11.)

[Firth 1957] J. R. Firth. *A synopsis of linguistic theory 1930-55.* vol. 1952-59, pages 1–32, 1957. (Cited on page 7.)

[Gialampoukidis 2016] Ilias Gialampoukidis, Anastasia Moumtzidou, Dimitris Liparas, Stefanos Vrochidis and Ioannis Kompatsiaris. *A hybrid graph-based and non-linear late fusion approach for multimedia retrieval.* In CBMI, pages 1–6. IEEE, 2016. (Cited on page 32.)

[Goyal 2014] Kartik Goyal and Eduard H. Hovy. *Unsupervised Word Sense Induction using Distributional Statistics.* In COLING, pages 1302–1310. ACL, 2014. (Cited on page 30.)

[Han 2009] Jiawei Han. *Mining Heterogeneous Information Networks by Exploring the Power of Links.* In Discovery Science, volume 5808 of *Lecture Notes in Computer Science*, pages 13–30. Springer Berlin Heidelberg, 2009. (Cited on page 13.)

[Harris 1954] Zellig Harris. *Distributional structure.* Word, vol. 10, no. 23, pages 146–162, 1954. (Cited on page 7.)

[Heintz 2014] Benjamin Heintz and Abhishek Chandra. *Beyond graphs: toward scalable hypergraph analysis systems.* ACM SIGMETRICS Performance Evaluation Review, vol. 41, no. 4, pages 94–97, 2014. (Cited on page 19.)

[Hope 2013a] David Hope and Bill Keller. *MaxMax: a graph-based soft clustering algorithm applied to word sense induction.* In Computational Linguistics and Intelligent Text Processing, pages 368–381. Springer, 2013. (Cited on page 16.)

[Hope 2013b] David Hope and Bill Keller. *UoS: A Graph-Based System for Graded Word Sense Induction.* In Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International

Workshop on Semantic Evaluation (SemEval 2013), pages 689–694, Atlanta, Georgia, USA, June 2013. Association for Computational Linguistics. (Cited on pages 13, 14, 16 and 17.)

[Jurgens 2011] David Jurgens. *Word Sense Induction by Community Detection*. In Proceedings of TextGraphs-6: Graph-based Methods for Natural Language Processing, TextGraphs-6, pages 24–28, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics. (Cited on pages 11, 16 and 17.)

[Kivimäki 2013] Ilkka Kivimäki, Alexander Panchenko, Adrien Dessy, Dries Verdegem, Pascal Francq, Cédrick Fairon, Hugues Bersini and Marco Saerens. *A graph-based approach to skill extraction from text*. Graph-Based Methods for Natural Language Processing, page 79, 2013. (Cited on pages 11, 14, 15 and 17.)

[Klapaftis 2007] Ioannis P. Klapaftis and Suresh Manandhar. *UOY: A Hypergraph Model for Word Sense Induction & Disambiguation*. In Proceedings of the 4th International Workshop on Semantic Evaluations, SemEval '07, pages 414–417, Stroudsburg, PA, USA, 2007. Association for Computational Linguistics. (Cited on pages 11, 15, 17 and 19.)

[Klapaftis 2008] Ioannis P. Klapaftis and Suresh Manandhar. *Word Sense Induction Using Graphs of Collocations*. In Proceedings of the 2008 Conference on ECAI 2008: 18th European Conference on Artificial Intelligence, pages 298–302, Amsterdam, The Netherlands, The Netherlands, 2008. IOS Press. (Cited on pages 10, 11, 16 and 17.)

[Klapaftis 2010] Ioannis P. Klapaftis and Suresh Manandhar. *Word Sense Induction & Disambiguation Using Hierarchical Random Graphs*. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10, pages 745–755, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. (Cited on pages 10, 14, 16 and 17.)

[Levy 2014] Omer Levy and Yoav Goldberg. *Dependency-Based Word Embeddings*. In ACL (2), pages 302–308. The Association for Computer Linguistics, 2014. (Cited on pages 30 and 33.)

[Liu 2011] Haishan Liu, Paea Le Pendu, Ruoming Jin and Dejing Dou. *A Hypergraph-based Method for Discovering Semantically Associated Itemsets*. In Proceedings of the 2011 IEEE 11th International Conference on Data Mining, ICDM '11, pages 398–406, Washington, DC, USA, 2011. IEEE Computer Society. (Cited on pages 12, 15, 16, 17 and 19.)

[Manning 2014] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard and David McClosky. *The Stanford CoreNLP Natural Language Processing Toolkit*. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pages 55–60, 2014. (Cited on page 31.)

[Matuschek 2013] Michael Matuschek and Iryna Gurevych. *Dijkstra-WSA: A Graph-Based Approach to Word Sense Alignment.* Transactions of the Association for Computational Linguistics (TACL), vol. 1, pages 151–164, May 2013. (Cited on pages 10, 14 and 17.)

