UNIVERSIDADE DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Large-Scale Semantic Relationship Extraction for Information Discovery

David Soares Batista

Supervisor: Doctor Mário Jorge Costa Gaspar da Silva

Thesis approved in public session to obtain the PhD Degree in
Information Systems and Computer Engineering
Jury final classification: Pass with Distinction

Jury

Chairperson: Chairman of the IST Scientific Board

Members of the Committee:

Doctor Mário Jorge Costa Gaspar da Silva
Doctor Paulo Miguel Torres Duarte Quaresma
Doctor Byron Casey Wallace
Doctor David Manuel Martins de Matos

2016

UNIVERSIDADE DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Large-Scale Semantic Relationship Extraction for Information Discovery

David Soares Batista

Supervisor: Doctor Mário Jorge Costa Gaspar da Silva

Thesis approved in public session to obtain the PhD Degree in
Information Systems and Computer Engineering
Jury final classification: Pass with Distinction

Jury

Chairperson: Chairman of the IST Scientific Board

Members of the Committee:

Doctor Mário Jorge Costa Gaspar da Silva, Professor Catedrático do Instituto Superior Técnica da Universidade de Lisboa

Doctor Paulo Miguel Torres Duarte Quaresma, Professor Associado (com agregação) da Escola de Ciências e Tecnologia da Universidade de Évora

Doctor Byron Casey Wallace, Assistant Professor, School of Information, University of Texas at Austin, USA

Doctor David Manuel Martins de Matos, Professor Auxiliar do Instituto Superior Técnico da Universidade de Lisboa

Funding Institutions

Fundação para a Ciência e Tecnologia

2016

Abstract

Semantic relationship extraction (RE) transforms text into structured data in the form of <e_1, *rel*, e_2> triples, where e_1 and e_2 are named-entities, and *rel* is a relationship type. Extracting such triples, involves learning rules to detect and classify relationships from text. Supervised learning techniques are a common approach, but they are demanding of training data, and are hard to scale to large document collections. To achieve scalability, I propose using an on-line classifier, based on the idea of nearest neighbor classification, and leveraging min-hash and locality sensitive hashing for efficiently measuring the similarity between instances. The classifier was evaluated with datasets from three different domains, showing that RE can be performed with high accuracy, using an on-line method based on efficient similarity search. To obtain training data, I propose using a bootstrapping technique for RE, taking a collection of documents and a few seed instances as input. This approach relies on distributional semantics, a method for capturing relations among words based on word co-occurrence statistics. The new bootstrapping approach was compared with a baseline system, also using a bootstrapping approach but relying on TF-IDF vector weights. The results show that the new bootstrapping approach based on distributional semantics outperforms the baseline. The classifier and the bootstrapping approach are combined into a framework for large-scale relationship extraction, requiring little or no human supervision. The proposed framework was empirically evaluated showing that relationship extraction can be efficiently performed in large-text collections.

Keywords

Semantic Relationship Extraction, Min-Hash, Bootstrapping, Distributional Semantics, Word Embeddings

Resumo Alargado

Os processos de extracção de conhecimento analisam grandes volumes de dados com o intuito de, a partir deles, inferir conhecimento. Um tipo de processos, conhecido como KDD (do inglês *Knowledge Discovery in Databases*), permite extrair conhecimento a partir de informação estruturada, que normalmente reside em bases de dados relacionais. Existe, no entanto, uma grande quantidade de informação não estruturada a partir da qual é difícil inferir conhecimento. Por exemplo, nos relatórios internos de empresas, arquivos de jornais e repositórios de artigos científicos, a informação importante encontra-se expressa em linguagem natural.

Transformar grandes colecções de documentos textuais em dados estruturados faz com que seja possível construir bases de conhecimento, que podem depois ser interrogadas. Este processo de construção automática de conhecimento tem aplicações em domínios diversos, como por exemplo, em bioquímica ou jornalismo computacional. Considere-se um exemplo de um bioquímico quer saber que proteínas interagem com uma dada proteína X mas não interagem com uma outra proteína Y. Poderia, para tal, interrogar uma base de conhecimento sobre interacções entre proteínas, construída a partir de um repositório de artigos científicos de onde foram extraídas as relações por análise do texto (Tikk et al., 2010).

Imaginemos ainda que um jornalista pretende investigar quais as localizações visitadas durante as campanhas eleitorais por todos os candidatos nos últimos 10 anos. Analisar manualmente centenas de artigos noticiosos seria um processo custoso, mas usar uma ferramenta para extrair todas as relações entre pessoas e locais tornaria esta tarefa muito mais simples. Além disso, um jornalista poderá estar interessado em analisar um arquivo de notícias para descobrir interacções entre pessoas e organizações.

Em engenharia informática, extracção de informação é a tarefa que trata de identificar e extrair dados a partir de texto, em concreto entidades-mencionadas (i.e., pessoas,

locais, organizações, eventos, proteínas, etc.), os seus atributos, e as possíveis relações semânticas entre elas (Sarawagi, 2008). A extracção de relações semânticas trata de transformar texto em dados estruturados, triplos da forma <e_1, *rel*, e_2>, onde e_1 e e_2 são entidades-mencionadas, e *rel* um tipo de relação. Os triplos representam relações semânticas, e.g.: relações de filiação entre pessoas e organizações, ou a interacção entre proteínas. Estes triplos podem ser organizados em bases de conhecimento representadas como grafos, onde as entidades mencionadas são vértices e as suas relações os arcos. Este tipo de representação permite explorar uma colecção de documentos através de interrogações considerando entidades e/ou relações, em vez de apenas correspondências entre palavras.

Os métodos de extracção de relações semânticas são muitas vezes avaliados e comparados em competições usando dados públicos. A maioria das abordagens actuais aplicam métodos baseados em *kernels* (Shawe-Taylor and Cristianini, 2004) explorando grandes espaços dimensionais. No entanto, os métodos baseados em *kernels* são altamente exigentes em termos de requisitos computacionais, especialmente se for necessário manipular grandes conjuntos de dados de treino. Estes métodos são usados em conjunto com outros algoritmos de aprendizagem, como o das Máquinas de Vectores de Suporte (SVM, do inglês, *Support Vector Machine*) (Cortes and Vapnik, 1995), que resolvem um problema de optimização de complexidade quadrática e que é tipicamente feito *off-line*. Além disso, as SVMs apenas conseguem fazer classificação binária, sendo necessário treinar um classificador diferente para cada tipo de relação a extrair. Este tipo de abordagens tende a não escalar para grandes colecções de documentos. Além disso, sendo um método de aprendizagem supervisionado, necessita de dados de treino, que são muitas vezes difícil de obter.

Uma alternativa, proposta nesta dissertação, para fazer a extracção de relações semânticas propõe, ao invés de aprender um modelo estatístico supervisionado, classificar uma relação semântica procurando pelos exemplos mais similares numa base de dados de exemplos de relações semânticas. Para um dado segmento de texto, contendo duas entidades-mencionadas, que pode ou não representar uma relação semântica, o algoritmo selecciona de uma base de dados os k exemplos mais similares (top-k). De seguida o algoritmo atribui o tipo de relação ao segmento de texto de acordo com o tipo mais frequente de entre os top-k exemplos seleccionados. Este procedimento simula de certa forma um classificador baseado nos vizinhos mais próximos (k-Nearest

Neighbors), onde cada exemplo é ponderado por um peso correspondente ao tipo de relação que representa.

Uma abordagem ingénua ou simples para encontrar os top-k exemplos mais similares, dado um segmento de texto r, numa base de dados contendo N exemplos de relações semânticas, obriga a computar $N \times r$ similaridades. Este tipo de abordagem facilmente constrange o desempenho de um sistema para grandes valores de N. Para garantir a escalabilidade do algoritmo é necessário ultrapassar esta limitação, e reduzir o número de comparações a fazer.

Esta dissertação propõe um classificador *on-line* baseado na ideia de encontrar exemplos similares, tirando partindo da técnica de *min-hash* (Broder et al., 2000) e de *locality sensitive hashing* (Gionis et al., 1999) para calcular de forma eficiente a similaridade entre as relações semânticas (Batista et al., 2013a). O classificador proposto foi avaliado com textos em inglês e português. Para a avaliação em inglês foram usados conjuntos de dados públicos de três domínios diferentes, enquanto que para o português foi criado um conjunto de dados (Batista et al., 2013b) tendo como base a Wikipedia e a DBpedia (Lehmann et al., 2015). Os ensaios de avaliação mostraram que a tarefa de extracção de relações semântica pode ser feita de forma rápida, escalável e com bons resultados usando um classificador *on-line* baseado em procura por similaridade.

Um aspecto crucial nas abordagens de aprendizagem supervisionadas é a necessidade de dados de treino em quantidade e variedade suficiente. Tipicamente, os dados de treino são escassos ou não existentes. O seu processo de criação é custoso, envolvendo uma anotação manual por parte de humanos, normalmente mais do que um. Para o caso específico de treinar um classificador para extrair relações semânticas de frases, é necessário construir um conjunto de dados de treino consistindo de frases onde as entidades e o tipo de relações semântica entre elas está anotado. Uma abordagem possível para gerar dados de treino, sem intervenção humana, consiste em recolher frases expressando relações semânticas por meio de *bootstrapping*. Um sistema de *bootstrapping* aplicado à extracção de relações semânticas recebe como dados de entrada uma colecção de documentos e um conjunto de exemplares de relações semânticas, denominados sementes (Agichtein and Gravano, 2000; Brin, 1999; Pantel and Pennacchiotti, 2006).

Por exemplo, <`Google, Mountain View`> é um exemplar semente para a relação semântica *sede-em*, entre organizações e localizações. O sistema começa por analisar

a colecção de documentos recolhendo todos os contextos (e.g. palavras na vizinhança) onde as entidades, parte de uma semente, ocorrem. Os contextos recolhidos são analisados e agrupados de forma a gerar padrões de extracção. De seguida, a colecção de documentos é novamente analisada, desta vez fazendo uso dos padrões de extracção gerados anteriormente para encontrar novos exemplares do mesmo tipo de relação semântica das sementes. Os novos exemplares, extraídas a partir dos padrões, e, com base num critério de inclusão definido sistema decide se os adiciona ou não ao conjunto de sementes inicial. Este processo retorna ao início e é repetido até um determinado critério de paragem ser observado.

O objectivo do *bootstrapping*, no âmbito da extracção de relações semânticas, é expandir o conjunto de sementes com novos exemplares de relações semânticas, mas limitando a deriva semântica (McIntosh and Curran, 2009), ou seja, evitando o desvio progressivo da semântica das novas relações extraídas da semântica das relações das sementes iniciais.

As técnicas de *bootstrapping* actuais usam vectores com pesos TF-IDF (Salton and Buckley, 1988) como representação do texto correspondente ao contexto de ocorrência das entidades envolvidas numa relação semântica. No entanto, o processo de expansão do conjunto inicial de sementes com este tipo de representação tem limitações. Usando a representação de vectores com pesos TF-IDF, a similaridade entre dois segmentos de texto (i.e., dois vectores) só é positiva se os mesmo segmentos tiverem pelo menos uma palavra em comum. Podem ser aplicadas técnicas de *stemming*, fazendo com que palavras com uma raiz ou origem comum fiquem com a mesma representação (Porter, 1997). No entanto, este tipo de técnicas apenas apenas resolve os casos para palavras cuja a raiz ou origem é a mesma.

A Hipótese Distribucional do Significado (Harris, 1954) afirma que cada língua pode ser descrita em termos da sua estrutura distribucional, i.e., em termos de ocorrências de partes relativas a outras partes. Mais concretamente, se duas palavras co-ocorrem com as mesmas palavras então há uma certa semelhança entre elas. Fazendo uma análise estatística dos contextos de co-ocorrência de palavras diferentes é possível gerar representações vectoriais que capturam as relações entre palavras, designadas por *embeddings* de palavras (Mikolov et al., 2013b).

Compondo *embeddings* de palavras é possível capturar a similaridade entre dois segmentos de texto, mesmo que não existam palavras em comum entre eles. Nesta

dissertação proponho uma abordagem para o *bootstrapping* de relações semânticas baseada na Hipótese Distribucional do Significado, mais concretamente, recorrendo a representações que capturam relações entre palavras com base em estatísticas de co-ocorrência, para representar os contextos de ocorrência das sementes. Este tipo de abordagem permite que na fase de recolha de novos exemplares, usando os padrões de extracção gerados, estes não necessitem de ocorrer em contextos contendo exactamente as mesmas palavras que nos padrões, bastando que ocorram em contextos com palavras semanticamente semelhantes.

Num ensaio de avaliação experimental, a nova abordagem de *bootstrapping* de relações semânticas foi comparada uma base de referência, um sistema também de *bootstrapping* baseado em vectores com pesos TF-IDF. Os resultados mostram que a utilização de representações de contextos baseados em *embeddings* supera o desempenho conseguido pela base de referência (Batista et al., 2015).

O classificador baseado em *min-hash* e a nova abordagem de *bootstrapping* baseada em *embeddings* foram combinados num ambiente para extracção de relações semânticas em larga-escala, que não requer supervisão humana. Neste ambiente, o sistema de *bootstrapping*, recolhe, com base em algumas sementes, exemplos de relações semânticas. Estes exemplos são depois indexados na base de dados do classificador *min-hash*. Após a geração da base de dados de exemplares de relações semânticas dos vários tipos a extrair, o classificador pode, de seguida, extrair vários de tipos de relações semânticas fazendo uma única análise sobre uma colecção de documentos. Este ambiente de extracção de relações semânticas foi avaliado empiricamente mostrando-se que a extracção de relações pode ser feita em larga-escala usando dados de treino gerados por *bootstrapping*.

Palavras Chave

Extracção de Relações Semânticas, *Min-Hash*, *Bootstrapping*, Semântica Distribucional, *Word Embeddings*

Acknowledgments

During the course of the work presented in this thesis I interacted with several people. Each of them contributed, in some way, to the final outcome.

I am extremely grateful to Professor Mário Silva for all the support he gave me during the course of this thesis. Although having a busy schedule, he would always find the time to review and discuss my work. I am also very grateful for the invitation to work with him: he introduced me, in 2008, to the fantastic world of Information Retrieval and Natural Language Processing, and it has been an amazing experience since then.

I am also sincerely grateful for all the help and insights I received from Bruno Martins, for the long time discussions and suggestions which indeed most of the times helped me unblock and attack problems from different perspectives. Obrigado Bruno!

There were several people, during my stay at IST/INESC, whose friendship and support gave me the strength and motivation to carry on. They made me feel better when things went wrong and were always there to cheer me up: Ana Neves, Diana Leitão, Diogo Mera, Edgar Felizardo, Pável Calado, Rui Pinto and Susana Brandão. In some way they are also part of this thesis.

Last but not least, I would like to thank my parents, José and Teresa, for all the opportunities they gave my in life, and for always encouraging me and supporting my decisions.

Contents

List of Figures

List of Figures

List of Tables

List of Tables

List of Algorithms

Acronyms

kNN k-Nearest-Neighbours.

AUC Area Under the Curve.

BREDS Bootstrapping Relationship Extraction with Distributional Semantics.

CBOW Continuous Bag-of-Words.

CRF Conditional Random Fields.

EL Entity Linking.

HAL Hyperspace Analogue to Language.

IE Information Extraction.

KB Knowledge Base.

LSA Latent Semantic Analysis.

LSH Locality-Sensitive Hashing.

LSI Latent Semantic Indexing.

MuSICo MinHash-based Semantic Relationship Classifier.

NE Named-Entity.

NER Named-Entity Recognition.

NLP Natural Language Processing.

NP Noun Phrase.

NPLM Neural Probabilistic Language Model.

OIE Open Information Extraction.

PMI Pointwise Mutual Information.

PoS Part of Speech.

PPMI Proximity Pointwise Mutual Information.

RE Relationship Extraction.

RI Random Indexing.

SLA Semantic Lexicon Acquisition.

SVD Singular Value Decomposition.

SVM Support Vector Machines.

TF-IDF Term Frequency - Inverse Term Frequency.

TOEFL Test of English as a Foreign Language.

TREMoSSo Triples Extraction with Min-Hash and diStributed Semantics.

VSM Vector Space Model.

1

Introduction

Knowledge discovery processes mine large volumes of data to build knowledge. One class of such processes, known as Knowledge Discovery in Databases, mines information stored in a structured form, usually a relational database. There is, however, important information, available in unstructured form, for which it is hard to uncover knowledge automatically. For instance, internal corporate reports, newspaper archives and scientific articles, all hold important information expressed in natural language.

Transforming large collections of unstructured textual documents into structured data makes it possible to construct knowledge bases, which can then be explored through queries and data analyses. Such data can have applications in different domains, such as biochemistry or computational journalism. A researcher in biochemistry who wants to know which proteins interact with some protein X but not with another protein Y, could query a knowledge base of known interactions, built by extracting relationships among proteins documented in the scientific literature.

Suppose that a journalist wants to investigate which locations were visited during election campaigns by all the candidates in the last 10 years. This would require manually analysing hundreds of articles, but having a tool to extract from the articles all the relationships between persons and locations would simplify this task. Moreover,

a journalist might be interested in mining news archives for the discovery of interactions between people and organisations.

Information Extraction concerns the task of automatically extracting structured information from document collections, namely named-entities (i.e., persons, locations, organisations, events), their attributes, and semantic relationships between entities (Sarawagi, 2008). Extracted relationships are represented by triples in the form **<e$_1$, rel, e$_2$>**, where **e$_1$** and **e$_2$** are noun phrases of a relationship, and **rel** is the type of relationship relating the two noun phrases.

Consider, for instance, the following sentence:

The linguist ***Noam Chomsky*** *was born in* ***East Oak Lane*** *neighbourhood of* ***Philadelphia***.

Two semantic relationships can be extracted between the the named-entities (identified in bold):

<**Noam Chomsky**, *place-of-birth*, **East Oak Lane**>
<**East Oak Lane**, *part-of*, **Philadelphia**>

Applying this procedure to extract triples from a large collection of documents, such as a news archive spanning over a period of several years, can generate millions of of triples. These triples hold different relationship types relating named-entities, such as persons, locations and organisations.

The extracted triples can be organised in special knowledge bases. Such knowledge bases can be represented as graphs, where named-entities are vertices and relationships are edges. This representation allows one to explore a document collection through queries considering entities and relationships, instead of simply performing keyword matching. For instance, given the graph shown in Figure 1.1, one could query the knowledge base to return a list of persons that studied at any university located in the city of Cambridge, Massachusetts. The knowledge base would return *Buzz Aldrin*, and *Barack Obama*.

1.1 Large-Scale Relationship Extraction

Relationship Extraction methods are comparatively assessed in evaluation competitions over public data-sets (Airola et al., 2008; Hendrickx et al., 2010). Most state-of-

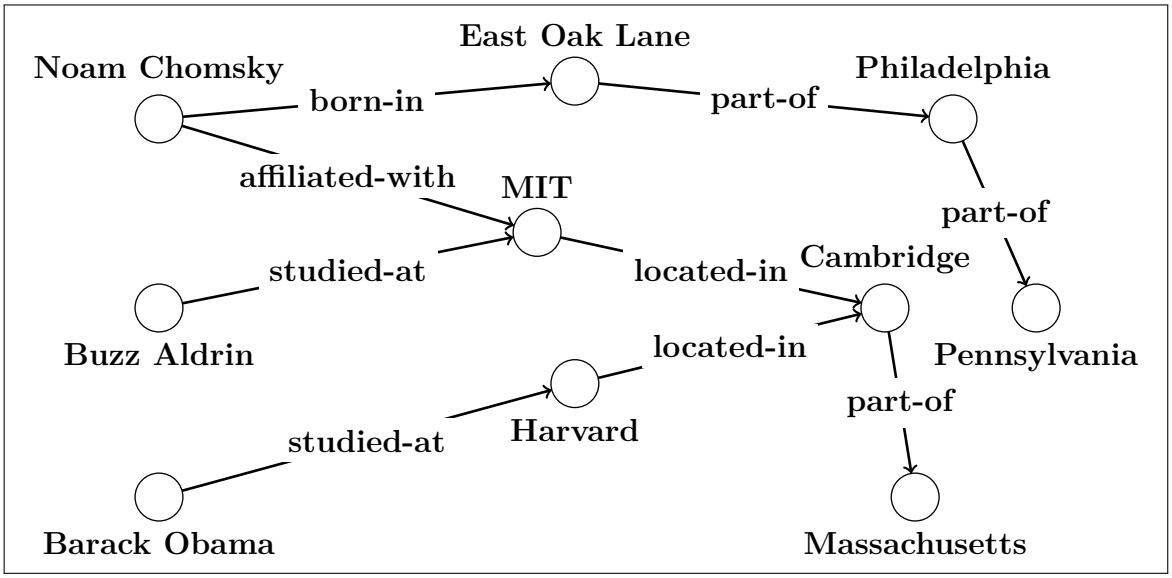


Figure 1.1: A sample of a knowledge-graph connecting relationship triples.

the-art approaches rely on kernel methods (Shawe-Taylor and Cristianini, 2004), which explore large feature spaces without requiring explicit representation of the features. Nonetheless, kernel methods are still highly demanding in terms of computational requirements whenever one needs to manipulate large training data sets. Kernel methods, even if relying only on simple kernels, are typically used together with learning algorithms, such as Support Vector Machines (SVM), proposed by Cortes and Vapnik (1995). The training of SVM is a quadratic programming optimization problem and is typically performed off-line. Moreover, given that SVM can only directly address binary classification problems, it is necessary to train several classifiers (i.e., in a one-versus-one or a one-versus-all strategy) to address multi-class relationship extraction tasks.

This type of approach is hard to scale to large collections of documents (those having a magnitude of tenths of thousands of documents or above). Moreover, these approaches rely on training data to learn extraction models for certain types of semantic relationships, and such data are expensive to obtain.

One alternative approach to perform relationship extraction could be based on the idea of classifying a new relationship by finding the most similar relationship instances from a given database of examples, instead of learning a statistical model. For a new

relationship instance, the algorithm would select, from a database of already indexed relationship instances, the top-k most similar instances. The algorithm would then assign the relationship type, to the instance being classified, according to the most frequent type at the top-k most similar relationship instances. This procedure essentially corresponds to a weighted kNN classifier, where each example instance has a weight corresponding to its similarity with respect to the instance being classified. A naive approach to find the most similar pairs of relationship instances, in a database of size N, given a relationship r, involves computing $N \times r$ similarities, which quickly becomes a bottleneck for large values of N. To overcome this complexity, it is necessary to achieve scalability. It is therefore highly important to devise appropriate pre-processing operations that facilitate relatedness computations quickly.

One bottleneck of supervised approaches for relationship classification is the need of training data, that is, sentences annotated with the semantic relationship that they express. One possible approach to collect sentences expressing a specific semantic relationship with minimal or no human supervision is by bootstrapping. A bootstrapping system starts with a large collection of documents and a few seed instances. A seed instance contains two arguments in a relationship. For instance, <`Google, Mountain View`> is a seed example of a *located-in* relationship, between the organisation Google and the location Mountain View. The document collection is scanned to collect the occurrence contexts (e.g., the sentence, surrounding tokens) of the seed instances. Then, the system analyses the collected contexts and generates extraction patterns. The collection of documents is scanned once again using the extraction patterns to match new relationships instances. These newly extracted instances are then added to the seed set, and the process is repeated, until a certain stop criteria is met. This kind of approach is appealing because it does not rely on manually annotated training data. Instead, only a few seed instances of the relationship type to be extracted are required.

The objective of bootstrapping is thus to expand the seed set with new relationship instances, while limiting the semantic drift, i.e. the progressive deviation of the semantics for the extracted relationships from the semantics of the seed relationships.

State-of-the-art bootstrapping approaches rely on word vector representations with TF-IDF weights (Salton and Buckley, 1988). However, expanding the seed set by relying on TF-IDF representations to find similar instances has limitations, since the similarity between any two relationship instance vectors of TF-IDF weights is only

positive when the instances share at least one term. For instance, the two phrases:

Microsoft *was founded by* **Bill Gates**.

Bill Gates *is the co-founder of* **Microsoft**.

do not have any words in common between the named-entities in bold, but both represent the semantics that a person was the founder of an organisation. Stemming techniques can aid in these cases, but such techniques would only work for variations of the same root word (Porter, 1997).

The *distributional hypothesis* by Harris (1954), states that each language can be described in terms of a distributional structure, i.e., in terms of the occurrence of parts relative to other parts. Firth (1957) explored this idea, based on a word context, popularized by the famous quote *you shall know a word by the company it keeps*. By performing statistical analysis of co-occurrence contexts for different words, one can generate representations that capture these relations among words, i.e., word embeddings. By relying on word embeddings, the similarity of two relational phrases can be captured even if no common words exist. For instance, the word embeddings for *co-founder* and *founded* should be similar, since these words tend to occur in the same contexts.

In this dissertation I propose to apply the distributional hypothesis, in the form of word embeddings, as an alternative to TF-IDF weighted vectors for bootstrapping semantic relationship instances. The word embeddings approach allows the extraction patterns to match new relationship instances, even if they do not occur in contexts with the same exact words as in the extraction patterns. The words in the context and in the extractions patterns just need to be semantically similar.

1.2 Research Questions and Methodology

This dissertation addresses the following research questions:

- *Can supervised large-scale relationship extraction be efficiently performed based on similarity search ?*
- *Can the distributional hypothesis increase the performance of bootstrapping relationship instances ?*

The first question was addressed by exploring an efficient similarity search algorithm and applying it in the context of semantic relationship extraction classification. The second question was addressed by researching existing approaches to induce word representations based on the *distributional hypothesis* and using these representations in bootstrapping semantic relationships from a large collection of documents.

The research strategy consisted of implementing the new algorithms and approaches and evaluating them against state-of-the-art approaches over public datasets.

I have developed and implemented a new algorithm for supervised relationship extraction based on similarity search. The algorithm was evaluated with English and Portuguese datasets. The English datasets consisted of three different document collections, from different domains, which are commonly used as benchmarks for semantic relationship extraction. The Portuguese dataset was derived from DBpedia and Wikipedia, and contains sentences in Portuguese expressing semantic relationship expressions between entities from DBpedia. The experiments with these datasets allowed me to explore different configuration parameters and features of the proposed algorithm, and to compare the performance of my approach against other algorithms, measured in terms of precision, recall and F_1.

I have developed and implemented new bootstrapping approach for discovering new relationship instances from text, relying on the *distributional hypothesis* to generate word vector representations. This approach was evaluated against another bootstrapping approach, which does not rely on the *distributional hypothesis*. For experimental validation, I used a large public collection of documents (Parker et al., 2011). Bootstrapping approaches typically extract a significant number of semantic relationships from a large collection of documents. Such collections are unannotated and therefore evaluating the performance is not straightforward. I used an approach proposed by Bronzi et al. (2012), which allows one to evaluate systems that perform relationship extraction at a large-scale. This evaluation procedure allowed to tune the performance of my approach and compare the results, in terms of precision, recall and F_1, with other bootstrapping approaches.

The two proposed algorithms were then combined in a framework to perform large-scale semantic relationships. The bootstrapping approach collects training data for the classifier. Since the supervised classifier works in an on-line fashion, new relationship examples can be added, even for new relationship types. The classifier extracts different

types of semantic relationships from large collection of documents. This framework requires little or no human supervision, since it relies only on a few seeds for each targeted relationship type to gather training examples. The framework was evaluated in a experiment evolving a collection of 10 million of news articles. The evaluation of this experiment was also done using the approach proposed by Bronzi et al. (2012).

1.3 Results and Contributions

The evaluation experiments of the proposed on-line supervised classifier show that relationship extraction can be performed with high accuracy, using a computationally efficient approach based on similarity search. The proposed algorithm is also fast because, instead of learning a statistical model, it looks only for the most similar relationship examples in a database to classify a new relationship. When measuring the scalability and processing time, I observed that the time taken to process grew linearly with the size of the dataset considered, with most of the processing time spent in feature extraction.

The new semi-supervised bootstrapping approach, relying on word vector representations of the contexts expressing relationships, achieved good results on the experiments, outperforming a baseline system that relies on TF-IDF vector weights. The experiments also shown that the relaxed semantic matching caused by using the word embeddings makes the system learn more extraction patterns and consequently extract more relationship instances.

TREMoSSo, the framework integrating the two new proposed algorithms, was also evaluated through an experiment. The results show that relationship extraction can be performed at large-scale with little or no human supervision.

In addition to the new algorithms for relationship extraction and their assessment, the main contributions resulting of this thesis are:

- A new framework TREMoSSo (Triples Extraction with Min-Hash and diStributed Semantics), which performs large-scale extraction of semantic relationships based on similarity search and the distributional hypothesis, requiring little or no human supervision.

- An annotated dataset of Portuguese relationships which was generated for the

evaluation of the experiments part of this dissertations and which is made publicly available.

1.4 Publications

The work developed and presented in this thesis was originally published in peer-reviewed international conferences and journals. Included and presented in greater detail in this thesis, is the research described in the following publications:

- *Semi-Supervised Bootstrapping of Relationship Extractors with Distributional Semantics*. David S Batista, Bruno Martins, and Mário J Silva. In Empirical Methods in Natural Language Processing-EMNLP 2015. ACL, 2015. (Honorable Mention for Best Short Paper)

- *A Minwise Hashing Method for Addressing Relationship Extraction from Text*. David S Batista, Rui Silva, Bruno Martins, and Mário J Silva. In Web Information Systems Engineering-WISE 2013. Springer Berlin Heidelberg, 2013.

- *Exploring DBpedia and Wikipedia for Portuguese Semantic Relationship Extraction*. David Soares Batista, David Forte, Rui Silva, Bruno Martins, and Mário J. Silva. Linguamática, 5(1), 2013.

1.5 Thesis Outline

The remaining chapters of this dissertation are organised as follows. In Chapter 2, I survey the different techniques used for relationship extraction, including proposed evaluation methods, metrics and public datasets available, in addition to proposing a taxonomy for organising the different approaches.

In Chapter 3, I introduce the *distributional hypothesis* and explain how the theory behind it was exploited to generate rich representations of word vectors. I describe models used to induce word vector representations, based on word-classes, matrix approaches, and based on neural networks.

In Chapter 4, I introduce MuSICo, the proposed on-line classifier to perform relationship extraction based on similarity search. I describe in detail the rationale behind

the construction of the classifier and also present an evaluation of the performance of the classifier over datasets from different domains comparing its performance with other approaches, and an evaluation of its scalability.

In Chapter 5, I describe BREDS, a new semi-supervised bootstrapping approach, which relies on word vector representations induced with distributional semantics and its experimental evaluation. The results show that the performance is better than a baseline obtained with another system using vectors weighted with the TF-IDF schema.

In Chapter 6, I describe the TREMoSSo framework, which integrates MuSICo and BREDS, for relationship extraction requiring little or no human supervision. I also describe an experiment where the framework was used to extract semantic relationships from a large collection of news articles.

In Chapter 7, I summarize the achieved results, drawing conclusions about the performance of the solutions proposed for relationship extraction, the conducted experiments, and how they can provide answers to the research questions of this thesis.

2

Relationship Extraction

This chapter reviews related work in semantic relationship extraction (RE) from textual documents, proposing a taxonomy as a way of organising previously proposed extraction methods. Each approach is characterized, along with a survey of the metrics and public datasets that have been created to evaluate RE methods. The chapter also reviews previous work in relationship extraction for Portuguese.

2.1 A Relationship Extraction Taxonomy

In general, a semantic relationship can be defined among multiple entities (n-ary). Within the scope of this work only binary relationships will be considered, as we can see, without loss of generality an n-ary relationship as a set of binary relationships.

Extracted binary relationships have the structure of a triple **<e$_1$, rel, e$_2$>**, where **e$_1$** and **e$_2$** are named-entities or noun phrases in a sentence from which the relationship is being extracted, and **rel** is a relationship type or class that connects the two other arguments. Depending on the context, the terms *entities*, *named-entities* or *nominals* are also commonly used to refer to the entity arguments of a relationship.

Different techniques have been proposed to tackle the problem of detecting and

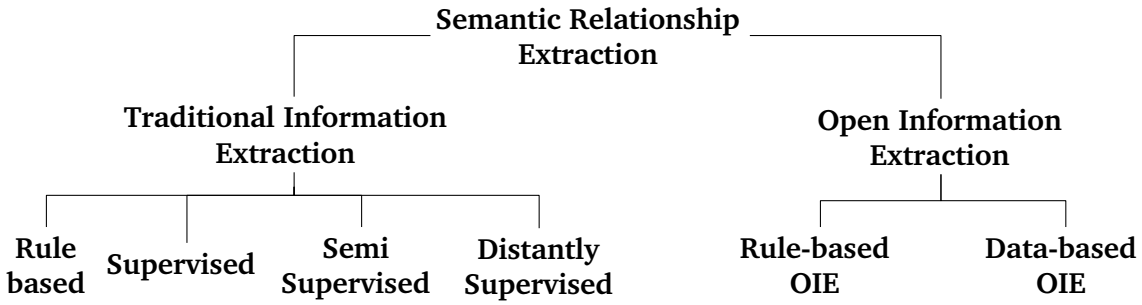


Figure 2.1: A taxonomy of relationship extraction approaches.

extracting relationships from textual documents. Figure 2.1 depicts my proposed taxonomy for organising such techniques. These techniques can be divided in two main branches, Traditional Information Extraction and Open Information Extraction.

Traditional RE techniques extract relationship instances that belong to a predefined set of relationship types. These techniques include:

Rule-based approaches, which usually aim at extracting one type of relationship by relying on manually-crafted rules. These were the first to be devised for extracting relationships from text.

Supervised approaches, which are based on manually annotated documents. For each pair of named-entities in a sentence, a label indicates the type of relationship between the two entities. An annotated collection of documents is used to train classifiers. Then, for any given sentence, a trained classifier can detect the presence of a relationship type.

Semi-Supervised approaches, which make use of known relationships to iteratively extract new relationships. From the textual contexts of seed relationships, the approaches derive patterns, which are used in turn to derive new relationships.

Distantly Supervised approaches use a knowledge base of known relationships to automatically collect large amounts of training data. The collected data is used to train RE classifiers. If a relation is expressed between two entities in a knowledge base, there is a high probability that the same relationship holds for a given sentence where those two same entities are referred. A supervised classifier can then be trained after collecting a large number of these sentences.

Another approach is Open Information Extraction (OIE), introduced by Etzioni et al. (2008). OIE is suited when the target relations are unknown and the textual data is heterogeneous. OIE is mainly directed to perform RE over massive and heterogeneous web corpora which are, in which the relations of interest are unanticipated, and their number can be large. OIE techniques typically make a single pass over a corpus and extract a large set of relational triples without requiring any human input. OIE can be divided into two main categories, data- and rule-based.

Rule-based OIE relies on hand-crafted patterns from PoS-tagged text or rules operating on dependency parse trees.

Data-based OIE generates patterns based on training data represented by means of dependency tree or PoS-tagged text.

Before any RE technique is applied, a pre-processing step exists where the text is analysed to extract features characterizing a relationship. The next section describes how the textual analysis is performed. The following section will then detail each of the methods for RE.

2.2 Feature Extraction with Textual Analysis

Features are informative and discriminative characteristics of a sentence, which facilitate the learning and generalization during the training of classifiers. Features used in RE can be of different types: lexical, syntactic or semantic.

Lexical

Lexical features typically include:

- a sequence of words occurring between the words of the relationship arguments;
- a sequence of words in a limited context before, and after the words of the relationship arguments;

A sequence can be represented as n contiguous sequence of items (e.g., words or characters) from a given text, denoted as n-grams (Suen, 1979).

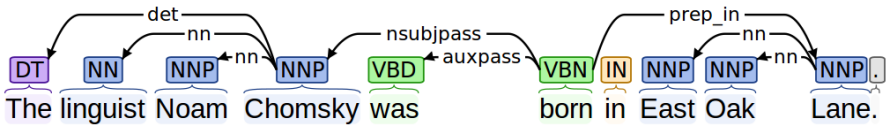


Figure 2.2: A syntactic dependency tree.

Words can also be reduced through lemmatisation or stemming (Porter, 1997), which groups together different inflection forms of a word. For instance, the words *answers*, *answered* and *answering*, all have has root the word *answer*.

Syntactic

Syntactic features are generated through grammatical analysis of the sentences. A process, known as parsing or syntactic analysis, identifies how words group together and relate to each other as heads and dependents (Manning and Schütze, 1999). The result of this process is presented as a tree structure over a sentence, known as a syntactic dependency tree. Figure 2.2 shows the syntactic dependency tree graph for the sentence:

The linguist Noam Chomsky was born in East Oak Lane.

The root node in the tree corresponds to the word 'born'. The leaves represent the words with the corresponding part-of-speech (PoS) tags. Each tag identifies the grammatical category of a word. In the above sentence, the grammatical analysis identifies six different tags:

- a determiner (DT): *The*;
- an adjective (JJ): *linguistic*;
- a proper noun (NNP): *Noam*, *Chomsky*, *East*, *Oak*, *Lane*;
- a verb in the past tense (VBD): *was*;
- a verb in the past participle (VBN): *born*;
- a preposition (IN): *in*;

Other grammatical categories (i.e., tags) exist (Marcus et al., 1993). Petrov et al. (2012) proposed a set of tags common to 22 languages. A PoS-tagging process scans all words in a sentence and assigns a PoS tag to each. Typically, this is a pre-processing step for parsing, which is faster than computing the whole tree with all the syntactic dependencies.

