Contributions to Hypernym Patterns Representation and Learning based on Dependency Parsing and Sequential Pattern Mining

Hamad Issa Alaa Aldine

DOCTORAT BRETAGNE LOIRE / MATHSTIC

Université Bretagne Sud

Université Libanaise
École Doctorale Sciences et Technologies
Doyen

THESE DE DOCTORAT DE

L'UNIVERSITE BRETAGNE SUD ET
L'UNIVERSITE LIBANAISE

ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : Informatique

Par

Ahmad ISSA ALAA ALDINE

Contributions to Hypernym Patterns Representation and Learning based on Dependency Parsing and Sequential Pattern Mining

Thèse présentée et soutenue à Vannes, 15 Décembre 2020
Unité de recherche : IRISA
Thèse N° : 575

Rapporteurs avant soutenance :

M. Matthieu Roche	Chercheur HDR, UMR TETIS, CIRAD
Mme. Nada Matta	Professeur des Universités, Université de Technologie de Troyes

Composition du Jury :

Président :	M. Fabrice Guillet	Professeur des Universités, Université de Nantes
Examinateurs :	M. Ahmad Faour	Maître de conférences, Université Libanaise
Dir. de thèse :	M. Giuseppe Berio	Professeur des Universités, Université de Bretagne-Sud
	M. Mohamed Dbouk	Professeur des Universités, Université Libanaise
Co-dir. de thèse :	Mme. Mounira Harzallah	Maître de conférences (HDR), Université de Nantes
	M. Nicolas Bechet	Maître de conférences, Université de Bretagne-Sud

Invité(s)

Mme. Haïfa Zargayouna	Maître de conférences, Université de Paris 13

ACKNOWLEDGEMENT

Foremost and forever, I thank god for giving me the strength and the ability to complete this thesis that was guided and supported by many people.

First, my sincere thanks go to supervisors in France Prof. Giuseppe BERIO, Dr. Mounira HARZALLAH, and Dr. Nicolas BECHET for allowing me to conduct this thesis under their supervision and for their great guidance, support, and hard work throughout all stages of this thesis. Their massive knowledge in the working field and their ideas have contributed greatly to the improvement of the thesis.

As well, I would like to thank my supervisors in Lebanon Prof. Mohamed DBOUK and Dr. Ahmad FAOUR for allowing me to conduct this thesis under their supervision and for their great support throughout the thesis. Their general advice and permanent encouragement greatly helped me to improve my research and communication skills.

I would like to thank the reporters Prof. Nada MATTA and Dr. Mathieu ROCHE for their acceptance to review this thesis and for their valuable remarks and discussion that improved well my knowledge.

I would like to thank the thesis committee members Prof. Fabrice GUILLET and Dr. Haifa. ZARGAYOUNA for their valuable suggestions throughout the thesis which have also contributed greatly to the improvement of the thesis.

I'm also grateful to the hosting laboratories LS2N, IRISA, and L'ARiCoD for allowing me to join their teams and giving me access to the laboratory and research facilities. Without their precious support, it would not be possible to conduct this research.

Finally, I would like to thank my family and my friends in France and Lebanon for their encouragement and for the great and happy time we spent together which helped me blow off stress and lifted my mood during the difficult times of the Ph.D. journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Hypernym relation is a semantic relationship between a specific term (hyponym) and a generic term of it (hypernym). For instance, "musical instrument" is hypernym of "piano". Hypernym relations play a central role in building taxonomies (i.e. semantic hierarchies relating concepts, expressing a classification), considered the backbone of building any ontology. Additionally, they are extremely important to address several open research problems and applications such as information retrieval and question answering.

Research Context

The term ontology was first used in philosophy and it stands for the study of existence and the systematic explanation of being that deals with the nature and structure of reality. In the last decades, ontology has been used highly in the field of computer science. In 1993, Gruber [36] originally defined ontology as an "explicit specification of a conceptualization". In 1997, Borst [67] defined an ontology as a "formal specification of a shared conceptualization". In 1998, Studer et al. [95] merged these two definitions into one definition: "**formal**, **explicit** specification of a **shared conceptualization**". **Formal** refers to the fact that the ontology must be machine-readable. **Explicit** means that the types of concepts used and the constraints on their use are explicitly defined. **Shared** means that ontology should capture knowledge that is accepted by the communities. **Conceptualization** is an abstract model of phenomena by identifying the relevant concepts of these phenomena. In other words, ontologies are rich repositories of knowledge represented in such a way that information systems including artificially expert systems are able to use them. The main components of ontology are concept, taxonomic relation, ad-hoc relation, axiom and instance. A Concept (also known as class) refers to abstract or concrete things (i.e. individuals, instances, objects) of the world, characterised by the same properties. A concept can also be defined as a set of terms that refer to it. A relation defines a semantic link between two concepts. Taxonomic relation is the most used and it expresses that one concept generalises (i.e. categorises) another concept. Axiom allows completing the definition of concepts and relations.

The manual building of ontology is a hard and tedious task that requires knowledge engineers and experts in the domain. Besides, human language texts (written for some purposes or specifically written as ontology requirement documents) can be considered the main sources of knowledge useful for building ontologies. The prominent advancement in natural language processing, machine learning, deep learning, and data mining have opened the way to ontology learning from texts, making this way of working hot and promising, providing effective support to the manual task.

The term ontology learning refers to the automatic or semi-automatic development of ontology even though full automation will be probably never possible. Buitelaar et. al [16] suggest that the ontology learning process is made of increasingly complex subtasks following the classical ontology layer cake shown in figure 1. Thus, the process starts by extracting relevant terms from texts. Synonym terms are clustered in synsets. A label is chosen for each synset having a unique meaning. This label will be the concept term representing this synset. Concept taxonomies are built, making possible a minimal support for reasoning over the ontology. This step is accomplished by identifying taxonomic relations between concepts. It often consists of hypernym relation extraction and then taxonomic relation induction. Ad-hoc (non-taxonomic) relationships between concepts are identified. Axioms comprising concepts, taxonomic and non-taxonomic relationships are extracted to make even more precise the represented knowledge. To populate ontology, instances of concepts and relations are extracted.

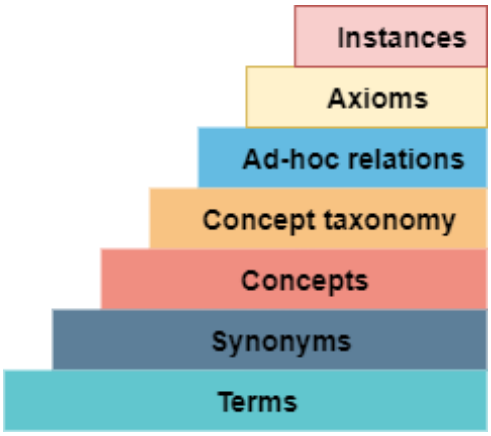


Figure 1: Ontology development layer cake from textual resources

Figure 2 provides an explicit view of the ontology learning process decorated with the several techniques proposed and used to accomplish the various subtasks for ontology learning [7, 104, 13]. These techniques fit into three main categories, i.e. **linguistic-based**, **statistical-**

based, and **logical-based**. Linguistic techniques are used to pre-process the text corpora. Terms, concepts and relations are extracted by using linguistic and statistical techniques. Logical techniques are used to identify axioms.