[Mihalcea 2004] Rada Mihalcea, Paul Tarau and Elizabeth Figa. *PageRank on Semantic Networks, with Application to Word Sense Disambiguation.* In Proceedings of the 20th International Conference on Computational Linguistics, COLING '04, Stroudsburg, PA, USA, 2004. Association for Computational Linguistics. (Cited on pages 10, 15 and 17.)

[Mihalcea 2011] R. Mihalcea and D. Radev. Graphbased natural language processing and information retrieval. Cambridge University Press, 2011. (Cited on pages 7 and 9.)

[Moro 2014] Andrea Moro, Alessandro Raganato and Roberto Navigli. *Entity Linking meets Word Sense Disambiguation: a Unified Approach.* Transactions of the Association for Computational Linguistics (TACL), vol. 2, pages 231–244, 2014. (Cited on pages 10, 15 and 17.)

[Murphy 2012] Kevin P. Murphy. Machine learning: A probabilistic perspective. The MIT Press, 2012. (Cited on page 15.)

[Navigli 2007] Roberto Navigli and Mirella Lapata. *Graph Connectivity Measures for Unsupervised Word Sense Disambiguation.* In Proceedings of the 20th International Joint Conference on Artifical Intelligence, IJCAI'07, pages 1683–1688, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc. (Cited on pages 10, 11, 14, 15 and 17.)

[Navigli 2010] Roberto Navigli and Giuseppe Crisafulli. *Inducing Word Senses to Improve Web Search Result Clustering.* EMNLP '10, pages 116–126, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. (Cited on pages 11, 14 and 17.)

[Panchenko 2017] A. Panchenko, S. Faralli, S. P. Ponzetto and C. Biemann. *Unsupervised Does Not Mean Uninterpretable: The Case for Word Sense Induction and Disambiguation.* In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017). The Association for Computer Linguistics, 2017. (Cited on page 30.)

[Qian 2014] Tao Qian, Donghong JI, Mingyao Zhang, Chong Teng and Congling Xia. *Word Sense Induction Using Lexical Chain based Hypergraph Model.* In Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 1601–1611, Dublin, Ireland, August 2014. Dublin City University and Association for Computational Linguistics. (Cited on pages 11, 16, 17 and 19.)

[Ratinov 2009] Lev-Arie Ratinov and Dan Roth. *Design Challenges and Misconceptions in Named Entity Recognition.* In CoNLL, pages 147–155. ACL, 2009. (Cited on page 31.)

[Saluja 2013] Avneesh Saluja and Jirı Navrátil. *Graph-Based Unsupervised Learning of Word Similarities Using Heterogeneous Feature Types.* Graph-Based Methods for Natural Language Processing, page 29, 2013. (Cited on pages 13, 14, 15 and 17.)

[Sang 2003] Erik F. Tjong Kim Sang and Fien De Meulder. *Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition.* In CoNLL, pages 142–147. ACL, 2003. (Cited on page 31.)

[Schaeffer 2007] Satu Elisa Schaeffer. *Survey: Graph Clustering.* Comput. Sci. Rev., vol. 1, no. 1, pages 27–64, August 2007. (Cited on page 16.)

[Silberer 2010] Carina Silberer and Simone Paolo Ponzetto. *UHD: Cross-lingual Word Sense Disambiguation Using Multilingual Co-occurrence Graphs.* In Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval '10, pages 134–137, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. (Cited on pages 10, 15, 16 and 17.)

[Sinha 2007] Ravi Sinha and Rada Mihalcea. *Unsupervised Graph-basedWord Sense Disambiguation Using Measures of Word Semantic Similarity.* ICSC '07, pages 363–369, Washington, DC, USA, 2007. (Cited on pages 10, 14 and 17.)

[Tsatsaronis 2007] George Tsatsaronis, Michalis Vazirgiannis and Ion Androutsopoulos. *Word Sense Disambiguation with Spreading Activation Networks Generated from Thesauri.* January 2007. (Cited on pages 10, 14, 15 and 17.)

[Véronis 2004] Jean Véronis. *HyperLex: lexical cartography for information retrieval.* Computer Speech & Language, vol. 18, no. 3, pages 223 – 252, 2004. (Cited on pages 11, 15, 16 and 17.)

[Yeh 2009] Eric Yeh, Daniel Ramage, Christopher D. Manning, Eneko Agirre and Aitor Soroa. *WikiWalk: Random Walks on Wikipedia for Semantic Relatedness.* In Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing, TextGraphs-4, pages 41–49, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics. (Cited on pages 10, 15 and 17.)

[Yu 2014] Shoou-I Yu, Lu Jiang, Zexi Mao, Xiaojun Chang, Xingzhong Du, Chuang Gan, Zhenzhong Lan, Zhongwen Xu, Xuanchong Li, Yang Cai*et al. Informedia@ trecvid 2014 med and mer.* In NIST TRECVID Video Retrieval Evaluation Workshop, volume 24, 2014. (Cited on page 27.)