The upper levels of a syntactic dependency tree explain how these words are grouped and related to each other. A noun can be combined together with another noun, forming a noun phrase. For instance, *Noam Chomsky*, might combine with an adjective to form yet another noun phrase, *linguist Noam Chomsky*.

A pair of words is connected by a syntactic dependency, commonly defined as a binary operation that takes as arguments the two related words:

$$dependency(head,\ dependent) \tag{2.1}$$

Dependencies are individualised by labelled links imposing some linguistic conditions on the linked words (Otero, 2008). McDonald et al. (2013) proposed a set of syntactic dependencies common to six different languages.

Another possible parsing tree is a *constituents tree*, where nodes are words grouped into sub-phrases, representing types of phrases (e.g., verb or noun phrases) and the edges are unlabelled.

The Stanford Parser (De Marneffe and Manning, 2008; De Marneffe et al., 2006) generates the following dependencies for the sentence:

The linguist Noam Chomsky was born in East Oak Lane.

- **det**: defines a relation between the head of an noun-phrase and its determiner: *det(Chomsky,The)*;
- **nn**: defines a noun compound modifier, any noun that serves to modify the head noun: *nn(Chomsky,Noam)*;
- **nsubjpass**: a passive nominal subject is a noun phrase which is the syntactic subject of a passive clause. *nsubjpass(Chomsky,born)*;

- **auxpass**: a passive auxiliary of a clause is a non-main verb of the clause which contains the passive information *auxpass(was,born)*;

- **prep_in**: a prepositional modifier of a verb, that serves to modify the meaning of the verb: *prep_in(born,Lane)*;

In a textual analysis, syntactic features typically include:

- the PoS tags associated with each word;

- the syntactic dependency tree between the noun phrases that represent the arguments in the relationship.

Computationally, syntactic parsing is much more expensive than PoS-tagging. Experiments in RE by Wu and Weld (2010) show that tagging a sentence with its PoS-tags can be as much as 30 times faster than syntactic parsing. In another experiment, Akbik and Löser (2012) compare the analysis of 500 sentences by PoS-tagging and by syntactic parsing, and report that PoS-tagging is about 25 times faster.

Semantic

Semantic features correspond to the semantic categories of words or noun phrases in a sentence, representing the semantic arguments of a relationship. For instance, *Noam Chomsky* is a PERSON, *Massachusetts Institute of Technology* in an ORGANISATION, and *Boston* is a LOCATION. This process is known as named-entity recognition (NER), and associates a word or sequence of words to a semantic category (Nadeau and Sekine, 2007).

Semantic features generated from a textual analysis process typically include:

- the semantic type of the noun phrases arguments in a relationship;

- the semantic type associated with other noun phrases occurring in the sentence, which are not the arguments of the relationship.

Example of linguistic features

Given the following sentence, which expresses a relationship between the named-entities **Noam Chomsky** and **East Oak Lane**:

*The linguist **Noam Chomsky** was born in **East Oak Lane**.*

a textual analysis would generate the following features:

- the words: *was*, *born*, *in*;
- the sequence of PoS-tags between the named-entities: verb in past tense (*VBD*), verb past participle (*VBN*), preposition (*IN*);
- the syntactic dependencies: *nsubjpass*, *auxpass*, *prep_in*;
- e_1:PERSON, e_2:LOCATION.

The following section describes how each RE approach makes use of lexical, syntactic and semantic features to detect and extract relationships from textual documents.

2.3 Rule-based

Rule-based methods employ a set of hand-made patterns to extract relationships. These are typically aimed at one specific relationship type.

Hearst (1992) proposed a method to automatically extract the hyponymy relationships (i.e., *is-a*) between two or more noun phrases across a wide range of text. Relationships are identified by patterns, which occur frequently and across different text genres. Examples of such patterns are:

Pattern 1: such NP as {NP,}* { (or | and) } NP

... works by such authors as Herrick, Goldsmith, and Shakespeare.

<Herrick, *is-a*, author>
<Goldsmith, *is-a*, author>
<Shakespeare, *is-a*, author>

Pattern 2: NP {, NP} * {,} or other NP

Bruises, wounds, broken bones or other injuries ...

<bruise, *is-a*, injury>
<wound, *is-a*, injury>
<broken bone, *is-a*, injury>

Pattern 3: NP {,} including {NP ,}* {or |and} NP

Some european countries, including Portugal and France.

<Portugal, *is-a*, european country>
<France, *is-a*, european country>

Muslea (1999) surveys different rule-based systems based on manually created extraction patterns. The patterns are mostly based on information derived from the text where the relationships are extracted from.

2.4 Supervised Methods

In supervised approaches, the extraction of relationships is modelled as a classification task. If labelled data of positive and negative examples of relationships are available, a classifier can be trained. The classifiers are trained after a process of textual analysis, which transforms the sentence into a set of features. Then, given a sentence, where two entities were previously identified, the classifier predicts whether the sentence contains a relationship between the two entities or not.

For each annotated example of a different relationship type, the extracted features $f_1, f_2, ..., f_N$ form an N-dimensional representational vector:

$$x = [f_1, f_2, ..., f_N]$$

Then, supervised learning algorithms learn how to classify each relationship by assigning weights to each feature and combining them effectively.

2.4.1 Logistic Regression

Different supervised learning algorithms have been explored to perform relationship extraction. For instance, in logistic regression (Cox, 1958) the hypothesis has the form expressed in Equation 2.2:

$$h_\theta(x) = \frac{1}{1 + \exp(-\theta^\top x)} \tag{2.2}$$

where x is a vector of features; θ is a vector of weighting parameters, which are learned to minimize a cost function J:

$$J(\theta) = -\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))\right] \tag{2.3}$$

where m is the total number of relationship instances, and $y^{(i)}$ is the true label for the instance x.

Kambhatla (2004) applied a multinomial logistic regression to RE. He explored lexical, syntactic, and semantic features, such as: the semantic types of the arguments, the words between the two arguments in a relationship, and the PoS-tags of the words on which the mentions are dependent in the dependency tree derived from the syntactic parse tree.

2.4.2 Support Vector Machines

Support Vector Machines (SVM) are a popular classification technique (Cortes and Vapnik, 1995). An SVM tries to find a linear hyperplane, in an N-dimensional space, with the largest distance to the nearest instances of positive and negative classes.

Figure 2.3 shows an example of a two dimensional space of points X_1 and X_2. There are, in general, a number of hyperplanes that separate the positive and the negative training data. The SVM algorithm determines the optimal w (i.e., a vector normal to the hyperplane), and b (i.e., bias) such that the corresponding hyperplane separates the positive and negative training data with the maximum margin.

For most real-case scenarios, there is no linear hyperplane which can separate the data. The *soft margin hyperplane* allows the data to be separated with a minimal

number of errors. The method introduces a parameter C, and the optimization becomes a trade-off between a large margin and a small error penalty. The parameter C is selected such that a larger C corresponds to assigning a higher penalty to errors.

It is important to identify which features are good indicators of a relationship and select only these when training a classifier. Feature engineering is a process where features are selected on a trial-and-error basis in order to maximize the performance of a classifier. For some datasets it can be difficult to arrive at an optimal subset of relevant features, or it can be difficult to find a separating hyperplane in the given input space.

Kernel Functions

One way to make the classes linearly separable is by embedding the points (i.e., features values) in a higher-dimensional feature space through a mapping function φ, where a maximal margin separation is possible, as shown in Figure 2.4.

Kernel functions allow the SVM algorithm to operate in a high-dimensional feature space without mapping each vector to that space, by simply computing the inner products between the images of all pairs of data in that feature space (Shawe-Taylor and Cristianini, 2004). More precisely, a kernel function K measures the similarity between two relationship instances, defined as the mapping from $K : X \times X \rightarrow [0, \infty)$, from the input space X to the similarity score, which corresponds to an inner product

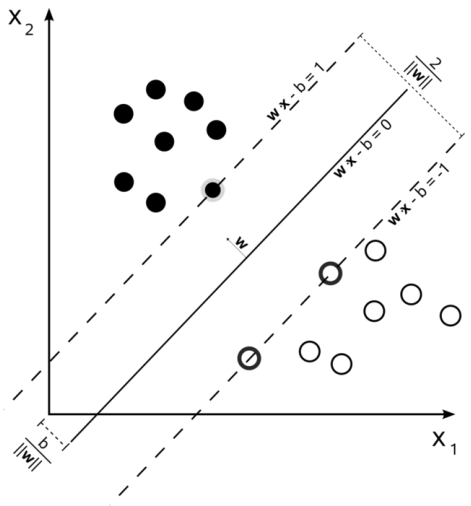


Figure 2.3: A maximum margin separating plane.

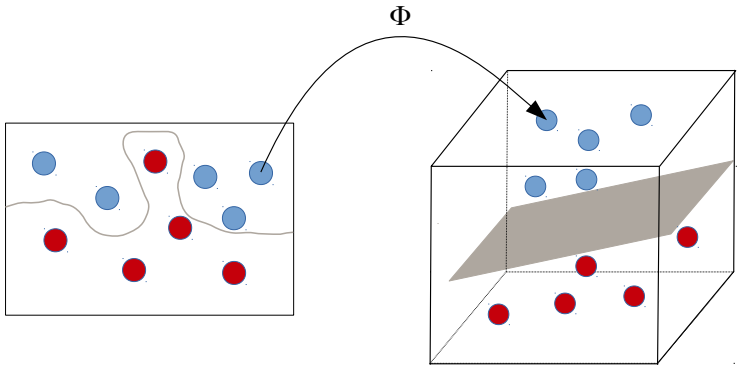


Figure 2.4: Mapping from a 2-D to 3-D space to find a linear separation.

between two vectors instances, in some feature space based on a mapping Φ:

$$K(x, y) = \Phi(x) \cdot \Phi(y) = \sum_{i=0}^{n} \Phi_i(x)\Phi_i(y) \tag{2.4}$$

where $\Phi_i(x)$ is a feature function over the instance x.

The idea behind kernel methods for relationship extraction is to explore exhaustively rich input representations, such as syntactic dependency parse trees, in a higher dimensional space, thus generating a much larger number of features than those given in the input. Different kernels were proposed for the task of semantic relationship extraction. Zelenko et al. (2003) described a kernel inspired in sub-sequence string kernel (Lodhi et al., 2002). The kernel receives as input two objects representing entity-augmented parse tree structures, and computes the similarity between two relationships in terms of a weighted sum of the number of sub-trees that are common between the two. The authors evaluated their approach with SVM and Voted Perceptron (Rosenblatt, 1958).

Culotta and Sorensen (2004) described a generalised version of the previous kernel, based on dependency trees. In their approach, a bag-of-words kernel is also used to compensate for errors in syntactic analysis. Every node of the dependency tree contains extra information like PoS-tags, phrase types (i.e., noun phrase, verb phrase), or entity semantic types.

A further extension is proposed by Zhao and Grishman (2005), using composite kernels to integrate information from different syntactic sources. They incorporate

tokenisation, parsing, and syntactic dependency analysis, so that processing errors occurring at one level may be overcome by information from other levels.

Bunescu and Mooney (2005b) presented another alternative approach, which uses information concentrated in the shortest path along a dependency tree between the two entities. The authors argue that the shortest path between the two nominals encodes sufficient information to infer the semantic relationship between them.

Bunescu and Mooney (2005a) presented a generalised sub-sequence kernel that works with sparse sequences, containing combinations of words and PoS-tags to capture the word-context around nominal expressions. Three sub-sequence kernels are used to compute the similarity between relationship instances at the word level, namely comparing sequences of words occurring (i) before and between, (ii) in the middle, and (iii) between and after the nominal expressions. A combined kernel is simply the sum of all three sub-kernels.

Zhou and Zhang (2007) employ diverse lexical, syntactic and semantic knowledge in feature-based relation extraction using SVM with a linear and a polynomial kernel. Features include: the words between, before and after the arguments of the relationship, the semantic type of the arguments, the full syntactic parse tree and semantic resources such as list of countries and WordNet (Miller, 1995).

Airola et al. (2008) introduced the All-Paths kernel. They use a representation based on a weighted directed graph that consists of two unconnected sub-graphs, one representing the dependency structure of the sentence, and the other representing the sequential ordering of the words.

Other works continue to explore combinations or extensions of the previously described kernel methods (Kim et al., 2010; Nguyen et al., 2009). However, most proposals have been evaluated on different data sets, making it difficult to assess which is better.

2.4.3 Multi-Class Classification

Supervised models for relationship extraction typically learn how to classify a relationship instance into one of K classes, where each class represents a type of relationship.

Some models can be adapted to multi-class classification. For instance, the logistic

regression algorithm introduced in 2.4.1 can be extended to a multi-class scenario. For a given feature vector x of dimensionality N, the model estimates probability that $P(y = k|x)$ for each value of $k = 1, ..., K$. The hypothesis $h_\theta(x)$, will output a K-dimensional vector, whose elements sum to 1, representing the K estimated probabilities, as shown in Equation 2.5:

$$P(y^{(i)} = k|x^{(i)}; \theta) = \frac{\exp(\theta^{(k)\top} x^{(i)})}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x^{(i)})} \tag{2.5}$$

where k is a class, x is a feature vector representing a relationship instance, and θ denotes all the parameters of the model, which are represented by a N by K matrix, obtained by concatenating $\theta(1), \theta(2), ..., \theta(K)$ into columns:

$$\theta = \begin{bmatrix} | & | & | & | \\ \theta^{(1)} & \theta^{(2)} & \dots & \theta^{(K)} \\ | & | & | & | \end{bmatrix}$$

The parameters θ are learned by minimizing the following cost:

$$J(\theta) = -\left[\sum_{i=1}^{m}\sum_{k=1}^{K} 1\left\{y^{(i)} = k\right\} \log \frac{\exp(\theta^{(k)\top} x^{(i)})}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x^{(i)})}\right] \tag{2.6}$$

where m is the number of training instances, K is the number of classes, y^i is the label associated with instance x, and 1 is the indicator function.

For other supervised algorithms like the SVM, which can only learn how to distinguish between two classes, the multi-class classification must be decomposed into multiple binary classifications. There are two typical strategies:

One-versus-All: one model is trained per class, where examples of one class are positive and all other examples from the remaining classes are negative. To classify a new relationship, the model selects the class which reports the highest confidence score.

One-versus-One: one model is trained between every possible pair of classes. Having K classes, $K(K-1)/2$ classifiers are trained. To classify a new relationship, a

voting scheme is applied. All trained classifiers are applied to a new relationship, and the classification with the highest number of predictions is attributed the the new relationship instance.

Aly (2005) presents a survey of supervised algorithms and techniques for solving multi-class classification problems.

Multiple SVM extensions to handle multi-class problems have been proposed, e.g.: Weston and Watkins (1999), Bredensteiner and Bennett (1999), and Crammer and Singer (2002). Nevertheless, these extensions introduce additional constraints that can result in a larger optimization problem, which may be impractical for a large number of classes.

2.4.4 Conditional Random Fields

Conditional Random Fields model the probability of a sequence of labels $y = (y_0, \ldots, y_T)$, given an input sequence of objects (e.g., words) $x = (x_0, \ldots, x_T)$ (Lafferty et al., 2001). The label sequence is modelled as a normalized product of feature functions:

$$p(y|x) = \frac{1}{Z(x)} \exp \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_{t-1}, y_t, x, t) \tag{2.7}$$

where f_k are feature functions, T is the length of the sequence, and $Z(x)$ is a function used to normalize the probabilities to 1. K is the number of feature functions, and λ_k are feature weights. The weights are learned during training, using techniques such as stochastic gradient descent or L-BFGS.

Culotta et al. (2006) model RE as a sequence labelling problem with CRF, and explore implicit relationships. For instance, familial relation patterns: one's sister is likely one's mother's daughter, or a cousin is a father's sister's son. They propose an integrated supervised machine learning method that learns both contextual and relational patterns to extract relationships. In addition to common linguistic features, such as neighbouring words and syntactic information, the work explored features that incorporate relational patterns between entities. The relational patterns are extracted from a graph that connects entities present in Wikipedia.

2.4.5 Deep Learning

A new approach has been applied to perform relationship extraction based on neural networks, called Deep Learning (Bengio et al., 2015; Schmidhuber, 2015).

Machine learning techniques for RE always introduce a textual analysis step which generates features from text. These features are then used to train classifiers, which learn to optimize the weight of each feature. Deep Learning takes a different approach. Instead of feature extraction, each each word is represented as a dense vector of real values in an n-dimensional space, referred to as word embeddings (Mikolov et al., 2013a; Turian et al., 2010).

The challenge of Deep Learning approaches for NLP is how to learn a function to compose these word embedding vectors to build a representation of multi-word units, such as phrases or sentences. Ideally that representation should hold the semantics of the sentence. Linguistic features can be explored and combined with the word embeddings to derive compositionality functions. But there are also approaches which learn the word embeddings specific for a task (Collobert et al., 2011).

The Matrix-Vector Recursive Neural Network (MV-RNN) model learns vector representations for phrases, assigning a vector and a matrix to every node in a syntactic parse tree (Socher et al., 2012). In each node, the vector captures the meaning of the constituent, while the matrix captures how it changes the meaning of neighbouring words or phrases. This approach was applied to relationship extraction by computing the dependency path between the entities whose relationship is to be classified. Then, the highest node in that path is selected and the relationship is classified using that node's vector as features for a classifier.

Hashimoto et al. (2013) followed the same approach but introduced a different composition function, distinguishing words with the same spelling but different PoS-tags and using different weight matrices dependent on the child nodes.

Ebrahimi and Dou (2015) noticed that, compared to constituency-based parse trees, dependency graphs can represent a relationship more compactly. This holds especially for sentences with distant entities, where the parse tree spans words that are not relevant to the relation. They proposed a new compositionality structure to incorporate dependency trees into a neural network based on the shortest path between the entities of a relationship in a dependency graph.

Dataset	Domain	Language	# Relationship Types
ACE 2002 (NIST, 2002)	News	English	24
ACE 2003 (NIST, 2002)	News	English	24
ACE 2004 (Doddington et al., 2004)	News	English	23
AImed (Bunescu and Mooney, 2005a)	Biomedical	English	2
Wikipedia (Culotta et al., 2006)	Wikipedia	English	47
BioInfer (Pyysalo et al., 2007)	Biomedical	English	23
ReRelEM (Freitas et al., 2009)	News	Portuguese	24
SemEval (Hendrickx et al., 2010)	Generic Web	English	19

Table 2.1: Manually annotated datasets for relationship extraction.

2.4.6 Evaluation

Supervised techniques for RE rely on manually annotated datasets for evaluation. These datasets also constitute reference benchmarks for comparisons among different systems. The two main domains of available datasets include news articles and biomedical topics, such as articles about protein interactions. Table 2.1 lists datasets that have been made available to the public.

In a RE evaluation, a dataset can be split in three parts: training, development, and testing. The system to be evaluated is trained on the training set; the development set is used for error analysis and parameter tuning; then, the system is evaluated over the test set. In some evaluations, each dataset is only split into training and test parts. To compare the performance of different systems over the same dataset, we need the metrics typically used in Information Retrieval (IR) such as Precision, Recall, and F_1:

$$\text{Precision} = \frac{\#\text{correctly extracted}}{\#\text{correctly extracted} + \#\text{incorrectly extracted}} \tag{2.8}$$

$$\text{Recall} = \frac{\#\text{correctly extracted}}{\#\text{relationships in the dataset}} \tag{2.9}$$

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.10}$$

Comparing the results and globally evaluating the RE methods described in the literature is not straightforward. Experiments with different kernels are evaluated over different datasets, and some proposed techniques were evaluated on datasets created by their proponents. For instance, in their seminal work, Zelenko et al. (2003) evaluate their sub-sequence kernel on 200 news articles over two types of relationships, *person-affiliation* and *organisation-location*, obtaining F_1 scores of 86.8% and 83.3% respectively. The work by Culotta et al. (2006), based on CRF achieves a F_1 score of 61% over the Wikipedia dataset.

Table 2.2 summarizes the scores achieved in terms of F_1 of the techniques described in previous sections, over public datasets. All the SVM-based approaches that conducted experiments over the ACE datasets used only the major relationship, except for Kambhatla (2004), which trained a multinomial logistic classifier over 24 relationship sub-types. Each major type in the ACE dataset is a relationship that aggregates some of the 24 sub-types. For instance, the relationships *subsidiary-of*, between organisations and *part-of* between locations, are both aggregated into a major *part-of* relationship. An SVM-based approach using only the five major relationship types, instead of the 24 relationship sub-types, substantially reduces the number of models to train in a multi-class scenario. The best approach on the ACE news domain dataset was obtained by Zhao and Grishman (2005), who report an F_1 score of 70.4% over 7 major relationship types.

In the biomedical domain, the All-Paths Kernel by Airola et al. (2008) achieves the best results both in the AIMed and the BioInfer dataset, reporting F_1 scores of 56.0% and 61.3%, respectively.

The SemEval 2010 dataset covers a broader type of relationships drawn from the Web (Hendrickx et al., 2010). Contrary to the other datasets, there is no semantic type associated to the entities in a relationship. The best system achieved an F_1 score of 82% (Rink and Harabagiu, 2010). Almost all participants took an SVM-based approach with different kernels and features derived from external resources such Cyc (Lenat, 1995), WordNet (Miller, 1995), Roget's Taxonomy (Jarmasz and Szpakowicz, 2003), Levin's verb classes (Levin, 1993) or Google's n-gram collection (Michel et al., 2010).

Kernel-based methods achieve the best results, but are typically very demanding in terms of computational requirements, especially for a large number of relationship types or when applied to large document collections. Choi et al. (2013) present an

Technique	ACE 2002	ACE 2003	ACE 2004	Aimed	BioInfer	SemEval 2010
Kambhatla (2004) (Multinomial Logistic Regression)		25.5%				
Culotta and Sorensen (2004) (Sub-Sequence/Bag-of-Words Kernel)	38.0%					
Bunescu and Mooney (2005a) (3 Sub-Sequence Kernel combined)	47.7%		54.0%			
Bunescu and Mooney (2005b) (Shortest Dependency Path Kernel)	52.5%					
Zhao and Grishman (2005) (Composite Kernel)		70.4%				
Airola et al. (2008) (All-Paths Kernel)			56.0%	61.3%		
Rink and Harabagiu (2010) (SVM (SMO algorithm) with 45 features)						82.2%
Socher et al. (2012) (PoS-tags, WordNet, Semantic types)						82.4%
Hashimoto et al. (2013) (Local syntactic features only)						79.4%
Ebrahimi and Dou (2015) (PoS-tags, WordNet, Semantic types)						82.6%

Table 2.2: Performance of supervised systems over different datasets.

extensive report over the kernel methods for RE.

Comparing Deep Learning approaches, Socher et al. (2012) report an F_1 score of 71.9% over the SemEval 2010 dataset without using any external resources, i.e., using only word embeddings, and an F_1 score of 82.4% when also adding PoS-tags, WordNet hypernyms, and named-entity semantic types as features. Hashimoto et al. (2013) achieved an F_1 score of 79.4%, also relying on word embeddings only. Finnaly, Ebrahimi and Dou (2015) achieved an F_1 score of 82.6%, incorporating the same features as Socher et al. (2012), but relying on a syntactic dependencies tree, instead of a constituents tree.

2.5 Semi-Supervised Bootstrapping

Supervised approaches are dependent on labelled data to train classifiers. However, labelled data is not always available, and annotating the needed data can represent a bottleneck in RE processes. Plus, the cost associated with its manual annotation can be prohibitive. On the other hand, unlabelled data is abundant and easily available (e.g., the information available on the Web).

Semi-supervised bootstrapping approaches are appealing because they do not rely on manually annotated training data. Instead, only a few seed instances of the relationship type to be extracted are required. A bootstrapping system for relationship extraction starts with a large collection of documents and a few seed instances. A seed instance contains two entities representing a relationship. For instance, <`Google, Mountain View`> is a seed example of a *located-in* relationship, between an organisation and a location. The document collection is scanned to collect the occurrence contexts (e.g., a sentence, surrounding tokens) of the seed instances. Then, the system analyses the collected contexts and generates extraction patterns. Next, the collection of documents is scanned once again using the extraction patterns to match new relationship instances. These newly extracted instances, are then added to the seed set, and the process is repeated until certain stop criteria are met.

A typical problem with these iterative approaches is semantic drift, the extraction of relationships whose semantics are different from the semantics of the seeds. This is mainly caused by collecting text segments where a seed occurs, but which do not represent the same semantics as the seed instances. For instance, the seed <`Google,`

`Mountain View`> would match contexts like: *Google's headquarters in Mountain View*, or *Google, based in Mountain View*, but could also match *Google's shareholders meeting in Mountain View*, which does not represent a *located-in* relationship. Collecting these erroneous contexts leads to generating extraction patterns that target other relationship types. As this type of errors propagate, the semantics of the extracted relationships rapidly drifts away from the original.

2.5.1 Bootstrapping Semantic Relationships

Several approaches have been proposed to perform relationship extraction based on semi-supervised bootstrapping.

DIPRE

One of the first systems to apply a semi-supervised bootstrapping approach to relationship extraction was the Dual Iterative Pattern Relation Expansion (DIPRE), developed by Brin (1999). DIPRE scans web pages looking for co-occurrences of <author, book> pairs that compose a seed. For each found co-occurrence it creates a tuple of 7 elements:

<author, book, order, url, prefix, suffix, middle>

where *order*, a Boolean, is true if *author* occurs before *book* and false otherwise; *url* is the URL of the document where the seed instance occurred. The *prefix* and *suffix* contain a context window of 10 characters to the left and right of the matched entities and *middle* represents the text between *author* and *book*. To generate extraction patterns, DIPRE uses the 7-element tuples created for each occurrence found. An extraction pattern *outpattern*, has the following structure:

<url, prefix, suffix, middle>

Tuples are grouped by matching *order* and *middle*. DIPRE verifies whether the *order* and *middle* are the same. Then, it sets *outpattern.prefix* to the longest matching suffix of the *prefix*'s of the occurrences. Similarly, it sets *outpattern.suffix* to the longest matching prefix of the *suffix*'s from all the occurrences. These patterns are further

generalized by introducing regular expressions. The new learned patterns are then used to search the corpus again and extract new <name, is-author-of, book> relationships.

DIPRE controls drifting by implementing a simple mechanism to avoid generating too general patterns, which can then extract relationship instances which do not represent an <name, is-author-of, book> relationship. This is based on estimating the specificity of a pattern:

$$\text{specificity}(p) = |p.middle| \cdot |p.urlprefix| \cdot |p.prefix| \cdot |p.suffix| \tag{2.11}$$

A pattern p is rejected if the specificity$(p) \times n > t$, where n is the number of books with occurrences supporting the pattern p and t is a threshold.

Snowball

Snowball relies on bootstrapping to extract relationship instances (Agichtein and Gravano, 2000; Yu and Agichtein, 2003). It follows DIPRE's approach of collecting three contexts for each seed occurrence, generating the tuple:

$$\text{<BEF, } e_1\text{, BET, } e_2\text{, AFT>}$$

where BEF contains the words occurring before the first entity, BET the words between the two entities, and AFT the words after the second entity. Each context is represented by a vector with the TF-IDF weighting schema (Salton and Buckley, 1988). The contexts are then clustered by a single-pass clustering algorithm, using the cosine similarity between the vectors representing the contexts as a similarity metric:

$$\begin{aligned} \text{Sim}(C_i, C_j) = \alpha \cdot \cos(BEF_i, BEF_j) \\ + \beta \cdot \cos(BET_i, BET_j) \\ + \gamma \cdot \cos(AFT_i, AFT_j) \end{aligned} \tag{2.12}$$

where constants α, β, γ weight each vector. Each resulting cluster contains several textual occurrences of seeds represented by three textual contexts. An extraction

pattern is generated from each cluster by computing the centroids for each context (i.e., BEF, BET, AFT) from all the occurrences.

The document collection is scanned once again, and, for each segment of text where two named-entities with the same semantic types as the seeds occur, a <BEF, e_1, BET, e_2, AFT> tuple is instantiated. This process requires the previous identification of the named-entities in the text.

Then, the similarity between the instantiated tuples and each extraction pattern is computed. If the score is greater than a threshold τ_{sim}, the instance is extracted.

Snowball ranks the learned patterns and extracted instances as a way to control the semantic drift. A pattern is ranked according to the instances it extracted. If an extracted instance contains an entity e_1, which is part of an instance in the seed set, and the associated entity e_2 is the same as in the seed set, the extraction is considered positive (i.e., *Positive*). If the relationship contradicts a relationship in the seed set (i.e., e_2 does not match), the extraction is considered negative (i.e., *Negative*). If the relationship is not part of the seed set, the extraction is considered unknown (i.e., *Unknown*). Each pattern p is scored:

$$\text{Conf}(p) = \text{Positive} \times \frac{\text{Positive}}{\text{Positive} + \text{Negative} \cdot W_{ngt} + \text{Unknown} \cdot W_{unk}} \tag{2.13}$$

where W_{ngt} and W_{unk} weight the negative and unknown extractions, respectively. The confidence of a relationship instance is calculated based on the similarity scores with the pattern that extracted it, weighted by the pattern's confidence:

$$\text{Conf}(i) = 1 - \prod_{i=0}^{|P|}(1 - \text{Conf}(P_i) \times \text{Sim}(C_i, P_i)) \tag{2.14}$$

where P is the set of patterns that extracted i, and C_i is the segment of text where i occurred. Instances with a confidence above a threshold τ_t are used as seeds in the next iteration. After the first iteration, Snowball updates the confidence score of each pattern, taking into consideration the confidence score in the previous iteration:

$$\text{Conf}(P) = \text{Conf}_{\text{new}}(P) \times W_{updt} + \text{Conf}_{\text{old}}(P) \times (1 - W_{updt}) \tag{2.15}$$

where W_{updt} is a weight term for the confidence values. When W_{updt} is greater than 0.5, more weight is given to the new confidence score.

Espresso

Espresso, developed by Pantel and Pennacchiotti (2006) is another bootstrapping system that relies on the similarity between extraction patterns and instances to control semantic drift. As with any other bootstrapping system, Espresso starts with a set of seed instances and collects text segments containing the seed terms. Expresso generates extraction patterns by applying a pattern learning algorithm proposed for question answering (Ravichandran and Hovy, 2002) to the collected sentences.

Two functions rank patterns (r_π) and instances (r_l) based on point-wise mutual information (PMI) (Church and Hanks, 1990). Both are recursively defined:

$$r_\pi(p) = \frac{\sum_{i \in I} \left(\frac{\text{PMI}(i,p)}{\max_{\text{pmi}}} \times r_l(i) \right)}{|I|} \tag{2.16}$$

$$r_l(i) = \frac{\sum_{i \in I} \left(\frac{\text{PMI}(i,p)}{\max_{\text{pmi}}} \times r_\pi(p) \right)}{|P|} \tag{2.17}$$

where the reliability of each manually supplied seed instance is $r_l = 1$; $\max_{\text{pmi}}$ is the maximum PMI between all patterns and instances, $|P|$ is the set of all patterns, and $|I|$ the set of all extracted instances. The PMI between an instance i with arguments e_1, e_2 and a pattern p is defined as:

$$\text{PMI}(i,p) = \log \frac{|e_1, p, e_2|}{|e_1, *, e_2| \times |*, p, *|} \tag{2.18}$$

where $|e_1, p, e_2|$ is the frequency of pattern p instantiated with e_1 and e_2, the wildcard (i.e., *) represents any possible contexts. The reliability of an instance, r_l, is the average PMI with each pattern, weighted by the reliability of each pattern.

Only the top-k patterns are retained for the next iteration, where k is the number of patterns from the previous iteration plus one. Next, the system extracts instances

that match any of the patterns in P and ranks each according to r_l keeping only the top-m instances are kept for the next iteration.

Blohm et al. (2007) et al. evaluated different pattern ranking functions over seven relationship types, comparing the pattern ranking used by Snowball against the PMI based ranking used by Expresso. The ranking used by Snowball outperformed PMI on five out of the seven relationship types. The experiments also showed that PMI-based functions did not significantly outperform a baseline which gives random scores to patterns. Moreover, a simple approach, such as scoring patterns according to the number of distinct seed instances from which they are generated, yields better results than elaborate measures, such as PMI.

2.5.2 Semantic Lexicon Acquisition

Semantic lexicon acquisition (SLA) systems extract concepts or terms and the associated semantic class, such as the semantic class of a proper-noun (i.e., male person, female person, organisation, locations), or biomedical categories for terms found in biomedical journals. SLA can been seen as a particular case of relationship extraction (i.e., *is-a* relationships).

Seminal works relied on co-occurrence statistics to determine the membership of a new term to the semantic class of the seeds (Riloff and Shepherd, 1997; Roark and Charniak, 1998). Mutual bootstrapping introduced the idea of learning not only terms of a particular semantic class, but also extraction patterns for a particular class (Riloff and Jones, 1999).

SLA systems proposed different techniques to deal with semantic drift. Curran et al. (2007) proposed a mutual exclusion bootstrapping algorithm to extract named-entities associated with a semantic class. The algorithm attempts to minimise semantic drift by using multiple bootstrapping instances, each with an exclusive term, and assumes terms have a single sense (i.e., semantic class). Each class is extracted in parallel using separate bootstrapping instances that compete to extract terms and contexts. If more than one class attempts to extract the same term, that term is discarded.

McIntosh and Curran (2009) hypothesized that semantic drift occurs when a candidate term is more similar to recently added terms than to the seed terms or high scored terms, added in the earlier iterations. Given a growing lexicon of size N, L_N, let

$L_{1...n}$, correspond to the first n terms extracted into L, and $L_{(N-m)...N}$ correspond to the last m terms added to L_N. In an iteration, let t be the next candidate term to be added to the lexicon. The drift ratio is defined as the average distributional similarity with the first n extracted terms over the average distributional similarity with the last m extracted terms.

$$\text{drift}(t, n, m) = \frac{\text{sim}(L_{1,...,n}, t)}{\text{sim}(L_{N-m,...N}, t)} \tag{2.19}$$

In their work, the distributional similarity between two terms is calculated using the context tokens around each term, and the weighted Jaccard measure (Curran, 2004). At each iteration, the set of candidate terms to be added to the lexicon is scored and ranked. If the term has zero similarity with the last m terms, but is similar to at least one of the first n terms, the term is selected. If the score is below a specified threshold the term is discarded from the extraction process.

2.5.3 Evaluation

Semi-supervised bootstrapping approaches use the same precision and recall metrics as supervised systems. However, bootstrapping approaches extract relationships from large collections of documents. Due to their size, such collections (i.e., newspapers archives, scientific articles), are typically not annotated, making the calculation of precision and recall hard.

One common approach is to rely on an external knowledge base (KB) to provide the ground-truth. The extracted relationship instances are compared against the known instances of the same relationship type in the KB, which enables computing how many extractions are correct.

Snowball's evaluation of the performance of an extraction over a collection of documents D was based on determining an *Ideal* set, which contains all the tuples that appear in D. The authors used as a knowledge base a structured directory of companies containing organisation-location pairs. They created the *Ideal* set by identifying all organisation names and possible variations in D, and then checked if the headquarters of each organisation were mentioned nearby. They then created the *Join* set, which is a join of the *Ideal* and the *Extracted* set (i.e., the output of the system) on the

organisation as key. For each tuple $< o, l > \in Ideal$, the authors find a matching tuple $< o', l' > \in Extracted$ if $o \simeq o'$ (i.e., allowing organisation name variations) and creating a new tuple $< o, l, l' >$. Precision and Recall are then calculated as follows:

$$\text{Precision} = \frac{\sum_{i=0}^{|Join|}[l_i \simeq l'_i]}{|Join|} \tag{2.20}$$

$$\text{Recall} = \frac{\sum_{i=0}^{|Join|}[l_i \simeq l'_i]}{|Ideal|} \tag{2.21}$$

where $l_i \simeq l'_i$ is equal to 1 if the test value l_i matches the extracted value l'_i, and 0 otherwise.

The authors of Snowball report a precision of 76% and a recall of 45% on a collection of 300,000 news articles, extracting *located-in* relationships between organisations and locations, given only five seed examples of relationships.

The evaluation of Espresso consisted of extracting relationships from two different domains, news articles and chemistry texts. The evaluation was performed over a sample of the output. For each instance, two human judges assigned a score of 1 for correct, 0 for incorrect, and ½ for partially correct. The precision for a given set of instances is the sum of the judges' scores divided by the number of instances.

From the news articles domains, Espresso extracted three relationship types, with the following precision values: *is-a*, 73%; *part-of*, 80%; *succession*, 49%. From the chemistry domain, four relationship types: *is-a*, 85%; *part-of*, 60%; *reaction*, 91%; *production*, 72.5%.