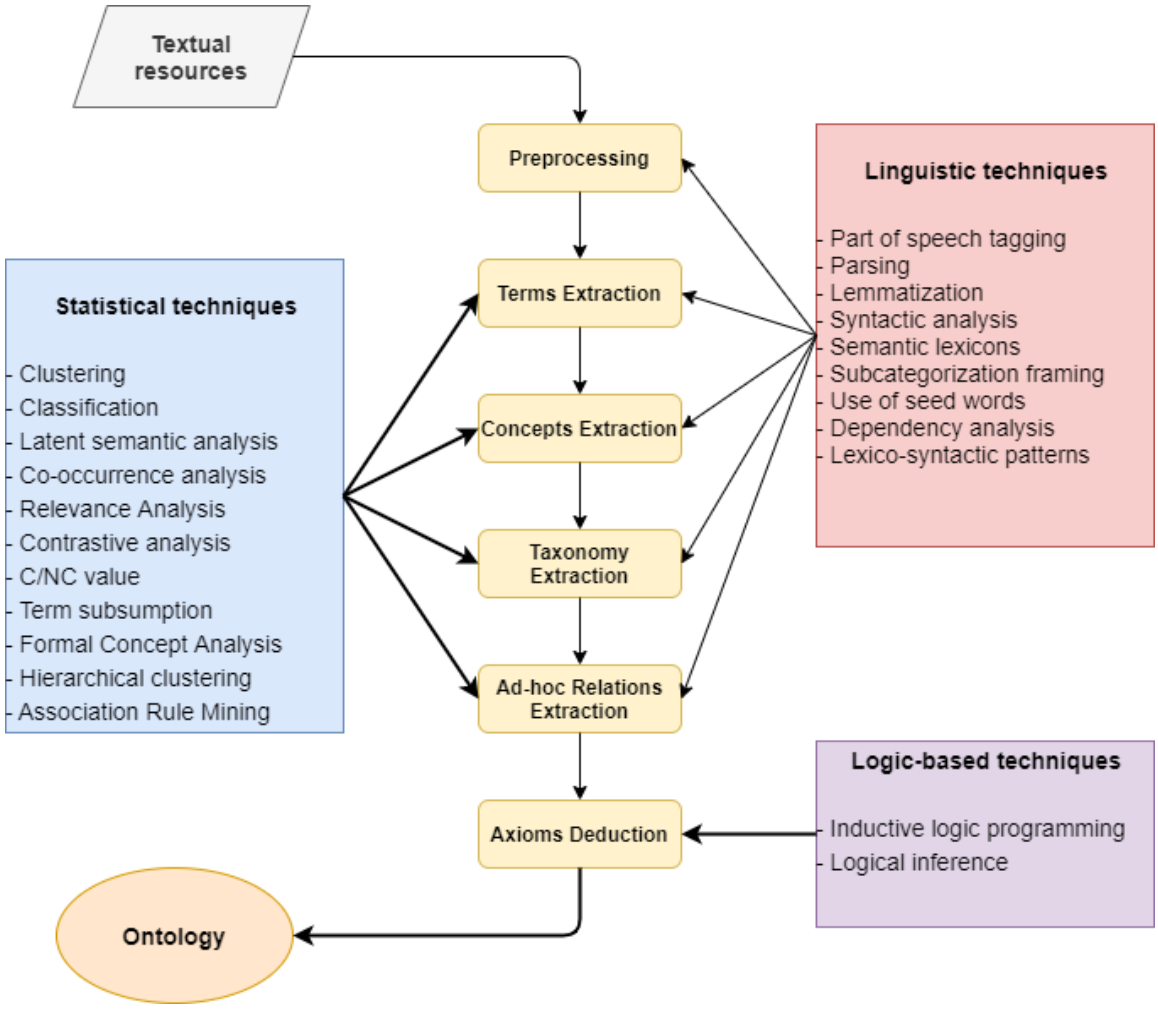


Figure 2: Ontology learning process

— **Linguistic techniques:** are basically used in the stage of pre-processing where techniques such as part of speech tagging [14, 62], parsing [57, 56, 91, 48], and lemmatization are applied to the text. To extract terms and concepts, linguistics techniques such as syntactic analysis [41], subcategorization framing [33], and use of seed words [42] are used. To extract relations, linguistic techniques such as dependency analysis [93] and lexico-syntactic pattern [40] are used.

— **Statistical techniques:** are mostly derived from information retrieval, machine learning,

and data mining. They are useful to extract terms, concepts, and relations. For term and concept extraction, techniques such as clustering [105, 47], latent semantic analysis [50, 97], co-occurrence analysis [15], contrastive analysis [70, 99], and C/NC value [32] are used. For taxonomic relation extraction, techniques such as term subsumption [31], classification [92], hierarchical clustering [109, 23], and association rule mining [94] are used. Classification and association rule mining are also used for ad-hoc relation extraction.

— **Logical techniques:** are mainly used to extract axioms using knowledge representation and reasoning of machine learning. Inductive logical programming [54, 58] and logical inference are the two main logical techniques used to extract axioms according to the literature review.

In the last two decades, various tools for supporting ontology learning have been developed. These tools take unstructured texts as inputs and build an ontology by either automatically or semi-automatically process. Some of these tools are: **ASIUM** [27, 28], **TextStorm/Clouds** [72], **OntoLearn** [65, 99, 98], **Text2Onto** [21], **CRCTOL** [45], and **OntoGain** [24].

However, current tools and techniques are still unsatisfactory for taxonomy extraction. One reason is that the quality of hypernym relations found and used is still unsatisfactory for the purpose. The other reason is that taxonomy extraction is not an easy task because several issues should be addressed. Let consider figure 3 showing an example of taxonomy in the domain of animal where each node refers to one concept and each edge refers to a taxonomic relationship.

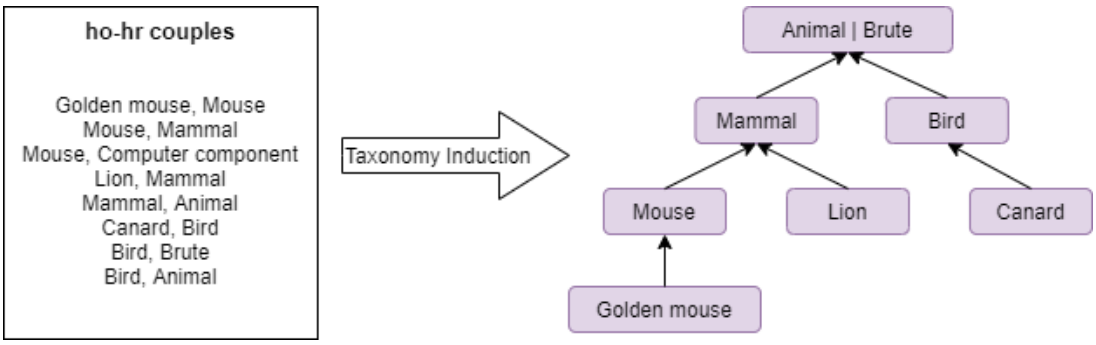


Figure 3: An example of taxonomy induction from ho-hr couples

First, hypernym relations do not necessarily satisfy the transitivity property due to the ambiguity of terms, while transitivity property is fundamental in taxonomies. For instance, the facts “Mouse” is hypernym of “Golden mouse” and “Computer component” is hypernym of

"Mouse" do not mean "Computer component" is hypernym of "Golden mouse". Second, hypernym relations hold between terms that are not necessary concepts, while taxonomic relations should hold between concepts in taxonomies. For instance, "Computer component" is hypernym of "Mouse" while "Computer component" is not a concept in the domain of Animal. Third, several hypernym relations may share synonym hyponym or hypernym terms, while synonym terms should be merged together in a single node in taxonomies. For instance, "Animal" is hypernym of "Bird" and "Brute" is hypernym of "Bird", but since "Animal" and "Brute" are synonyms they are merged together in one node (see figure 3).

In our work, however, we will focus on the quality of hypernym relations extracted from the given texts, trying to achieve better completeness and correctness of those relationships.

Motivations

A state of the art analysis reveals that two main types of approaches have been proposed for extracting hypernym relations: distributional and pattern-based. Despite the high interest in distributional approaches, we have oriented our research on the pattern-based approaches. The main reasons motivating our choice are presented hereinafter.

Patterns are interesting when dealing with ontology building because they are easily understandable and directly related to the most explicit but informally expressed knowledge in texts; distributional approaches are, on the contrary, conceptually oriented to hidden knowledge, which naturally completes the explicit knowledge found in texts. This is a good reason to assume that pattern-based and distributional approaches are not alternative but complementary. Thus, through this thesis, we will try to make evident this complementary assumption by performing an insightful comparison between what patterns and recent distributional approaches can achieve using common corpora and benchmark training datasets.

The interest of patterns is also closely related to their potential generality. Patterns are often applicable to various corpora even if they were learned from one corpus using supervised learning. On the contrary, the latest and highly effective distributional approaches using supervised learning tend to be highly dependent on the training data, so that requiring each time to learn new models. Thus, through this thesis, we will try to confirm the generality of the learned patterns and the high dependence of the supervised distributional approaches on the training data.

Patterns have a long and well-recognized tradition for building ontologies (and even concep-

tual models in the larger context of information systems), especially when specific requirement documents are used as source texts.

Unfortunately, current pattern-based approaches suffer from a low recall (as better justified in Section 1.1.5), and increasing recall without degrading precision is not an easy task. Through this thesis, we suggest to understand and proof how recall can be increased without degrading precision by analysing and experimenting three key pattern related aspects: pattern representation, pattern learning, and anti-pattern learning.

According to the state of the art, pattern representation ranges from lexico-syntactic representation to grammatical dependency representation. Grammatical dependency representations are semantically richer than lexico-syntactic representations. They seem to be more interesting for extracting hypernyms because they are linking terms based on their grammatical relations without considering their position in a sentence as with lexico-syntactic patterns. Nevertheless, using dependencies requires a computational effort, which is quite huge when compared to the one needed for lexico-syntactic representations. Therefore, there is the need to carefully study to what extent dependencies are interesting in improving recall and/or precision in comparison with lexico-syntactic representations.

Analysis of the state of the art reveals that following any pattern learning approach is interesting because recall can be systematically increased (with a rich enough corpus). However, precision may fall down if not controlled. There is therefore the need to develop a learning process where both precision and recall are put under control. The learning process should be therefore constrained to find patterns increasing both precision and recall. However, probably, precision cannot be kept if recall dramatically increases.

Some referred and reviewed works have proposed the usage of anti-patterns for filtering false hypernym relations. Anti-patterns are, for instance, meronym patterns representing when two related terms describe composed things. More generally, anti-patterns can be defined as patterns for hypernym relation counterexamples. We think that this is an effective way to proceed for mastering the trade-off between precision and recall. Thus, there is the need to address the anti-patterns (i.e. patterns for catching hypernym relation counterexamples) in the learning process.

Contributions

In the following, we briefly describe the contributions achieved by the performed works in this thesis addressing the motivations stated above.

1. Hearst's patterns [40] are the most popular patterns that have been extensively used to

extract hypernym relations from texts. Traditionally, they are lexico-syntactic patterns characterized by lexical and syntactical information that is obtained by performing shallow linguistic techniques such as tokenization and POS tagging. We first contribute by providing a new representation of Hearst's patterns based on grammatical dependencies obtained by performing dependency parsing. Then, we study to what extent this new representation improves Hearst's patterns in terms of precision and recall. The results confirm that the new representation leads to a considerable improvement in terms of recall with a slight reduction in terms of precision.

2. Our second contribution is an approach for learning patterns. To the best of our knowledge, existing approaches for learning patterns use machine learning techniques, and they suffer from sparsity issue (described in Section 1.1.5). To avoid sparsity issue, we propose to learn patterns using sequential pattern mining techniques. To avoid the fall down in the precision or recall, both metrics are put under control by splitting the approach into 3-phases. In each phase, the learning process is restricted to learn patterns that either improve precision or recall without degrading the other metric. In general, the first phase tries to learn patterns that better replace given set of seed patterns. The second phase tries to learn new patterns other than that correspond to the given set of seed patterns. And finally, in the third phase, anti-patterns are learned to filter counterexamples matched by the patterns learned in the first and second phases. In this contribution, we also provide a new representation of patterns. They are represented as sequential patterns i.e. sequence of itemsets where each itemset comprises lexical and/or syntactical information about terms, it also may comprise grammatical dependencies linking between terms. The results confirm that using our approach we were able to learn patterns and anti-patterns that combined together to outperform the given set of seed patterns by both precision and recall.

3. In the third contribution, we propose two pattern-based methods, unsupervised and supervised, for hypernymy detection based on the defined patterns in the first contribution and the learned patterns in the second contribution. For each method, experiments are performed to compare their performance to existing unsupervised and supervised distributional methods. Additionally, we perform an analysis to confirm the complementary between our pattern-based method and distributional method. Finally, we do an experiment to confirm the generality of the learned patterns on various corpora and the high dependence of the distributional method on the training data.

Thesis Structure

In addition to this introduction, this thesis is divided into 4 chapters and a conclusion:

Chapter 1: Literature Review. In this chapter, due to our interest in extracting hypernym relations, we first provide a detailed description of existing approaches for hypermym relations extraction. We also describe the evaluation processes, corpora, and datasets used to evaluate these approaches in the literature review. Finally, we provide a synthesis discussion to better understand the various features of the described approaches. Secondly, we introduce sequential pattern mining and then describe approaches for learning patterns using sequential pattern mining.

Chapter 2: Dependency Patterns. This chapter presents our first contribution. It describes the process of reformulating Hearst's patterns as dependency patterns using dependency parsing. It explains the usage of dependency patterns for extracting hypernym relations from texts. It also details our proposed evaluation protocol to evaluate hypernym patterns. Finally, we present the performed experiment to evaluate our patterns, the results of the experiment, and the analyses to the results.

Chapter 3: Sequential Hypernym Patterns. In this chapter, we present our second contribution: an approach to extract patterns automatically using sequential pattern mining and dependency parsing. We describe the 3 phases of the approach and the general workflow of steps to learn sequential patterns in each phase. We also describe our proposed representation of sentences as sequences. After that, we describe the experiments done to evaluate the approach. And finally, we provide the results and the analysis of the results.

Chapter 4: Methods for Hypernymy Detection Using our Patterns. This chapter describes the proposed pattern-based methods to compare our two main contributions to existing unsupervised and supervised methods for hypernymy detection. It also provides an analysis to confirm that pattern-based and distributional methods are complementary to each other. Finally, it describes the performed experiment to study the generality of the pattern-based method and distributional method.

Finally, a conclusion of the thesis is given and some points for future works are provided.

CHAPTER 1

LITERATURE REVIEW

This chapter provides a literature review of approaches for hypernym relations extraction and the processes of the their evaluation (section 1.1). In section 1.2, we provide a brief introduction about sequential pattern mining and describe some approaches for learning patterns using sequential pattern mining.

1.1 Approaches for Hypernym Relations Extraction

In this section, we describe approaches for extracting hypernym relations between terms from texts. Corpora, datasets, and evaluation metrics used to evaluate them are presented. Finally, a synthesis discussion is provided to better understand their various characteristics. Throughout the thesis, we also refer to hypernym relations by ho-hr couples (e.g. *ho-hr(Apple, Fruit)*).

These approaches are divided into three major types: **Pattern-based** approaches, **Distributional** approaches, and **Structural-based** approaches. Pattern-based and structural-based approaches are part of of linguistic techniques while distributional approaches are part of statistical techniques.

These approaches deal with hypernym relations by addressing a specific task: hypernymy extraction, hypernymy detection, or hypernymy discovery.

— **Hypernymy extraction task.** It consists on extracting ho-hr couples from scratch i.e. whenever only the corpus text is provided as input. It is formulated as: Given a corpus C as input, to extract all ho-hr couples from C.

— **Hypernymy detection task.** It consists on detecting whether candidate ho-hr couples are hypernyms or not. It is formulated as: Given, as input, both a corpus C and a dataset $D = \{(x_i, y_i)\}_{i=1}^{n}$ of n candidate ho-hr couples, to detect if each $(x_i, y_i) \in D$ represents an hypernym relation or not.

— **Hypernymy discovery task.** This task is recently addressed by some approaches. It consists on discovering hypernyms for a given set of hyponyms. It is formulated as: Given, as input, both a corpus C and a dataset $D = \{x_i\}_{i=1}^{n}$ of n candidate hyponym terms, to discover for each $x_i \in D$ a set of hypernym terms ($\{y_{i,j} : j \geq 1\}$) from C.

Pattern-based and structural-based approaches are often used for the hypernym extraction task; the distributional approaches are often used for hypernym detection or discovery tasks. However, in this section, we do not distinguish between the approaches according to the addressed task. This is because an approach specifically targeting hypernymy detection or discovery can be easily used for hypernymy extraction and vice-versa. A hypernymy detection approach can be used for hypernymy extraction from scratch by first identifying candidate ho-hr couples from the input corpus; a hypernymy discovery approach can be used for hypernym extraction by first identifying hyponym candidates from any term found in the input corpus.