2.6 Distantly Supervised

Another paradigm for relationship extraction relies on a large knowledge base (KB) holding relationships. If a relationship between two entities exists in the KB, then there is a high likelihood that given sentence, from the same domain as the KB, mentioning the two same entities also expresses the same relationship. Collecting sentences or text segments with this procedure generates a large amount of training data, which can be used to train classifiers. For instance, having the following fact from DBpedia (Lehmann et al., 2015):

< **Noam Chomsky**, *affiliated-with*, **MIT** >

and looking for mentions of both entities in a document collection, one can match segments of text like:

... *leftist* **MIT** *professor* **Noam Chomsky** *once wrote* ...

... **Noam Chomsky**, *the* **MIT** *linguistics professor and political activist...*

Then, the same procedure of a supervised approach follows: features are extracted by textual analysis and a classifier can be trained.

Mintz et al. (2009) applied this paradigm using Freebase (Bollacker et al., 2008) as KB, and collected sentences from Wikipedia articles that have pairs of entities expressed in a Freebase relationship. The training was performed with a multi-class logistic regression classifier using syntactic and lexical features. The authors report that syntactic features can be helpful for relationships that are ambiguous or distant in their expression in the sentence. Features from different mentions of the same relationship were combined and used in conjunction.

Hoffmann et al. (2010) developed LUCHS, introducing a new technique: dynamic lexicon features. These lexicons form Boolean features which, along with lexical and dependency parser-based features, were used to train a CRF extractor for each relation. While training a CRF extractor for a given relation, LUCHS uses a corpus of lists to automatically generate a set of semantic lexicons, specific to that relation.

Nguyen and Moschitti (2011) created training data defined in YAGO (Suchanek et al., 2007) and sentences from Wikipedia documents mentioned in Freebase. The training data was used to learn an extractor based on combining a syntactic tree kernel and a polynomial kernel, noting that using both dependency and constituent structures within the combined kernel improve the performance of the system.

The drawback of these approaches are the noisy sentences; sentences that, despite mentioning both entities, do not express the same relationships as in the knowledge base. For instance, the first relationship instance introduced above, < **Noam Chomsky**, *affiliated-with*, **MIT** >, could also match a text segment such as:

... **Noam Chomsky** *had a meeting at the* **MIT** *with ...*

System	Knowledge Base	#Extractors	F_1
Mintz et al. (2009)	Freebase	102	67.6%
Nguyen and Moschitti (2011)	YAGO	52	74.3%
LUCHS	Wikipedia	5,025	61.0%

Table 2.3: F_1 scores for distantly supervised systems.

which does not express the *affiliated-with* relationship. Roth et al. (2013) surveyed and classified approaches to deal with noise reduction in three types: at-least-one (Hoffmann et al., 2011; Riedel et al., 2010), Hierarchical Topic Models (Alfonseca et al., 2012) and Patterns Correlations (Takamatsu et al., 2012)

2.6.1 Evaluation

Distantly supervised systems may be evaluated in two ways. One is to use only part of the relationships in the database during training, and compare the extracted relationship instances against the data that was held out during training. Another is having humans evaluate a sample of the output results. Table 2.3 shows the used KB, the number of relationship extractors learned and the F_1 scores for the systems presented in this section.

The work by Mintz et al. (2009) was pioneer in using semantic KB to collect large amounts of training data. With a logistic classifier the authors report an F_1 score of 67.6% for 102 relationship types. Nguyen and Moschitti (2011) combine a tree kernel and a polynomial kernel, reporting F_1 of 74.29% on 52 relationships.

LUCHS learns a much larger number of extractors compared to the other approaches and introduces dynamic lexicons as features. The authors report an F_1 score of 61% for approximately 5,000 relationship types.

2.7 Rule-based OIE

In the above presented approaches, the relationship types to be extracted are known *a priori*, enabling the manual building of specific patterns based on heuristics, or automatically inferring patterns from training examples. Open Information Extraction (OIE) techniques, on the other hand, extract all possible relationship types from a given

collections of documents. Rule-based OIE techniques rely on hand-crafted heuristics based on textual features, such as PoS-tagged or dependency parse trees.

ReVerb

ReVerb extracts relationships based on a simple constraint: every relational phrase must be either a verb (e.g., *invented*), a verb followed immediately by a preposition (e.g., *located in*), or a verb followed by nouns, adjectives, or adverbs ending in a preposition (e.g., *has atomic weight of*) (Fader et al., 2011). This corresponds to the PoS-tags pattern shown in Figure 2.5.

If there are multiple possible matches for a single verb, the longest possible match is chosen. If the pattern matches multiple adjacent sequences, ReVerb merges them into a single relation phrase.

ReVerb is based on patterns expressed in terms of PoS-tags and noun phrase chunks, making the extraction process fast. During extraction, the system first looks for a matching relational phrase and then for the arguments (e_1, e_2) of the relationship, thus avoiding confusing a noun in the relational phrase for an argument.

The downside of ReVerb is that it can only extract relationships that are mediated by a verb. Moreover, it also fails to capture more complex forms of expressing a relationship. For instance:

- a non-contiguous phrase structure: $\mathbf{e_1}$ *is produced and maintained by* $\mathbf{e_2}$;
- phrasal verbs: $\mathbf{e_1}$ *turned* $\mathbf{e_2}$ *off*;
- when the relational phrase does not occur between the arguments: *the* $\mathbf{e_1}$ *that* $\mathbf{e_2}$ *discovered*, *discovered by* $\mathbf{e_1}$, $\mathbf{e_2}$ ....

R2A2 improves ReVerb with an argument learning component, which identifies the arguments of a relationship (Etzioni et al., 2011). Through experiments, the authors

V | V P | V W* P

V = verb particle? adv?
W = (noun | adj | adv | pron | det)
P = (prep | particle | inf. marker)

Figure 2.5: ReVerb patterns for relationship extraction.

note that most arguments of a relationship fit into a small number of syntactic categories. These categories are then captured by specific patterns based on PoS-tags. The patterns capture noun phrases with prepositional phrases or lists among others.

DepOE

Gamallo et al. (2012) developed DepOE, a multilingual OIE system that first discovers clause constituents in the dependency parse tree, and then applies a set of rules over the clause constituents to extract relational triples. A clause is the smallest part a of sentence with a coherent piece of information, consisting of one subject, one verb and, optionally, an indirect object, a direct object, a complement, and one or more adverbials phrases.

DepOE relies on DepPattern (Otero and González, 2012), a dependency parser, to compute the syntactic dependencies of a sentence. Based on the dependencies, the system identifies the clause constituents, concretely, the verb and its dependency constituents: *subject, direct object, verb propositions, or attributes*. DepOE selects the dependent lemma of the clause verb and then lists all the dependent lemmas linked to the target lemma (as a head) through the syntactic dependency path. This results in the constituents of the clause, including information about the head of the phrase. The clause constituents and the verb phrase of each clause are the input for a set of extraction rules, extracting one triple per clause.

ClausIE

ClausIE, developed by Del Corro and Gemulla (2013), reasons over the information given by a dependency parser to extract relationships. After detecting a clause, ClausIE identifies the clause type and the verb type using two insights. First, only seven combinations of the clause constituents appear in the English language. Once the clause type is identified, an extraction rule can be applied. The second insight is that each occurrence of a verb in an English sentence can be classified into five types: intransitive, copular, monotransitive, ditransitive, complex transitive.

The verb type along with the presence of a direct object, indirect object or a complement, uniquely identifies the type of clause. Conversely, the verb type is uniquely determined by the type of the constituents and the type of the clause. ClausIE uses

these observations to detect the clause type. It then applies rules specific to each clause to extract relationships.

2.8 Data-based OIE

Data-based techniques for OIE generate extraction patterns based on training data. The extraction patterns are represented by PoS-tags or by syntactic dependencies.

TextRunner

TextRunner which was used by Etzioni et al. (2008) to introduce the paradigm of OIE, generates a model in two phases, capable of extracting generic relationships. In the first phase, a syntactic parser is applied to several thousand sentences, generating the corresponding syntactic dependencies. For each parsed sentence, TextRunner applies a set of heuristic constraints to label the sentence as a positive example of a relationship, for instance:

- the dependency path between e_1 and e_2 must be no longer than a certain length;
- the path from e_1 to e_2 along the dependency path does not contain relative clauses;
- neither e_1 nor e_2 consist solely of a pronoun.

The sentences for which the constraints fail are labelled as negative. In the second phase, the labelled sentences are mapped into a feature vector, with domain-independent features that can be evaluated at extraction time without the use of a parser. Examples of included features are:

- the sequence of PoS tags between e_1 and e_2;
- the PoS tag to the left of e_1;
- the PoS tag to the right of e_2.

The features are used to train a Naïve Bayes classifier. O-CRF (Banko and Etzioni, 2008) follows the exact same approach as TextRunner, but the RE is seen as a label sequencing problem, training a CRF model instead.

Wikipedia-based Open Extractor

The Wikipedia-based Open Extractor (WOE) developed by Wu and Weld (2010) exploits Wikipedia infoboxes to collect examples of relationships. An infobox is a structured template associated with an Wikipedia article, holding information about the subject which the article describes.

WOE gathers sentences by matching attribute values from the infobox with sentences in the text, keeping only sentences that contain references to both the subject of the article and attribute values from the infobox. The system has two variants, one working with PoS-tags (WOE^{PoS}), and another with syntactic dependencies ($\text{WOE}^{\text{Parse}}$). For each sentence, $\text{WOE}^{\text{Parse}}$ computes the shortest dependency path connecting the subject and the attribute value. This process results in a set of patterns represented as syntactic dependencies, which are scored based on their frequency.

WOE extracts relationships based on these patterns, scoring the relationships according to the pattern that extracted it. WOE^{PoS} extracts PoS-tags and a trains a Conditional Random Field (CRF) to extract relationships.

WOE can also be considered a distantly supervised system, since it uses a knowledge base to gather training data. Nevertheless, WOE aims at extracting generic relationship instances, whereas supervised systems aim at a large but closed set of relationship types.

OLLIE

OLLIE, developed by Mausam et al. (2012), learns extraction patterns by combining information taken from triples extracted with ReVerb and syntactic dependencies trees. This enables mapping the syntactic dependency tree to an extraction pattern that identifies the arguments of the relationship and the relation phrase, that is the sequence of words that captures the relationship.

OLLIE first collects sentences from a corpus containing words that are part of a ReVerb triple, including variations of the verb. For instance, given the triple <Paul Annacone; is the coach of; Federer> this would include sentences such as *Now coached by Annacone, Federer is winning more titles than ever.*

For each sentence, OLLIE computes the syntactic dependencies connecting the two relationship arguments and the relational word. Next, it annotates the relation node in the syntactic dependency path with the exact relation word and the PoS-tag, taken

from the ReVerb triple associated with this sentence.

By checking some constraints over the syntactic dependency tree, OLLIE generates *extraction patterns*. For instance, checking if the relational node, in the syntactic dependency tree, is between the arguments of the relationship or if a proposition edge, in the syntactic dependency tree, matches the preposition in the associated relational triple. For patterns that fail to match the constraints, OLLIE generates semantic and lexical patterns. First, it removes the relational word and then aggregates the patterns based on the syntactic structure. Then, the relational word is replaced into a list of words with which the pattern was seen. The *extraction templates* are generated by replacing, in the ReVerb triples associated with each sentence the relational word with *rel*, and by normalizing auxiliary verbs.

In the extraction phase, the system extracts the dependency path for any given sentence and matches it with one of the *extraction patterns*. Then, the associated *extraction templates* are used to identify the arguments of the relationship and the relational word.

2.9 OIE Evaluation

Evaluating OIE systems is not straightforward, mainly due to the fact that the systems produce a diverse and open set of relationships from a large collection of documents. A common approach is based on human judges, no less than two, analysing a sample of the output. Each output triple $<e_1$, rel, $e_2>$ is judged as correct or incorrect according to a common criteria to all judges. The degree of agreement among judges is also measured using a statistical measure of agreement (Cohen, 1960; Fleiss et al., 1971; Krippendorff, 2011). In addition, this process can be performed over the output of several OIE systems, so that a comparative evaluation can be reported. Nevertheless, it is not easy to summarize a global quantitative comparison of each presented OIE system. Different authors perform experiments on different datasets, and use different metrics, such as F_1 scores or the area under the curve (AUC), typically on a precision-recall graph, while others report precision only.

The evaluation of TextRunner consisted on analysing a sample of 400 triples with a confidence score above 0.8, and extrapolating the results to triples extracted from a 9 million Web pages corpus. The authors report 7.8 million well-formed triples (i.e.,

triples which represent relationships), 1 million of which are concrete triples (i.e., the arguments are grounded to real-world entities). Of these 88.1% have been assessed as correct.

O-CRF was compared with TextRunner on a dataset of 500 sentences (Bunescu and Mooney, 2007), where O-CRF achieved an F_1 score of 59.8% and TextRunner 36.6%.

WOE^{Parse} and WOE^{PoS} are compared against TextRunner on 300 randomly selected sentences from three different datasets: news articles, Wikipedia, and the Web. The authors report only comparative results. WOE^{PoS} achieves an F_1 score between 18% and 34% better than TextRunner, while WOE^{Parse} achieves an improvement between 72% and 91% over TextRunner. On average, it takes WOE^{Parse} 0.68 seconds to process a sentence, while for TextRunner and WOE^{Pos}, it only takes 0.022 seconds, showing that WOE^{Parse} is about 30 times slower, due to dependency parsing.

ReVerb was evaluated against WOE^{PoS} and WOE^{Parse} on 500 randomly chosen sentences from the Web. The performance for each system is evaluated in terms of AUC in a precision-recall graphic. The AUC for ReVerb is 30% higher than WOE^{Parse} and more than doubles the AUC for WOE^{PoS}. ReVerb takes 16 minutes to process the 500 sentences, while WOE^{PoS} need 21 minutes, and WOE^{Parse} takes 11 hours, due to dependency parsing.

DepOE was compared against ReVerb on 200 randomly selected sentences from the English Wikipedia. Both the relationship type and the arguments were considered in the evaluation; the authors report a precision of 68% for DepOE and of 52% for ReVerb.

OLLIE was evaluated against ReVerb and WOE^{Parse} on a dataset of 300 sentences from three domains: news articles, Wikipedia and biology textbook. Each system associates a confidence score with an extraction. Ranking the extractions based on confidence, generates a precision-yield curve and each system is evaluated by measuring the area under the precision-yield curve. According to the authors, OLLIE finds 4.4 times more correct extractions than ReVerb and 4.8 times more than WOE^{Parse} at a precision of about 75%. Overall, OLLIE has 2.7 times larger area under the curve than ReVerb and 1.9 times larger than WOE^{Parse}. The good performance of OLLIE is due to extracting relationships where the relation is not expressed between the named-entities, and the ability to handle relationships mediated by nouns and adjectives.

ClausIE was evaluated against TextRunner, WOE^{Parse}, ReVerb and OLLIE over

System	500 Web	200 Wikipedia	200 NYT
ClausIE	1 706 / 2 975	598 / 1 001	696 / 1 303
OLLIE	547 / 1 242	234 / 565	211 / 497
ReVerb	388 / 727	165 / 249	149 / 271
WOE^{Parse}	447 / 1 028		
TextRunner	286 / 798		

Table 2.4: Correct extractions and total number of extractions on a comparative ClausIE evaluation.

three datasets: 500 sentences taken from the Web, 200 from the Wikipedia, and 200 sentences from The New York Times (NYT). Table 2.4 shows the correct extractions and total number of extractions, form each of the three datasets. Although ClausIE outperformed the other systems the authors note that it was also the slowest one, not giving specific details about computation times.

2.10 Relationship Extraction in Portuguese

The methods and techniques presented before were evaluated with experiments considering English textual documents. In this section, I briefly review some of the approaches taken and the developed systems to perform relationship extraction in Portuguese textual documents. A more extended review of research done in semantic relationship extraction for Portuguese can be found in Collovini et al. (2013).

2.10.1 Systems and Approaches

The ReRelEM evaluation task (Freitas et al., 2008), which was held in the scope of the second HAREM event (Mota and Santos, 2008), published a dataset with annotated relationship between named-entities. The dataset considered 24 relationship types. Three systems participated on the task.

The SEI-Geo system recognizes only *part-of* relationships between geographic entities (Chaves, 2008). The system uses hand-crafted patterns based on linguistic features to detect geographic entities in text. Then, pairs of entities are mapped into a geographic ontology and classified as correct if the relationship is already expressed in the ontology. The best run of SEI-Geo achieved an F_1 score of 45% on *part-of* relationship.

The SeRELeP system by Bruckschen et al. (2008) recognizes three different types of relationships, *occurred*, *part-of*, and *identity*. SeRELeP is also based on heuristic rules applied over linguistic and syntactic features generated by PALAVRAS (Bick, 2000). SeRELeP achieved an F_1 score of 31% for the *occurred* relationship, 18% for *part-of*, and 68% for *identity*.

The REMBRANDT system recognizes all the 24 different relationship types in the dataset (Cardoso, 2008, 2012). REMBRANDT uses hand-crafted rules based on linguistic features, relying on DBpedia (Lehmann et al., 2015) and Wikipedia as knowledge bases. The system explores the categories, in-links and out-links of a Wikipedia page associated with an entity for detecting and classifying relationships between named-entities. Considering the 24 relationship types, REMBRANDT achieved an F_1 score of 45%.

All the three systems that participated in the ReRelEM are rule-based. Each uses a set of hand-crafted heuristics to trigger the extraction of different types of relationships. Each system opted for the recognition of different types of relationships, which makes it difficult to draw conclusions about their relative performance. To the best of my knowledge, the ReRelEM dataset has not been reported as used by any other system or relationship extraction experiment. It would be interesting to apply supervised approaches, like kernel methods, which have shown very good results for English. There may be a reason for this. Unlike other datasets, which contain annotated relationships at a sentence level, the ReRelEM dataset contains annotations considering relationships between named-entities within the whole scope of a document. This requires a complex pre-processing step, such as dealing with anaphora references and co-reference resolution.

Oliveira et al. (2010) present a system that extracts five types of relationships: *synonymy*, *hyperonymy*, *part-of*, *cause* and *propose*, between terms modified by adjectives or prepositions. The system is also based on hand-crafted patterns using lexical and syntactic features. The syntactic features are extracted with the OpeNLP toolkit (Morton et al., 2005) trained with a Portuguese Treebank corpus containing syntactic annotations (Afonso et al., 2002). The system was applied to Wikipedia texts, obtaining information that which was later used to build Onto.PT (Oliveira and Gomes, 2014) a lexical network for Portuguese in the fashion of WordNet (Miller, 1995).

García and Gamallo (2011) compared the impact of different features in extracting

occupation relationship instances over Portuguese texts. The authors used a supervised approach, based on distant supervision, following the approach by Mintz et al. (2009) which collects training sentences from Wikipedia infoboxes and texts. Each training sentence is analysed extracting for each word the lemma and the PoS-tag. A syntactic parser computes the syntactic dependencies between words. The training sentences are then used to train an SVM classifier. The experiments show that lexico-syntactic features achieve a higher performance than bags of lemmas and PoS-tags or syntactic dependencies.

DepOE, which was already described in section 2.7, is an OIE system capable of extracting relationships from Portuguese texts, but, as mentioned before, the authors only report experiments for English (Gamallo et al., 2012).

Collovini et al. (2014) proposed a system to extract relational descriptors between named-entities within the organisation domain (i.e., one of the entities in the relationship is an organisation). A relational descriptor is a word or sequence of words that describes a relationship between two named-entities. The proposed system learns a CRF model, with features extracted by PALAVRAS. Among the features considered are: PoS-tags, syntactic dependencies, semantic types associated to the named-entities, and lexical features based on dictionaries of professions and occupations. The author reports an F_1 of 63%.

Souza and Claro (2014) report on developing a supervised OIE approach, similar to ReVerb (Fader et al., 2011), for extracting relational triples from Portuguese texts. They train and evaluate different classifiers with 500 annotated sentences, where positive and negative examples of relationships are labelled. The best F_1 score is of 84% obtained with a C4.5 decision tree classifier (Quinlan, 1993).

2.10.2 Discussion

Table 2.5 shows a comparison of the techniques and approaches used by the systems surveyed in this section. Most of the relationship extraction methods used with Portuguese apply a traditional approach using hand-crafted heuristics that explore different linguistic features. Nevertheless, recent works began to employ supervised techniques and taking OIE approaches.

The systems presented and discussed throughout this chapter were evaluated over

System	Technique	Approach
SEI-Geo (Chaves, 2008) SeRELeP (Bruckschen et al., 2008) REMBRANDT (Cardoso, 2008) Oliveira et al. (2010)	Rule-based	Traditional
Collovini et al. (2014)	Supervised	
DepOE (Gamallo et al., 2012)	Rule-based	OIE
Souza and Claro (2014)	Supervised	

Table 2.5: Comparison of RE systems for Portuguese.

English datasets. However, some are language independent and could be easily adapted to Portuguese. An exception is the ClausIE system, which specifically exploits properties of the English language. To adapt these techniques to Portuguese, one would need to replace their core components that perform textual analysis such as: PoS-tagging, syntactic parsing, NER.

There are software toolkits and annotated datasets that can be useful to perform textual analysis for Portuguese. One valuable resource is the PALAVRAS (Bick, 2000), a syntactic parser, used by some of the approaches presented before. Unfortunately, PALAVRAS is not available as open source software and its use for experiments depends on the author's permission.

Another valuable resource is the CINTIL Corpus (Barreto et al., 2006; Branco and Silva, 2006), which can be used to train classifiers for different NLP tasks. CINTIL contains several syntactic and lexical annotations such as PoS-tags, semantic class of named-entities, and syntactic dependencies (Branco et al., 2012). CINTIL is a public resource, but its use requires buying a license.

The Portuguese Treebank Floresta Sintáctica (Afonso et al., 2002) is a free resource for Portuguese, which can be used to train a syntactic parser or a PoS-tagger. Many other resources can be found in Linguateca (Santos, 2009), which produces and maintains a list of resources and software toolkits to perform NLP tasks for Portuguese.

I believe that if more resources were made publicly available and free for use, it would boost the NLP research for Portuguese. Plus, if a software toolkit to perform NER, PoS-tagging and dependency parsing for Portuguese was available to download and use out-of-the-box this would increase the drive to produce and research text

	#Relationship Types	Learning Data
Rule-based	At least one rule per relationship type	—
Supervised	Depends on the number of annotated relationships types	Annotated sentences with different relationship types
Semi-supervised	A few seed instances per relationship type	A document collection
Distantly Supervised	Depends on the KB	A KB and a document collection
OIE	Extracts all possible relationship types	None or automatically labels data to learn patterns

Table 2.6: Comparison of different techniques for relationship extraction.

mining applications in Portuguese, like RE. An alternative to a software toolkit could be statistical models generated from Portuguese corpus-based resources, which could be used in popular software toolkits such as Python NLTK (Bird et al., 2009), Stanford CoreNLP (Manning et al., 2014) or OpenNLP (Morton et al., 2005).

2.11 Conclusion

This chapter reviewed existing approaches to perform relationship extraction. I proposed a taxonomy as a way to organise the different methods, with two main branches: Traditional RE and Open Information Extraction. Traditional RE techniques extract instances of specific relationship types, while OIE techniques extract all possible relationships. Table 2.6 characterizes the approaches presented in this chapter in terms of the number of distinct relationship types extracted and the use of learning data.

Rule-based techniques require strong linguistic expertise and are practicable if the

goal is to extract just a few specific types of relationships. Rule-based methods typically achieve a high precision but with a low recall.

Regarding supervised classifiers, kernel-methods, combined with SVM and incorporating external resources of knowledge to help the model generalize the extraction patterns, achieve very good results. Also, recent techniques based on Deep Learning for NLP achieve state-of-the art results. Both approaches are computationally demanding. A fundamental component of supervised systems is training data. Even with a good statistical model, the performance of a classifier is always dependent on the training data and good performance requires in general a large number of examples of good quality training data.

Semi-supervised bootstrapping approaches were mainly motivated by the lack of training data. The challenge of this technique is to deal with semantic drift, i.e., the progressive deviation of the semantics of extracted relationships from the semantics of the seed relationships. The techniques to cope with semantic drift involve ranking the extraction patterns and relationship instances as the bootstrapping progresses. Nevertheless, different ranking methods and representations of the extractions patterns and relationship instances can be explored to better differentiate between valid and invalid instances.

Distant Supervision is another technique to alleviate the lack of training datasets. The process involves selecting pairs of entities in a relationship, expressed in some knowledge base (KB), and then collecting sentences where the same two entities co-occur, automatically generating large amounts of training data. The challenge is to select only sentences which truly express the same relationship as in the KB and discard the noise. The process is related with semi-supervised bootstrapping. However, this technique has a global vision of all the sentences where relationships exist, in contrast with an iterative process.

OIE approaches extract all possible relationships and can be catalogued into two dimensions: how the extraction patterns are built, and which features are used to describe these same patterns. Table 2.7 organises the OIE systems described in this chapter in this way.

Extraction patterns can be inferred based on data or manually built, and these patterns can be represented by PoS-tags or by syntactic dependencies. PoS-tags are computationally fast, but only capture local relations between words, only enabling

	PoS-tags	Syntactic Dependencies
Data-based	WOE^{PoS} TextRunner	WOE^{Parse} OLLIE
Rule-based	ReVerb R2A2	ClausIE DepOE

Table 2.7: A comparision of OIE systems.

the extraction of local relationships. Syntactic dependencies capture long distance relations between words in a sentence. Thus, they enable the extraction of long distance relationships between entities. However, computing the syntactic dependencies is much more computationally demanding than PoS-tagging.

To further use the output triples of OIE systems in other applications, for instance knowledge base population, the triples need to be normalised (Kok and Domingos, 2008; Min et al., 2012; Soderland and Mandhani, 2007). In normalisation operations, two problems need to be solved:

Polysemy occurs when the same relational phrase has different meanings. For instance, given the triples <*Euro, currency of, Germany*> and <*authorship, currency of, science*>, although both triples have the same relational phrase, the meaning is different. In the first, the relational phrase indicates the currency of a country, while in the second it is used as metaphor to express a factor of importance in an area.

Synonymy occurs when the same semantic relationship can be expressed by different relational phrases. For instance, in <*Euro, currency used in, Germany*> and <*Dinar, legal tender in, Iraq*>. Although both relations have different relational phrases, they express the same relationship.

Which OIE approach (i.e., data-based or rule-based) performs better is a matter of continuous debate in the research community. Manually devising a set of rules is in some cases enough to achieve good performance (Gamallo et al., 2012; Mausam

et al., 2012). These approaches are appealing since there is no need to gather data and train complex machine learning models. But efficiently devising these rules requires linguistic expertise and language-specific knowledge, and not all rule based approaches can be universally applied to every language.

In contrast, relying on large collections of documents and training statistical models to infer extractions rules requires less linguistic expertise. Also, by performing language-specific accommodations, most machine learning techniques can be adapted to different languages, and many resources for performing syntactic analysis in many languages are available (McDonald et al., 2013; Petrov et al., 2012).

Most of the recent published work for relationship extraction starts to explore Deep Learning approaches. At the core of Deep Learning for NLP are vector based representations, which map words to an n-dimensional space. The next chapter describes the theory behind these representations and how these vectors are generated.

3

Distributional Semantics

This chapter describes some of the existing approaches to inducing a semantic model based on co-occurrences. It starts by introducing the *distributional hypothesis* and then describing methods that exploit it to generate semantic models. The methods fall into three distinct groups: class-based word representations, which are based on word clusters; semantic vector space models, where word vector representations are built from matrices of co-occurrences; and word vectors, induced using neural language models.

3.1 Distributional Hypothesis

The *distributional hypothesis* by Harris (1954), states that each language can be described in terms of a distributional structure, i.e., in terms of the occurrence of parts relative to other parts. Firth (1957) explored this idea, based on a word context, popularised by the famous quote *you shall know a word by the company it keeps*. Later, Rubenstein and Goodenough (1965) have shown that a pair of words is highly synonymous if their contexts show a relatively high amount of overlap.

The idea behind the distributional hypothesis is that there is a correlation between

... with the Les Paul	**guitar**	that Gibson began manufacturing ...
... own solid-body electric	**guitar**	made by Gibson in 1940 or 1941 ...
... acoustic and electric	**guitar**	are used in a variety of musical genres ...
... the classical	**guitar**	is a nylon-string guitar ...

Figure 3.1: Example of possible contexts for the word *guitar*.

distributional similarity and meaning similarity (Sahlgren, 2008). This encouraged different lines of research to estimate the meaning of a word based on its co-occurrence with other words. Note that the hypothesis does not require that the words co-occur with each other, it only requires that words co-occur within the same context. Figure 3.1 gives an example of possible contexts for the word *guitar*. The word *guitar* can be described by words such as: *Les Paul, Gibson, solid-body, electric, acoustic, classic, nylon-string.*

3.2 Class-based Word Representations

A simple technique to induce a distributional semantics model consists of assigning a class to each word, in a way that semantically similar words will belong to the same class. This can be performed by inducing clusters over words which tend to occur in the same contexts.

3.2.1 Brown Clustering

Brown clustering is a bottom-up hierarchical agglomerative clustering algorithm, which generates clusters of semantically similar words, maximizing the mutual information of bi-grams (Brown et al., 1992). The algorithm takes as input a corpus, seen as long sequence of words. Initially, Brown clustering starts with a partition C of $|V|$ clusters, with each distinct word in the vocabulary V assigned to a cluster. Then, it considers all pairs of possible pairwise merges of clusters, and selects the merged pair that maximizes the quality of the current partition C:

$$\text{Quality}(C) = \sum_{i=1}^{V} log\ \text{P}(C(w_i)|C(w_{i-1})) \times \text{P}(w_i|C(w_i)) \tag{3.1}$$

where w_i is a current word, w_{i-1} is the previous word, $C(w_i)$ is the cluster of the current word and $C(w_{i-1})$, is the cluster of the previous word.

The model considers two probabilities, $\mathrm{P}(c|c')$ of a transition to c, given that the previous cluster was c', and $\mathrm{P}(w|c)$, the probability that cluster c generates the word w. Equation 3.1 decomposes into:

$$\mathrm{Quality}(C) = \sum_{c,c'} \mathrm{P}(c,c') \cdot \log \frac{\mathrm{P}(c,c')}{\mathrm{P}(c)\mathrm{P}(c')} \; + \; G \tag{3.2}$$

where the term G is a constant and therefore ignored. The remaining corresponds to the mutual information (MI) of C, which defines $\mathrm{Quality}(C)$. The MI of C is estimated by counting the relative frequencies of unigrams and bi-grams. So, at each iteration, the algorithm merges the clusters a, b having the maximum score:

$$\mathrm{L}(a,b) = \sum_{d \in C'} \mathrm{MI}(a \cup b, d) - \sum_{d \in C} (\mathrm{MI}(a,b) + \mathrm{MI}(a,c)) \tag{3.3}$$

where, $a \cup b$ is the new cluster, resulting from the merge of a with b, C is the current set of clusters, and $C' = C - a, b + a \cup b$ is the set of clusters after merging a with b. This procedure runs iteratively until a pre-determined number of clusters M is reached.

3.3 Semantic Vector Spaces

Semantic Vector Spaces represent each word as a vector, based on the analysis of the co-occurrences of each word in its different contexts. This process involves the following steps:

1. Building a co-occurrence matrix.
2. Weighting the matrix elements.
3. Applying a dimensionality reduction technique.
4. Comparing vectors to estimate relatedness.

Building a Co-Occurrence Matrix

The process starts by creating from a large collection of documents a matrix $M_{W \times C}$, where W is the vocabulary size, that is, the number of distinct words in the collection of documents, and C represents the different contexts where a word or a group of words occurs. The type and size of context may vary. Each row M_w represents a word, and each column M_c is some defined context where w occurs. The Hyperspace Analogue to Language (HAL), proposed by Lund and Burgess (1996), generates a matrix considering every word-word co-occurrence within a context window. Columns and rows represent every word in vocabulary, and each cell represents the summed co-occurrence counts for each word pair. This procedure is sensitive to direction. A row contains co-occurrence information for words appearing before the word in consideration, the column for the same word contains co-occurrence information for words appearing after.

Element Weighting

Each entry $M_{w,c}$ has a score which represents an association between a word and a context. This score can be calculated by different approaches, from a simple count of co-occurrence frequency to more complex weighting schema, like TF-IDF (Jones, 1972), Pointwise Mutual Information (PMI) (Church and Hanks, 1990), among others, as described by Kiela and Clark (2014).

Dimensionality Reduction

The vectors M_w representing a word, typically have high dimensionality and are sparse. Sparsity is due to the fact that the majority of the words in a language occur only in a limited number of contexts, comparing to all possible contexts. On the other hand, only a very small number of words are distributed uniformly by a large number of contexts. This is a particular example of a more general phenomena known as Zipf's law (Zipf, 1949).

Popular methods to reduce the sparsity of the matrix include Latent Semantic Analysis (LSA) (Landauer and Dutnais, 1997), which corresponds to a singular-value decomposition (SVD) (Golub and Kahan, 1965). The SVD eliminates any linear combination and decomposes a matrix M into three matrices: $M = U \Sigma V^T$, where U and V are are orthogonal and have unit length, and Σ is a diagonal matrix with the

singular values of M. A matrix $\hat{M} = U_k \Sigma_k V_k^T$ is the matrix of rank k that best approximates M, formed by the top k singular values, and U_k and T_k the matrices produced by selecting the corresponding columns from U and T.

Random Indexing (RI) incrementally generates a dimensionally-reduced matrix (Kanerva et al., 2000; Sahlgren, 2005). Each context is associated with a unique randomly generated vector. The vector is sparse, with D dimensions, each having three possible values, +1, -1, and 0. Next, context vectors are generated by scanning the text, one word at the time. Each time a word occurs in a context, that context's vector is added to the already collected context vector for the word in question. Words are thus represented by D-dimensional context vectors that are effectively the sum of the word contexts. The D-dimensional random index vectors are *nearly orthogonal*, meaning that a RI-generated matrix $M'_{W \times D}$ is an approximation of the standard co-occurrence matrix $M_{W \times C}$. The corresponding rows are similar or dissimilar to the same degree, but with $D \ll C$.

Comparing Vectors

Each vector row represents a word, and the relatedness between two words can be measured by comparing their vectors. The most popular measure to compare two vectors is probably the cosine of the angle between them (Salton et al., 1975), which corresponds to the inner product of the vectors, after they have been normalised to unit length. Kiela and Clark (2014) survey similar metrics for measuring the relatedness between two words including the Euclidean distance, the Manhattan distance, among others. Any distance measure can be converted to a measure of similarity by inversion or subtraction.

3.4 Language Models

One problem with the vectors generated by matrix-based approaches is the sparsity and their high dimensionality. Vectors will have a large number of dimensions, typically in the order of 10^4 to 10^6, even when applying dimensionality reduction techniques. An alternative are word embeddings, which are words vectors learned by a neural network. Such vectors, typically, have a lower number of dimensions (e.g., between 10^2 and 10^3), comparing to matrix-based approaches, and are dense.

This section starts by introducing n-gram language models and discussing their limitations. Then, it presents a seminal approach to generate language models based on neural networks, which attacked some of the limitations of n-gram language models and introduced the idea of representing a word by a dense vector containing real values. Finally, neural network approaches that focus only on learning word embeddings are discussed.

3.4.1 n-Gram Models

Language models estimate the probability of a given sequence of words. They are typically represented by the conditional probability of the next word in a sequence of n-words, given all the previous ones. The n-gram model assigns probabilities to a sequence of words by factorising the joint likelihood of the sequence into conditional likelihoods of a word given a context of previous words (Manning et al., 2008):

$$\mathrm{P}(w_1, w_2, \ldots, w_m) = \prod_{i=1}^{m} \mathrm{P}(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}) \tag{3.4}$$

This conditional probability can be estimated with counts of n-grams frequency:

$$\mathrm{P}(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}) = \frac{\mathrm{count}(w_{i-(n-1)}, \ldots, w_{i-1}, w_i)}{\mathrm{count}(w_{i-(n-1)}, \ldots, w_{i-1})} \tag{3.5}$$

Traditionally, n-grams of words are used to represent the previous context of a word. A larger context gives a better prediction, but also increases the model complexity and data sparsity. Typically, models take into account trigrams ($n = 3$). Another problem with this approach is how to estimate the probability of sequences which have not been observed in the training data. Proposed solutions include smoothing techniques, for instance by combining smaller contexts of n-grams to model an unseen context (Chen and Goodman, 1996).

3.4.2 Neural Network Language Models

Another approach to generate language models is based on neural networks (McCulloch and Pitts, 1943). In the process of estimating the model parameters, the network

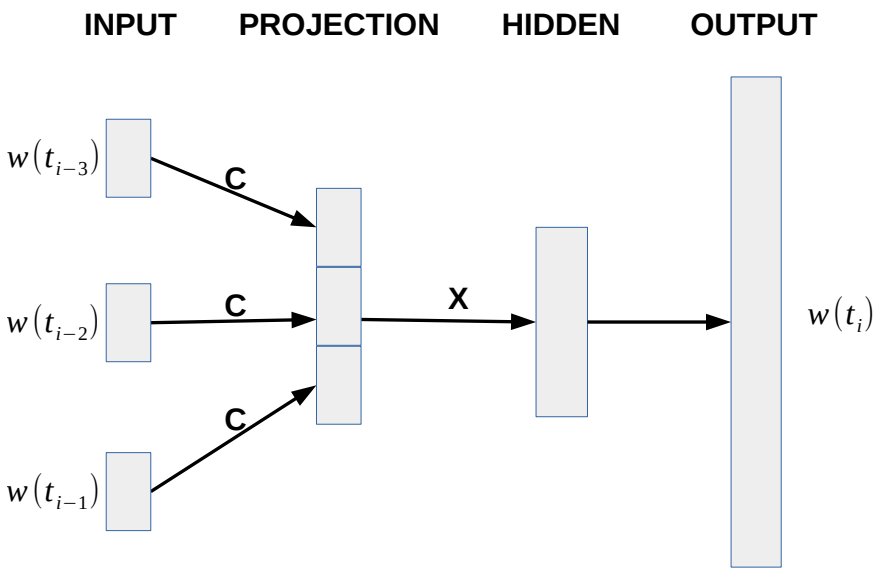


Figure 3.2: The Neural Probabilistic Language Model.

learns word vector representations called word embeddings. These embeddings are the result of projections, which transform sparse and integer-valued vectors to real-valued vectors of a lower dimension.

Neural Probabilistic Language Model

The seminal work of Bengio et al. (2003) proposed the Neural Probabilistic Language Model (NPLM), introducing two major improvements to the n-gram language models: considering larger contexts and exploring the *distributional hypothesis* to generalize to contexts not seen during training.

In the NPLM, the probabilistic prediction of $\mathrm{P}(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1})$ is obtained as follows. Each word w_i, from a vocabulary V, is represented by a real vector $v_i \in \mathbb{R}^m$. A matrix C of dimension $|V| \times m$ represents all the vectors of the vocabulary V, which can be randomly initialized or based on prior knowledge. The parameter m defines the dimension of the vectors.

The model maps an input sequence of words to a conditional probability distribution over all the words in V. The input is a vector X, corresponding to the concatenation of each word vector in the word sequence, and the output is a vector of dimension V whose i-th element is $\mathrm{P}(w_t = i | w_{t-n+1}, \ldots, w_{t-1})$, the probability of w_i being the next

word in the sequence.

The NPLM can be implemented by a feed-forward neural network with linear projection layer, which simply concatenates the input words, a non-linear hidden layer, and a softmax function at the ouput, as shown in Figure 3.2. The softmax function at the output guarantees positive probabilities summing to 1:

$$\mathrm{P}(w_t|w_{t-n+1},\ldots,w_{t-1}) = \frac{\exp(y_{w_t})}{\sum_{i=1}^{|V|}\exp(y_i)} \tag{3.6}$$

where the vectors y_i are the unnormalized log-probabilities for each output word i computed by the hidden layer. The non-linear hidden layer is trained by stochastic gradient descent with backpropagation (Rumelhart et al., 1988), looking for the parameters θ that maximize $L(\theta)$:

$$L(\theta) = \frac{1}{|T|}\sum_{t=1}^{T}\log\ \mathrm{P}(w_t|w_{t-n+1},\ldots,w_{t-1};\theta) \tag{3.7}$$

NPLM jointly learns word vector representations and the language model. Bengio (2008) introduces in greater detail the neural network language model describing the NLPM.

Continuous Skip-Gram and Continuous Bag-of-Words

Mikolov et al. (2009) proposed to train a language model in two steps: first, word vectors representations are learned using a simple model, and then the language model is trained on top of these distributed representations of words.

Inspired by this idea of learning word vector representations detached from the language model, Mikolov et al. (2013a,b) proposed the Skip-Gram model, focusing only on learning word embeddings. The main idea behind this model is to predict the most probable surrounding words in a context window of length c for every given word w in a corpus, for every possible context windows t in a corpus. Formally, to maximize the average log probability of any context word given the current center word:

$$L(\theta) = \frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log\ \mathrm{P}(w_{t+j}|w_t;\theta) \tag{3.8}$$

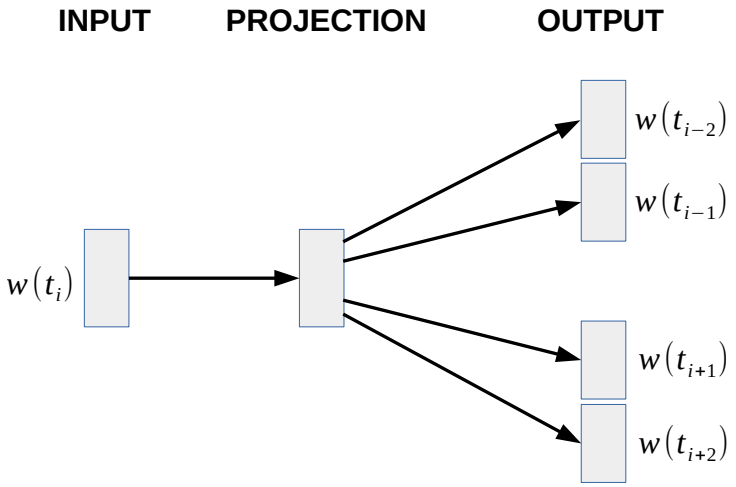


Figure 3.3: The Skip-gram model.

in the formula above, the second summation represents the sum over all the words in contexts windows up to c many words away from the center word, ignoring the center word.

Similar to the Skip-Gram, the Continuous Bag-of-Words (CBOW) model, which sums the outside words and predicts the center word (Mikolov et al., 2013a), instead of predicting the surrounding words given a center word.

Both the Skip-Gram and the CBOW have a single-layer architecture based on the inner product between two word vectors, and both use the context of a word, instead of only the preceding words. Figure 3.3 and Figure 3.4 show the architecture of the Skip-Gram and CBOW the models, respectively. Rong (2014) comprehensively explains the parameter learning process of both models in greater detail.

3.5 Evaluation

The models presented above can be evaluated in two different scenarios. An intrinsic evaluation involves word similarities tasks. An extrinsic evaluation involves altering a system to solve a specific NLP task. A system has its original or word or features representation model replaced or augmented with a new model, then the performance of this new system is compared with the performance of the original system.

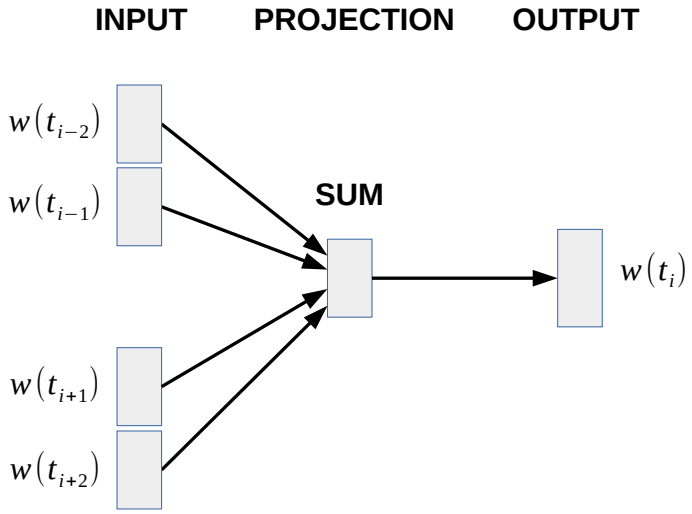


Figure 3.4: The continous bag-of-words model.

Intrinsic

The synonym part of Test of English as a Foreign Language (TOEFL) consists of given a word, choosing from four alternative words the most similar. The LSA and the RI were both evaluated on this dataset. The LSA choices were determined by computing cosines between the vector for the given word and each of the four alternatives, choosing the word with the highest cosine. LSA achieved human-level scores of 64.4% accuracy, and RI achieved similar scores between 64.5% and 67%. This is close to human scores, since a large sample of applicants who took the tests averaged an accuracy of 64.5% on the same questions.

Word embeddings are typically evaluated in terms of semantic and syntactic regularities (Mikolov et al., 2013c), which are captured by different word analogies. For instance, to evaluate how well the embeddings capture plurals, an analogy can be formulated, such as: "the word *apple* is similar to *apples* in the same sense that *car* is similar to *cars*", which can be evaluated by a simple equation:

$$\mathrm{W}(\text{“apple”}) - \mathrm{W}(\text{“apples”}) = \mathrm{W}(\text{“car”}) - \mathrm{W}(\text{“cars”}) \tag{3.9}$$

where $\mathrm{W}(x)$ corresponds to embedding for the word x. The equation is equivalent to:

$$\mathrm{W}(\text{“apple”}) - \mathrm{W}(\text{“apples”}) + \mathrm{W}(\text{“cars”}) = \mathrm{W}(\text{“car”}) \tag{3.10}$$

Model	Semantic	Syntactic	Average
NPLM	23%	53%	38%
CBOW	24%	64%	44%
Skip-gram	55%	59%	52.5%

Table 3.1: NPLM, Skip-Gram and CBOW accuracy in semantic and syntactic regularities.

The vector X = W("apple") + W("apples") - W("car") is computed. Then, the evaluation procedure consists in finding, among all the generated vectors, the one that maximizes the cosine similarity with X. If that vector corresponds to the word "cars" the analogy is considered correct.

Many other types of regularities can be formulated and evaluated. Mikolov et al. (2013a) compared the NPLM, Skip-Gram and CBOW in terms of semantic and syntactic regularities, results in terms of accuracy and averaged accuracy are presented in Table 3.1.

Extrinsic

The most popular method of extrinsic evaluation is probably the incorporation of word clusters as features in NLP supervised learning tasks. The features representing a word class help the algorithm to generalize beyond the training examples used during the learning process. This approach has been applied in many different NLP and IE tasks. For instance, Miller et al. (2004) used word-clusters in a supervised approach for performing NER. The system incorporated word cluster membership as features. Turian et al. (2010) make an extensive evaluation of the impact of incorporating word clusters and word embeddings in NLP systems for the tasks of NER and chunking. The results show that these word representations improve the accuracy of state-of-the-art supervised baselines.

Sun et al. (2011) trained an SVM classifier for relationship extraction, adding word-clusters as additional features. The results show that when combined with certain features, word clusters can improve the performance of the classifier.

Koo et al. (2008) evaluated dependency parsers with lexical features that incorporate word clusters, demonstrating that it achieved substantial improvement over a competitive baseline.

3.6 Conclusion

This chapter introduced existing approaches to capturing the semantics of a word based on the *distributional hypothesis*, i.e., assuming that similar words are used in similar contexts. Three main approaches were described: generating classes of words through clustering, factorization methods of co-occurrences counts, and language models based on neural networks.

Inducing word classes through word clustering was one of the first approaches to represent the semantics of a word, the Brown clustering algorithm being one of the most popular approaches. The grouped words, representing the class of a word, have proven to be useful features in different NLP supervised learning tasks.

Approaches based on co-occurrence matrices, such as SVD and LSA, generate a vector for each word, leveraging on the global statistics of co-occurrence between words and contexts. The generated vectors are somehow interpretable, since each dimension represents a context associated with the word. Nevertheless, the vectors do not capture relationships between words, such as genre or plural, only word similarity. A problem with co-occurrence matrices approaches is sparsity and the huge dimensions of the generated vectors. Also, the process of generating such vectors involves choosing between many different techniques to construct the matrix. Sahlgren (2006), Turney and Pantel (2010), and Kiela and Clark (2014) describe several of these techniques and detail design decisions regarding choice of context types, weighting schemas, and similarity measures.

Approaches based on neural networks learn word vectors trough backpropagation. They do not make use of global co-occurrence statistics. Even if the co-occurrence counts were already computed by another process, neural networks approaches still have to go through every possible context window in a given corpus. Although lower in dimension, when compared with vectors generated from matrix approaches, the dimensions in each word embedding vector are not easy interpretable. The most recent approaches scale with the corpus size, and word vectors induced by them are good at capturing complex patterns beyond simple word similarity.

Of all the approaches, word embeddings are the most promising. This approach is used by the bootstrapping relationship extraction system to be detailed in Chapter 5.

4

MinHash-based Relationship Classification

The MinHash-based Semantic Relationship Classifier (MuSICo), is the method for large-scale relationship extraction based on nearest neighbour classification (k-NN) that I propose in this dissertation. This chapter starts by introducing the two techniques that are the foundations for the classifier: Min-Hash and Locality-Sensitive Hashing. Next, it describes how the classifier was built and the results of validation experiments performed with English and Portuguese datasets.

4.1 Min-Hash

The Min-Hash technique was first introduced in the seminal work of Broder (1997), where it was successfully applied to the task of detecting duplicate Web pages. Given a vocabulary Ω of size D, that is, the set of all possible representative elements occurring in a collection of documents and two sets, A and B, where:

$$A, B \subseteq \Omega = \{1, 2, \ldots, D\} \tag{4.1}$$

The Jaccard similarity coefficient between the sets of elements is given by the ratio of the size of the intersection between A and B, to the size of the union of both datasets, as show in Formula 4.2:

$$\text{Jaccard}(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{4.2}$$

Calculating the similarity between two sets, using the Jaccard similarity, requires computations in the order of n^2, since every element in set A has to be compared with every other element in set B. However, suppose that a random permutation π is performed on the ordering considered for the elements in the vocabulary Ω:

$$\pi : \Omega \longrightarrow \Omega$$

An elementary probability argument shows that the Jaccard similarity can be estimated from the probability of the first (i.e., the minimum) values of the random permutation π, for sets A and B, being equal, given that the Jaccard coefficient is the number of common elements to both sets over the number of elements that exist in at least one of the sets:

$$\text{P}\left(\min(A) = \min(B)\right) = \frac{|A \cap B|}{|A \cup B|} = \text{Jaccard}(A, B)$$

After the creation of k minwise independent permutations (i.e., $\pi_1, \pi_2, \ldots, \pi_k$) one can efficiently estimate Jaccard(A, B) as $\hat{\text{J}}(A, B)$, without bias, as a binomial distribution. Equation 4.3 shows the expected value of the binomial distribution used for estimating the Jaccard coefficient from the k random permutations:

$$\hat{\text{J}}(A, B) = \frac{1}{k} \sum_{j=1}^{k} \begin{cases} 1, & \min(\pi_k(A)) = \min(\pi_k(B)) \\ 0, & \text{otherwise} \end{cases} \tag{4.3}$$

For larger k, the estimation $\hat{\mathrm{J}}(A, B)$ approximates Jaccard(A, B). Equation 4.4 shows the corresponding variance for the distribution, which decreases for larger values of k:

$$\text{Variance}(\hat{\mathrm{J}}(A, B)) = \frac{1}{k}\, \hat{\mathrm{J}}(A, B) \left(1 - \hat{\mathrm{J}}(A, B)\right) \tag{4.4}$$

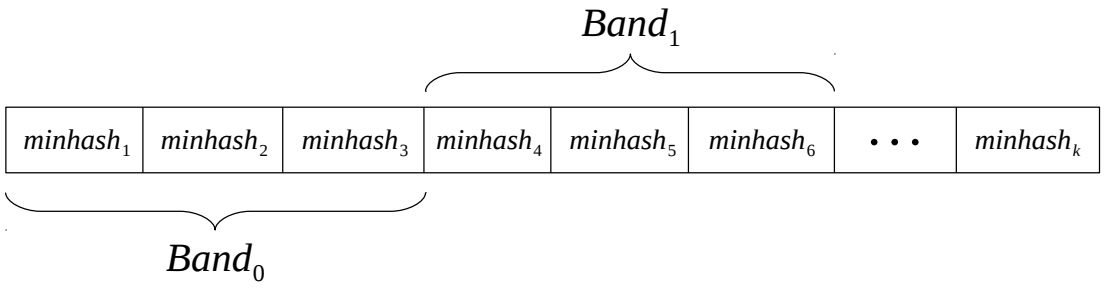
In practice, the π_k permutations are achieved by applying k hashing functions to each element of a given set S. These transform each element of the set into a representation which allows the applications of an ordering function over the elements. After applying the hashing schema and ordering the elements, only the element whose hashing value is the minimum is kept. Repeating this process for k different hashing functions, $h^k_{min}(S)$, results in k min-hash values. Each document is then represented by a min-hash signature, that is, a k-size vector containing all the min-hash values, that is, a min-hash value for each hashing function.

4.2 Locality-Sensitive Hashing

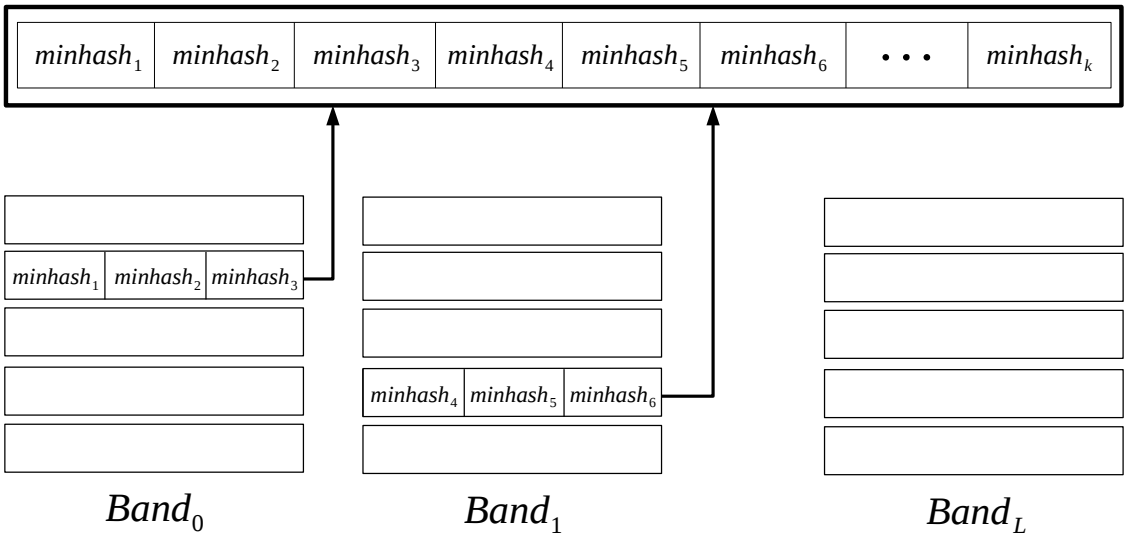
Locality-Sensitive Hashing (LSH) is a technique to reduce the dimensionality of data. This technique hashes input objects in such a way that similar objects are stored with high probability in the same bucket (Gionis et al., 1999).

Each object to be stored in the LSH structure, in this case a vector of min-hash signatures, is split into L smaller chunks, composed by n min-hash signatures, as shown in Figure 4.1a. Each chunk is then indexed into L different hash-tables, where the key is an n-chunk from the object and the value is the full min-hash signature of the object, as shown in Figure 4.1b.

The size k of the min-hash signature of an object and the number of bands L, must follow the constrain that $k \mod L = 0$. The idea behind this schema is that objects with a common sub-structure, and therefore with equal min-hash signatures, are going to be hashed into the same bands. Given any object, similar objects can be found by retrieving objects that were hashed into the same bands.



(a) A min-hash signature vector representing an object.



(b) LSH Bands indexing chunks of min-hash signatures of objects.

Figure 4.1: Locality-Sensitive Hashing schema for storing a vector of min-hash signatures in different bands.

```
Noam_,oam__ ,am_C_,m_Ch_,_Cho_,Chom_,homs_,omsk_,msky_,sky__,ky_i_,
y_is_,_is__,is_a_,s_a__,_a_p_,a_pr_,_pro_,prof_,rofe_,ofes_,fess_,
esso_,ssor_,sor__,or_a_,r_at_,_at__,at_M_,t_MI_,_MIT_
```

Figure 4.2: A 5-gram character representation for the sentence *Noam Chomsky is a professor at MIT*

4.3 MuSICo

By relying on machine learning algorithms, a statistical model can be inferred from annotated examples containing different relationships types. Then, given a new sentence, the statistical model can make the decision of whether the sentence holds a certain relationship type.

Another possibility is, instead of learning a complex statistical model, finding the most similar relationship instances in a database and using these similarities to make the decision of whether the sentence holds a certain relationship type. That is, the relationship expressed in the sentence can be classified according to the relationship type of the most similar relationship instances in a database. A naive approach to find the most similar pairs of relationship instances in a database of size N, with a given relationship r, involves computing all possible r with N pair similarities, which quickly becomes a bottleneck for large N. Even if the task is done in a parallel fashion, overcoming this complexity is necessary to achieve good scalability.

MuSICo is a system that compares the most similar relationship instances in an efficient way, given a new relationship instance. The similarity comparison of relationship instances is calculated with the Jaccard similarity, which estimates the similarity between any two given objects A and B by comparing its constituent parts. If A and B are textual documents, a common way is to transform each document into a set of shingles. A n-shingle (or n-gram) of a document is a sequence of n characters that appear in the document (see Figure 4.2).

If the set of n-grams from a document x is given by shingles(x), then comparing A with B involves $|\text{shingles}(A)| \times |\text{shingles}(B)|$ comparison operations. This can be too computationally intensive, especially if the sets a very large number of elements. Moreover, given a task of trying to estimate the similarity between a given document and each document in a large collections of documents, the comparison operation has

to be done between all the documents in the collection.

MuSICo approximates the Jaccard coefficient through a Min-Hash procedure, and leveraging the LSH method for rapidly finding the k-Nearest-Neighbours (k-NN) with most similar relationship instances. The min-hash signatures are generated from a set of features, extracted from the sentences, namely: character quadgrams, prepositions, verb forms in the past participle tense, infinitive forms of verbs, and relational patterns.

With traditional k-NN classifiers, training takes virtually zero computation time, since it just involves storing the example instances, but classification is highly demanding. MuSICo, by relying on a LSH technique for indexing the training instances, allows classification to be made efficiently, since it only has to measure the similarities of a small set of candidate instances. The technique leverages on the min-hash signatures to compress the relationship instances and preserve the expected similarity of any pair of instances.

MuSICo operates in two phases. The indexing phase involves processing an annotated dataset of relationships instances, and storing them in LSH bands. The classification phase classifies a new relationship instance based on the examples indexed in the LSH bands. A textual analysis step is common to both phases.

4.3.1 Textual Analysis

MuSICo represents each relationship instance as a set of features, by performing a textual analysis of the sentence. This analysis starts by identifying three contexts:

BEF-BET: Words occurring before the first entity and between the two entities.

BET: Words between the two entities that constitute the binary relationship.

BET-AFT: Words occurring between the two entities and after the second entity.

For instance, given the following sentence, where the related entities are in bold:

*The **micropump** is fabricated by **anisotropic etching**, considering orientation.*

MuSICo would identify the following contexts:

BEF-BET: *The micropump is fabricated by*

BET: *is fabricated by*

BET-AFT: *is fabricated by anisotropic etching, considering orientation.*

This representation follows the observation that a relationship between two named-entities is generally expressed using only words that appear in one of the three contexts defined above (Bunescu and Mooney, 2005a). For each context, MuSICo considers lexical and syntactic features. The lexical features are essentially based on quadgrams of characters. The syntactic features are derived from PoS-tags and include:

- prepositions;
- verb forms in the past participle;
- infinitive forms of verbs, except auxiliary verbs;
- relational patterns corresponding to: a verb, followed by nouns, adjectives, or adverbs, and ending with a preposition.

The relational pattern is inspired by one of the features used in the ReVerb OIE system by Fader et al. (2011). The described features are extracted from the three contexts (i.e., the BEF-BET, BET, and BET-AFT).

MuSICo tries to identify the presence of the passive voice when it captures a relational pattern in a relationship. The idea is to use the presence or absence of the passive voice to detect the direction of the relationship. The technique to detected the passive is based on PoS-tags. Concretely, it considers any form of the verb *to be*, followed by a verb in the past tense or the past participle and ending in the word *by*, which is than followed by one of the named-entities or nominals of the relationship. This constraint is relaxed, allowing for the occurrence of adverbs, adjectives or nouns between the two verbs and the preposition *by*. For instance:

Sun Microsystems was acquired by business software giant **Oracle**.

The software company **Netscape**, which was later bought by **AOL**.

Example

Each generated feature is represented by a string, and is assigned to a globally unique identifier associating it to the context from where it was extracted. For example, given the sentence:

"The software company **Netscape** was bought by **AOL**."

besides characters quadgrams, the following features are also generated by MuSICo:

- by_PREP_BEF-BET, by_PREP_BET
- bought_BEF-BET, bought_BET
- buy_VVN_BEF-BET, buy_VVN_BET
- was_bought_by_RVB_PASSIVE_BEF-BET
- was_bought_by_RVB_PASSIVE_BET

4.3.2 Indexing

The indexing operation consists of extracting the described features from a dataset of annotated relationship instances, calculating the min-hash signatures for each individual feature and indexing the signatures in the LSH bands. Given an annotated dataset of relationship instances, a complete outline of the indexing operation is as follows:

1. Identify, for each sentence, the three contexts.
2. Extract from each context: sets of character quadgrams, prepositions, verb forms in the past participle tense, infinitive forms of verbs, and relational patterns.
3. Compute the min-hash signature vector for each relationship instance, based on the features extracted in the previous step, as described in 4.1. After this, each relationship instance is represented by a min-hash signature made from k min-hashes.
4. Split the min-hash signature vector into bands, and hash each vector into the L bands, as described in 4.2.

4.3.3 Classification

Once the database is populated with examples of indexed relationship instances, the classification operation, given a target sentence, is as follows:

1. Identify the three contexts and extract from each a set of features.
2. Compute the min-hash signatures based on the set of extracted features.
3. Retrieve relationship instances from the collection of examples, having at least one identical LSH band.
4. Estimate the Jaccard similarity coefficient of the instances with the target instance using the available min-hash signatures.
5. Order the retrieved instances by their similarity towards the target instance.
6. Assign the relationship type based on the weighted votes from the top-k most similar instances.

The classification is computationally efficient. The similarity is only computed between the target instance and the retrieved candidates, using the complete min-hash signatures to approximate the Jaccard similarity coefficient. In this way, the classifier avoids the pairwise similarity comparisons against all example relationship instances in the database. This classification procedure essentially corresponds to a weighted k-NN classifier, where each example instance has a weight corresponding to its similarity towards the instance being classified, and where the more similar instances have therefore a higher vote, in the classification, than the ones that are more dissimilar.

4.4 Evaluation

MuSICo was evaluated considering three configuration parameters: the size of the min-hash signatures, the number of LSH bands, and the number k of nearest neighbours which are considered in the classification.

The experiments involved datasets consisting of sentences where a specific type of relationship is expressed or no relationship at all, considering popular datasets for English and a Portuguese dataset gathered from Wikipedia and DBpedia.

In the implementation of the minwise hashing scheme, each of the independent permutations is a hashed value. Each of the k independent permutations is associated with a polynomial hash function $h^k(x)$ that maps the members of Ω to distinct values.

In the experiments, the textual analysis involving part-of-speech tagging was performed for the English dataset with the MorphAdorner NLP package (Burns, 2013). The L bands in the LSH structure are implemented with MapDB, a persistence storage structure developed by Kotek (2013)

A PoS-tagger for Portuguese was developed based on the OpenNLP (Morton et al., 2005) software package and trained with morphologically annotated datasets for Portuguese, namely, the CINTIL Corpus of Modern Portuguese (Barreto et al., 2006; Branco and Silva, 2006) and Floresta Sintáctica (Afonso et al., 2002). The two datasets were normalized into one by converting the representation formats to a simple tagset following the work of Petrov et al. (2012).

4.4.1 Portuguese Dataset Creation

Wikipedia is a comprehensive resource that contains diverse content in many languages, Portuguese included. In Wikipedia, besides textual descriptions about concepts and entities from different fields of knowledge, there is also structured information in the form of infoboxes. An infobox is a manually-created table that holds the main facts in the form of attributes and values for many Wikipedia articles, as in Figure 4.3.

Nome completo	Otis Ray Redding, Jr.
Nascimento	9 de setembro de 1941
Origem	Dawson, na Geórgia
País	Estados Unidos
Data de morte	10 de dezembro de 1967 (26 anos) Madison, no Wisconsin
Gênero(s)	*Soul*, *Southern Soul*, *Soul Blues*
Instrumento(s)	Vocal
Gravadora(s)	Stax Records, Volt, Atco, Rhino, Sundazed
Página oficial	http://www.otisredding.com/

Figure 4.3: Infobox on Ottis Redding in the Portuguese Wikipedia.

Projects like DBpedia explored the automatic construction and the knowledge graphs derived from facts expressed in infoboxes of Wikipedia pages in several languages, including Portuguese (Lehmann et al., 2015).

The same facts are often expressed both in the text of Wikipedia articles and associated infoboxes, and consequently, in derived resources like DBpedia. By combining relationships expressed in DBpedia with articles from Wikipedia, where both entities co-occur, we can collect large amounts of training data. For example, the article of the Portuguese Wikipedia about the artist *Ottis Redding* contains the sentence:

Ottis Redding *nasceu na pequena cidade de* ***Dawson, Georgia***
(in English: ***Ottis Redding*** *was born in the small city of* ***Dawson, Georgia***)

The infobox of this article contains the attribute:

origem (i.e., *origin*) = *Dawson, Georgia*

and, hence, the DBpedia knowledge graph contains a relationship of the type *origem* between the entities *Otis Redding* and *Georgia*.

By combining the information from DBpedia with phrases from Wikipedia articles, as in the above example, we automatically generate training data for extracting *origem* relationships. This process can introduce noise: a pair of entities in a DBpedia relationship may also co-occur in a sentence expressing a different relationship or no relationship at all. However, it is expected that the large volume of data extracted in this way compensates the noise present in the training data (Mintz et al., 2009).

The general procedure for generating a dataset of semantic relationships from Wikipedia and DBpedia is thus the following:

1. Gather from DBpedia all the semantic relationships expressed among subjects (i.e., Wikipedia pages) which correspond to persons, locations or organisations.

2. For each relationship, record the Wikipedia pages of the entities involved and their type.

3. Extract all the sentences from the two Wikipedia pages associated with each of the entities in the relationship.

4. Filter the sentences gathered in the previous step, keeping only those where both entities involved in a relationship are mentioned.

5. Keep the sentences gathered from this filtering process as relationship instances of a given relationship type.

In step 3, to improve the coverage of the number of gathered examples, the filtering process considers small variations on the entity names in the mapping from DBPedia and in the Wikipedia article's title, to the text in the sentences. Besides the original names, the process also considers sequences of characters up to the first comma or parenthesis, since many Wikipedia entities are disambiguated by adding more information to the article's title. For instance, although most of the sentences only refer the state by *Georgia*, the Wikipedia page for the state of Georgia in the USA is identified by *Georgia_(United_States)*.

The procedure described above generates many sentences expressing the various types of semantic relationships in DBpedia, which were in turn derived from information in the infoboxes of Wikipedia. Many of the relationship types expressed in the generated sentences correspond to slight variations of the same semantic concept. For instance, given the following sentence, which expresses the DBpedia relationship *localizado-em* (i.e., *locatedInArea*):

East Oak Lane *é um bairro localizado em* ***Philadelphia***.
(in English: ***East Oak Lane*** *is a neighbourhood located in* ***Philadelphia***.)

and the sentence where the DBpedia relationship *capital* is expressed:

Lisboa *é a capital e a cidade mais populosa de* ***Portugal***.
(in English: ***Lisboa*** *is the capital and the largest city of* ***Portugal***.)

although the two sentences are annotated with different relationship types, both are variations of the same concept that can be generalized to *localizado-em*. Given this observation, the different types of relationships present in DBpedia were mapped into 10 general relationships types. Table 4.1 details the mappings from DBpedia relationships into a more generalized semantic concept.

Relationship	DBpedia Relationships
localizado-em (*located-in*)	locatedInArea, archipelago, location, locationCity locationCountry, municipality, subregion, federalState district, region, province, state, county map;campus, garrison, department, country capitalCountry, city, capital, largestCity
origem-de (*origin*)	origin, birthPlace, foundationPlace, sourcePlace nationality, residence, hometown, sportCountry
local-de-enterro-ou-falecimento (*death-or-burial-place*)	deathPlace, placeOfBurial
parte-de (*part-of*)	currentMember, pastMember, type parentOrganisation, distributingCompany broadcastNetwork, affiliation, university, youthClub, party, pastMember, team, associatedMusicalArtist, member
antepassado-de (*ancestor-of*)	parent, child
sucessor-de (*successor-of*)	successor, predecessor
pessoa-chave-em (*key-person-in*)	keyPerson, president, leaderName president, monarch, foundedBy leader, leaderName, founder
influenciado-por (*influenced-by*)	influenced, doctoralAdvisor
parceiro (*partner*)	spouse, partner
não-relacionado (*other*)	all the other relationships

Table 4.1: Mappings of DBpedia relationships into 10 general relationship types in the created Portuguese Wikipedia dataset.

Relationship	Number of Examples
localizado-em (*located-in*)	46,236
origem-de (*origin*)	23,664
local-de-enterro-ou-falecimento (*death-or-burial-place*)	6,726
parte-de (*part-of*)	5,142
antepassado-de (*ancestor-of*)	266
sucessor-de (*successor-of*)	496
pessoa-chave-em (*key-person-in*)	355
influenciado-por (*influenced-by*)	147
parceiro (*partner*)	128
não-relacionado (*other*)	6,441

Table 4.2: Number of relationship instances gathered by distant supervision for the Portuguese Wikipedia dataset.

During the mapping process, the order of the entities in some relationship types was swapped in order to keep the semantics consistent with the other relationship instances also mapped into the same generic relationship.

The mappings generated a dataset containing 10 different types of relationships, as illustrated in Table 4.2. All the relationships consider the order of the arguments in the relationship, they are asymmetric, except the relationships *parceiro* and *não-relacionado* which are symmetric.

A small subset of the automatically generated training data was manually reviewed to create a curated dataset for evaluation. During the review process it was found that the distant supervision method achieved an accuracy of about 80% in assigning the relationship type that is actually expressed in the sentence. This result is in agreement with previous work of García and Gamallo (2011). Several problems were also found concerning the segmentation of Wikipedia articles into sentences (e.g., it is common to see phrases that include, at the beginning or end, words from the title section immediately before the sentence). In the construction of the manually curated dataset for the evaluation of results, all the identified problems were manually corrected.

Table 4.3 shows the statistical characterization of the subset of data which was manually reviewed (i.e., column Test), as well as the sub-set of data which was not manually reviewed (i.e., column Train), for the whole dataset. This dataset is available online at `http://dmir.inesc-id.pt/project/DBpediaRelations-PT_01_in_English`.

	Train	Test	Total
# Sentences	97,363	625	97,988
# Terms	2,172,125	14,320	2,186,445
# Relationship Types	10	10	10
# Relationship Instances	89,054	547	89,601
# Named-Entities	70,716	838	71,119
Avg. sentence length (terms)	22.42	24.12	22.43
StDev. sentence length (terms)	11.39	11.00	11.39
Avg. instances/class	8,905.4	54.7	8,960.1
StDev. instances/class	14,109.33	64.18	14,172.38

Table 4.3: Statistical characterization of the Portuguese Wikipedia relationships dataset.

4.4.2 Experiments with English Datasets

In the experiments considering English texts, MuSICo was evaluated with three datasets containing relationship collections from different domains, which are commonly used as benchmarks (see a statistical characterization of the three datasets in Table 4.4):

SemEval dataset (Hendrickx et al., 2010): consists of 10,717 sentences, annotated according to 19 possible relationship types between two nominals, 9 non-symmetric relations types, plus a label for denoting that no relationship is being expressed. Compared with the other datasets , this dataset is well balanced among classes, and is split into 8,000 instances for training and 2,717 for testing.

Wikipedia dataset (Culotta et al., 2006): consists of paragraphs from 441 Wikipedia pages, containing annotations for 4,681 relation mentions of 53 different relation types like *job-title*, *birth-place*, or *political-affiliation*. The dataset is split into training and testing subsets, with about 70% of the paragraphs for testing, and the remaining 30% for training. In the Wikipedia dataset, the distribution of the examples per class is highly skewed: *job-title* is the most frequent relation (379 instances), whereas *grandmother* and *discovered* have just one example in the dataset. Moreover, although the full dataset contains annotations of 53 different relationship types, only 46 types are included in both the training and testing subsets. Still, of these 46 relation types, 14 of them have less than 10 examples.

	SemEval		Wikipedia		AImed
	Train	**Test**	**Train**	**Test**	**Data**
# Sentences	8,000	2,717	2,199	926	2,202
# Terms	137,593	46,873	49,721	20,656	75,878
# Relationship types	19	19	47	47	2
# Relation instances (except *other*)	6,590	2,263	15,963	6,386	1,000
# Nominals	16,001	5,434	5,468	2,258	4,084
Avg. sentence length (terms)	119.8	119.4	177.2	172.8	184.2
StDev. sentence length (terms)	45.0	44.4	104.5	100.1	98
Avg. instances/class	421	143	295.6	135.9	1 961.5
StDev. instances/class	317.5	105.5	1707.3	728.2	1 372.5
Max. instances/class (except *other*)	844	22	268	113	1 000
Min. instances/class	1	1	1	1	1 000

Table 4.4: The English datasets used in the MuSICo evaluation.

Therefore, in the experiments with the Wikipedia dataset, only a subset of 46 relationship types was considered. Additionally, the direction of the relationship was disregarded, only the problem of predicting the relationship type was considered (i.e., classifying according to one of the 46 semantic relationship types, or as *other*). Of all the three datasets, this was the one that least fitted the general approach for modelling the relationship extraction task, requiring significant adaptations.