1.1.1 Pattern-based Approaches

Pattern-based approaches are based on sets of patterns matching sentences, thus extracting ho-hr couples comprised in those sentences. Each pattern comprises specific placeholders for hyponym (ho) and hypernym (hr), needed for suggesting ho-hr couples found in any sentence matching with that pattern. The earliest and most popular patterns to extract hypernym relations were defined as **Lexico-syntactic Patterns**, and they are called Hearst's patterns. After that, several approaches have been proposed to improve the performance of these patterns in terms of precision and recall. Recent pattern-based approaches have a large interest to get benefits of machine learning techniques for the purpose of **Pattern Learning**.

Lexico-syntactic Patterns

The earliest and most popular lexico-syntactic patterns were introduced by Hearst [40, 39], they are known as Hearst's patterns. In general, these patterns are expressed as regular expressions and they are used to identify hypernym relations between noun phrases (NP) in a sentence. Hearst's patterns were initially defined for English language. Later on, similar patterns were developed for other languages such as French [66], Dutch [83], and Japanese [5]. Table 1.1 shows the 5 Hearst's patterns that were originally defined for the English language and the similar patterns defined for the French language (NPs refers to list of noun phrases). Recently, several works [87, 81] have added the pattern 'NP_{ho} was|were|is|are a NP_{hr}" to the set of original

Hearst's patterns, and it is considered as one of the Hearst's patterns. Similarly, through this thesis, we also consider it as one of the Hearst's patterns.

English	**French**
NP_{hr} such as NPs_{ho}	NP_{hr} tel que NPs_{ho}
	NP_{hr} comme NPs_{ho}
Such NP_{hr} as NPs_{ho}	NP_{hr} tel NPs_{ho}
NPs_{ho} and/or other NP_{hr}	NPs_{ho} et/or d'autre NP_{hr}
NP_{hr} especially NPs_{ho}	NP_{hr} particulièrement NPs_{ho}
NP_{hr} including NPs_{ho}	
NP_{ho} was|were|is|are a NP_{hr}	

Table 1.1: The Hearst's patterns and the similar patterns for the French language [66]

Let us consider the following sentences, Hearst's patterns, and the hypernym relations extracted by matching patterns on the sentences:

Sentences:

1. Musical instruments such as piano and guitar are very popular.
2. It features piano and other keyboards and is achingly beautiful in a slow and haunting way.
3. Belafonte went on to record in many genres, including blues, american folk, gospel, and more.

Hearst's Patterns:

i. NP_{hr} such as NPs_{ho}.

ii. NPs_{ho} and other NP_{hr}.

iii. NP_{hr} including NPs_{ho}.

Extracted hypernym relations:

— ho-hr(piano, musical instrument)

— ho-hr(guitar, musical instrument)

— ho-hr(piano, keyboard)

— ho-hr couple(blues, genre)

— ho-hr(american folk, genre)

— ho-hr(gospel, genre)

The pattern "NP_{hr} such as NPs_{ho}" means that a hypernym noun phrase (NP_{hr}) must be followed by the terms "such" and "as", that must be followed by one or more hyponym noun phrases (NPs_{ho}) to match a sentence and extract one or more hypernym relations. For instance, the pattern matches the above example sentence "Musical instrument such as piano and guitar are very popular" and extract two hypernym relations: ho-hr(piano, musical instrument) and ho-hr(guitar, musical instrument). The extracted hypernym relation "ho-hr(piano, musical instruments)" indicates that "piano" is a specific kind of "musical instrument". In other words, "musical instrument" is the hypernym of "piano" or "piano" is the hyponym of "musical instrument".

Based on lexico-syntactic patterns, several approaches were proposed to extract hypernym relations from web pages for the purpose of taxonomy induction. *Etzioni et al.* [25] introduced KNOWITALL, a fully automated system to extract hypernym relations from the web using Hearst's patterns. *Kozareva et al.* [49] also proposed an approach to induce taxonomy from

Lexico-syntactic patterns	Precision
NPs_{ho} and other NP_{hr}	0.7
NP_{hr} including NPs_{ho}	0.44
NPs_{ho} and any other NP_{hr}	0.76
NP_{ho} is a NP_{hr}	0.44
NPs_{ho} like other NP_{hr}	0.31
NP_{hr} such as NPs_{ho}	0.58
NPs_{ho} other than NP_{hr}	0.44
NP_{hr} like NPs_{ho}	0.17

Table 1.2: Lexico-syntactic patterns and their precision presented in [87].

the web using Hearst's patterns. In their papers, an evaluation using the precision and recall metrics was performed to show the performance of using Hearst's patterns to build taxonomies from Animals and Planets domains. The results showed good precisions in both domains (0.98 and 0.97 respectively), but their recalls were low (0.38 and 0.39 respectively). In 2012, *Wu et al.* [106] constructed Probase, a large database of concepts and hypernym relations extracted from billions of web pages using Hearst's patterns. It consists of 2.65 million terms and 20.76 million hypernym relations (precision in average = 0.92). More recently, *Seitner et al.* [87] also proposed a similar approach to build a large database of hypernym relations from the web using a set of 59 lexico-syntactic patterns collected from the literature review (instead of using only the 6 patterns of Hearst). Their constructed database consists of 401 million hypernym relations. In their evaluation, they computed a precision for each used pattern. Some patterns have a considerable precision, while others have a very low precision. Table 1.2 shows some of the lexico-syntactic patterns and their precision that were used to build the large database in [87].

A major issue of lexico-syntactic patterns is their low recall. Indeed, they suffer from several limitations that highly affect their recall and slightly affect their precision:

1. Patterns are limited in the scope i.e. they are able to only extract hypernym relations between terms occurring in the same sentence.
2. Usually, only few patterns are available (Hearst's patterns are only 6 patterns) while there are several ways to express the same hypernym relation in a text.
3. Patterns are prone to make errors due to parsing errors and/or sentence ambiguity. For instance, a sentence "Birds live in forest such as cardinals and snowy owls" matches the pattern "NP such as NPs" leading to wrongly identify "forest" (instead of "Birds") as hypernym of "cardinals" and "snowy owls".

Approaches to Improve Recall of Lexico-syntactic Patterns

Several approaches have been proposed to improve the recall of lexico-syntactic patterns by extending existing patterns or discovering new ones. Other approaches proposed to use hypernym inference rules to improve recall.

Jacques et al. [44] introduced new patterns by adapting or extending already existing patterns. For instance, they introduced the new pattern "NP_{hr} like NPs_{ho}" raising the existing pattern "NP_{hr} such as NPs_{ho}" by replacing "such as" by "like", the latter having a close meaning to the former. They also introduced the new pattern "NP_{ho} is the most NP_{hr}" by adding "the most" in the existing pattern "NP_{ho} is a NP_{hr}". Table 1.3 shows the original Hearst's patterns and the adapted or extended new patterns.

Existing patterns	New patterns
NP_{hr} such as NPs_{ho}	NP_{hr} like NPs_{ho}
NP_{ho} is a NP_{hr}	NP_{ho} is the most NP_{hr} NPs_{ho} which are kinds of NP_{hr}
NP_{hr} especially NPs_{ho}	NP_{hr} particularly NPs_{ho}

Table 1.3: New introduced patterns in [44] from existing patterns

In [75], a manual search was performed to extract lexico-syntactic patterns (paths) linking the hypernym "building(s)" to a given list of 132 hyponyms from Construction Textbooks Corpus (CTC). The CTC consists of 176 samples of textbooks published in the fields of construction engineering and architecture (approximately consists of one million words). No evaluation process was followed in the original paper to compute the performance of the extracted patterns. However, extracted patterns, after generalization by replacing the hypernym word "building(s)" by any noun phrase (NP), have been used by *Seitner et al.* [87] to build a large database of hypernym relations, providing a precision value for each pattern. Table 1.4 shows some of the extracted originally patterns, their occurrence frequency in the corpus (CTC), and their precision as presented in [87]. The results confirm that hypernym patterns can be extracted from a given corpus by searching for paths linking hyponyms to hypernyms. However, manual search is a hard and tedious task which prevents the usage of large corpora. Thus, an automatic process is significant to facilitate the task and allow the usage of large corpora. Furthermore, the results in table 1.4 show a low precision for most of the discovered new patterns (especially the ones not corresponding to Hearst's patterns), which confirms the difficulty and unreliability of the manual procedure.