AImed dataset (Bunescu and Mooney, 2005a): consists of 225 MEDLINE abstracts, 200 of which describing interactions between human proteins. There are 4,084 protein references and approximately 1,000 tagged interactions. In this dataset, there is no distinction between genes and proteins, and the relations are symmetric and of a single type. The experiments were made with a 10-fold cross validation methodology, using the same splits as in the study that originally used this dataset (i.e., the paper referenced above).

Results

The evaluation experiments used the full set of features described in subsection 4.3.1, with different parameters for the minwise hashing-based scheme:

- k nearest neighbours considered: 1, 3, 5 or 7;

	Sigs./	1 kNN			3 kNN			5 kNN			7 kNN		
	Bands	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
SemEval	200/25	0.662	0.622	0.641	0.683	0.642	0.662	0.698	0.652	0.674	0.698	0.637	0.666
	200/50	0.662	0.621	0.640	0.683	0.643	0.662	0.698	0.651	0.673	0.698	0.636	0.666
	400/25	0.664	0.636	0.650	0.685	0.668	0.676	0.708	0.672	0.690	0.691	0.667	0.679
	400/50	0.663	0.635	0.649	0.684	0.664	0.674	**0.708**	0.674	**0.690**	0.694	0.670	0.682
	600/25	0.657	0.631	0.644	0.677	0.660	0.669	0.697	0.674	0.685	0.695	0.660	0.677
	600/50	0.657	0.631	0.644	0.676	0.658	0.667	0.699	**0.678**	0.688	0.694	0.664	0.678
	800/25	0.654	0.630	0.642	0.675	0.656	0.665	0.694	0.662	0.678	0.696	0.658	0.677
	800/50	0.654	0.632	0.643	0.677	0.658	0.667	0.698	0.665	0.681	0.696	0.658	0.676
Wikipedia	200/25	0.410	0.336	0.369	0.434	0.335	0.378	0.439	0.310	0.363	0.489	0.323	0.389
	200/50	0.409	0.336	0.369	0.435	0.336	0.379	0.440	0.310	0.364	0.489	0.321	0.387
	400/25	0.453	0.350	0.394	0.472	0.354	0.405	0.507	0.348	0.413	0.485	0.323	0.388
	400/50	0.450	0.349	0.393	0.468	0.354	0.403	0.503	0.350	0.412	0.509	0.328	0.399
	600/25	0.419	0.344	0.378	0.439	0.352	0.391	0.492	0.364	0.419	0.522	**0.365**	**0.430**
	600/50	0.419	0.343	0.377	0.444	0.354	0.394	0.485	0.353	0.408	**0.532**	0.353	0.425
	800/20	0.416	0.344	0.377	0.431	0.348	0.385	0.493	0.351	0.410	0.513	0.343	0.411
	800/50	0.419	0.345	0.378	0.433	0.350	0.387	0.515	0.346	0.414	0.517	0.338	0.409
AImed	200/25	0.405	0.545	0.465	0.430	0.509	0.466	0.480	0.484	0.482	0.507	0.460	0.482
	200/50	0.405	0.545	0.465	0.430	0.509	0.466	0.480	0.484	0.482	0.507	0.460	0.482
	400/25	0.420	0.589	0.491	0.451	0.554	0.497	0.481	0.524	0.501	0.516	0.502	0.509
	400/50	0.420	0.588	0.490	0.455	0.561	0.502	0.484	0.529	0.505	**0.519**	0.505	0.512
	600/25	0.409	0.605	0.488	0.445	0.571	0.500	0.475	0.529	0.500	0.511	0.513	0.512
	600/50	0.409	0.605	0.488	0.445	0.571	0.500	0.475	0.530	0.501	0.511	0.513	0.512
	800/25	0.416	0.613	0.496	0.453	0.595	0.514	0.481	0.547	0.512	0.490	0.512	0.501
	800/50	0.418	**0.614**	0.498	0.454	0.596	**0.515**	0.482	0.545	0.511	0.489	0.514	0.501

Table 4.5: Precision (P), Recall(R) and F_1 scores obtained with various configurations of MuSICo in the English datasets.

- size of the min-hash signatures: 200, 400, 600 or 800 integers;
- number of LSH bands: 25, 50.

The performance was measured in terms of macro-averaged precision, recall and F_1-scores over the relationship labels, apart from the *not-related/other* labels. This corresponds to calculating macro-averaged scores over 18 classes in the case of the SemEval dataset, over 46 classes in the case of the Wikipedia dataset, and over the single *is-related* class in the AImed dataset.

Table 4.5 presents the obtained results, showing that using the 5 or the 7 nearest

Relationship	Instances Direction	(train/test)	Asymmetrical Precision	Recall	F_1	Symmetrical Precision	Recall	F_1
Cause-Effect	(e_1,e_2)	344/134	0.843	0.843	0.843	0.798	0.902	0.847
	(e_2,e_1)	659/194	0.735	0.902	0.810			
Component-Whole	(e_1,e_2)	470/162	0.572	0.759	0.653	0.628	0.670	0.648
	(e_2,e_1)	150/129	0.609	0.520	0.561			
Entity-Destination	(e_1,e_2)	844/291	0.744	0.911	0.819	0.747	0.901	0.817
	(e_2,e_1)	1/1	1.000	0.000	0.000			
Entity-Origin	(e_1,e_2)	568/211	0.789	0.815	0.802	0.756	0.795	0.775
	(e_2,e_1)	148/47	0.667	0.723	0.694			
Product-Producer	(e_1,e_2)	323/108	0.670	0.602	0.634	0.673	0.589	0.628
	(e_2,e_1)	394/123	0.654	0.569	0.609			
Member-Collection	(e_1,e_2)	78/32	0.778	0.438	0.560	0.767	0.777	0.772
	(e_2,e_1)	612/201	0.776	0.791	0.783			
Message-Topic	(e_1,e_2)	490/210	0.751	0.733	0.742	0.778	0.778	0.778
	(e_2,e_1)	144/51	0.750	0.706	0.727			
Content-Container	(e_1,e_2)	374/153	0.726	0.778	0.751	0.706	0.802	0.751
	(e_2,e_1)	166/39	0.627	0.821	0.711			
Instrument-Agency	(e_1,e_2)	97/22	0.429	0.545	0.480	0.605	0.667	0.634
	(e_2,e_1)	407/134	0.615	0.679	0.645			
Other	—	1 410/454	—	—	—	0.442	0.293	0.352
Macro-average	—	—	0.708	0.674	0.690	0.718	0.764	0.740

Table 4.6: Results obtained by MuSICo for each relationship type in the SemEval dataset.

neighbours, instead of just the most similar example, results in an increased performance for the SemEval and Wikipedia datasets, while a better F_1 score was obtained for the AImed dataset when considering the 3 nearest training examples.

Table 4.6 presents per-class results in the case of the SemEval dataset, considering the configuration that achieved the best performance in the results from Table 4.5 (i.e., a configuration using the 5 nearest neighbours, with a min-hash size of 400, and with 50 bands in the LSH method). Besides the results on the regular SemEval classification setting, involving relation types with direction, a setting that ignores the relationship directions (i.e., considering 8 different relationship types) was also evaluated. The results show that some classes, such as *cause-effect*, are relatively easy to classify, whereas classes such as *instrument-agency* are much harder. For the class corresponding to *entity-destination*(e_1,e_2), the dataset only contains one instance for training and one instance for testing.

Other features representations for the relationship instances have been considered, such as:

- using different textual windows and different n-gram sizes;
- n-grams of tokens, after a lemmatization process;
- WordNet-based features.

However, the features described before achieved the best trade-off between accuracy and computational performance.

Comparison with other approaches

The best F_1 score with MuSICo was 0.69. This is in line with the state-of-the-art, where the best participating system in the SemEval 2010 task achieved a performance of over 0.82, whereas the second best system reported an F_1 score of 0.77, and the median F_1 score was of 0.68.

Table 4.7 shows the F_1 scores of the top ranked participants and summarizes the features employed by each participant system. Analysing in detail the components of the participating systems, most used a variety of features built by relying on external resources. The winning system, by Rink and Harabagiu (2010), derived in

F_1	Approach	Syntactic Dependencies	PoS-tags	External Resources
0.82	2 SVM classifers	YES	YES	YES
0.77	4 Kernels (SVM)	NO	YES	YES
0.77	Logistic Regression	NO	NO	YES
0.75	SVM	YES	YES	YES
0.69	MuSICo	NO	YES	NO

Table 4.7: MuSICo versus the best SemEval 2010 Task 8 systems.

total 45 different features from external resources such as WordNet (Miller, 1995), VerbNet (Schuler, 2005), Levin Verb Classes (Levin, 1993), Google's N-gram collection (Michel et al., 2010). The system also uses features generated from semantic parsing (Johansson and Nugues, 2007), which identify predicates in text and their semantic roles, and also from syntactic dependencies and PoS-tags. The system consists of two classifiers, one for detecting the relationship type and another for the direction of the relationship.

The second best participant, Tymoshenko and Giuliano (2010), relied on shallow syntactic processing (i.e., PoS-tags) and semantic information derived from the Cyc knowledge base (Lenat, 1995), providing different sources of information which are represented by different kernel functions. The final system is based on a linear combination of four kernels.

The third best participant, Tratz and Hovy (2010), used more simpler and straightforward approach, consisting of a single multi-label Logistic Regression classifier using a large number of boolean features. The features were derived from external resources, such as the U.S. Census 2000's most common names and surnames list, WordNet, Roget's Thesaurus (Jarmasz and Szpakowicz, 2003), and Web 1T N-gram (Thorsten Brants, 2006). Interestingly, this system, although not relying on any syntactic features, still achieved the 3rd place. Nevertheless, the authors claim to use the output of a in-house noun compound interpretation system as features, which associates semantic topics to nouns.

The fourth best participant, Chen et al. (2010), relied on PoS-tags, syntactic dependencies and features derived from WordNet, and applying a one-versus-all approach, training 10 SVM classifiers on for each relationship type.

All the top-ranked systems in the SemEval evaluation task derived features by

F_1	Kernel Type	Syntactic Dependencies	PoS-tags
0.56	All-Paths Graph Kernel	YES	NO
0.55	Shallow Linguistic Kernel	NO	YES
0.52	MuSICo	NO	YES

Table 4.8: Comparison of MuSICo with other approaches for the AImed dataset.

relying on different external resources and tools, and all systems involved learning a statistical model, by using an SVM with complex kernels or by relying on multi-class logistic regression.

With the AImed dataset, Tikk et al. (2010) compared different kernel-based methods, which quantify the similarity of two instances through counting the similarities of their substructures, with a common cross-validation methodology. Table 4.8 shows the results for different kernels with the AImed dataset. The All-Paths Graph Kernel by Airola et al. (2008) achieves an F_1 score of 0.56. The Shallow Linguistic Kernel, which is essentially a simplified version of the sub-sequence kernel from Bunescu and Mooney (2005a), combining words and PoS-tags, achieves an F_1 score of 0.55. MuSICo has only slightly inferior results, with an F_1 score of 0.52.

The All-Paths Graph Kernel (Airola et al., 2008) is based on a weighted directed graph that consists of two unconnected sub-graphs, one representing the dependency structure of the sentence, the other representing the sequential ordering of the words. Weights are determined by dependency weights, which are the higher the shorter the distance of the dependency to the shortest path between the candidate entities is.

Compared to other supervised approaches for relationship extraction, MuSICo is based on a much simpler set of features, which are common across domains and mostly language independent. PoS-tags are necessary for computing some of the features, but PoS-tagging can be made efficiently and accurately for most languages (Petrov et al., 2012). Comparing with other approaches over the same datasets, this approach directly supports multi-class and on-line learning while still attaining competitive results.

Processing Times

A direct comparison against other approaches, in terms of processing times, cannot be easily made. This would require a common set of tools for performing feature

extraction, as well as exactly the same implementations of the different algorithms.

All kernel-based approaches use an SVM as a classifier, which typically involve three main steps:

1. Generate the linguistic structures to be used within the kernel, for instance, calculating the syntactic dependencies or computing features derived from external knowledge resources.

2. Determine the substructures used by the kernel, and compute pairwise similarities.

3. Apply the SVM algorithm.

The approach used by MuSICo is significantly different, relying mostly on character quadgrams and PoS-tagging. Compared to syntactic parsing, PoS-tagging can be 20 to 30 times faster (Akbik and Löser, 2012; Wu and Weld, 2010) for the task of RE. Running on a single thread, MuSICo achieves an F_1 score of 0.69 with the SemEval dataset taking 172 seconds to process all three stages (i.e., feature extraction, indexing and classification) considering the 5 nearest neighbours, min-hash signatures of size 400, and 50 LSH bands.

For the AImed dataset, and without considering feature extraction Tikk et al. (2010) reports times of approximately 66.4 and 10.8 seconds, for training and testing, using the Shallow Linguistic Kernel, and times of approximately 4,517.4 and 3.7 seconds for training and testing, using the All-Paths Graph Kernel. MuSICo, using a single thread, takes on average about 161 seconds to process all three stages for each AImed fold, considering 3 nearest neighbours, min-hash signatures of size 800 and 50 bands.

The charts on Figure 4.4 present the processing times in seconds for each processing stage: feature extraction, indexing and classification, for all the parameter configuration presented in Table 4.5. For the AImed dataset, the charts show the average time for 1 fold.

The total processing time involved in each experiment naturally increases with the size of the dataset being considered. The time needed to perform feature extraction is independent of the LSH configuration and min-hash signatures size being used. The results indicate that these values represent a significant amount of the total processing time that is involved in each experiment. The results also show that the indexing

times increase significantly as the size of the min-hash signatures gets larger, since more hash functions need to be computed, and more min-hash values have to be stored and compared. Augmenting the number of bands in turn increases the classification time, since the number of hash tables where we have to look for candidate instances, and possibly also the number of candidates increases.

Scalability

The Min-Hash scheme, the core of MuSICo's architecture, was initially developed for detection of duplicate web pages in a search engine system (Broder et al., 2000). Therefore it was designed with scalability in mind. I also carried experiments to evaluate the scalability of MuSICo regarding the training and testing phases.

The first experiment evaluated the scalability of MuSICo when populating the database of relationships, i.e., indexing the training data. In this experiment, I considered different sizes of the SemEval dataset, specifically, 25%, 50%, 75% and 100%. For each partition the time taken in performing feature extraction and indexing was measured. Feature extraction corresponds to computing n-grams of characters, and performing part-of-speech tagging. The indexing phase involves only calculating the min-hash signatures for relationship features and indexing them in the different bands.

Figure 4.5a shows the processing time of as a function of the size of the dataset. Most of the processing time in the indexing phase is spent on feature extraction, as expected. Nevertheless, in terms of processing time, it increases linearly with dataset size.

Another experiment was carried to evaluate the scalability of the classification phase on the full SemEval training dataset (see Figure 4.5b). The database of LSH bands was populated with all the relationships part of the training set. As in the previous experiment, the classification time was measured for different partition sizes of the test dataset of SemEval, specifically, 25%, 50%, 75% and 100%. As expected, most of the processing time is also spent in feature extraction, but the processing time in classification grows linearly with the dataset.

Both training and classification times were measured with the configuration that achieved the best performance on SemEval in terms of F_1 score, concretely, min-hash signatures of size 400 and 50 LSH bands and considering the 5 nearest neighbours.

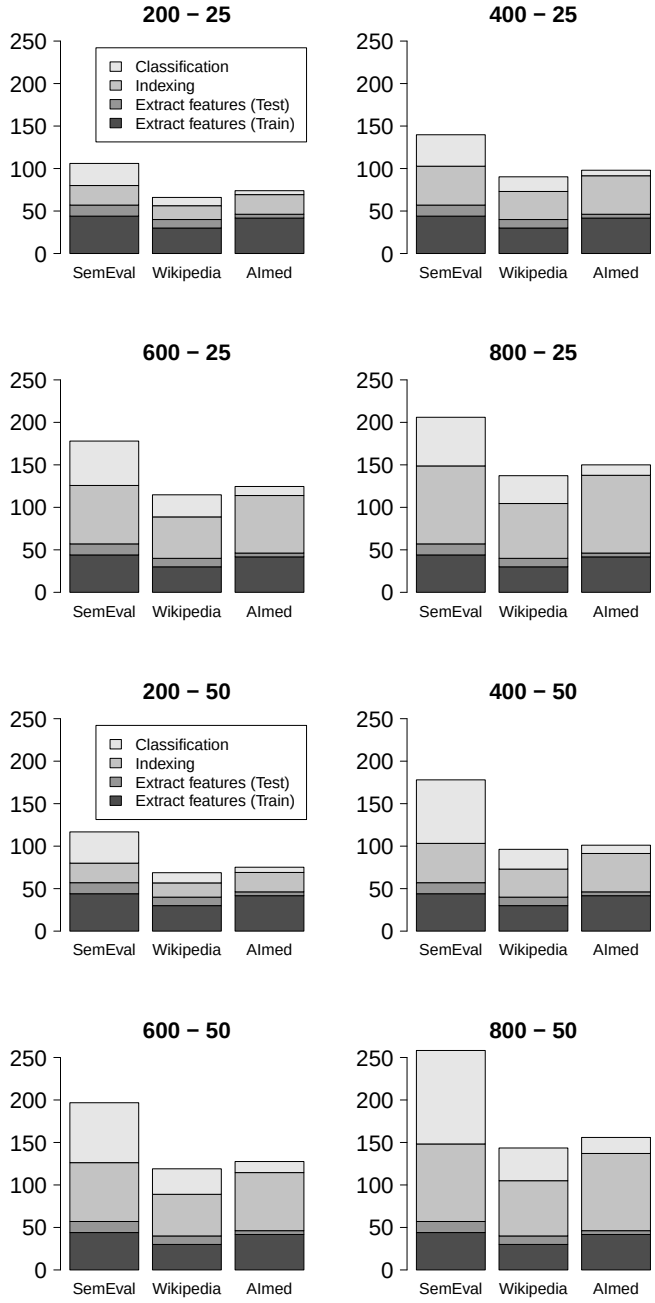
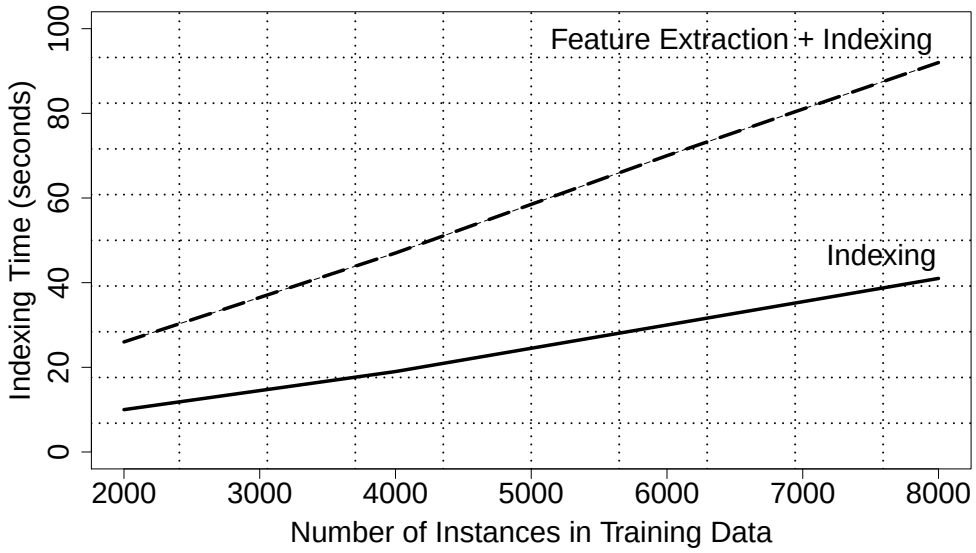
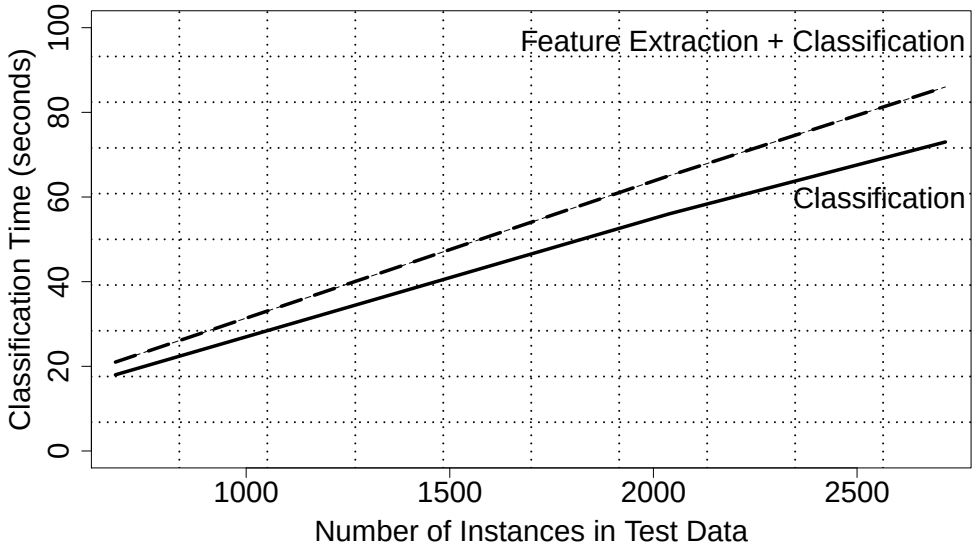


Figure 4.4: MuSICo processing times, in seconds, for each dataset and configuration. The numbers in bold, at the top of each graph, represent the size of the min-hash signatures and the number of bands.



(a) Indexing Time



(b) Classification Time

Figure 4.5: MuSICo processing times, in seconds, for scalability evaluation in indexing and classification phases.

As the graphics in Figure 4.5 show, processing time in both the indexing and the classification phase grows linearly with dataset size, which demonstrates the linear scalability of MuSICo's approach to relationship extraction. Most of the processing time is spent in feature extraction; this can be improved if the feature extraction is performed in parallel, leveraging on multi-core CPUs.

4.4.3 Experiments with Portuguese Datasets

MuSICo was also evaluated for Portuguese, using a dataset of Portuguese semantic relationships created by relying on distant supervision over the Portuguese Wikipedia and DBpedia. Also, as the part-of-speech (PoS) tagging component in MuSICo is language specific, a PoS-tagger for Portuguese was constructed based on a morphologically annotated Portuguese corpus.

Results

MuSICo was evaluated with the dataset generated from the Wikipedia/DBpedia for Portuguese in two experiments. The first experiment left out of the indexing phase the relationship instances that were manually reviewed, and these were later used for evaluation. The second experiment considered the whole dataset, with 25% of the instances for each relationship type held-out in the indexing phase to be later used for evaluation.

As with the previous experiments with English datasets of semantic relationships, experiments with a Portuguese dataset also considered different features for relationship representation and configuration parameters:

- Relationship features:
  - quadgrams of characters;
  - verbs;
  - prepositions;
  - relational patterns;
- k nearest neighbours considered: 1, 3, 5 or 7;

Set	Features
I	Quadgrams
II	Quadgrams Verbs
III	Quadgrams Verbs Prepositions
IV	Quadgrams Verbs Prepositions Relational Patterns

Table 4.9: Groups of features used in experiments with the Portuguese dataset.

- size of the min-hash signatures: 200, 400, 600 or 800 integers;
- number of LSH bands: 25, 50;

Table 4.10 describes the results for the first experiment. The features corresponding to each set on the left of the table are described in Table 4.9. The results show that the combination of character quadgrams, verbs, prepositions, and relational patterns, provides the best classification performance. Also, using the 5 or 7 first neighbours, rather than just the most similar relationship instance, improves the performance.

Table 4.11 presents the obtained results for the second experiment, showing that the method using distant supervision, along with the proposed classifier, allows extracting relationships with an F_1 score of 0.56. We can also see that the values of the different evaluation metrics are slightly lower for tests with 25% of the dataset. This indicates that measured results with the manually annotated collection may be regarded as an upper limit to an approximation of the true accuracy of the system.

The results of the second experiment were further analysed. Table 4.12 shows the results for relationship type and considering an evaluation where the direction of the relationship is ignored, as well as the results obtained for the relationship type *non-related/other*. The configuration for this evaluation considers the features and indexing parameters that had the best performance in the results of Tables 4.10 and 4.11:

- all features: quadgrams, verbs, prepositions, and relational patterns;

	Sigs./	1 kNN			3 kNN			5 kNN			7 kNN		
	Bands	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Set I	200/25	0.492	0.400	0.441	0.627	0.426	0.507	0.716	0.423	0.532	0.724	0.429	0.539
	200/50	0.489	0.400	0.440	0.625	0.425	0.506	0.716	0.423	0.532	0.726	0.430	0.540
	400/25	0.476	0.405	0.438	0.559	0.418	0.478	0.724	0.434	0.543	**0.736**	**0.443**	**0.553**
	400/50	0.474	0.405	0.437	0.557	0.423	0.481	0.715	0.434	0.540	0.731	0.441	0.550
	600/25	0.609	0.435	0.508	0.645	0.437	0.521	0.688	0.440	0.537	0.663	0.440	0.529
	600/50	0.583	0.435	0.498	0.646	0.437	0.521	0.686	0.433	0.531	0.719	0.441	0.547
	800/25	0.545	0.426	0.478	0.610	0.430	0.504	0.651	0.434	0.521	0.640	0.442	0.523
	800/50	0.541	0.423	0.475	0.611	0.432	0.506	0.652	0.436	0.523	0.643	0.444	0.525
Set II	200/25	0.476	0.414	0.443	0.628	0.437	0.515	0.713	0.429	0.536	0.718	0.432	0.539
	200/50	0.474	0.414	0.442	0.628	0.437	0.515	0.713	0.429	0.536	0.718	0.432	0.539
	400/25	0.499	0.417	0.454	0.563	0.430	0.488	0.725	0.437	0.545	0.729	0.442	0.550
	400/50	0.497	0.417	0.453	0.565	0.436	0.492	0.674	0.440	0.532	0.729	0.443	0.551
	600/25	0.580	0.425	0.491	0.640	0.442	0.523	0.669	0.439	0.530	0.728	0.435	0.545
	600/50	0.553	0.425	0.481	0.641	0.442	0.523	0.724	0.439	0.547	0.728	0.441	0.549
	800/25	0.549	0.424	0.479	0.615	0.433	0.508	0.720	0.443	0.549	**0.736**	0.441	**0.551**
	800/50	0.549	0.424	0.479	0.615	0.433	0.508	0.712	**0.447**	0.549	0.731	0.438	0.548
Set III	200/25	0.477	0.403	0.437	0.628	0.431	0.511	0.720	0.432	0.540	0.723	0.438	0.546
	200/50	0.478	0.404	0.438	0.628	0.431	0.511	0.666	0.432	0.524	0.670	0.438	0.530
	400/25	0.522	0.431	0.472	0.574	0.432	0.493	0.732	0.446	0.554	0.731	0.442	0.551
	400/50	0.522	0.431	0.472	0.578	0.441	0.500	0.679	0.446	0.538	0.732	0.445	0.554
	600/25	0.581	0.427	0.492	0.630	0.432	0.513	0.673	0.446	0.536	0.677	0.441	0.534
	600/50	0.554	0.427	0.482	0.631	0.432	0.513	0.726	0.439	0.547	0.731	0.442	0.551
	800/25	0.548	0.426	0.479	0.616	0.435	0.510	0.721	**0.449**	0.553	**0.733**	0.447	**0.555**
	800/50	0.545	0.423	0.476	0.620	0.446	0.519	0.721	0.445	0.550	0.732	0.446	0.554
Set IV	200/25	0.472	0.404	0.435	0.629	0.436	0.515	0.724	0.436	0.544	0.723	0.440	0.547
	200/50	0.474	0.404	0.436	0.575	0.436	0.496	0.671	0.436	0.529	0.670	0.440	0.531
	400/25	0.521	0.429	0.471	0.572	0.429	0.490	0.730	0.443	0.551	0.731	0.441	0.550
	400/50	0.521	0.429	0.471	0.573	0.436	0.495	0.680	0.447	0.539	**0.732**	0.444	0.553
	600/25	0.579	0.423	0.489	0.628	0.429	0.510	0.673	0.446	0.536	0.678	0.437	0.531
	600/50	0.552	0.423	0.479	0.629	0.428	0.509	0.728	0.446	0.553	0.731	0.438	0.548
	800/25	0.547	0.423	0.477	0.616	0.433	0.509	0.715	0.445	0.549	0.723	0.444	0.550
	800/50	0.544	0.420	0.474	0.618	0.439	0.513	0.716	0.444	0.548	0.731	**0.449**	**0.556**

Table 4.10: Precision (P), Recall(R) and F_1 results obtained with various configurations of MuSICo in the Portuguese dataset.

	Sigs./	1 kNN			3 kNN			5 kNN			7 kNN		
	Bands	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Set IV	200/25	0.448	0.353	0.395	0.460	0.345	0.394	0.492	0.331	0.396	0.487	0.325	0.390
	200/50	0.450	0.354	0.396	0.459	0.347	0.395	0.489	0.332	0.395	0.507	0.328	0.398
	400/25	0.440	0.350	0.390	0.448	0.344	0.389	0.468	0.328	0.386	0.479	0.320	0.384
	400/50	0.439	0.351	0.390	0.445	0.343	0.387	0.465	0.327	0.384	0.483	0.321	0.386
	600/25	0.461	0.358	0.403	0.466	0.353	0.401	0.482	0.337	0.397	0.469	0.324	0.383
	600/50	0.461	**0.360**	0.404	0.463	0.353	0.401	0.490	0.340	0.401	0.492	0.329	0.394
	800/25	0.446	0.358	0.397	0.462	0.350	0.398	0.492	0.338	0.401	**0.516**	0.333	**0.405**
	800/50	0.445	0.358	0.397	0.453	0.349	0.394	0.484	0.336	0.397	0.510	0.333	0.403

Table 4.11: MuSICo results over 25% of the instances for each semantic relationship for the Portuguese dataset.

- min-hash signatures of size 800;
- locality-Sensitive-Hashing of size 25;
- 7 nearest neighbours.

Table 4.12 also presents an assessment of the results obtained in terms of accuracy. Unlike macro-averages, which weight the types of relationships by the numbers of occurrences in the corpus, accuracy measures the portion of correct classifications. Results show that some classes, such as *origem-de* (i.e., *source*) and *parte-de* (i.e., *part-of*), are relatively easy to identify and classify, while classes such as *influenciado-por* (i.e., *influenced-by*) or *successor-de* (i.e., *successor-of*) are more difficult to identify and classify correctly. It is also important to notice that for the class *influenciado-por* the training data contains only 110 relationship instances, and 35 instances for testing.

4.5 Conclusions

This chapter described MuSICo, a new scalable on-line supervised method for relationship extraction based on nearest neighbour classification (k-NN). The nearest neighbour search is computationally feasible since the k-NN classifier leverages on min-wise hashing and on Locality-Sensitive Hashing (LSH).

Most supervised methods for relationship extraction have a high computational complexity. MuSICo is fast, because, instead of learning a statistical model, it looks

Relationship	Direction	Instances (train/test)	Assymetrical			Symmetrical		
			P	A	F_1	P	A	F_1
local-de-enterro-ou-falecimento	(e_1,e_2)	4 788/1 596	0.802	0.595	0.683	0.806	0.574	0.671
	(e_2,e_1)	257/85	0.375	0.035	0.065			
influenciado-por	(e_1,e_2)	84/28	0.000	0.000	0.000	0.000	0.000	0.000
	(e_2,e_1)	26/9	1.000	0.111	0.199			
pessoa-chave-em	(e_1,e_2)	106/35	0.500	0.086	0.146	0.233	0.079	0.117
	(e_2,e_1)	161/53	0.200	0.113	0.145			
localizado-em	(e_1,e_2)	33 639/11 213	0.916	0.929	0.922	0.924	0.922	0.923
	(e_2,e_1)	1 038/346	0.395	0.087	0.142			
origem-de	(e_1,e_2)	16 784/5 594	0.723	0.806	0.807	0.733	0.908	0.811
	(e_2,e_1)	965/321	0.664	0.567	0.612			
antepassado-de	(e_1,e_2)	151/50	0.471	0.800	0.593	0.545	0.727	0.623
	(e_2,e_1)	49/16	0.000	0.000	0.000			
parte-de	(e_1,e_2)	2 590/863	0.541	0.544	0.543	0.680	0.576	0.623
	(e_2,e_1)	1 267/422	0.574	0.275	0.372			
sucessor-de	(e_1,e_2)	117/39	0.400	0.051	0.091	0.541	0.161	0.248
	(e_2,e_1)	255/85	0.359	0.165	0.226			
parceiro	—	96/32	—	—	—	0.600	0.188	0.286
não-relacionado	—	4 831/1 610	—	—	—	0.767	0.543	0.636
Macro-Average	—	—	0.516	0.333	0.405	0.583	0.468	0.494
Accuracy	—	—		0.813			0.834	

Table 4.12: Results per relationship type over 25% of the instances of each relationship type.

for the most similar relationship examples in a database in order to classify a new relationship. Besides simplicity, computational efficiency, and direct support for multi-class, MuSICo also has the advantage of being an on-line classifier: to consider new training instances, one only needs to compute their min-hash signatures and store them in the LSH bands.

Experiments made with datasets from three different application domains, Semeval (i.e., generic web text), Wikpedia (i.e, wikipedia articles) and AImed (i.e., protein interactions), have shown that relationship extraction can be performed with high accuracy, using this method based on computationally efficient similarity search.

MuSICo achieves a competitive accuracy comparing to state-of-the-art results, concretely F_1 scores of 0.69 and 0.52 for the SemEval and AImed datasets, without relying on any external resources for feature extraction. Although these scores are below state-of-the-art, the global processing time, including feature extraction, training and classification, is done in less than 3 minutes (i.e., 172 seconds for SemEval and 161 seconds for AImed), using a single thread of execution.

The classifier was also evaluated with Portuguese data. Experiments with a dataset based on Wikipedia show the suitability of the proposed method, extracting 10 different types of semantic relationships, eight of them being asymmetrical, with an average F_1 of 0.56.

When measuring the scalability and processing time experiments, the observed time taken to process grew linearly with the size of the dataset considered, with most of the processing time spent in feature extraction, which can be easily computed in parallel, levering on multi-core CPUs.

The software implementation of MuSICo used in the experiments presented in this chapter is available at https://github.com/davidsbatista/MuSICo.

5

Bootstrapping Relationships with Distributional Semantics

BREDS (Bootstrapping Relationship Extraction with Distributional Semantics) is a new semi-supervised bootstrapping system for relationship extraction relying on distributional semantics proposed in this dissertation. BREDS relies on word vector representations (i.e., word embeddings) together with a simple compositionality function to bootstrap relationships. This chapter introduces BREDS, describing its architecture and workflow, and reports the results of a validation experiment.

5.1 BREDS

A bootstrapping system for relationship extraction starts with a collection of documents and a few seed instances. The system scans the documents, collecting textual segments containing occurrences of the seed instances. Then, based on these contexts, the system generates extraction patterns. The document collection is scanned once again using the extraction patterns to match new relationship instances. These newly extracted instances are then added to the seed set, and the process is repeated until a

certain stop criterion is met.

A key aspect in the bootstrapping process is the expansion of the seed set with new relationship instances while limiting the semantic drift, i.e., the progressive deviation of the semantics of the extracted relationships from the semantics of the seed relationships. BREDS addresses this challenge with a new approach based on word embeddings.

State-of-the-art bootstrapping approaches rely on word vector representations with TF-IDF weights, such as Snowball by Agichtein and Gravano (2000). However, expanding the seed set by relying on TF-IDF representations to find similar instances has limitations, since the similarity between any two relationship instance vectors of TF-IDF weights is only positive when the instances share at least one term. For instance, the two relational phrases:

Microsoft *was founded by* **Bill Gates**.

Bill Gates *is the co-founder of* **Microsoft**.

do not have any words in common, but both represent the same semantics, that is, a person is the founder of an organisation. Stemming techniques can aid in these cases (Porter, 1997). However, such techniques would only work for variations of the same root word. By relying on word embeddings, the similarity of two relational phrases can be captured even if no common words exist. For instance, the word embeddings for *co-founder* and *founded* should be similar, since these words tend to occur in the same contexts.

Word embeddings can nonetheless also introduce semantic drift. For instance, when relying on word embeddings, relational phrases like:

John *studied history at* **Lisbon University**.

Mary *is an history professor at* **Lisbon University**.

can both have a high similarity. BREDS controls the semantic drift by ranking the extracted relationship instances and scoring the generated extractions patterns.

Note that TF-IDF approaches represent a sentence or a relational phrase as a single vector, whereas in word embeddings approaches each word is represented by a single vector. BREDS combines the embedding vectors of a relational phrase into a single vector, and computes the similarity of relationship instances based on that single vector.

BREDS has the same processing phases as Snowball (see Figure 5.1):

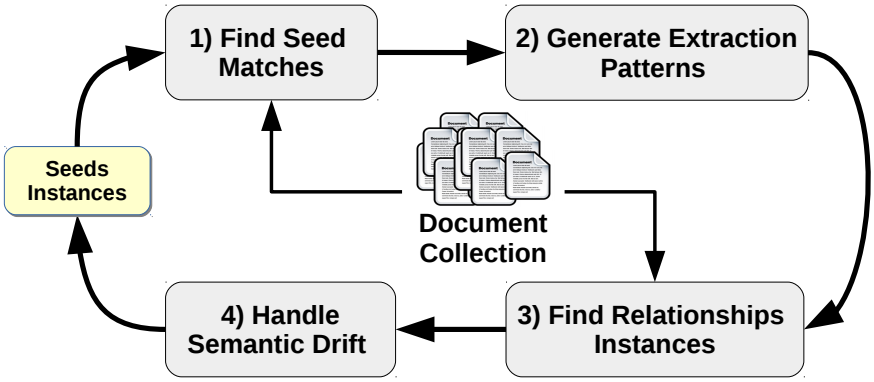


Figure 5.1: BREDS general workflow procedure.

1. Find Seed Matches
2. Generate Extraction Patterns
3. Find Relationship Instances
4. Handle Semantic Drift

It differs, however, in attempting to find similar relationships using word embeddings, instead of relying on TF-IDF representations. The remainder of this section details each of the four processing phases.

5.1.1 Find Seed Matches

As any other bootstrapping system, BREDS is initialized with seed instances of a given relationship type. Then, BREDS scans the document collection and, if both entities of a seed instance co-occur in a text segment within a sentence, that segment is considered and BREDS extracts textual contexts as in Snowball:

BEF: the words before the first entity;

BET: the words between the two entities;

AFT: the words after the second entity.

For instance, in the sentence:

The tech company **Soundcloud** *is based in* **Berlin**, *capital of Germany.*

the three textual contexts correspond to:

BEF: The tech company

BET: is based in

AFT: capital of Germany

In the BET context, BREDS tries to identify a relational pattern based on a heuristic originally proposed in the ReVerb OpenIE system, by Fader et al. (2011). The relational pattern limits a relation context to:

- a verb (e.g., *invented*);
- a verb followed by a preposition (e.g., *located in*);
- a verb followed by nouns, adjectives, or adverbs ending in a preposition (e.g., *has atomic weight of*).

These patterns will nonetheless only consider verb-mediated relationships. If no verbs exist between two entities, BREDS extracts all the words between the two entities, to build the representations for the BET context. For instance, given the sentence:

Google is based in Mountain View.

the relational pattern would correspond to: *is based in*, and in the sentence:

Google headquarters in Mountain View.

the pattern would be: *headquarters in*. Each context is transformed into a single vector by a simple compositional function that starts by removing stop-words and adjectives and then sums the word embedding vectors of each individual word. Mikolov et al. (2013a,b) showed that representing small phrases by summing each individual word's embedding results in good representations for the semantics in the phrase.

Relationship Representation

A relationship instance i is thus represented by three embedding vectors: V_{BEF}, V_{BET}, V_{AFT}. For instance, in the sentence:

The tech company **Soundcloud** *is based in* **Berlin**, *capital of Germany.*

the relationship instance expressed by the sentence will be represented by the following embedding vectors:

- V_{BEF} = embedding("tech") + embedding("company")
- V_{BET} = embedding("is") + embedding("based")
- V_{AFT} = embedding("capital")

where, embedding(x) is a function that represents the embedding vector for word x.

For the BET context, BREDS also tries to identify the passive voice, using part-of-speech (PoS) tags, which can help detecting the correct order of the entities in a relational triple. For instance, using the seed <`Google, DoubleClick`> expressing the relationship that the organisation `Google` owns the organisation `DoubleClick`, if BREDS extracts relationship instances between two organisations and detects patterns like:

ORG$_1$ *agreed to be acquired by* **ORG$_2$**

ORG$_1$ *was bought by* **ORG$_2$**

it will swap the order of the entities when producing a relational triple from the instance being expressed in the phrase. Hence, it will output the triple <ORG$_2$, *owns*, ORG$_1$>, instead of <ORG$_1$, *owns*, ORG$_2$>.

BREDS identifies the presence of the passive voice by considering any form of the verb *to be*, followed by a verb in the past tense or the past participle and ending in the word *by*, followed by a named-entity. This constrain is relaxed, allowing for the occurrence of adverbs, adjectives or nouns between the two verbs and the preposition *by*.

5.1.2 Extraction Patterns

After collecting all the seed contexts from the document collection and generating instances, BREDS generates extraction patterns by applying a single-pass clustering algorithm to the relationship instances gathered in the previous step. Each resulting cluster contains a set of relationship instances, where each instance is represented by three embedding vectors.

Algorithm 1 describes the clustering approach, which takes as input a list of relationship instances and assigns the first instance to a new empty cluster. Next, it iterates through the list of instances, computing the similarity between an instance x and every cluster Cl. The instance x is assigned to the first cluster with similarity higher or equal to τ_{sim}. If all the clusters have a similarity lower than τ_{sim}, a new cluster Cl_{new} is created, containing the instance x.

Algorithm 1: Single-Pass Clustering.

Input: $Instances = \{x_1, x_2, x_3, ..., x_n\}$, τ_{sim}
Output: $Patterns = \{\}$
$Cl_1 = \{x_1\}$
$Patterns = \{Cl_1\}$
for $x \in Instances$ **do**
  for $Cl \in Patterns$ **do**
    if $Sim(x, Cl) >= \tau_{sim}$ **then**
      $Cl = Cl \cup \{x\}$
    else
      $Cl_{new} = \{x\}$
      $Patterns = Patterns \cup \{Cl_{new}\}$

The similarity function $\text{Sim}(i_n, Cl_j)$, between an instance i_n and a cluster Cl_j, returns the maximum of the similarities between an instance i_n and any of the instances in a cluster Cl_j, if the majority of the similarity scores is higher than τ_{sim}. A value of zero is returned otherwise. As a result, clustering in Algorithm 1 differs from the original Snowball method, which computes similarities towards cluster centroids instead.

The computation of the similarity is illustrated in Figure 5.2. An instance is compared with every other instance inside the cluster, and the majority of the scores decides whether that instance is added to the cluster or not.

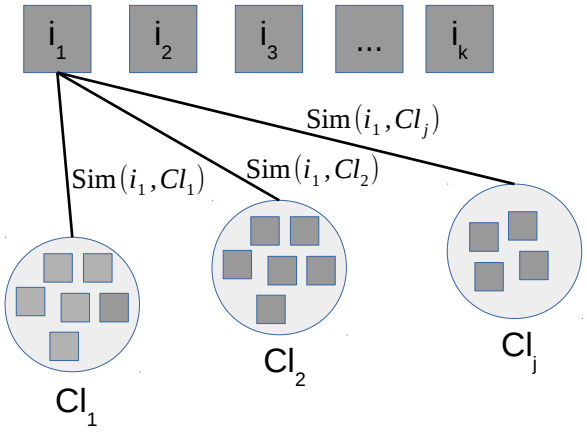


Figure 5.2: Comparison between an instance and a cluster of instances.

The similarity between any two relationship instances is calculated by measuring the cosine similarity between each instances's contexts embedding vectors:

$$\begin{aligned} \text{Sim}(S_n, S_j) = \alpha \cdot \cos(BEF_i, BEF_j) \\ + \beta \cdot \cos(BET_i, BET_j) \\ + \gamma \cdot \cos(AFT_i, AFT_j) \end{aligned} \tag{5.1}$$

where the parameters α, β and γ define the weight associated to the embedding vector of each context.

5.1.3 Find Relationship Instances

After the generation of extraction patterns, BREDS scans the documents once again, collecting all segments of text containing entity pairs whose semantic types match the semantic types of the seed instances. For instance, for the seed <`Google, DoubleClick`> BREDS collects all text segments of text containing a pair of named-entities tagged as organisations.

For each collected segment, the three contexts BEF, BET, AFT are extracted and an instance x is generated, as explained in 5.1.1. Then, the similarity with all previously generated extraction patterns (i.e., clusters) is calculated. If the similarity between x and a pattern Cl is equal or above τ_{sim}, then x is considered a candidate instance, and the confidence score of the pattern Cl is updated.

A pattern's confidence score is computed based on the extracted relationship instances. If an extracted relationship instance contains an entity e_1, which is part of an instance in the seed set, and the associated entity e_2 is the same as in the seed set, the extraction is considered positive (i.e., included in set P). If the relationship contradicts a relationship in the seed set (i.e., e_2 does not match), the extraction is considered negative (i.e., included in set N); if the relationship is not part of the seed set, the extraction is considered unknown (i.e., included in set U). A confidence score is assigned to each pattern p according to its extractions as defined in Equation 5.2:

$$\text{Conf}_\rho(p) = \frac{|P|}{|P| + W_{ngt} \cdot |N| + W_{unk} \cdot |U|} \tag{5.2}$$

In the equation, W_{ngt} and W_{unk} are weights associated to the negative and unknown extractions, respectively.

The pattern which has the highest similarity (i.e., $pattern_{best}$) is associated with i, along with the similarity score (i.e., sim_{best}). This information is kept in a history of *Candidates*. Algorithm 2 describes this process in detail. Note that, as the histories of *Candidates* and *Patterns* are kept through all the bootstrap iterations, new patterns or instances can be added, or the scores of existing patterns or instances can change.

Algorithm 2: Find Relationship Instances.

```
Input: Sentences = {s_1, s_2, s_3, ..., s_n }, τ_sim
Input: Patterns = {Cl_1, Cl_2, ..., Cl_n }
Output: Candidates
for s ∈ Sentences do
    x = create_instance(s)
    sim_best = 0
    p_best = None
    for Cl ∈ Patterns do
        sim = Sim(x, Cl)
        if sim >= τ_sim then
            Conf_ρ(Cl)
            if sim >= sim_best then
                sim_best = sim
                p_best = Cl
    Candidates[x].patterns[p_best] = sim_best
```

5.1.4 Handle Semantic Drift

To control semantic drift, BREDS follows the framework of Snowball, ranking the extracted instances and discarding the least ranked. At the end of each iteration, all the instances in *Candidates* are ranked according to their current confidence scores. The confidence score of an instance is based on the similarity of all scores towards the patterns that extracted it, weighted by the pattern's confidence score:

$$\text{Conf}_\iota(i) = 1 - \prod_{j=0}^{|\xi|}(1 - \text{Conf}_\rho(\xi_j) \times \text{Sim}(C_i, \xi_j)) \tag{5.3}$$

In the above equation, ξ is the set of patterns that extracted i, and C_i is the textual context where i occurred. Only the relationship instances with a confidence score equal or above τ_{min} are added to the seed set, and subsequently used in the next bootstrapping iteration.

5.2 Evaluation

This section describes an experiment evaluating the performance of BREDS against Snowball. The experiment compares the performance of BREDS with Snowball, essentially comparing how word embeddings perform against TF-IDF vectors in bootstrapping relationship instances.

5.2.1 Document Collection Pre-Processing

The document collection used in the experiment consisted of all the news articles published by the AFP and the APW between 1994 and 2010, which are part of the English Gigaword collection (Parker et al., 2011), totalling 5.5 millions of articles.

The pre-processing pipeline is depicted in Figure 5.3. The pipeline is based on the models provided by the NLTK 3.0.0 toolkit (Bird et al., 2009): sentence segmentation[1], tokenisation[2], PoS-tagging[3] and NER. The NER component in NLTK is a wrapper over the Stanford NER 3.5.2 toolkit (Finkel et al., 2005).

[1] nltk.tokenize.punkt.PunktSentenceTokenizer
[2] nltk.tokenize.treebank.TreebankWordTokenizer
[3] taggers/maxent_treebank_pos_tagger/english.pickle

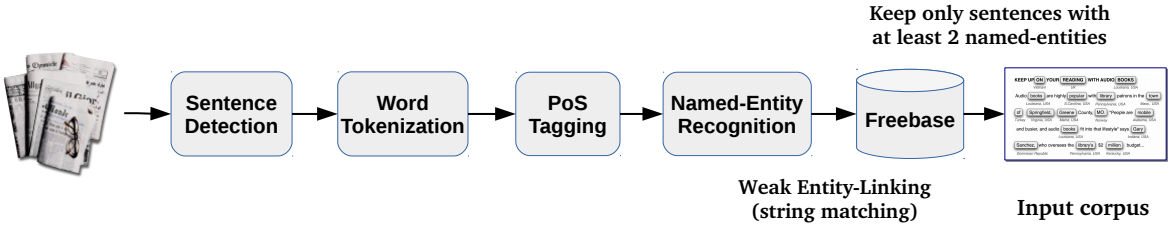


Figure 5.3: Document collection pre-processing pipeline.

The recognized entities in each sentence were associated by simple string matching with entity names in FreebaseEasy (Bast et al., 2014), a processed version of Freebase (Bollacker et al., 2008). FreebaseEasy contains a unique meaningful name for every entity, together with canonical binary relations. This facilitates the use of the relationships expressed in Freebase to evaluate the extracted relationships. For the experiment, only the sentences containing at least two entities mentioned in FreebaseEasy were considered, which corresponds to 1.2 million sentences.

The word embeddings were computed with the skip-gram model (Mikolov et al., 2013a), configured for skip length of 5 tokens and vectors of 200 dimensions using the *word2vec*[4] implementation. The corpus for generating the embeddings was the full set of AFP and APW articles, that is, all the articles published between 1994 and 2010. The TF-IDF representations used by Snowball were calculated over the same articles set.

5.2.2 Evaluation Framework

Evaluating a relationship extractor on a realistically-sized corpora, i.e. containing hundreds of thousands of documents, is not humanly feasible. Bronzi et al. (2012) proposed a framework for automatic evaluation of large-scale relationship extractors, that estimates precision and recall.

Precision is measured by calculating how many relationships extracted by the system are correct. For the experiment, a relationship $<e_1, rel, e_2>$ is considered correct if a knowledge base (KB) also contains the same relationship rel between e_1 and e_2, or if the frequency of occurrence of e_1, rel, e_2 on a large corpus is above some threshold.

To calculate recall, the ground-truth of the input corpus needs to be estimated.

[4] https://code.google.com/p/word2vec/

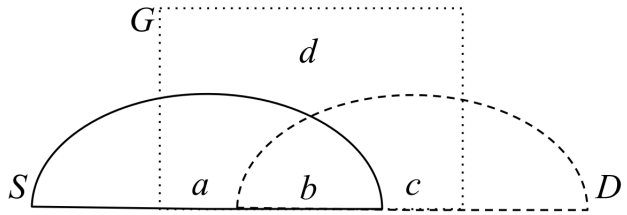


Figure 5.4: Venn diagram of the intersections among system output (S), knowledge base (D) and the estimated ground truth (G).

Figure 5.4 shows the Venn diagram of the intersections between a KB D, the estimated ground truth G and the system output S. Four different sets are generated based on these intersections:

- a contains correct relationships from the system output that are not in the KB;
- b is the intersection between the system output and the KB;
- c contains relationships which are in the KB and are also described in the input corpus, but that were not extracted by the system;
- d contains relationships described in the input corpus that are not in the system output nor in the KB.

Having calculated the size of each region, precision and recall can be computed as:

$$P = \frac{|a| + |b|}{|S|} \tag{5.4}$$

$$R = \frac{|a| + |b|}{|a| + |b| + |c| + |d|} \tag{5.5}$$

To calculate the precision the size of regions a and b needs to be estimated. Region b is calculated by determining whether an extracted relationship is in the KB D. Since entity linking was performed between the entities in the sentences and Freebase, this is obtained by direct string comparison. The size of the region a is estimated by leveraging on proximity PMI (PPMI), which measures the likelihood of observing a relationship given that e_1 and e_2 were observed:

$$\text{PPMI}(\text{e}_1, \text{rel}, \text{e}_2) = \frac{\text{count}(\text{e}_1 \text{ NEAR:}X \textit{ rel } \text{NEAR:}X \text{ e}_2)}{\text{count}(\text{e}_1 \text{ AND e}_2)} \tag{5.6}$$

in the equation above, X is the maximum number of words between the named-entities e_1 and e_2. The intuition behind this heuristic is that relationships with high (relative) frequency occurrence in a large corpus, such as the Web, are more likely to be correct.

The full Gigaword collection without the AFP and APW collections was used as the large corpus to estimate the frequencies needed to calculate the PPMI. This dataset was indexed with a full-text indexing library[5]. The proximity PMI threshold was set to 0.7, and X to 6 tokens.

To estimate recall, the size of regions c and d needs to be estimated. Region c (i.e., relationships which are in the KB and also described in the input corpus) is estimated by first generating a super set G' containing true and false relationships from the input corpus. This is achieved by performing the cartesian product at a sentence level, of all the possible relationships between two entities. The set of true relationships in G' is approximately the same as the set of relationships that a perfect extractor would find in the input corpus (i.e., the sentences containing at least two named-entities linked to Freebase).

Having G', $|G \cap D|$ is estimated by matching all the G' relationships in D. Since $|G \cap D| = |b| + |c|$, then $|c| = |G \cap D| - |b|$. To calculate d we apply the PPMI to the relationships found in the input corpus which are not in the KB (i.e., $G' \setminus D$). We then determine $|G \setminus D|$, and finally $d = |G \setminus D| - |a|$.

5.2.3 Experiment

BREDS and Snowball have several configuration parameters. Table 5.1 shows settings of all these parameters, which are the same both for BREDS and Snowball.

The experiment only considers relationship instances at a sentence level, formed by pairs of named-entities no further away than 6 tokens, and with at least 1 token between them, and context windows of 2 tokens before the first entity and after the second entity. Regarding the single-pass clustering, clusters with less than 2 relationship instances are discarded. In both systems, the bootstrapping ran for 4 iterations. The W_{unk} and W_{unk}

[5] https://pypi.python.org/pypi/Whoosh/

Parameter	Value
BET context maximum number of tokens	6
BET context minimum number of tokens	1
BEF context size	2
AFT context size	2
Minimum number of clustered instances	2
Number of bootstrapping iterations	4
W_{ngt}	2
W_{unk}	0.1

Table 5.1: Configuration parameters used in the experiment.

parameters were set to 2 and 0.1, respectively, based on the results reported by Yu and Agichtein (2003). In what regards the values for the similarity threshold τ_{sim}, and the confidence threshold τ_t, 36 different bootstrapping runs were performed varying these two parameters and combining all possible different values within the interval [0.5,1.0].

The experiment also evaluated two different schemas regarding the weighting of the context vectors. Table 5.2 describes the weighting attributed to each context vector (i.e., BEF, BET, AFT) as used in the similarity Formula 5.1. Conf_1 only considers the vector representing the BET context, while Conf_2 uses the three contexts, giving more weight to the BET context.

Besides Snowball (Classic) as proposed by Agichtein and Gravano (2000) and BREDS, the experiment also considered an alternative implementation of Snowball in which a relational pattern based on ReVerb is used to select the words for the BET context, instead of just using the TF-IDF scores of all the words in that context.

Configuration	Context Weighting
Conf_1	$\alpha = 0.0$
	$\beta = 1.0$
	$\gamma = 0.0$
Conf_2	$\alpha = 0.2$
	$\beta = 0.6$
	$\gamma = 0.2$

Table 5.2: Context vectors weighing for the experiment.

Relationship	Seeds
acquired	<Adidas, Reebok> <Google, DoubleClick>
founder-of	<CNN, Ted Turner> <Amazon, Jeff Bezos>
headquarters	<Nokia, Espoo> <Pfizer, New York>
affiliation	<Google, Marissa Mayer> <Xerox, Ursula Burns>

Table 5.3: Seeds for each relationship type used in the experiment.

The experiment considered four different types of semantic relationships, and, for each relationship type, two seeds were considered, as described in Table 5.3.

Some of the extracted relationships are not correctly evaluated either because there is not sufficient statistical evidence in the corpus to classify them as correct with the PPMI measure, or because the relationship is not present in the KB.

The PPMI estimates whether a given triple <e_1, rel, e_2> is valid or not by measuring the occurrence of two entities regarding a relational phrase *rel* in their proximity, over the global frequency of co-occurrence of the two entities disregarding the words in their proximity.

When analysing the output of the experiment, we notice that some relationship instances are being classified as incorrect, although they are correct. This happens because there is not sufficient statistical evidence to classify them as correct with the PPMI measure. For instance, of all the 996 occurrences of the pair <Microsoft, Steve Ballmer> only four occur with word *head*, and only two with the word *executive*, in their proximity. The evaluated systems output these two triples as part of the extraction process of *affiliation* relationships, but the PPMI formula will give them a low score, $^2/_{996}$=0.002 and $^4/_{996}$=0.004, respectively. Therefore, the extracted relationship is classified as incorrect.

In order to avoid for such triples being wrongly classified as false, I analysed the results and gathered a list of relational phrases (i.e., unigrams and bigrams). Then, in

Relationship	**Valid relational phrases** (*rel*)
acquired	'owns', 'acquired', 'bought', 'acquisition',
founder-of	'founder', 'co-founder', 'cofounder', 'co-founded', 'cofounded', 'founded', 'founders', 'started by'
headquartered	'headquarters', 'headquartered', 'offices', 'office', 'building', 'buildings', 'based in', 'head offices' 'located in', 'main office', ' main offices',
affiliation	'chief', 'scientist', 'professor' 'CEO', 'employer'

Table 5.4: Relational phrases used in calculating the PPMI.

the evaluation process, all the triples with a PPMI value below the defined threshold were checked against the list of relational phrases. If there is a direct match between the relational phrase in the triple and the relational phrase in the list, the triple is considered correct. Table 5.4 presents these phrases, specific for each of the four evaluated relationship types.

Another aspect is that the corpus consists of sentences taken from news articles, which span for a period of over fifteen years (i.e., 1994-2010). Many of the facts written in these articles are no longer correct, since they have changed and the KB used for evaluation only contains the most recent changes. For instance, in 1997, Boeing headquarters were in Seattle, but in 2001 the company moved to Chicago. Also, until 1998 the president of FIFA was João Havelange, but then Sepp Blatter became president. To account for these cases, a manual check was done over some of the extracted relationships and the correctly extracted relationship instances were added to a manually created knowledge base.

The precision and recall values were computed using both the FreeBaseEasy KB and the manually inserted relationships to avoid counting these extractions as incorrect.

Table 5.5 shows, for each relationship type, the best F_1 score of all combinations of τ_{sim} and τ_t values, considering only extracted relationship instances with confidence scores equal or above 0.5. The last row on each table shows the average precision,

BREDS

	Conf_1				Conf_2			
Relationship	**#Instances**	**(P)recision**	**(R)ecall**	**F_1**	**#Instances**	**(P)recision**	**(R)ecall**	**F_1**
acquired	132 (2.1%)	0.73	**0.77**	**0.75**	5 (0.3%)	**1.00**	0.15	0.26
founder-of	413 (6.6%)	**0.98**	**0.86**	**0.91**	261 (16.2%)	0.97	0.79	0.87
headquartered	870 (14.0%)	0.63	**0.69**	**0.66**	614 (38.1%)	**0.64**	0.61	0.62
affiliation	4806 (77.3%)	**0.85**	**0.91**	**0.88**	730 (45.3%)	0.84	0.60	0.70
Weighted Avg. for P, R and F_1	0.83	0.87	0.85	———	0.79	0.63	0.70	

(a) Precision, Recall and F_1 over the extracted instances with the two different configurations of BREDS

Snowball (ReVerb)

	Conf_1				Conf_2			
Relationship	**#Instances**	**(P)recision**	**(R)ecall**	**F_1**	**#Instances**	**(P)recision**	**(R)ecall**	**F_1**
acquired	53 (3.5%)	0.83	0.61	0.70	11 (1.8%)	0.73	0.22	0.34
founder-of	241 (16.1%)	0.96	0.77	0.86	212 (35.3%)	0.97	0.75	0.85
headquartered	891 (59.4%)	0.48	0.63	0.55	322 (53.7%)	0.55	0.42	0.47
affiliation	316 (21.1%)	0.52	0.29	0.37	55 (9.2%)	0.36	0.05	0.08
Weighted Avg. for P, R and F_1	0.58	0.58	0.58	———	0.68	0.50	0.57	

(b) Precision, Recall and F_1 over the extracted instances with the two different configurations of Snowball (ReVerb)

Snowball (Classic)

	Conf_1				Conf_2			
Relationship	**#Instances**	**(P)recision**	**(R)ecall**	**F_1**	**#Instances**	**(P)recision**	**(R)ecall**	**F_1**
acquired	38 (2.8%)	0.87	0.54	0.67	43 (5.0%)	0.77	0.54	0.63
founder-of	222 (16.6%)	0.97	0.76	0.85	187 (21.6%)	0.98	0.73	0.84
headquartered	743 (55.7%)	0.52	0.61	0.57	551 (63.8%)	0.53	0.54	0.54
affiliation	332 (24.9%)	0.49	0.29	0.36	83 (9.6%)	0.42	0.08	0.13
Weighted Av for P, R and F_1	0.60	0.55	0.57	———	0.63	0.54	0.57	

(c) Precision, Recall and F_1 over the extracted instances with the two different configurations of Snowball (Classic)

Table 5.5: (P)recision, (R)ecall and F_1 scores for BREDS, Snowball (ReVerb), and Snowball (Classic) over the total number of instances extracted for four different relationship types.

recall and F_1 weighted by the number of extracted instances. The results show that BREDS achieves better F_1 scores than both versions of Snowball. The F_1 score of BREDS is higher, mainly as a consequence of much higher recall scores, which is due to the relaxed semantic matching caused by using the word embeddings.

For some relationship types, the recall more than doubles when using word embeddings instead of TF-IDF. For instance, in the *affiliation* relationship BREDS outperforms Snowball, extracting a much larger number of correct relationships. This is due to the to high semantic similarity between words/phrases, represented as word embeddings, that mediate the relationship of affiliation between a person and an organization, as opposed to TF-IDF weighted vectors.

For the *acquired* relationship, when considering $Conf_1$, the precision of BREDS drops compared with the other versions of Snowball, but without affecting the F_1 score, since the higher recall compensates for the small loss in precision.

Regarding the context weighting configurations, $Conf_2$ produces a lower recall when compared to $Conf_1$. This might be caused by the sparsity of both BEF and AFT contexts, which contain many different words that in most cases do not contribute to capture the relationship between the two entities. Although, sometimes, the phrase or word that indicates a relationship occurs on the BEF or AFT contexts, it is more often the case that these phrases or words occur in the BET context.

Comparative performance results of Snowball (Classic) and Snowball (ReVerb) suggests that selecting words based on a relational pattern to represent the BET context, instead of using all the words, works better for TF-IDF representations.

Analysing the relationships extracted by each system, one can notice that word embeddings generate more extraction patterns. Table 5.6 shows the most common words occurring in the BET context for the top-ranked instances extracted by each system.

In terms of the threshold parameters τ_{sim} and τ_t, there is no global configuration that yields the best results across all the evaluated relationship types. As explained before, Table 5.5 shows the best F_1 scores from all the possible combinations of the parameters τ_{sim} and τ_t values within the interval [0.5,1.0], corresponding to 36 different bootstrap runs for each relationship type.

Analysing the configurations settings for the four relationship types, the best performance of BREDS is achieved when τ_{sim} is set to 0.6 or 0.7. With a similarity threshold

Relationship	BREDS	Snowball (ReVerb and Classic)
acquired	acquired acquisition purchased by 's purchase of	acquisition acquired
founder-of	founder co-founder co-founders founded	founder
headquartered	based in headquarters in headquartered in offices in	based in headquarters in
affiliation	president chief executive vice-president general manager CEO chairman	president chief executive

Table 5.6: Most common phrases/words included in the patterns which extracted the top ranked instances.

of τ_{sim}=0.5 the system extracts too many incorrect relationships, and with $\tau_{sim} \geq 0.8$, it is too conservative and it becomes hard to expand the seed set.

Regarding the parameter τ_t, which controls the confidence threshold of relationships instances to be added to the seed set, when τ_{sim} is set to values of 0.6 or 0.7 the best F_1-scores are achieved when τ_t is set to 0.7 or 0.8.

5.3 Conclusions

Overall, the BREDS approach for representing a sentence expressing a relationship as a unique embedding vector achieves good results, outperforming Snowball, in an evaluation experiment. The main advantage of BREDS over Snowball is the relaxed semantic matching due to the word embeddings, which causes BREDS to learn more extraction patterns and consequently achieving a higher recall.

There are several parameters that affect the performance of the system. In addition to the heuristic threshold aspect, the initial seed set of relationship instances has an impact on the performance of BREDS. In the experiment, there was no formal or exhaustive procedure to select the seed instances. Instead, for each relationship type, I randomly choose a few entities in relationships from the KB, then calculated their co-occurrence frequency in the corpus, selecting those seen at least 5 times. The corpus used in the experiment consisted of 5.5 millions news articles published between 1994 and 2010, which are part of the English Gigaword collection (Parker et al., 2011).

In the experiment, BREDS ($Conf_1$) achieves the best weighted average F_1, with a score of 0.85, outperforming all the other systems and configurations. It is also worth noticing that, in this experiment BREDS, achieved a good balance between precision and recall, with weighted average scores of 0.83 and 0.87, respectively.

Although achieving better results than Snowball, BREDS still has some limitations. Since the identification of relationships is based only on part-of-speech tags, BREDS cannot capture capture long-distance relationships. Most of these cases could only be handled by computing the syntactic dependencies of the words in a sentence.

BREDS, being a bootstrapping system, can be used to generate relationships to be used as an input to a supervised RE system. In the next Chapter, I introduce TREMoSSO, a framework which performs large-scale relationship extraction integrating both BREDS and MuSICo.

The software implementations of BREDS and Snowball, as used in the experiments presented in this chapter, are publicly available on-line at https://github.com/davidsbatista/BREDS and https://github.com/davidsbatista/Snowball respectively.

6

Large-Scale Relationship Extraction

TREMoSSo (Triples Extraction with Min-Hash and diStributed Semantics) is a framework integrating MuSICo (see Chapter 4) and BREDS (see Chapter 5) along with other NLP tools. TREMoSSo performs large-scale extraction of semantic relationships based on similarity search and the distributional semantics, requiring little or no human supervision. This chapter describes the architecture of TREMoSSo, and reports an experiment, where it was applied to a large collection of news articles.

6.1 TREMoSSo Architecture

TREMoSSo relies on MuSICo and BREDS. MuSICo is a scalable on-line supervised classifier, which can extract instances of many different relationship types. Given a sentence, MuSICo classifies it as one of many possible relationship types based on its database of relationship examples. Being a supervised classifier, it needs training data. BREDS is a bootstrapping system based on word embeddings, which can automatically collect large amounts of instances of a specific relationship type, given just a few seed instances.

The architecture of TREMoSSo is depicted in Figure 6.1. The two main compo-

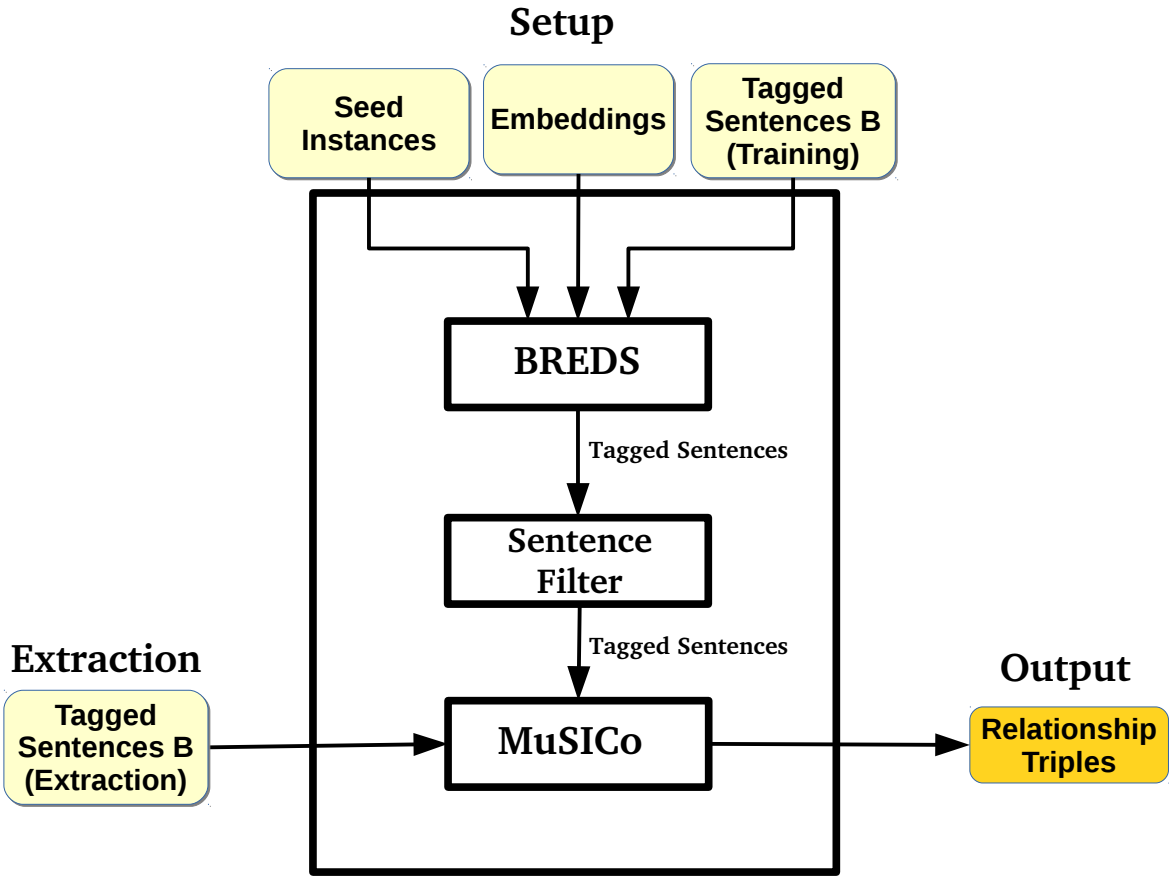


Figure 6.1: TREMoSSo architecture.

nents work in tandem: BREDS generates data for training (i.e., sentences holding relationships between named-entities) and MuSICo classifies new sentences. The sentences, collected by bootstrapping, hold relationship instances and are used by MuSICo to populate its database of examples, which is the basis of MuSICo's classification schema. BREDS could also be used to directly extract relationship instances from a document collection, without relying on MuSICo, but a different bootstrapping process would have to be performed for each relationship type. By relying on MuSICo, several different types of relationships can be extracted with a single-pass over a document collection. Plus, new relationships instances can be continuously added to MuSICo's database, since it is an on-line classifier. There is an optional filter between the BREDS and MuSICo. This component takes the output of BREDS and selects only sentences containing valid relationships. TREMoSSo runs in two main processing steps:

1. **Setup:** BREDS collects training data from tagged sentences by bootstrapping, based on a few seeds and word embeddings. Next, MuSICo extracts features from the training data, and calculates the min-hash signatures. Finally the min-hash signatures are indexed by the Locality-Sensitive-Hashing (LSH) component of MuSICo. This step needs to run for each different relationship type.

2. **Extraction:** Large-scale relationship extraction using the sentences indexed in MuSICo's LSH tables to compare and classify the sentences from a document collection.

The input for TREMoSSo consists of:

1. Seed instances.

2. Previously trained word vector embeddings.

3. A set of sentences, tagged with named-entities, for generating training data.

4. A set of sentences, tagged with named-entities, from which it extracts different types of relationship instances.

6.2 Experiment Preparation

This section details the experimental setup and the preparation of the input dataset, including: the pre-processing of a collection of English news articles, the generation of word embeddings, the relationship types considered, the seeds selected, the tuning of parameters and a description of the framework used to evaluate the relationships extracted.

Figure 6.2 shows how the English Gigaword collection (Parker et al., 2011) was spilt into different datasets used as input for TREMoSSO.

6.2.1 Setup and Extraction Datasets

The English Gigaword is a comprehensive archive of newswire text, it contains close to 10 million of news articles published between 1994 and 2010 by seven distinct international sources of English newswire: Agence France-Presse (AFP), Associated Press World (APW), the New York Times (NYT), Central News Agency of Taiwan English Service (CNA), Xinhua News Agency English Service (XIN), Los Angeles Times/Washington Post Newswire Service (LTW), and Washington Post/Bloomberg Newswire Service (WPB).

For the experiment I created two document sets:

Set A: the articles published by the AFP and the APW.

Set B: the articles published by the NYT.