Other approaches have been proposed to improve recall based on hypernym inference. It

Lexico-syntactic patterns	Frequency	Precision
buildings such as NPs$_{ho}$	29	0.58
NPs$_{ho}$ and/or other building	15	0.7
building example of this is NP$_{ho}$	4	0.18
example of building is NP$_{ho}$	2	0.33
NP$_{ho}$ is an example of building	2	0.36
building including NPs$_{ho}$	4	0.44
building types NPs$_{ho}$	3	0.17
NP$_{ho}$ is a building	3	0.44
building whether NPs$_{ho}$	3	0.13
buildings like NPs$_{ho}$	2	0.17
compare building with NPs$_{ho}$	1	0.16
NP$_{ho}$ sort of building	1	0.18

Table 1.4: Lexico-syntactic patterns extracted in [75]

should be noted that these approaches are not pure pattern-based approaches but they were proposed to overcome the scope limitation of lexico-syntactic patterns (i.e. hyponym and hypernym terms must occur in the same sentence). In [80], new hypernym relations were extracted based on the following rule: if term y is hypernym of term x and term z is similar to the term x, then y is hypernym of term z. For instance:

is-a(violin, instrument), similarity(fiddle, violin) :- is-a(fiddle, instrument)

In [6], Syntactic Contextual Subsumption (SCS) method was proposed to infer hypernym relations. The method is based on assumption that if a term y is a hypernym of term x, then $S_r(y)$ that denotes the set of objects that have a non-taxonomic relation r with y mostly contains $S_r(x)$ and not vice versa. For instance:

Given r = "eat" a non-taxonomic relation,
y = "animal" a candidate hypernym term,
x = "tiger" a candidate hyponym term,
$S_r(y = animal)$ = {grass, potato, mouse, insects, meat, wild boar, deer, buffalo},
and $S_r(x = tiger)$ = {meat, wild boar, deer, buffalo}.
Since $S_r(y = animal)$ mostly contains $S_r(x = tiger)$, we can induce "animal" is hypernym of "tiger".

Approaches to Improve Precision of Lexico-syntactic Patterns

Few approaches have been proposed to increase the precision of lexico-syntactic patterns. In [79], *Ponzetto et al.* proposed to filter non-hypernym relations by using meronym patterns. They denied hypernym relations if they were extracted by meronym patterns more frequently than by hypernym patterns. For instance, let us suppose the couple (x, y) is extracted once by using some hypernym patterns and the same couple (x, y) is extracted twice by using meronym patterns: (x, y) is removed from the list of extracted hypernym relations. Table 1.5 shows the lexico-syntactic patterns used to extract hypernym relations and meronym (non-hypernym) relations.

Hypernym patterns	Meronym patterns
NP_{hr} such as | like | especially NPs_{ho}	NP's NP
Such NP_{hr} as NPs_{ho}	NP in NP
NPs_{ho} and | or | like other NP_{hr}	NP with NP
NP_{ho}, one of the NP_{hr}	NP contain NP
NP_{hr} , the NPs_{ho} which	NP of NP
NP_{ho} is | are a NP_{hr}	NP are used in NP
NP_{hr} include | including NPs_{ho}	NP has | have an NP
	NP is | are part of NP

Table 1.5: Hypernym and mernonym patterns used in [79]

Other approaches to improve precision are based on statistical measures to estimate confidence score for the extracted ho-hr couples. Then, ho-hr couples with scores below a given threshold are ignored.

In [25], a confidence score for a couple (x, y) is estimated based on the co-occurrence of x within specific phrases called discriminator phrases DP (the phrases are extracted automatically from the corpus for each hypernym). For instance, let us consider ho-hr couple (piano, instrument), then "play x" is expected to be a discriminator phrase for the hypernym "instrument". Then, a confidence score for a couple (x, y) is computed as:

$$score(x, y) = \frac{f(x, DP(y)))}{f(x)} \tag{1.1}$$

where $DP(y)$ is the set of discriminator phrases related to y, $f(x, DP(y)))$ is the co-occurrence of x as part of any phrase of $DP(y)$ in the corpus, and $f(x)$ is the occurrence of x in the corpus.

Recently, *Roller et al.* [81] have proposed several measures to estimate a confidence score. The first one is a simple measure computing the extraction probability $p(x, y)$ (equation 1.2) of a given ho-hr couple (x, y).

First, let $\rho = \{(x, y)\}^n$ denote the set of n ho-hr couples that have been extracted from a given corpus using Hearst's patterns. Let $w(x, y)$ denote how often (x, y) has been extracted by Hearst's patterns and $W = \sum_{(x,y)\in\rho} w(x, y)$ the total number of extractions. Then:

$$p(x, y) = \frac{w(x, y)}{W} \tag{1.2}$$

The extraction probability as defined in Equation 1.2 is skewed by the occurrence frequency of each term x and y. Then, positive pointwise mutual information (PPMI, equation 1.3) has been used as a second measure to compute a confidence score. PPMI is the positive part of PMI, corresponding to the maximum between 0 and the value of PMI.

$$PPMI(x, y) = max(0, PMI(x, y)) \tag{1.3}$$

$$PMI(x, y) = log\frac{p(x, y)}{p^-(x)p^+(y)} \tag{1.4}$$

where $p^-(x)$ is the probability that x occurs as hyponym, and $p^+(y)$ is the probability that y occurs as hypernym.

$$p^-(x) = \frac{\sum_{(x,y)\in\rho} w(x, y)}{W} \tag{1.5}$$

$$p^+(y) = \frac{\sum_{(x,y)\in\rho} w(x, y)}{W} \tag{1.6}$$

The third measure is based on Singular Value Decomposition (SVD). In linear algebra, SVD is the factorization of a given matrix into three matrices U ,Σ, and V (equation 1.7). In computer science, it is used to perform a low-rank embedding (dimensionality reduction) of a given matrix. In [81], SVD is used to perform low-rank embedding of the PPMI matrix $X \in \mathbf{R}^{m\times m}$ with matrix entries $M_{x,y} = PPMI(x, y)$, where m is the total number of unique terms that have been extracted as hyponym or hypernym.

$$SVD(X) = U\Sigma V^T \tag{1.7}$$

Measure \ Dataset	BLESS	EVAL	LEDS	SHWARTZ	WBLESS
p	0.49	0.38	0.71	0.29	0.74
PPMI	0.45	0.36	0.7	0.28	0.72
spmi	**0.76**	**0.48**	**0.84**	**0.44**	**0.96**

Table 1.6: Average precision results for the three measures as presented in [81]

Hence, a confidence score of one ho-hr couple (x, y) is computed:

$$spmi(x, y) = u_x^T \Sigma_r v_y \tag{1.8}$$

where u_x, v_y denote the x-th and y-th row of U and V, respectively, and Σ_r is the diagonal matrix Σ with the r largest values are set to zero.

Table 1.6 shows the results presented in [81] using 5 datasets and average precision (AP) as evaluation metric (more details about datasets and AP evaluation metric are provided in Section 1.1.4). For all datasets, the best results are achieved by spmi. In addition to the improvement achieved in terms of precision, spmi also improves recall since it allows to infer hypernym relations between term couples that do not occur together in the same sentence. However, spmi and the other measures do not perform uniformly and their performances fall down for some datasets. Specifically, EVAL and SHWARTZ datasets comprise several ho-hr couples that cannot be discovered by using Hearst's patterns because these couples cannot be explicitly found in the corpus.

Hypernym Pattern Learning

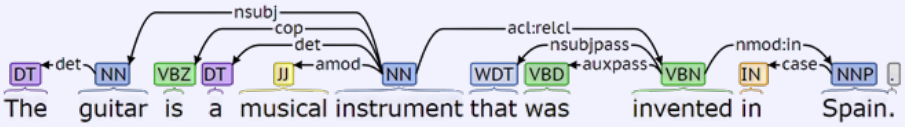
The manual definition of hypernym patterns is a difficult and tedious task, often leading to low precision (see table 1.4). Instead of manually defining patterns, other pattern-based approaches have been proposed to automatically learn patterns. These approaches learn patterns given a list of known hypernym and non-hypernym relations as input. The first approach was proposed by Snow et al. [92], they first suggest to collect all noun couples occurring in a corpus and label these couples as hypernym related or not by using WordNet [29]. Then, all shortest dependency paths linking between those noun couples in the corpus sentences are extracted by using MINIPAR dependency parser [56]. The dependency path is a set of dependency relations linking a couple of nouns in a dependency tree (more details are provided about dependency parsing and relations in Section 2.2.1). It may exist multiple dependency paths linking the couple of nouns, so shortest dependency path refers to the path with the minimum number of

dependency relations. In the following, we show an example of how we get the shortest dependency path linking a couple of nouns from the dependency tree of a given sentence. It should be noted that the couple of nouns are replaced in the shortest dependency paths by their POS tag (such as NN).