The datasets are disjoint; the training data is exclusively from Set A and is used to extract relationships from Set B. The same NLP pipeline used for the experiment of Chapter 5, depicted in Figure 5.3, processed both sets. This included, segmenting the articles into sentences and then applying the Stanford NER (Finkel et al., 2005) to tag organisations, persons and locations, and linking by direct string matching the entities to the KB. In this experiment, I generated a KB from three different knowledge bases: FreebaseEasy (Bast et al., 2014), YAGO (Hoffart et al., 2013) and DBpedia (Lehmann et al., 2015). The reason for merging these three knowledge bases (KBs) lies in that although the KBs might hold the same relationships, they can be expressed using different surface strings to refer to the same entities. For instance, the relationship <Bill

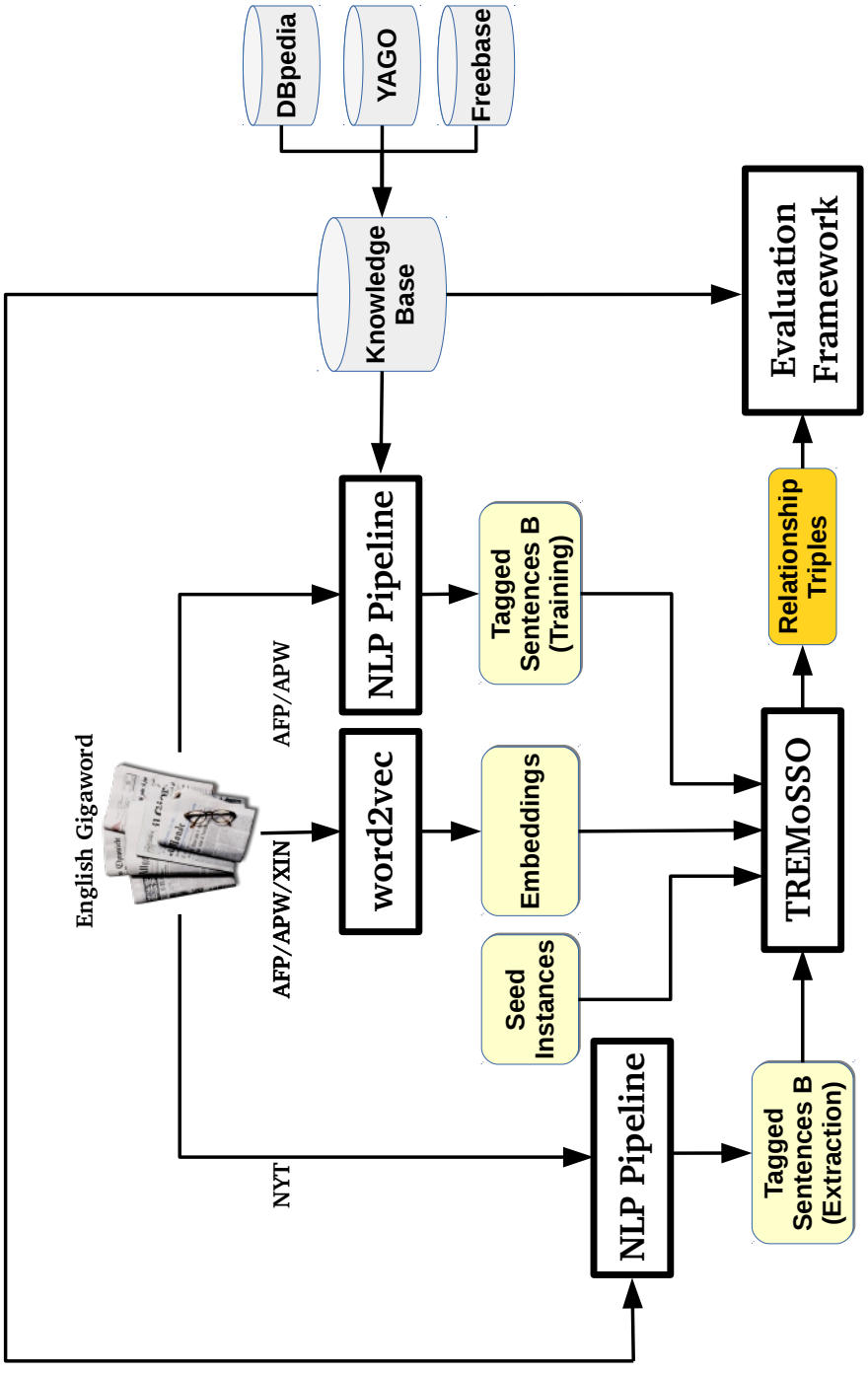


Figure 6.2: Pre-processing of the English Gigaword collection for TREMoSSO.

	Set A	Set B
News Source	AFP and APW	NYT
# documents	5 587 401	1 962 178
# sentences	52 871 331	27 761 097
# selected sentences	2 012 601	848 134

Table 6.1: Statistical characterization of the datasets used in the experiments.

`Gates`, *founder*, `Microsoft`> is contained in all three KBs. However, different KBs use different surface strings. FreebaseEasy uses the string *Microsoft Corporation*, while in DBpedia the company is simply referred to as *Microsoft*.

Using these three KBs enables the collection of more sentences. Then, only the sentences containing at least two entities mentioned in any of the knowledge bases were considered. This transforms Set A and Set B into the Tagged Sentences A and Tagged Sentences B, which are taken by TREMoSSo as input.

Having only sentences with named-entities represented in a KB is a necessary pre-processing step. It enables the training data, gathered by bootstrapping to be filtered, selecting only the correct relationships as training data for MuSICo. Also, it allows to evaluate the output of TREMoSSo, and to calculate the performance over the Set B. Table 6.1 describes the statistical characterization of both sets.

6.2.2 Word Embeddings

BREDS relies on word embeddings to bootstrap relationship instances. In this experiment, I used the Skip-Gram model (Mikolov et al., 2013a) to compute the word embeddings, configured for skip length of 5 tokens and vectors of 200 dimensions using the *word2vec*[1] implementation. The corpus from which the embeddings were generated consisted of all the news articles published by the AFP, the APW and XIN (see Table 6.2).

6.2.3 Seeds

The seeds used to bootstrap relationship are dependent on relationship types that one which to extract. The experiment considered the following types of semantic

[1] https://code.google.com/p/word2vec/

	# Documents
AFP	2 479 624
APW	3 107 777
XIN	1 744 025
Total	7 331 426

Table 6.2: News articles collections used to generate the word embeddings.

relationship between three types of name-entities, organization (ORG), person (PER) and location (LOC):

affiliation: relates a person with an organization; the person might work for the organization or is a representative of that organization (e.g. a professor affiliated with an university, a president or a worker of a company)

has-part-in: indicates that one organization owns in total or in part another organization (e.g, a company was acquired by another company, a company is a subsidiary of another company, or a company holds shares of another organization)

founder-of: indicates the person that founded an organization

installations-in: indicates the geographic location of an organization installations (e.g.: headquarters, offices, or any other form of compound that represents the organization)

spouse-of: relates two persons living in a marriage or a civil union

studied-at: indicates that a person graduated, studied or currently studies at an organization

Table 6.3 summarizes the relationship types to be extracted in terms of the entities' semantic types and according to the direction of the relationship. The reason for choosing these specific relationship types is related to the type of named-entities that the used NER system can identify, and the relationship types among these types of named-entities (i.e., PER, LOC, and ORG) that can be mapped to relationships in the knowledge base.

Relationship	Direction
affiliated-with	(PER,ORG) (ORG,PER)
owns/has-parts-in	(ORG_1,ORG_2) (ORG_2,ORG_1)
founded-by	(ORG,PER) (PER,ORG)
has-installations-in	(ORG,LOC) (LOC,ORG)
spouse	(PER,PER)
studied-at	(PER,ORG) (ORG,PER)

Table 6.3: Evaluated relationships and arguments type.

Table 6.4 describes the seeds used to bootstrap relationship instances for each relationship type. No formal process was followed to evaluate which seeds would achieve the best results. This is not an easy task, given the size of the document collection and all the possible pairs of named-entities. Therefore, the selection of seeds was performed ad-hoc, and this approach is probably non-generalizable to other bootstrapping problems.

The process consisted on pre-selecting, for each relationship type, a few pairs of named-entities, mostly based on common sense, and confirming that both co-occurred in the corpus. Next, through a series of queries performed over Set A, I selected only those which had more co-occurrences with a phrase that expressed the relationship of interest than with any other phrase.

The considered phrases explicitly express the relationship of interest. For instance, for the *has-installations* relationship, apart from *headquarters*, it also aims at extracting other physical locations of organisation assets. Therefore, pairs of entities were also selected as seeds which occurred with phrases like "*plant in*" or "*fabric in*" in the corpus.

Relationship	Direction	Seeds
affiliated-with	(ORG,PER)	<Google, Eric Schmidt> <OPEC, Edmund Daukoru> <UEFA, Michel Platini> <WikiLeaks, Julian Assange>
	(PER,ORG)	<Dominique Strauss, IMF> <Henning Kagermann, SAP> <Gianni Agnelli, Fiat> <John Sauven, Greenpeace>
owns/has-parts-in	(ORG_1,ORG_2)	<Adidas, Reebok > <Volkswagen, Audi>
	(ORG_2,ORG_1)	<Mercedes-Benz, Daimler AG> <Airbus, EADS> <Audi, Volkswagen>
founded-by	(ORG,PER)	<CNN, Ted Turner> <Google, Sergey Brin>
	(PER,ORG)	<Dietmar Hopp, SAP AG> <Chung Ju-yung, Hyundai>
has-installations-in	(ORG,LOC)	<Opel, Spain> <Nokia, Espoo> <Volkswagen, Portugal> <Siemens, Munich>
	(LOC,ORG)	<Berlin, Deutsche Welle> <New York, NBC News> <Miami, National Hurricane Center> <Seoul, Samsung Group> <San Jose, Cisco> <London, Unilever>
spouse	(PER,PER)	<George W. Bush, Laura Bush> <Jennifer Lopez, Marc Anthony> <Britney Spears, Kevin Federline>
studied-at	(PER,ORG)	<Barack Obama;Columbia University> <Barack Obama;Harvard University> <Al Gore;Vanderbilt University> <Al Gore;Harvard University>
	(ORG,PER)	<Stanford, Larry Page> <Harvard, Barack Obama> <Harvard, Mark Zuckerberg> <Harvard, Steve Ballmer>

Table 6.4: Seeds per relationship type.

6.2.4 BREDS and MuSICo Configuration

The function of BREDS within TREMoSSo is to collect sentences holding a relationship of interest. The τ_{sim} parameter in BREDS controls the similarity threshold between instances and patterns, and for the experiment it was set to $\tau_{sim} = 0.6$. This choice is somehow relaxed. However, this is deliberate, since even if BREDS extracts some invalid relationship instances, the output of BREDS will be filtered and only the correct relationships will be selected as training data for MuSICo.

The bootstrapping ran four iterations, for each relationship type, and the context weighting parameters were set to $\alpha = 0.0$, $\beta = 1.0$ and, $\gamma = 0.0$ (i.e., BREDS only considered the words between the named-entities). This choice of parameters was based on the experiment described in Chapter 5.

For this experiment, a different version of MuSICo, as presented in Chapter 4, was implemented. This version is coded in Python and the bands in the Locality-Sensitive Hashing (LSH) structure are implemented with REDIS (Carlson, 2013), a in-memory key-value persistence storage structure. Also, in this version, both the indexing and classification phases are performed in parallel, leveraging multi-core hardware architectures. In the classification phase, MuSICo considers every pair of named-entities in a sentence, generating three contexts (i.e., BEF,BET and AFT). From each context it extracts the following features:

- The semantic type of each entity.
- ReVerb Patterns from the BET context.
- ReVerb Patterns from the BET context with passive voice.
- ReVerb Patterns from the BET context with verbs normalized.
- Verbs occurring in all contexts except for auxiliary verbs.
- All the nouns from the BET context.
- All the prepositions from BET context.
- n-grams of characters from the BEF,BET and AFT contexts.

Regarding MuSICo's configuration signatures, bands and kNN, the experiment only took into consideration the best configuration results reported on Chapter 4.

- min-hash bands of 25 and 50;
- signatures of size 400, 600 and 800;
- consider the 3, 5 and 7 nearest neighbours;

6.3 Running TREMoSSo

As described in Section 6.1 TREMoSSo runs in two main processing steps. The **Setup** step consists of the following operations:

- Bootstrap relationship instances.
- Filter the relationship instances gathered by bootstrapping.
- Index the relationship instances in MuSICo's LSH tables.

The **Extraction** step consists of only one operation:

- Extract relationship instances based on the populated MuSICo's LSH tables.

The Setup step starts by invoking BREDS giving the document collection and seeds to process as input. In the experiment, this operation was repeated 11 times with different sets of seeds for each relationship types/direction, as shown in Table 6.4. This operation resulted in a set of sentences. Next, a filter was applied to the sentences by invoking the evaluation framework presented before in Chapter 5. This operation was also repeated 11 times, each to filter the collected sentences for relationship type. Finally, after the filtering phase, the selected sentences are indexed in MuSICo's LSH tables. This operation is performed by invoking MuSICo in indexing mode.

After the Setup phase is complete, the system is ready to perform multi-class extraction of relationships. In the extraction phase, MuSICo was invoked in classification mode.

Relationship	Freebase	DBpedia	Yago
affiliation	employment	-	isAffiliatedTo
	governance	-	worksAt
	leader_of	-	-
owns/has-parts-in	acquired	subsidiary	owns
founded-by	founded	founder	created
has-installations-in	place_founded	headquarter	isLocatedIn
	-	location	-
	-	locationCountry	-
	-	locationCity	-
located-in	location_citytown	capital	isLocatedIn
		largestCity	
spouse-with	married_to	-	-
	spouse_partner	-	-
studied-at		almaMater	graduatedFrom

Table 6.5: Relationships from the Freebase, DBpedia and Yago used for evaluation.

6.4 Experiment Results

This section reports the results and performance of each of the two running phases of TREMoSSo: setup and classification.

I used the framework proposed by Bronzi et al. (2012), described in Chapter 5, to evaluate the extracted relationships. This framework depends on a KB holding the relationship types to be evaluated. To build my evaluation KB, I selected relationships types from the three KBs described before. The relationship types from each KB are described in Table 6.5.

The evaluation framework leverages a large corpus to estimate the proximity Pointwise Mutual Information (PPMI) (see Formula 5.6). For this, I used all the collections from Gigaword except the ones contained in sets A and B. This corresponds to the CNA, XIN, LTW, WPB collections (see Table 6.6). The StanfordNER (Finkel et al., 2005) tagged these four collections and then I created an index allowing to perform queries to calculate the PPMI.

Besides evaluating the final output of TREMoSSO, this framework was also used as the filter, between the output of BREDS and the input for MuSICo.

	# Documents
CNA	145 317
LTW	411 032
WPB	26 143
XIN	1 744 025
Total	2 326 517

Table 6.6: Collections used to create the full text index to calculate the PPMI.

6.4.1 TREMoSSo Setup Evaluation

The setup phase essentially corresponds to bootstrap relationship instances, filtering only the correct ones, and finally indexing them in the LSH tables of MuSICo. Table 6.7 shows precision recall and F_1 computed by the evaluation framework for the relationships bootstrapped by BREDS.

The results vary for different relationship types, and, for the same relationship type, the results also vary according to the direction. The *affiliated-with* relationship, considering the (ORG,PER) direction, achieves the best F_1. The news articles contain many references to affiliations of persons with organisations, and these are well captured, since many phrases/words have a high similarity among them, e.g.: *CEO*, *president*,

Relationship	Direction	Precision	Recall	F_1
affiliated-with	(ORG,PER)	0.97	0.82	0.89
	(PER,ORG)	0.52	0.53	0.53
owns	(ORG_1,ORG_2)	0.51	0.71	0.60
	(ORG_2,ORG_1)	0.41	0.47	0.44
founded-by	(ORG,PER)	1.00	0.76	0.86
	(PER,ORG)	0.87	0.33	0.48
has-installations-in	(ORG,LOC)	0.82	0.55	0.66
	(LOC,ORG)	0.93	0.58	0.71
spouse	(PER,PER)	0.59	0.59	0.59
studied-at	(PER,ORG)	0.89	0.74	0.81
	(ORG,PER)	0.88	0.41	0.56

Table 6.7: Precision, Recall and F_1 for evaluated relationships and directions considered.

chief, *financial chief*.

The *founded-by* relationship is mostly expressed through the verb and sometimes noun *founder* and *co-founder*.

The *owns* relationship is more difficult to handle. Detecting the correct direction, in particular, is hard. BREDS learns several patterns, such as: *owned by*, *a subsidiary of* or *unit of*, which capture <ORG_2, *owned-by*, ORG_1> relationships, but all these patterns have a high similarity with other patterns, such as *unit*, which captures <ORG_1, *owns*, ORG_2> relationships. Also, simply detecting the passive voice to discover the direction for this relationship type is not sufficient. For instance, the direction of the relationship changes by simple adding an *of* to the pattern *unit*. Another problem is due to a high similarity with patterns such as *merged with*, which do not express that one organisation owns another. Also, many entities tagged as organisations correspond to sports teams, and there are many sentences that express that one team bought an athlete from another team, like:

Bob Wickman *his fifth game since* **Cleveland** *acquired him from* **Milwaukee** .

The relationship *has-installations* is problematic for the direction (LOC,ORG), due to the way the patterns are ranked. For each extracted <e_1, rel, e_2> instance, in which e_1 is in the seed set, but e_2 does not correspond to the e_2 in the extracted relationship, BREDS classifies the extraction as negative (see Formula 5.2). For instance, given a few seeds, such as:

<Berlin; Deutsche Welle>
<New York; NBC News>
<London; Unilever>

BREDS can, for instance, learn the following extraction patterns:

LOC_{e_1} **headquarters of** ORG_{e_2}
LOC_{e_1} **- based** ORG_{e_2}
LOC_{e_1} **offices of** ORG_{e_2}

Next, based on these extraction patterns, BREDS can extract instances, such as:

<London; BBC>
<London; HSBC>
<Paris; EADS>
<London; Greenpeace>
<New York; United Nations>

among many others. The problem is with the relationship instances that share a location with a seed instance, in which e_1 is in the seed set, but associated with another e_2. Each will be marked as a negative extraction, although most of them are correct. This will then cause the confidence score of the pattern to be low (see Formula 5.2). Consequently, this will also cause the confidence score associated with each extracted instance to also have a low confidence score (see Formula 5.3), and no new seeds are added to the seed set. Something similar happens for the relationship *studied at* relationship, for the direction (ORG,PER).

There are two ways to cope with this problem: adding more seed instances or lowering the confidence threshold for instances to be added to the seed set.

The *spouse* relationship also achieves good results, although BREDS wrongly learns some extraction patterns based on words/phrases such as: *'divorced'*, *is the widow of.*

More generally, and by analysing all the results, most of relationship instances are wrongly extracted due three main causes of errors: shallow parsing, named-entity recognition and relational patterns.

Shallow parsing

One type of error is caused by the shallow parsing approach, which fails to detect long distance relationships between named-entities. For instance, in the sentence:

The **ICJ**, *which is part of the* **United Nations** *is based in* **The Hague**.

when bootstrapping *has-installations* relationship instances, BREDS discovers, among other patterns, the pattern *based in*. This pattern matches the sentence above, between the entities **United Nations** and **The Hague**, causing BREDS to wrongly extract the relationship <United Nations, *has-installations*, The Hague> .

Named-Entity Recognition

Errors originated by the NER sub-system, also cause the extraction of wrong relationship instances. For instance, considering the following tagged sentences:

Barotillo, a 22-year-old native of the **Philippines** *now based in* **Australia**.

An inspection of six **Boeing** *747-200s owned by* **Lufthansa** *uncovered no problems.*

in the first sentence, the named-entity **Philippines** is wrongly classified as an organization, resulting in the system wrongly extracting the instance <`Philippines`, *has-installations*, `Australia`>. In the second sentence, the NER system classified the string *Boeing* as an organization, but *Boeing* belongs to the sequence *Boeing 747-200s* which is refers six airplanes. When analyzing the sentence, BREDS wrongly extracts the relationship instance <`Boeing`, *owned-by*, `Lufthansa`>.

Relational Patterns

Another type of error is related to the matching of relational patterns represented as an unique embedding vector. Considering, as before, the relational pattern *based in* to extract *has-installations* relationship instances, that pattern causes BREDS to wrongly extract instances from the following sentences:

Anthony Shadid *is an* **Associated Press** *newsman based in* **Cairo**.

Ravi Nessman *is an* **Associated Press** *correspondent based in* **Jerusalem**.

In other cases, relational patterns have a high similarity, for the same relationship type, but in a different direction. For instance, for the relationship type <ORG_1, *owns*, ORG_2> which states that ORG_2 is part of ORG_1, a learned pattern is *'unit'*. For instance,

Unions at **General Motors** *unit* **Opel** *warned about a widespread strike action.*

but for the other direction <ORG_2, *owned-by*, ORG_1> a typical pattern learned by BREDS is *'a unit of'*. For instance,

Mercedes-Benz *a unit of* **Daimler-Benz**.

Both embedding vectors, generated for the relational patterns *unit* and *a unit of* , have high similarity. Some of these cases are due to generation of the unique vector by BREDS, which removes stop words before summing the embeddings of each word.

Another cause of wrong extractions related to the relational patterns is due to the use of negation. For instance, when bootstrapping *studied-at* relationship instances, BREDS learns the patterns *'graduated from'*, and *'never graduated from'*, which have high similarity between them, although the semantics are completely different.

Table 6.8 shows the number of relationship instances for each relationship type that were considered correct by the evaluation framework, and the total number of relationship instances. These relationships compose the training set that is indexed in MuSICo's database of examples.

The generated training data are highly skewed. The relations *affiliated-with* and *has-installations-in* account for almost 88% of all the relationship instance examples. The relation *affiliated-with* itself accounts for more than 50% of the examples, and *has-installations-in* more than 20%. Others have too few examples compared with the total number of gathered examples. For instance, the relationship *studied-at* has less than 1% of the total number of relationship instances, and *owns-has-parts* close to 3%.

6.4.2 Extraction

Next, MuSICo extracted relationships from Set B, containing total of 848,134 sentences. This step consisted on analysing all possible pairs of named-entities in a sentence. For each pair within a certain distance, MuSICo extracts the features, calculates the min-hash signatures, and using the the signatures finds the most similar examples in its database and classifies the relationship. The framework of Bronzi et al. (2012) calculated the performance of the extraction in terms of Precision, Recall and F_1 for each relationship type. Table 6.9 shows the results for the MuSICo's configuration

Relationship	Direction	# Relationship Instances
affiliated-with	(PER,ORG)	2 708 (13.9%)
	(ORG,PER)	9 775 (50.2%)
owns/has-parts-in	(ORG_1,ORG_2)	501 (2.6%)
	(ORG_2,ORG_1)	100 (0.5%)
founded-by	(ORG,PER)	802 (4.1%)
	(PER,ORG)	92 (0.5%)
has-installations-in	(ORG,LOC)	4 259 (21.9%)
	(LOC,ORG)	362 (1.9%)
spouse	(PER,PER)	725 (3.7%)
studied-at	(PER,ORG)	104 (0.5%)
	(ORG,PER)	36 (0.2%)
Total		19 464 (100%)

Table 6.8: Number of relationship instances per relationship type and direction in Tagged Sentences A, used as MuSICo's training data.

which achieved the best performance: min-hash signatures of size 400, 50 bands and considering the 5 nearest neighbours.

There is relation between the performance of the classifier for a specific relationship type and the number of examples indexed for that relationship type. Comparing the F_1 scores in Table 6.9 with the number of examples, as show in Table 6.8, one can notice a trend: the higher the relative number of examples for a relationship type, the better the classifier's performance for that relationship type.

6.4.3 Running Times of MuSICo

The running times of MuSICo for the two phases were also measured. The operation of extracting the features mentioned above from 19,464 sentences, part of the Tagged Set A, and indexing them in MuSICo's LSH took 572 seconds using 12 cores, considering min-hash signatures of 400 and 50 bands. By leveraging multi-core hardware architectures, and using the 12 cores, MuSICo processed, on average, 34.1 sentences per second. Note that in the training data each sentence contains just a single relationship.

In the classification phase, MuSICo analysed a total of 848,134 sentences, part of the

Relationship	Direction	Precision	Recall	F_1
affiliated-with	(ORG,PER)	0.490	0.736	0.588
	(PER,ORG)	0.070	0.293	0.113
owns/has-parts-in	(ORG_1,ORG_2)	0.423	0.194	0.265
	(ORG_2,ORG_1)	0.233	0.095	0.135
founded-by	(ORG,PER)	0.327	0.191	0.241
	(PER,ORG)	0.036	0.020	0.026
has-installations-in	(ORG,LOC)	0.836	0.655	0.734
	(LOC,ORG)	0.386	0.182	0.248
spouse	(PER,PER)	0.486	0.139	0.217
studied-at	(PER,ORG)	0.096	0.394	0.154
	(ORG,PER)	0.250	0.067	0.105

Table 6.9: Precision, Recall and F_1 for evaluated relationships and directions considered in Tagged Sentences B.

Tagged Set B. Since each sentence MuSICo considers all possible pairs of relationships, this resulted in a classification of 980,106 relationships. The classification operation including extracting features, calculating the min-hash signatures, and computing the similarity with the 5 closest neighbours, took approximately 6,050 seconds using the same 12 cores. In this scenario MuSICo processed on average, 3.2 sentences per second. Note, however, that each sentence can contain an arbitrary number of relationships.

6.5 Conclusions

This chapter presented TREMoSSo, a framework for performing large-scale relationship extraction, based on similarity search and distributional semantics. TREMoSSo requires little or no human supervision. The chapter illustrated, through an experiment, how BREDS can be used to populate the database of examples of MuSICo. Then, with its database of examples populated, MuSICo can extract different relationship types with a single pass from a large document collection.

A limitation to TREMoSSo's extraction capability is related to the number of indexed relationship examples per type. Relationship types with a considerably smaller number of examples then the other types will yield a smaller number of extractions, or

incorrect extractions.

Starting from only 40 seed relationship instances of 11 different types, it was able to extract around 4,700 correct relationship instances from a text with about 850,000 sentences.

The software implementation of TREMoSSo and MuSICo used in this experiment are available on-line at http://www.github.com/davidsbatista/TREMoSSo.

7

Conclusions

The goal of the research described in this thesis was to address challenges regarding the scalability of relationship extracting software for crawling large collections of documents, and how bootstrapping methods can be used to automatically generate training data. I addressed these two challenges separately, designing and implementing new algorithms and performing experimental evaluations in both cases. The proposed solutions for these two challenges were later combined in a scalable framework for semantic relationship extraction which requires little or no human supervision. This chapter reviews the main findings of this dissertation, discusses the limitations of the proposed solutions, and outlines directions for future work.

7.1 Main Findings

I explored new methods for semantic relationship extraction based on two research questions. The first addressed the scalability of relationship extraction:

Can supervised large-scale relationship extraction be efficiently performed based on similarity search ?

To answer this first question, I have developed MuSICo, a new supervised classifier, based on the idea of nearest neighbour classification, which is trained with textual representations of relationships. Supervised machine learning algorithms for classification infer a statistical model based on annotated examples. Alternatively, instead of learning a model, MuSICo's algorithm can find the most similar examples in a database, based on k nearest neighbour (kNN) search. Based on the similarities scores with the closest examples, the classifier makes a classification decision. This approach, however, relies in finding the most similar examples in a database in a fast and efficient way.

MuSICo was empirically evaluated through experiments with well known datasets from three different application domains, showing that the task can be performed with high accuracy, using a simple on-line method based on similarity search, that is also computationally efficient. Through experiments, it was shown that this classifier scales in terms of processing time almost linearly as the size of the dataset grows.

The second question addressed the problem of bootstrapping relationships instances from a large collection of documents:

Can distributional semantics improve the performance of bootstrapping relationship instances ?

To answer the second question, I have developed BREDS, a software for bootstrapping relationship extractors, which relies on distributional semantics to learn patterns for extracting relationship instances. The distributional hypothesis is exploited through word embeddings, i.e., dense vector representations for each word. The performance of BREDS was evaluated by comparing it to a baseline system which relies on TF-IDF weighted vectors. The obtained results show that relying on word embeddings achieves better performance than a similarly configured TF-IDF baseline. The relative increase in performance is mostly due to the higher recall, which is caused by the relaxed semantic matching enabled by computing similarities based on word embeddings.

7.2 Limitations and Future Work

The research activities described in this dissertation were mainly in the design, development, and evaluation of two tools. I will address their limitations and ideas for future developments separately.

7.2.1 MuSICo

MuSICo is a min-hash-based method for fast extraction of semantic relationships from textual documents. MuSICo's performance evaluation has shown that its scalable and, for some datasets, it achieves competitive results with the state-of-the-art. Nevertheless, some of the identified limitations could be overcome.

MuSICo relies on part-of-speech (PoS) tags and lexical features to represent relationships. However, these features do not capture long distance relationships in a sentence. Some of the state-of-the-art systems use graph kernel methods for relationship extraction, exploring the similarity between graph-based representations of the relationship instances, derived from both lexical information and from constituency or dependency parsing trees.

Teixeira et al. (2012) proposed an algorithm for graph fingerprints generation based on min-hash values vectors of the graph substructures. The algorithm enables efficient computation of the similarity between large sets of graphs using a small amount of data. This method could also be applied in MuSICo, allowing it to perform similarity search by relying on graph-based representations of the syntactic dependencies extracted from a sentence. This would enable MuSICo to capture long-distance relationships.

Since the seminal work of Broder (1997), there have been considerable theoretical and methodological developments on the application of minwise hashing techniques. In future work, the b-bit minwise hashing approach by Li and König (2010) for improving storage efficiency on very large datasets, or the extension proposed by Chum et al. (2008) for approximating weighted set similarity measures, could be considered.

7.2.2 BREDS

Through an experimental evaluation, BREDS outperformed a baseline system based on TF-IDF vector weights. Nevertheless, it has limitations that are at the heart of many of the wrongly extracted relationship instances. I identified four aspects in which BREDS could be improved.

Sentence parsing

A crucial limitation in BREDS is that it performs a shallow parsing of the sentence (i.e., PoS-tags), limiting the type of relationships that can be captured. Simply

PoS-tags are not enough to capture long distance relationships between entities in a sentence. Plus, relying solely on PoS-tags can introduce errors in the extraction of relationships. BREDS uses a relational pattern, introduced by ReVerb (Fader et al., 2011), to discover relationships between named-entities, for instance, given the following sentence:

The **ICJ**, *which is part of the* **United Nations** *is based in* **The Hague**.

BREDS wrongly extracts the relationship <`United Nations`, *has-installations*, `The Hague`>. This type of error can be avoided by computing the syntactic dependencies among words in the sentence, and relying on the words in the path between the two entities to identify a relationship. Bunescu and Mooney (2005b) showed that the path of dependencies between the entities are good indicators of the relationship.

Figure 7.1a shows the syntactic dependencies for the sentence above. By analysing the dependencies path one notices that the entities **ICJ** and **The Hague** are connected through the word *based*, which is a good indication of the *headquartered* relationship. In another example, given the sentence:

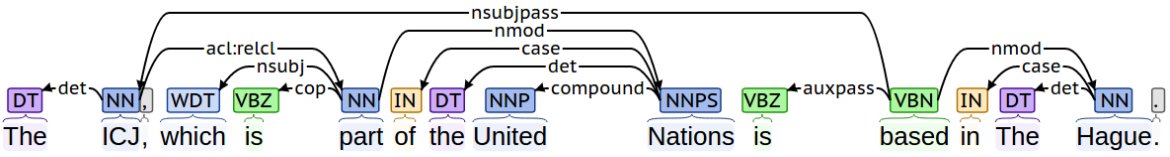
Anthony Shadid *is an* **Associated Press** *newsman based in* **Cairo**.

Due to the relational pattern based on PoS-tags, BREDS wrongly extracts the relationship <`Associated Press`, *has-installations*, `Cairo`>. By analysing the words in the dependency path (see Figure 7.1b) between the entities, one notices the word *newsman* instead of the phrase *based in*.

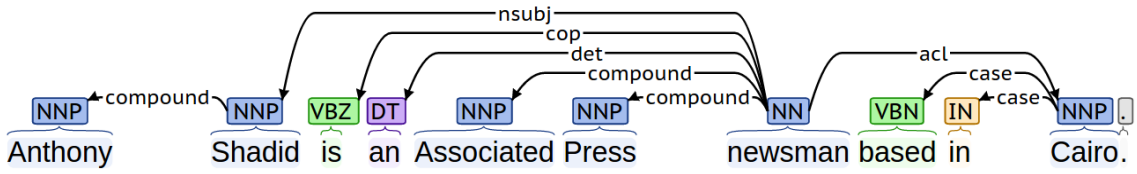
Finally in another example, given the sentence:

Fiat *'s acquisition of* **General Motors** *subsidiary* **Opel** *in Germany.*

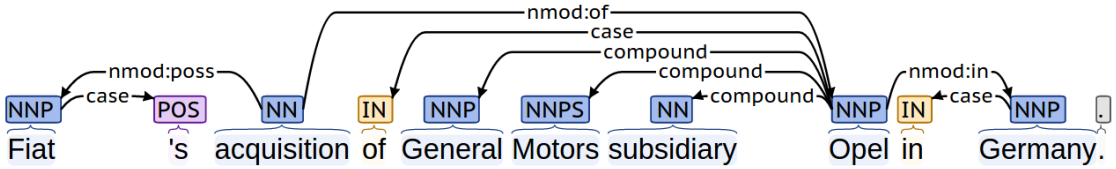
BREDS wrongly extracts the relationship <`Fiat`, *acquired*, `General Motors`>, due to finding the phrase *acquisition of* between the two entities. By analysing the dependencies path in Figure 7.1c, the path between **Fiat** and **Opel**, contains the word



(a) *The ICJ, which is part of the United Nations is based in The Hague*



(b) *Anthony Shadid is an Associated Press newsman based in Cairo*



(c) Fiat's acquisition of General Motors subsidiary Opel in Germany

Figure 7.1: Syntactic dependencies for different sentences.

acquisition, which might indicate that the sentence expresses the relationship <`Fiat`, *acquired*, `Opel`>.

Nevertheless, computing syntactic dependencies is much more computationally expensive than simply identifying the PoS-tags associated with each word.

Compositional function

After identifying the words that mediate a relationship, BREDS uses a simple compositional function to generate a single embedding vector. This function simply removes stopwords and then sums the word embedding vectors for the remaining words. There are, however, richer compositional functions which could be explored. Gormley et al. (2015) proposed a compositional model which combines word embeddings and NLP features, such as syntactic dependencies.

Semantic Drift

BREDS uses the Snowball Agichtein and Gravano (2000) mechanism to control semantic drift. This mechanism ranks a pattern by evaluating its extracted instances. The evaluation compares the extracted instances against instances in the seed set. But, as shown in Chapter 6, this approach can have some problems. When an extracted relationship $<e_1, e_2>$ shares e_1 with a seed instance, if e_2 does not match, the extracted instance is classified as incorrect. This will then cause the confidence score of the pattern to be low, and consequently, the confidence score of the extracted instance, resulting in no new seeds being added to the seed set.

Different mechanisms to detect semantic drift could be explored and incorporated in BREDS. For instance, McIntosh and Curran (2009) hypothesized that semantic drift occurs when a candidate instance is more similar to recently added instances than to the seed instances or high scored instances added in the earlier iterations.

Document Collection Pre-Processing

BREDS depends on a pre-processing step, which consists of identifying the named-entities in the collection of documents from which it extracts semantic relationships. In the experiments carried in this dissertation, the pre-processing step consisted of simply identifying three types of named-entities: persons, locations and organizations. This task was performed by a named-entity recognizer (NER) (Finkel et al., 2005) that associates a string or a sequence of strings with a class label. As shown in Chapter 6, errors generated by the NER sub-system will cause BREDS to extract wrong relationships.

A more robust approach to this task would be to replace the NER sub-system in the pre-processing pipeline by an entity-linking approach. Entity Linking (EL) involves the disambiguation of an entity according to a database. The main goal is to identify the different senses for a same entity. For instance, an EL system could infer that the strings *Bush*, *G. W. Bush*, and *George Bush* all refer to the same entity. This could alleviate some of the errors generated by simple NER. The biggest advantage, compared with simple NER, would be that BREDS could capture more contexts where the same entity is mentioned, but with a different surface string. Moreover, some EL systems, such as the one proposed by Hoffart et al. (2011), can identify more fine-grained categories of named-entities (e.g., categories from Wikipedia), going beyond

the person, location and organisation entities, which would enable BREDS to extract many more different relationship types.

7.3 Final Words

Knowledge bases (KBs), such as knowledge graphs, are essential tools for machine reasoning in many NLP tasks, like question answering. Manual construction and curation of a KB is costly, although they can be highly accurate. Automatic extraction of information from text is the obvious alternative. Relationship extraction (RE) is one way of achieving that goal of automating the extraction of structured information from text, particularly from large collections of documents.

I believe that, over the coming years, new RE techniques and the availability of large collections of text will make RE more accurate and improve the coverage of automatic generated KBs. As of the writing of this dissertation, Deep Learning based techniques dominate most of the current recent research in relationship extraction (dos Santos et al., 2015; Gormley et al., 2015; Xu et al., 2015a,b). However, these are supervised learning approaches requiring labelled datasets for training, which is always a bottleneck. In my view, future RE research will explore techniques combining semi-supervised or distantly supervised methods with the new Deep Learning approaches, efficiently extracting many different types of relationship instances from large document collections such as the Web.

Bibliography

Susana Afonso, Eckhard Bick, Renato Haber, and Diana Santos. **Floresta Sintá(c)tica: a Treebank for Portuguese**. In *In Proceedings of the Third International Conference on Language Resources and Evaluation*, LREC'02. European Language Resources Association, 2002. (Cited on pages 46, 48, and 74.)