Shortest dependency path example

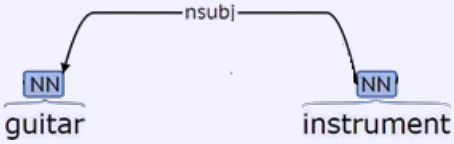
Example sentence: The guitar is a musical instrument that was invented in Spain

Dependency tree using enhanced Stanford parser[a] [86]:



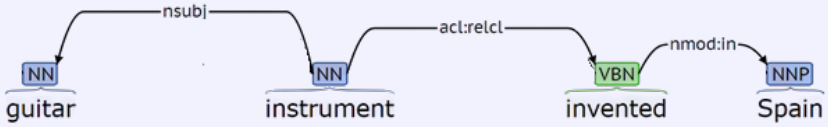
Noun couple (guitar, instrument)

Shortest dependency path: NN←nsubj, NN



Noun couple (guitar, Spain)

Shortest dependency path: NN←nsubj, instrument, acl:relcl→invented, nmod:in→NNP



a. The dependency tree is obtained by using the Stanford parsing online tool: `http://nlp.stanford.edu:8080/corenlp/process` (MiniPar is not accessible).

To capture some words that are important for hypernyms but not comprised in the shortest dependency path linking the noun couple (like "such" in "such NP as NP"), they suggest to extend the shortest dependency paths by satellite links where a word can be added to the path on either side of each noun. Additionally, to capture in a sentence multiple hyponyms for one hypernym as with Hearst's patterns, they suggest to distribute the same shortest dependency path across nouns that in conjunctions ("and" or "or") with other nouns. For instance, consider the following sentence "Animals such as tigers and cats belong to the same species"; "tigers" and "cats" nouns related by "and" conjunction share the common shortest dependency path with noun "Animals". All shortest dependency paths fit as features to train a logistic regression classifier. This classifier has the ability to predict hypernym relations between noun couples based

on the shortest dependency paths connecting them in the corpus. Finally, they consider paths with high weights in the classifier as patterns suggesting hypernym relations. By using a corpus of 6 million newswire sentences, they were able to rediscover Hearst's patterns and discover new hypernym patterns. Table 1.7 shows some new patterns discovered by the approach. In their experiment, they train and test multiple classifiers using multinomial Naive Bayes, complement Naive Bayes, and logistic regression. The logistic regression classifier showed the best F-score (equal to 0.348, while the F-score of Hearst's patterns is only 0.15).

NP_{hr} like NPs_{ho}
NP_{ho} is a NP_{hr}
NP_{ho} called NP_{hr}
NP_{ho}, a NP_{hr} (appositive)

Table 1.7: New discovered patterns by Snow approach [92]

A similar approach was later proposed by Sheena et al. [88] for learning both hypernym and meronym patterns using Stanford parser [61] instead of using Minipar [56]. Their learned classifier model predicts whether the nouns in a couple are hypernymy related, mernonymy related, or not related. Classifiers are learned by using ADTree and Naive Bayes. Table 1.8 shows some new discovered patterns.

Relation	**Pattern**
Hypernym	NP_{ho} a kind of NP_{hr}
Meronym	NP is a part/member of NP
Meronym	NP inside NP
Meronym	NP above NP

Table 1.8: New discovered hypernym and meronym patterns in [88]

A major limitation of approaches relying on shortest dependency paths to learn a hypernym classifier is the sparsity of the feature space. Indeed, the number of extracted shortest dependency paths and used as features is very large with few non-zero values. To overcome this limitation, Nakashole at al. [68] propose a method to reduce the number of shortest dependency paths by generalizing them. They first apply Stanford parser [61] to extract dependency paths, then they identify the shortest paths between named entities. These paths are syntactically and semantically generalized by replacing some words by POS tags, wildcards, and ontological type. POS tags stand for part-of-speech tags, wildcards (*) stands for zero to many possible

words, and ontological type is the semantic class name of a word. An example of generalized pattern is "$< person > [adj]$ voice * $< song >$'. This pattern matches sequence of words that start with an entity of ontological type "person" followed by a word of POS tag "adj", followed by a word "voice", followed by zero or many words (any words), and then finally end by an entity of ontological type "song".

1.1.2 Distributional Approaches

Distributional approaches are ground in the *distributional hypothesis*, which suggests that words sharing the same context tend to have a similar meaning [38]. These approaches predict if term y is a hypernym of term x by using the distributional representation of both terms. The distributional representation of a term is a vector in a **distributional semantic space**. Earlier distributional approaches are **unsupervised** that use statistical measures to predict hypernymy between terms. Recently, **supervised methods** have gained popularity to overcome the limitations of unsupervised approaches, especially after the availability of word embedding.

Distributional Semantic Spaces

A distributional semantic space is a vector space where each vector in the space stands for the distributional representation of a term. The distributional representation of terms is either contextual or syntactic. In contextual representation, the term is represented by its context words (Bag-Of-Words) [101]. In syntactic representation, the term is represented by the syntactic relations of the term [55]. Tables 1.9 and 1.10 show examples of contextual and syntactic distributional representation of terms respectively.

Term \ Context	music	Italy	guitar	band	rock
piano	3	2	0	1	0
instrument	1	0	3	0	2

Table 1.9: Contextual representation example

We can describe a distributional semantic space as a matrix M where each row stands for a term while each column stands for a context. The value of the entry $M_{i,j}$ represents the association between the term t_i and the context c_j [90]. This value has been widely represented by frequencies measures i.e. the occurrence number of c_j as a context of t_i in the corpus. For instance, in table 1.9, the value of $M_{1,1}$ is 3 and it means that c_1 "music" occurs 3 times as a

Term \ Syntactic	nsubj(include)	dobj(include)	dobj(eat)	nsubj(move)	nsubj(chrip)
Animal	2	0	1	1	0
Bird	0	1	3	1	1

Table 1.10: Syntactic representation example

context of t_1 "piano". Hereinafter, we mention two feature weightings that have been also used to represent the association between t_i and c_j:

— **Positive Pointwise Mutual Information (PPMI):** is the maximum value between zero and the log ratio between the joint probability of t and c and the product of their marginal probabilities [17]:

$$PPMI(t,c) = max(0, log\frac{P(t,c)}{P(t)P(c)}) \tag{1.9}$$

where $P(t)$, $P(c)$, and $P(t,c)$ are the relative frequency of t, c, and the pair (t,c) respectively.

— **Positive Local Mutual Information (PLMI):** is PPMI multiplied by the co-occurrence frequency of t and c to balance the values [26]:

$$PLMI(t,c) = freq(t,c) * PPMI(t,c) \tag{1.10}$$

More recently, most distributional approaches have shifted to take benefits of word embedding, instead of using traditional distributional semantic spaces, to get the distributional representation of words. The word embedding is the process of encoding words into low dimensional and continuous vector space model. The encoding of words by capturing the extent to which words occur in similar contexts enables the semantic and syntactic encoding of words. In other meaning, words that share similar contexts will be near each other in the encoded semantic space.

Word2Vec is the most popular method for word embedding, it was proposed by Mikolov et al. [63]. They apply a machine learning trick to learn a neural network with a single hidden layer, where the weights of the hidden layer are the word vectors. They propose two models: Continuous Bag-Of-Words (CBOW) and continuous Skip-gram. CBOW model uses the context words to predict the main word, while skip-gram model uses the main word to predict the context words. As an alternative to Word2Vec, Pennington et al. [78] introduce Glove, a word embedding method based on contextual windows. It builds a regression model based on global word co-occurrence counts.

Unsupervised Methods

Earlier unsupervised methods mostly used symmetric measures such as cosine similarity [82] and Lin similarity [55] to predict hypernymy. These measures are based on the assumption that hyponym and hypernym share similar contexts. In the following equations, $\overrightarrow{v}_x$ and $\overrightarrow{v}_y$ denote the feature vectors of terms x and y respectively, and F_x and F_y denote the sets of context words of term x and y respectively.

$$Cos(x, y) = \frac{\overrightarrow{v}_x \cdot \overrightarrow{v}_y}{\|\overrightarrow{v}_x\| \cdot \|\overrightarrow{v}_y\|} \tag{1.11}$$

$$Lin(x, y) = \frac{\sum_{c \in F_x \cap F_y}[\overrightarrow{v}_x[c] + \overrightarrow{v}_y[c]]}{\sum_{c \in F_x} \overrightarrow{v}_x[c] + \sum_{c \in F_y} \overrightarrow{v}_y[c]} \tag{1.12}$$

Hypernym relation is a directional relation where if x is a hypernym of y it does not mean that y is a hypernym of x. Thus, symmetric measures are not able to distinguish between hypernym and hypoynm ($Cos(x, y) = Cos(y, x)$). Consequently, asymmetric measures are then introduced to overcome this limitation. Most of these measures are inclusion measures since they are based on *Distributional Inclusion Hypothesis* (DIH). The hypothesis assumes that hypernym (y) contexts include most of the hyponym (x) contexts. Hereinafter, we list some inclusion measures. $(x \rightarrow y)$ denotes y is a hypernym of x.

- **Weeds Precision [102]:** a precision-based measure that computes the quantity of inclusion of x's contexts by y's contexts.

$$WeedsPrec(x \rightarrow y) = \frac{\sum_{c \in F_x \cap F_y} \overrightarrow{v}_x[c]}{\sum_{c \in F_x} \overrightarrow{v}_x[c]} \tag{1.13}$$

- **CosWeeds [52]:** corresponds to the geometrical mean of $cosine$ similarity (eq. 1.11) and $WeedsPrec$ (eq. 1.13).

$$cosWeed(x \rightarrow y) = \sqrt{cos(x, y) \cdot weedsPrec(x \rightarrow y)} \tag{1.14}$$

- **ClarkeDE [22]:** is a new variation of Weeds precision measure where it quantifies the weighted coverage of x's contexts by those of y'.

$$ClarkeDE(x \rightarrow y) = \frac{\sum_{c \in F_x \cap F_y} min(\overrightarrow{v}_x[c], \overrightarrow{v}_y[c])}{\sum_{c \in F_x} \overrightarrow{v}_x[c]} \tag{1.15}$$

- **invCL [52]:** is a measure that takes into account not only the inclusion of x's contexts by y's contexts, but also the non-inclusion of y's contexts by x's contexts. Both of inclusion and non-inclusion are measured by $ClarkeDE$ (eq. 1.15).

$$invCL(x \rightarrow y) = \sqrt{ClarkeDE(x \rightarrow y) \cdot (1 - ClarkDE(y \rightarrow x))} \tag{1.16}$$

In a recent work [84], Santus et al. point that DIH is not correct for all cases. For instance, "American" is a hypernym of "Barack Obama" but the (political) contexts of "Barack Obama" cannot be covered by those of "American".

Thus, they propose an entropy-based measure ($SLQS$) based on assumption that the most contexts of a hypernym are more general and less informative than the contexts of its hyponym. For instance, "move" and "eat", the context words of the hypernym "animal", are more general and less informative than "bark" and "has fur", the context words of the hyponym "dog".

$$SLQS(x \rightarrow y) = 1 - \frac{E_x}{E_y} \tag{1.17}$$

where E_x and E_y correspond to the median entropy of x and y for top N contexts (N is a hyper-parameter that is given as input).

$$E_x = median_{i=1}^{N}(H(c_i)) \tag{1.18}$$

where $H(c_i)$ is the entropy of the $i - th$ context.

$$H(c_i) = -\sum_{i=j}^{n} p(c_j|c_i)\ log_2(p(c_j|c_i)) \tag{1.19}$$

where $p(c_j|c_i)$ is the probability of the context c_j given the context c_i.

In [90], Shwartz et al. measure the ability of many unsupervised measures to detect hypernym relations from other types of relations in four different datasets (datasets are explained in section 1.1.4). They compute each measure using different context types (contextual and syntactic) and different feature weightings (Frequency, PPMI, and PLMI). They found that there is no common combination of measure, context type, and feature weighting that performs best in all datasets. However, they concluded that measures mostly show better performance when using syntactic contexts due to the fact that these contexts are semantically richer.

Datasets	Symmetric		Asymmetric	
	Addition	Multiplication	Concatenation	Difference
BLESS	0.66	0.56	0.68	**0.74**
WordNet	0.37	0.45	0.74	**0.75**

Table 1.11: Results comparing different representations in supervised approaches using term embedding as presented in [103]

Supervised Methods

Supervised approaches rely on a training dataset to learn a supervised model. The model is then used to predict hypernym relations between couple of terms. Most of these approaches are based on term embedding [63, 78], where the terms x and y of the couple are represented each one by its term embedding vector.

Term embedding vectors are combined in either symmetric or asymmetric representations. Pointwise addition and pointwise multiplication are examples of symmetric representations. Concatenation $\vec{x} \oplus \vec{y}$ [9] and difference $\vec{y} - \vec{x}$ [81, 103] are examples of asymmetric representations.

In [103], Weeds et al. have experimented different types of representations (symmetric and asymmetric) on two datasets: BLESS and a dataset constructed using WordNet. They used the linear SVM with RBF kernel to learn various classifiers. Table 1.11 shows the resulting accuracy (accuracy metric is described below in section 1.1.4) for each classifier on the two datasets as presented in [103]. They concluded that the two classifiers based on asymmetric representation perform better than those based on symmetric representation: more specifically, the representation based on the difference between term embedding vectors yields the best result.

In a recent work [108], Yu et al. assume that most of the previous supervised models that are based on term embedding are still based on the DIH, thus they are not strong enough to indicate hypernym relations. They propose an approach to encode hypernym properties in order to indicate hypernym relations using term embedding. They introduce a dynamic distance-margin model to learn two embedding models: hyponym embedding model and hypernym embedding model. Hyponym embedding model is trained using all terms that occur as hyponyms, while hypernym embedding model is trained using all terms that occur as hypernyms. Then, they learn an SVM classifier by using as a feature vector the concatenation of the hyponym emdedding vector, hypernym embedding vector and the norm distance between the two vectors ($\vec{x} \oplus \vec{y} \oplus ||\vec{y} - \vec{x}||$). The SVM classifier is trained using RBF Kernel. In their experiment, the result shows that their classifier has better accuracy than a classifier using traditional term embedding (word2vec).

Models		word2vec	Yu [108]	Luu [60]
Datasets	BLESS	84.0%	90.4%	**93.6%**
	ENTAILMENT	83.3%	87.5%	**91.7%**

Table 1.12: The result of comparing the various supervised classifiers on two datasets as presented in [60].

More recently, Luu et al. [60] said that the approach proposed in [108] is heavily dependent on the training data. In other words, their model is unable to predict hypernym relations between a couple of terms if it is not in the training data. They propose an approach to encode hypernym properties by learning term embedding that not only indicate the information of hyponym and hypernym, but also the contexts between them. First, they extract hypernym relation from WordNet to use them as training data. Then, they extract all sentences from Wikipedia that contains at least one hypernym relation. Consequently, they define triples of hypernym, hyponym and the contexts words between them. Finally, they use these triples as training data to learn the embedding model using dynamic weighting neural network. The trained embedding model is then used to build an SVM model by using as a feature vector the concatenation of the embedding vector of hyponym, the embedding vector of hypernym, and the difference between the two vectors ($\vec{x} \oplus \vec{y} \oplus \vec{y} - \vec{x}$). The SVM classifier is trained using RBF kernel. Similarly, they train two other classifiers using the term embedding of Yu et al. [108] and the traditional term embedding (Word2Vec [63]). Table 1.12 shows the accuracy of the three classifiers as presented in [60]. The best results are achieved by their approach.