Eugene Agichtein and Luis Gravano. **Snowball: Extracting Relations from Large Plain-Text Collections**. In *Proceedings of the ACM Conference on Digital libraries*, DL'00. Association for Computing Machinery, 2000. (Cited on pages v, 31, 98, 109, and 142.)

Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Salakoski. **All-paths Graph kernel for Protein-Protein Interaction Extraction with Evaluation of Cross-Corpus Learning**. *BMC Bioinformatics*, 9(Suppl 11), 2008. (Cited on pages 2, 22, 27, 28, and 85.)

Alan Akbik and Alexander Löser. **KrakeN: N-ary Facts in Open Information Extraction**. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX'12. Association for Computational Linguistics, 2012. (Cited on pages 16 and 86.)

Enrique Alfonseca, Katja Filippova, Jean-Yves Delort, and Guillermo Garrido. **Pattern Learning for Relation Extraction with a Hierarchical Topic Model**. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12. Association for Computational Linguistics, 2012. (Cited on page 38.)

Mohamed Aly. **Survey on Multi-Class Classification Methods**. Technical report, Caltech, USA, 2005. (Cited on page 24.)

Bibliography

Michele Banko and Oren Etzioni. **The Tradeoffs Between Open and Traditional Relation Extraction**. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-HLT'08. Association for Computational Linguistics, 2008. (Cited on page 41.)

Florbela Barreto, António Branco, Eduardo Ferreira, Amália Mendes, Maria Fernanda Nascimento, Filipe Nunes, and Joao Silva. **Open Resources and Tools for the Shallow Processing of Portuguese: the TagShare Project**. In *Proceedings of LREC 2006*, LREC'06. European Language Resources Association, 2006. (Cited on pages 48 and 74.)

Hannah Bast, Florian Bäurle, Björn Buchhold, and Elmar Haussmann. **Easy Access to the Freebase Dataset**. In *Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion*, WWW Companion '14. Association for Computational Linguistics, 2014. (Cited on pages 106 and 120.)

David S Batista, Rui Silva, Bruno Martins, and Mário J Silva. **A Minwise Hashing Method for Addressing Relationship Extraction from Text**. In *Web Information Systems Engineering–WISE 2013*, WISE'13. Springer Berlin Heidelberg, 2013a. (Cited on page v.)

David S Batista, Bruno Martins, and Mário J Silva. **Semi-Supervised Bootstrapping of Relationship Extractors with Distributional Semantics**. In *In Empirical Methods in Natural Language Processing*, EMNLP'15. ACL, 2015. (Cited on page vii.)

David Soares Batista, David Forte, Rui Silva, Bruno Martins, and Mário J. Silva. **Exploring DBpedia and Wikipedia for Portuguese Semantic Relationship Extraction**. *Linguamática*, 5(1), 2013b. (Cited on page v.)

Yoshua Bengio. **Neural Net Language Models**. *Scholarpedia*, 3(1), 2008. (Cited on page 60.)

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. **A Neural Probabilistic Language Model**. *Journal Machine Learning Research*, 3, 2003. (Cited on page 59.)

Yoshua Bengio, Ian J. Goodfellow, and Aaron Courville. **Deep Learning**. Book in preparation for MIT Press, 2015. URL http://www.iro.umontreal.ca/~bengioy/dlbook. (Cited on page 25.)

Eckhard Bick. **The Parsing System "PALAVRAS". Automatic Grammatical Analysis of Portuguese in a Constraint Grammar Framework**. University of Arhus, 2000. (Cited on pages 46 and 48.)

Steven Bird, Ewan Klein, and Edward Loper. **Natural Language Processing with Python**. O'Reilly Media, Inc., 2009. (Cited on pages 49 and 105.)

Sebastian Blohm, Philipp Cimiano, and Egon Stemle. **Harvesting Relations from the Web: Quantifiying the Impact of Filtering Functions**. In *Proceedings of the National Conference on Artificial Intelligence*, AAAI'07. AAAI Press, 2007. (Cited on page 34.)

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. **Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge**. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08. Association for Computing Machinery, 2008. (Cited on pages 37 and 106.)

António Branco and Joao Ricardo Silva. **A Suite of Shallow Processing Tools for Portuguese: Lx-Suite**. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations*, EACL'06. Association for Computational Linguistics, 2006. (Cited on pages 48 and 74.)

António Branco, Catarina Carvalheiro, Sílvia Pereira, Sara Silveira, João Silva, Sérgio Castro, and João Graça. **A PropBank for Portuguese: the CINTIL-PropBank.** In *Proceedings of International Conference on Language Resources and Evaluation*, LREC'12. European Language Resources Association, 2012. (Cited on page 48.)

Erin J. Bredensteiner and Kristin P. Bennett. **Multicategory Classification by Support Vector Machines**. *Computational Optimization and Applications*, 12(1-3), 1999. (Cited on page 24.)

Bibliography

Sergey Brin. **Extracting Patterns and Relations from the World Wide Web**. In *Selected Papers from the International Workshop on The World Wide Web and Databases*, WebDB '98. Springer, 1999. (Cited on pages v and 30.)

Andrei Broder. **On the Resemblance and Containment of Documents**. In *Proceedings of the Conference on Compression and Complexity of Sequences*, SEQUENCES '97. IEEE Computer Society, 1997. (Cited on pages 65 and 139.)

Andrei Broder, Moses Charikar, Alan M. Frieze, and Michael Mitzenmacher. **Min-wise Independent Permutations**. *Journal of Computer and System Sciences*, 60(3), 2000. (Cited on pages v and 87.)

Mirko Bronzi, Zhaochen Guo, Filipe Mesquita, Denilson Barbosa, and Paolo Merialdo. **Automatic Evaluation of Relation Extraction Systems on Large-scale**. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, AKBC-WEKEX '12. Association for Computational Linguistics, 2012. (Cited on pages 6, 7, 106, 128, and 133.)

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. **Class-based N-gram Models of Natural Language**. *Computational Linguistics*, 18(4), 1992. (Cited on page 54.)

Mírian Bruckschen, José Guilherme Camargo de Souza, Renata Vieira, and Sandro Rigo. **Desafios na Avaliação Conjunta do Reconhecimento de Entidades Mencionadas: O Segundo HAREM**, Chapter: 14. Sistema SeRELeP para o Reconhecimento de Relações Entre Entidades Mencionadas. Linguateca, 2008. (Cited on pages 46 and 48.)

Razvan Bunescu and Raymond Mooney. **Subsequence Kernels for Relation Extraction**. In *Proceedings of the Conference on Neural Information Processing Systems*, NIPS'05. MIT Press, 2005a. (Cited on pages 22, 26, 28, 71, 80, and 85.)

Razvan Bunescu and Raymond Mooney. **A Shortest Path Dependency Kernel for Relation Extraction**. In *Proceedings of the Human Language Technology*

Conference and the Conference on Empirical Methods in Natural Language Processing, HLT/EMNLP'05. Association for Computational Linguistics, 2005b. (Cited on pages 22, 28, and 140.)

Razvan C. Bunescu and Raymond J. Mooney. **Learning to Extract Relations from the Web Using Minimal Supervision**. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, ACL'07. Association for Computational Linguistics, 2007. (Cited on page 44.)

Philip R Burns. **MorphAdorner v2: a Java Library for the Morphological Adornment of English Language Texts**, 2013. http://morphadorner.northwestern.edu. (Cited on page 74.)

Nuno Cardoso. **Desafios na Avaliação Conjunta do Reconhecimento de Entidades Mencionadas: O Segundo HAREM**, Chapter: 11. REMBRANDT - Reconhecimento de Entidades Mencionadas Baseado em Relações e ANálise Detalhada do Texto. Linguateca, 2008. (Cited on pages 46 and 48.)

Nuno Cardoso. **REMBRANDT - A Named-Entity Recognition Framework**. In *Proceedings of International Conference on Language Resources and Evaluation*, LREC'12. European Language Resources Association, 2012. (Cited on page 46.)

Josiah L. Carlson. **REDIS in Action**. Manning Publications Co., 2013. (Cited on page 126.)

Marcírio Chaves. **Desafios na Avaliação Conjunta do Reconhecimento de Entidades Mencionadas: O Segundo HAREM**, Chapter: 13. Geo-Ontologias para Reconhecimento de Relações Entre Locais: a participação do SEI-Geo no Segundo HAREM. Linguateca, 2008. (Cited on pages 45 and 48.)

Stanley F. Chen and Joshua Goodman. **An Empirical Study of Smoothing Techniques for Language Modeling**. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*, ACL'96. Association for Computational Linguistics, 1996. (Cited on page 58.)

Yuan Chen, Man Lan, Jian Su, Zhi Min Zhou, and Yu Xu. **ECNU: Effective Semantic Relations Classification Without Complicated Features or**

Multiple External Corpora. In *Proceedings of the Fifth International Workshop on Semantic Evaluation*, SemEval '10. Association for Computational Linguistics, 2010. (Cited on page 84.)

Sung-Pil Choi, Seungwoo Lee, Hanmin Jung, and Sa-kwang Song. **An Intensive Case Study on Kernel-Based Relation Extraction**. *Multimedia Tools and Applications*, 71(2), 2013. (Cited on page 27.)

O. Chum, J. Philbin, and A. Zisserman. **Near Duplicate Image Detection: min-Hash and tf-idf Weighting**. In *British Machine Vision Conference*, BMVC'08. Springer, 2008. (Cited on page 139.)

Kenneth Ward Church and Patrick Hanks. **Word Association Norms, Mutual Information, and Lexicography**. *Computational Linguistics*, 16(1), 1990. (Cited on pages 33 and 56.)

J. Cohen. **A Coefficient of Agreement for Nominal Scales**. *Educational and Psychological Measurement*, 20(1), 1960. (Cited on page 43.)

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. **Natural Language Processing (Almost) from Scratch**. *Journal of Machine Learning Research*, 12, 2011. (Cited on page 25.)

Sandra Collovini, Tiago Bonamigo, and Renata Vieira. **A Review on Relation Extraction With an Eye on Portuguese**. *Journal of the Brazilian Computer Society*, 19(4), 2013. (Cited on page 45.)

Sandra Collovini, Lucas Pugens, AlineA. Vanin, and Renata Vieira. **Extraction of Relation Descriptors for Portuguese Using Conditional Random Fields**. In *Proceedings of Fourteenth edition of the Ibero-American Conference on Artificial Intelligence*, IBERAMIA'14. Springer, 2014. (Cited on pages 47 and 48.)

Corinna Cortes and Vladimir Vapnik. **Support-Vector Networks**. *Machine Learning*, 20(3), 1995. (Cited on pages iv, 3, and 19.)

David R Cox. **The Regression Analysis of Binary Sequences**. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1958. (Cited on page 19.)

Koby Crammer and Yoram Singer. **On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines**. *Journal of Machine Learning Research*, 2, 2002. (Cited on page 24.)

Aron Culotta and Jeffrey Sorensen. **Dependency Tree Kernels for Relation Extraction**. In *Proceedings of the Annual Meeting of the ACL*, ACL'04. Association for Computational Linguistics, 2004. (Cited on pages 21 and 28.)

Aron Culotta, Andrew McCallum, and Jonathan Betz. **Integrating Probabilistic Extraction Models and Data Mining to Discover Relations and Patterns in Text**. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL'06. Association for Computational Linguistics, 2006. (Cited on pages 24, 26, 27, and 79.)

James R Curran, Tara Murphy, and Bernhard Scholz. **Minimising Semantic Drift with Mutual Exclusion Bootstrapping**. In *Proceedings of the Conference of the Pacific Association for Computational Linguistics*, PACLING'07. Association for Computational Linguistics, 2007. (Cited on page 34.)

James Richard Curran. **From Distributional to Semantic Similarity**. PhD Thesis, University of Edinburgh. College of Science and Engineering. School of Informatics., 2004. (Cited on page 35.)

Marie-Catherine De Marneffe and Christopher D. Manning. **The Stanford Typed Dependencies Representation**. In *Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, CrossParser '08. Association for Computational Linguistics, 2008. (Cited on page 15.)

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. **Generating Typed Dependency Parses From Phrase Structure Parses**. In *Proceedings of International Conference on Language Resources and Evaluation*, LREC'06. European Language Resources Association, 2006. (Cited on page 15.)

Luciano Del Corro and Rainer Gemulla. **ClausIE: Clause-based Open Information Extraction**. In *Proceedings of the 22nd International Conference on*

World Wide Web, WWW '13. Association for Computating Machinery, 2013. (Cited on page 40.)

George Doddington, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel. **The Automatic Content Extraction (ACE) Program Tasks, Data, and Evaluation**. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation*, LREC'04. European Language Resources Association (ERLA), 2004. (Cited on page 26.)

Cicero dos Santos, Bing Xiang, and Bowen Zhou. **Classifying Relations by Ranking with Convolutional Neural Networks**. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, ACL'15. Association for Computational Linguistics, 2015. (Cited on page 143.)

Javid Ebrahimi and Dejing Dou. **Chain based RNN for Relation Classification**. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-HLT'15. Association for Computational Linguistics, 2015. (Cited on pages 25, 28, and 29.)

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. **Open Information Extraction from the Web**. *Communications of the ACM*, 51(12), 2008. (Cited on pages 13 and 41.)

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. **Open Information Extraction: The Second Generation**. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume One*, IJCAI'11. AAAI Press, 2011. (Cited on page 39.)

Anthony Fader, Stephen Soderland, and Oren Etzioni. **Identifying Relations for Open Information Extraction**. In *Proceedings of the Conference of Empirical Methods in Natural Language Processing*, EMNLP'11. Association for Computing Machinery, 2011. (Cited on pages 39, 47, 71, 100, and 140.)

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. **Incorporating Non-local Information into Information Extraction Systems by**

Gibbs Sampling. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*, ACL'05. Association for Computational Linguistics, 2005. (Cited on pages 105, 120, 128, and 142.)

J.R. Firth. **A Synopsis of Linguistic Theory 1930-1955**. *Studies in Linguistic Analysis*, Special Volume of the Philological Society, 1957. (Cited on pages 5 and 53.)

J.L. Fleiss et al. **Measuring Nominal Scale Agreement Among Many Raters**. *Psychological Bulletin*, 76(5), 1971. (Cited on page 43.)

Cláudia Freitas, Diana Santos, Hugo Gonçalo Oliveira, Paula Carvalho, and Cristina Mota. **Desafios na Avaliação Conjunta do Reconhecimento de Entidades Mencionadas: O Segundo HAREM**, Chapter: Relaçoes semânticas do ReRelEM: além das entidades no Segundo HAREM. Linguateca, 2008. (Cited on page 45.)

Cláudia Freitas, Diana Santos, Cristina Mota, Hugo Gonçalo Oliveira, and Paula Carvalho. **Relation Detection Between Named Entities: Report of a Shared Task**. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, SEW'09. Association for Computational Linguistics, 2009. (Cited on page 26.)

Pablo Gamallo, Marcos García, and Santiago Fernández-Lanza. **Dependency-Based Open Information Extraction**. In *Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*, ROBUS-UNSUP'12. Association for Computational Linguistics, 2012. (Cited on pages 40, 47, 48, and 51.)

Marcos García and Pablo Gamallo. **Evaluating Various Linguistic Features on Semantic Relation Extraction**. In *Proocedings of the Conference on Recent Advances in Natural Language Processing*, RANLP'11. Association for Computational Linguistics, 2011. (Cited on pages 46 and 78.)

Aristides Gionis, Piotr Indyk, and Rajeev Motwani. **Similarity Search in High Dimensions via Hashing**. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB '99. Morgan Kaufmann Publishers Inc., 1999. (Cited on pages v and 67.)

Bibliography

Gene Golub and William Kahan. **Calculating the Singular Values and Pseudo-Inverse of a Matrix**. *Journal of the Society for Industrial & Applied Mathematics, Series B: Numerical Analysis*, 2(2), 1965. (Cited on page 56.)

Matthew R. Gormley, Mo Yu, and Mark Dredze. **Improved Relation Extraction with Feature-Rich Compositional Embedding Models**. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'15. Association for Computational Linguistics, 2015. (Cited on pages 141 and 143.)

Zellig S. Harris. **Distributional Structure**. *Word*, 10, 1954. (Cited on pages vi, 5, and 53.)

Kazuma Hashimoto, Makoto Miwa, Yoshimasa Tsuruoka, and Takashi Chikayama. **Simple Customization of Recursive Neural Networks for Semantic Relation Classification.** In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, EMNLP'13. Association for Computational Linguistics, 2013. (Cited on pages 25, 28, and 29.)

Marti A. Hearst. **Automatic Acquisition of Hyponyms from Large Text Corpora**. In *Proceedings of the Conference on Computational Linguistics*, COLING'92. Association for Computational Linguistics, 1992. (Cited on page 17.)

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó. Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. **SemEval-2010 Task 8: Multi-way Classification of Semantic Relations Between Pairs of Nominals**. In *Proceedings of the Fifth International Workshop on Semantic Evaluation*, SemEval'10. Association for Computational Linguistics, 2010. (Cited on pages 2, 26, 27, and 79.)

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. **Robust Disambiguation of Named Entities in Text**. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11. Association for Computational Linguistics, 2011. (Cited on page 142.)

Johannes Hoffart, Fabian M. Suchanek, Klaus Berberich, and Gerhard Weikum. **YAGO2: A Spatially and Temporally Enhanced Knowledge Base from Wikipedia**. *Artificial Intelligence*, 194, 2013. (Cited on page 120.)

Raphael Hoffmann, Congle Zhang, and Daniel S. Weld. **Learning 5000 Relational Extractors**. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, ACL'10. Association for Computational Linguistics, 2010. (Cited on page 37.)

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. **Knowledge-based Weak Supervision for Information Extraction of Overlapping Relations**. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, HLT'11. Association for Computational Linguistics, 2011. (Cited on page 38.)

Mario Jarmasz and Stan Szpakowicz. **Roget's Thesaurus and Semantic Similarity**. In *Proocedings of the Conference on Recent Advances in Natural Language Processing*, RANLP'03. Association for Computational Linguistics, 2003. (Cited on pages 27 and 84.)

Richard Johansson and Pierre Nugues. **LTH: Semantic Structure Extraction Using Nonprojective Dependency Trees**. In *Proceedings of the Fourth International Workshop on Semantic Evaluations*, SemEval '07. Association for Computational Linguistics, 2007. (Cited on page 84.)

Karen Spärck Jones. **A Statistical Interpretation of Term Specificity and its Application in Retrieval**. *Journal of Documentation*, 28, 1972. (Cited on page 56.)

Nanda Kambhatla. **Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Extracting Relations**. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Posters and Demonstrations*, ACL'04. Association for Computational Linguistics, 2004. (Cited on pages 19, 27, and 28.)

Pentti Kanerva, Jan Kristoferson, and Anders Holst. **Random Indexing of Text Samples for Latent Semantic Analysis**. In *Proceedings of the 22nd Annual Conference of the Cognitive Science Society*, CogSci'00. Erlbaum, 2000. (Cited on page 57.)

Douwe Kiela and Stephen Clark. **A Systematic Study of Semantic Vector Space Model Parameters**. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC) at EACL*, EACL'14. Association for Computational Linguistics, 2014. (Cited on pages 56, 57, and 64.)

S. Kim, J. Yoon, J. Yang, and S. Park. **Walk-Weighted Subsequence Kernels for Protein-Protein Interaction Extraction**. *BMC Bioinformatics*, 11 (107), 2010. (Cited on page 22.)

Stanley Kok and Pedro Domingos. **Extracting Semantic Networks from Text Via Relational Clustering**. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, ECML PKDD '08. Springer, 2008. (Cited on page 51.)

Terry Koo, Xavier Carreras, and Michael Collins. **Simple Semi-Supervised Dependency Parsing**. In *Proceedings of the Association for Computational Linguistics: Human Languages Technology*, ACL'08. Association for Computational Linguistics, 2008. (Cited on page 63.)

Jan Kotek. **MapDB**, 2013. http://www.mapdb.org/. (Cited on page 74.)

Klaus Krippendorff. **Agreement and Information in the Reliability of Coding**. *Communication Methods and Measures*, 5(2), 2011. (Cited on page 43.)

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. **Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data**. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML'01. Association for Computing Machinery, 2001. (Cited on page 24.)

Thomas K Landauer and Susan T. Dutnais. **A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and**

Representation of Knowledge. *Psychological Review*, 104(2), 1997. (Cited on page 56.)

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. **DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia**. *Semantic Web Journal*, 6(2), 2015. (Cited on pages v, 36, 46, 75, and 120.)

Douglas B. Lenat. **CYC: A Large-scale Investment in Knowledge Infrastructure**. *Communications of the ACM*, 38(11), 1995. (Cited on pages 27 and 84.)

Beth Levin. **English Verb Classes and Alternations: A Preliminary Investigation**. University of Chicago Press, 1993. (Cited on pages 27 and 84.)

Ping Li and Christian König. **b-Bit Minwise Hashing**. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10. ACM, 2010. (Cited on page 139.)

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. **Text Classification Using String Kernels**. *Journal of Machine Learning Research*, 2, 2002. (Cited on page 21.)

Kevin Lund and Curt Burgess. **Producing high-dimensional semantic spaces from lexical co-occurrence**. *Behavior Research Methods, Instruments, & Computers*, 28(2), 1996. (Cited on page 56.)

Christopher D. Manning and Hinrich Schütze. **Foundations of Statistical Natural Language Processing**. MIT Press, 1999. (Cited on page 14.)

Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. **Introduction to Information Retrieval**. Cambridge University Press, 2008. (Cited on page 58.)

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, ACL'14. Association for Computational Linguistics, 2014. (Cited on page 49.)

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. **Building a Large Annotated Corpus of English: The Penn Treebank**. *Computational Linguistics*, 19(2), 1993. (Cited on page 15.)

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. **Open Language Learning for Information Extraction**. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL'12. Association for Computational Linguistics, 2012. (Cited on pages 42 and 51.)

Warren McCulloch and Walter Pitts. **A Logical Calculus of Ideas Immanent in Nervous Activity**. *Bulletin of Mathematical Biophysics*, 5(4), 1943. (Cited on page 58.)

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. **Universal Dependency Annotation for Multilingual Parsing**. In *Proceedings of the ACL 2013*, ACL'13. Association for Computational Linguistics, 2013. (Cited on pages 15 and 52.)

Tara McIntosh and James R. Curran. **Reducing Semantic Drift with Bagging and Distributional Similarity**. In *Proceedings of the Joint Conference of the Annual Meeting of the ACL and International Joint Conference on Natural Language Processing of the AFNLP*, ACL'09. Association for Computational Linguistics, 2009. (Cited on pages vi, 34, and 142.)

Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, The Google Books Team, Joseph P. Pickett, Dale Holberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden. **Quantitative Analysis of Culture Using Millions of Digitized Books**. *Science*, 331(6014), 2010. (Cited on pages 27 and 84.)

Tomas Mikolov, Jiri Kopecky, Lukas Burget, Ondrej Glembek, and Jan Cernocky. **Neural Network Based Language Models for Highly Inflective Languages**. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, ICASSP'09. IEEE Computer Society, 2009. (Cited on page 60.)

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. **Efficient Estimation of Word Representations in Vector Space**. In *Proceedings of Workshop at International Conference on Learning Representations*, ICLR'13, 2013a. (Cited on pages 25, 60, 61, 63, 100, 106, and 122.)

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. **Distributed Representations of Words and Phrases and their Compositionality**. In *Advances in Neural Information Processing Systems*, NIPS'13. Curran Associates, Inc., 2013b. (Cited on pages vi, 60, and 100.)

Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. **Linguistic Regularities in Continuous Space Word Representations.** In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLTC'13. Association for Computational Linguistics, 2013c. (Cited on page 62.)

George A. Miller. **WordNet: A Lexical Database for English**. *Communications of the ACM*, 38(11), 1995. (Cited on pages 22, 27, 46, and 84.)

Scott Miller, Jethran Guinness, and Alex Zamanian. **Name Tagging with Word Clusters and Discriminative Training**. In *Proceedings of the Human Language Technology Conference / North American Chapter of the Association for Computational Linguistics Annual Meeting*, HLT/NAACL'04. Association for Computational Linguistics, 2004. (Cited on page 63.)

Bonan Min, Shuming Shi, Ralph Grishman, and Chin-Yew Lin. **Ensemble Semantics for Large-scale Unsupervised Relation Extraction**. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12. Association for Computing Machinery, 2012. (Cited on page 51.)

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. **Distant Supervision for Relation Extraction Without Labeled Data**. In *Proceedings of the Joint Conference of Annual Meeting of the ACL and the Joint Conference on Natural Language Processing of the AFNLP*, ACL-IJCNLP'09. Association for Computational Linguistics, 2009. (Cited on pages 37, 38, 47, and 75.)

Thomas Morton, Joern Kottmann, Jason Baldridge, and Gann Bierner. **OpenNLP: A Java-based NLP Toolkit**, 2005. http://opennlp.apache.org. (Cited on pages 46, 49, and 74.)

Cristina Mota and Diana Santos. **Desafios na Avaliação Conjunta do Reconhecimento de Entidades Mencionadas: O Segundo HAREM**. Linguateca, 2008. (Cited on page 45.)

Ion Muslea. **Extraction Patterns for Information Extraction Tasks: A Survey**. In *National Conferenc on Artificial Intelligence Workshop on Machine Learning for Information Extraction*, AAAI-99. AAAI Press, 1999. (Cited on page 18.)

David Nadeau and Satoshi Sekine. **A Survey of Named Entity Recognition and Classification**. *Lingvisticae Investigationes*, 30(1), 2007. (Cited on page 16.)

Truc-Vien Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. **Convolution Kernels on Constituent, Dependency and Sequential structures for Relation extraction**. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'09. Association for Computational Linguistics, 2009. (Cited on page 22.)

Truc-Vien T. Nguyen and Alessandro Moschitti. **End-to-End Relation Extraction Using Distant Supervision from External Semantic Repositories**. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers*, HLT'11. Association for Computational Linguistics, 2011. (Cited on pages 37 and 38.)

ACE NIST. **Automatic Context Extraction**, 2002. http://www.itl.nist.gov/iad/mig//tests/ace/2002. (Cited on page 26.)

Hugo Oliveira and Paulo Gomes. **Onto.PT: Recent Developments of a Large Public Domain Portuguese Wordnet**. In *Proceedings of the Seventh Global WordNet Conference*, GWC'14. Global WordNet Association, 2014. (Cited on page 46.)

Hugo Gonçalo Oliveira, Hernani Costa, and Paulo Gomes. **Extracção de Conhecimento Léxico-semântico a Partir de Resumos da Wikipédia**. In *Actas do II Simpósio de Informática*, INFORUM'10. Simpósio de Informática, 2010. (Cited on pages 46 and 48.)

Pablo Gamallo Otero. **The Meaning of Syntactic Dependencies**. *Linguistik Online*, 35(3/08), 2008. (Cited on page 15.)

Pablo Gamallo Otero and Isaac González. **DepPattern: a Multilingual Dependency Parser**. In *Demo Session of the International Conference on Computational Processing of the Portuguese Language*, PROPOR'12. Springer, 2012. (Cited on page 40.)

Patrick Pantel and Marco Pennacchiotti. **Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations**. In *Proceedings of the International Conference on Computational Linguistics and Annual Meeting of the Association for Computational Linguistics*, ACL'06. Association for Computational Linguistics, 2006. (Cited on pages v and 33.)

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. **English Gigaword Fifth Edition LDC2011T07**, 2011. Linguistic Data Consortium: https://catalog.ldc.upenn.edu/LDC2011T07. (Cited on pages 6, 105, 115, and 120.)

Slav Petrov, Dipanjan Das, and Ryan T. McDonald. **A Universal Part-of-Speech Tagset**. In *Proceedings of the Conference on Language Resources and Evaluation*, LREC'12. European Language Resources Association (ERLA), 2012. (Cited on pages 15, 52, 74, and 85.)

M. F. Porter. **Readings in Information Retrieval**, Chapter: An Algorithm for Suffix Stripping. Morgan Kaufmann Publishers Inc., 1997. (Cited on pages vi, 5, 14, and 98.)

Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Björne, Jorma Boberg, Jouni Järvinen, and Tapio Salakoski. **BioInfer: a Corpus for Information Extraction in the Biomedical Domain**. *BMC Bioinformatics*, 8 (1), 2007. (Cited on page 26.)

J. Ross Quinlan. **C4.5: Programs for Machine Learning**. Morgan Kaufmann Publishers Inc., 1993. (Cited on page 47.)

Deepak Ravichandran and Eduard Hovy. **Learning Surface Text Patterns for a Question Answering System**. In *Proceedings of the Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2002. (Cited on page 33.)

Sebastian Riedel, Limin Yao, and Andrew McCallum. **Modeling Relations and Their Mentions Without Labeled Text**. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, ECML PKDD'10. Springer, 2010. (Cited on page 38.)

Ellen Riloff and Rosie Jones. **Learning Dictionaries for Information Extraction by Multi-level Bootstrapping**. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*, AAAI '99/IAAI '99. American Association for Artificial Intelligence, 1999. (Cited on page 34.)

Ellen Riloff and Jessica Shepherd. **A Corpus-Based Approach for Building Semantic Lexicons**. In *In Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, EMNLP'97. Association for Computational Linguistics, 1997. (Cited on page 34.)

Bryan Rink and Sanda Harabagiu. **UTD: Classifying Semantic Relations by Combining Lexical and Semantic Resources**. In *Proceedings of the Fifth International Workshop on Semantic Evaluation*, SemEval '10. Association for Computational Linguistics, 2010. (Cited on pages 27, 28, and 83.)

Brian Roark and Eugene Charniak. **Noun-phrase Co-occurrence Statistics for Semiautomatic Semantic Lexicon Construction**. In *Proceedings of*

the Seventeenth International Conference on Computational Linguistics - Volume 2, COLING '98. Association for Computational Linguistics, 1998. (Cited on page 34.)

Xin Rong. **word2vec Parameter Learning Explained**. *Computing Research Repository*, abs/1411.2738, 2014. (Cited on page 61.)

Frank Rosenblatt. **The Perceptron: a Probabilistic Model for Information Storage and Organization in the Brain.** *Psychological review*, 65(6), 1958. (Cited on page 21.)

Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. **A Survey of Noise Reduction Methods for Distant Supervision**. In *Proceedings of the 2013 workshop on Automated Knowledge Base Construction*, AKBC'13. Association for Computing Machinery, 2013. (Cited on page 38.)

Herbert Rubenstein and John B. Goodenough. **Contextual Correlates of Synonymy**. *Communications of the ACM*, 8(10), 1965. (Cited on page 53.)

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. **Neurocomputing: Foundations of Research**, Chapter: Learning Representations by Back-Propagating Errors. MIT Press, 1988. (Cited on page 60.)

Magnus Sahlgren. **An Introduction to Random Indexing**. In *Methods and Applications of Semantic Indexing Workshop at the International Conference on Terminology and Knowledge Engineering*, TKE 2005. Infoterm, 2005. (Cited on page 57.)

Magnus Sahlgren. **The Word-Space Model: Using Distributional Analysis to Represent Syntagmatic and Paradigmatic Relations Between Words in High-Dimensional Vector Spaces**. PhD Thesis, Stockholm University, Faculty of Humanities, Department of Linguistics., 2006. (Cited on page 64.)

Magnus Sahlgren. **The Distributional Hypothesis**. *Italian Journal of Linguistics*, 20(1), 2008. (Cited on page 54.)

G. Salton, A. Wong, and C. S. Yang. **A Vector Space Model for Automatic Indexing**. *Communications of the ACM*, 18(11), 1975. (Cited on page 57.)

Bibliography

Gerard Salton and Christopher Buckley. **Term-Weighting Approaches in Automatic Text Retrieval**. *Information Processing & Management*, 24(5), 1988. (Cited on pages vi, 4, and 31.)

Diana Santos. **Caminhos Percorridos no Mapa da Portuguesificação: A Linguateca em Perspectiva**. *Linguamática*, 1(1), 2009. (Cited on page 48.)

Sunita Sarawagi. **Information Extraction**. *Foundations and Trends in Databases*, 1(3), 2008. (Cited on pages iv and 2.)

J. Schmidhuber. **Deep Learning in Neural Networks: An Overview**. *Neural Networks*, 61, 2015. (Cited on page 25.)

Karin Kipper Schuler. **Verbnet: A Broad-coverage, Comprehensive Verb Lexicon**. PhD Thesis, University of Pennsylvania, 2005. (Cited on page 84.)

John Shawe-Taylor and Nello Cristianini. **Kernel Methods for Pattern Analysis**. Cambridge University Press, 2004. (Cited on pages iv, 3, and 20.)

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. **Semantic Compositionality Through Recursive Matrix-vector Spaces**. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12. Association for Computational Linguistics, 2012. (Cited on pages 25, 28, and 29.)

Stephen Soderland and Bhushan Mandhani. **Moving from Textual Relations to Ontologized Relations**. In *AAAI Spring Symposium: Machine Reading*, AAAI Spring Symposium Series. AAAI Press, 2007. (Cited on page 51.)

Erick Nilsen Pereira Souza and Daniela Barreiro Claro. **Extração de Relações utilizando Features Diferenciadas para Português**. *Linguamática*, 6(2), 2014. (Cited on pages 47 and 48.)

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. **Yago: A Core of Semantic Knowledge**. In *Proceedings of the 16th International Conference on World Wide Web*, WWW'07, 2007. (Cited on page 37.)

Ching Y. Suen. **n-Gram Statistics for Natural Language Understanding and Text Processing**. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2), 1979. (Cited on page 13.)

Ang Sun, Ralph Grishman, and Satoshi Sekine. **Semi-supervised Relation Extraction with Large-scale Word Clustering**. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT'11. Association for Computational Linguistics, 2011. (Cited on page 63.)

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. **Reducing Wrong Labels in Distant Supervision for Relation Extraction**. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12. Association for Computational Linguistics, 2012. (Cited on page 38.)

Carlos H. C. Teixeira, Arlei Silva, and Wagner Meira Jr. **Min-Hash Fingerprints for Graph Kernels: A Trade-off among Accuracy, Efficiency, and Compression**. *Journal of Information and Data Management*, 3(3), 2012. (Cited on page 139.)

Alex Franz Thorsten Brants. **Web 1T 5-gram Version 1**, 2006. Linguistic Data Consortium: https://catalog.ldc.upenn.edu/LDC2006T13. (Cited on page 84.)

Domonkos Tikk, Philippe Thomas, Peter Palaga, Jörg Hakenberg, and Ulf Leser. **A Comprehensive Benchmark of Kernel Methods to Extract Protein–Protein Interactions from Literature**. *PLoS Computational Biology*, 6(7), 2010. (Cited on pages iii, 85, and 86.)

Stephen Tratz and Eduard Hovy. **ISI: Automatic Classification of Relations Between Nominals Using a Maximum Entropy Classifier**. In *Proceedings of the Fifth International Workshop on Semantic Evaluation*, SemEval '10. Association for Computational Linguistics, 2010. (Cited on page 84.)

Joseph Turian, Lev Ratinov, and Yoshua Bengio. **Word Representations: A Simple and General Method for Semi-supervised Learning**. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*,

ACL'10. Association for Computational Linguistics, 2010. (Cited on pages 25 and 63.)

Peter D. Turney and Patrick Pantel. **From Frequency to Meaning: Vector Space Models of Semantics**. *Journal of Artificial Intelligence Research*, 37(1), 2010. (Cited on page 64.)

Kateryna Tymoshenko and Claudio Giuliano. **FBK-IRST: Semantic Relation Extraction using Cyc**. In *Proceedings of the Fifth International Workshop on Semantic Evaluation*, SemEval '10. Association for Computational Linguistics, 2010. (Cited on page 84.)

J. Weston and C. Watkins. **Support Vector Machines for Multi-Class Pattern Recognition**. In *In Proceedings of the Seventh European Symposium on Artificial Neural Networks*, ESANN'99. Association for Computing Machinery, 1999. (Cited on page 24.)

Fei Wu and Daniel S Weld. **Open Information Extraction Using Wikipedia**. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, ACL'10. Association for Computational Linguistics, 2010. (Cited on pages 16, 42, and 86.)

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. **Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling**. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'15. Association for Computational Linguistics, 2015a. (Cited on page 143.)

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. **Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths.** In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'15. Association for Computational Linguistics, 2015b. (Cited on page 143.)

Hong Yu and Eugene Agichtein. **Extracting Synonymous Gene and Protein Terms from Biological Literature**. *Bioinformatics*, 19(suppl 1), 2003. (Cited on pages 31 and 109.)

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. **Kernel Methods for Relation Extraction**. *Journal of Machine Learning Research*, 3, 2003. (Cited on pages 21 and 27.)

Shubin Zhao and Ralph Grishman. **Extracting Relations With Integrated Information Using Kernel Methods**. In *Proceedings of the Annual Meeting of the ACL*. Association for Computational Linguistics, 2005. (Cited on pages 21, 27, and 28.)

Guodong Zhou and Min Zhang. **Extracting Relation Information From Text Documents by Exploring Various Types of Knowledge**. *Information Processing and Management*, 43(4), 2007. (Cited on page 22.)

George K. Zipf. **Human Behaviour and the Principle of Least-Effort**. Addison-Wesley, 1949. (Cited on page 56.)