1.1.3 Structural-based Approaches

Structural-based approaches extract hypernym relations based on the structure of terms. According to Sparck Jones [46], the head word in a compound noun (such as a noun phrase) refers to a more general semantic category, whereas the modifier distinguish this member from other members of the same category. For this reason, he introduced head-modifier principle where the head of the compound noun is considered as the hypernym of the complete compound. For example, “instrument” is the hypernym of “musical instrument”. In compound nouns that contain three or more words, they identify multiple hypernyms. For example, from “keyboard percussion instrument”, we can infer two hypernym relations: ho-hr(keyboard percussion instrument, percussion instrument) and ho-hr(percussion instrument, instrument). Moreover, hypernym relations are identified from single-word nouns where the hypernym occurs as suffix substring in the nouns. For example, we may infer the hypernym relation ho-hr(songwriter, writer) from

the single-word noun “songwriter”. Figure 1.1 shows examples of hypernym relations identified from compound nouns based on their structure.

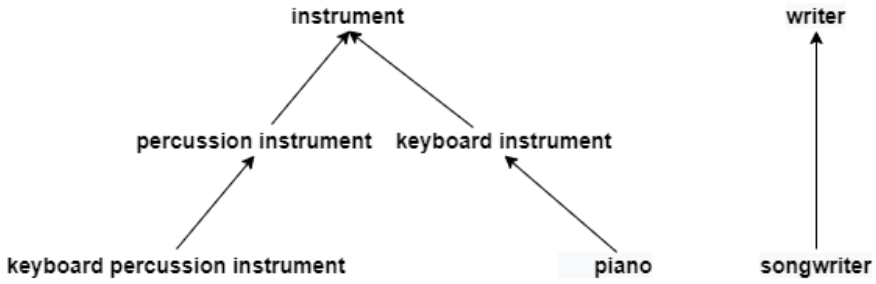


Figure 1.1: Examples of hypernym relations identified from compound nouns based on their structure

In [51], Lefever used head-modifier principle to identify hypernym relations. They propose to increase its precision by adding a threshold of at least minimum three characters for the identification of hypernyms. Using this threshold, they filtered out wrongly identified hypernym relations such as (sarattu, tu). In [76], Panchenko et al. propose to calculate a substring-based hypernymy score $\sigma(x, y)$ in order to detect hypernymy between a couple of terms (x, y). The score is computed only if y is substring of x and $length(y)$ is greater than 3; otherwise the score is equal to 0. In addition to English language, the score is used to detect hypernymy from Dutch, French, and Italian languages. For English and Dutch, the hypernym y should occur as suffix substring of x (e.g. “natural sciene” is-a “science”). For French and Italian, the hypernym y should occur as prefix substring of x (e.g. “algebre lineaire” is-a “algebre”, not “lineaire”). They also consider the case where the hyponym contains a preposition in English and Dutch languages. In this case, the hypernym should occur as prefix (e.g. “toast with bacon” is-a “toast”). In general, structural-based approaches show a good precision with an acceptable recall. In [51], they evaluated their structural-based module on six datasets, the average precision and recall on the six datasets are equal to 0.676 and 0.252 respectively. In [76], they plot the precision-recall curve of the score calculated on a testing dataset, the plot shows a high precision equal to 0.91 at the recall of 0.29.

$$\sigma(x, y) = \frac{length(y)}{length(x)} \tag{1.20}$$

1.1.4 Evaluation Process

The objective of the evaluation process is to evaluate the performance of the adopted approach in the context of the task (hypernym extraction from scratch, hypernym detection, hypernym discovery) to be accomplished. The process is indeed dependent on the task; figure 1.2 shows the main differences of the evaluation process according to the task. These differences are due to the task inputs and to the metrics used to assess the performance. Remainder of this section is first devoted to present the various corpora and datasets that have been used, as resulting from the literature review. Then, we describe the specific aspects of the evaluation process and related metrics in the context of each task mentioned above.

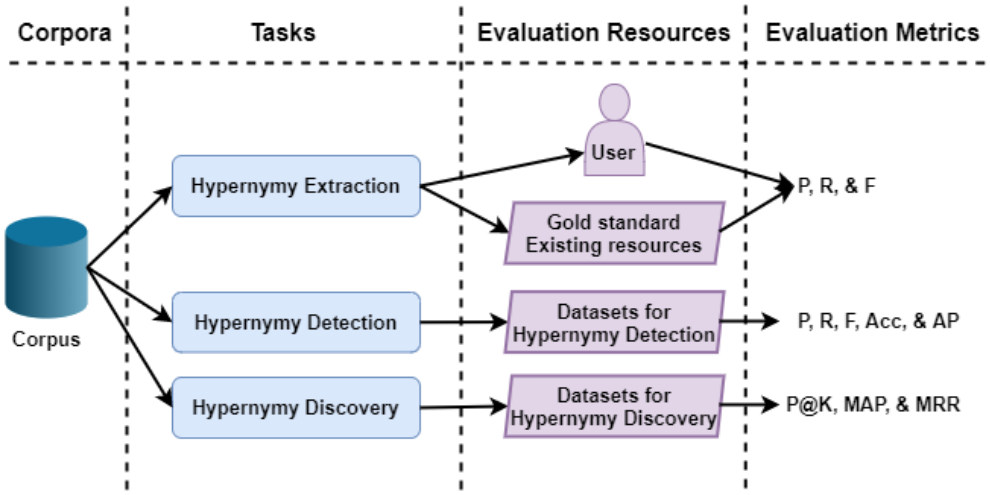


Figure 1.2: The evaluation processes.

Corpora

Corpora is the plural form of corpus. Corpus is a large collection of texts often collected from newspapers, journals, Wikipedia, web pages, and domain databases. It is either a domain-specific corpus or general-purpose corpus. In domain-specific corpus, texts are related to one specific domain such as Music, Medical, and so on. On the contrary, texts in general-purpose corpora are not related to any specific domain. In the literature review, corpora are used to extract ho-hr couples by matching patterns to their sentences [40], to learn patterns [92], and to build distributional semantic spaces that are useful by distributional approaches [90].

Hearst [40] used Grolier's American Academic Encyclopedia [35] to extract hypernym relations. In [92], Snow et al. used a corpus of 6 million newswire sentences. It contains articles col-

lected from the Associated Press, Wall Street Journal, and Los Angeles Times. Nakashole et al. [68] applied their algorithm to extract hypernym relations on two corpora: the New York Times archive (1.8 Million newspaper articles from the years 1987 to 2007), and the English edition of Wikipedia (3.8 Million articles as of June 21, 2011). In [69], the experiments were performed using two different corpora: a corpus of 4,916 Wikipedia sentences, and a subset of ukWaC Web corpus [30] (300,000 sentences). ukWaC is a 2-billion word corpus constructed by crawling the .uk domain of the web. In [90], Shwartz et al. used a concatenation of ukWaC corpus and a 2009 dump of the English Wikipedida [10]. While in [89], they only used a 2015 Wikipedia dump. Recently, Roller et al. [81] used a concatenation of Gigaword [34] and Wikipedia dump. In the hypernym discovery task of SemEval2018 [18], the task organizers provided 5 corpora: 3 general-purpose corpora and 2 domain-specific corpora. Here after, we detail these corpora:

1. **English corpus (UMBC):** a 3-billion word English corpus extracted from the web and it contains information from different domains [37].
2. **Italian corpus (itWac):** a 1.3-billion word Italian corpus extracted from different sources of web from the *.it* domain [10].
3. **Spanish corpus:** a 1.8-billion word Spanish corpus extracted from different sources of web [19]
4. **Medical corpus:** a 130-million word English corpus for Medical domain. It is a combination of abstracts and research papers provided by the MEDLINE (Medical Literature Analysis and Retrieval System).
5. **Music corpus:** a 100-million word English corpus for Music domain. It is a concatenation of music biographies contained in ELMD 2.0 [73], the music branch from Wikipedia, and a corpus of album customer reviews from Amazon [74].

Evaluation process and metrics for the Hypernymy Extraction Task

The task of hypernym extraction from scratch is defined as: Given a corpus C as input, to extract all ho-hr couples from C. Let $HC = \{(x, y)\}_{i=1}^{n}$ denote the n extracted couples from C.

The evaluation process can be performed manually or automatically. The manual evaluation is performed by domain experts labelling each $(x, y) \in HC$ as hypernym (i.e. ho-hr couple) or not hypernym (i.e. not ho-hr couple). In automatic evaluation, the labeling of each (x, y) is performed thankful to existing general resources such as WordNet [29] or specific-domain gold standards (GS). In general, hypernymy extraction methods are usually evaluated in terms of precision due to the difficulty of knowing the expected number of ho-hr couples ($AllPos$)

to be extracted from the given corpus. Recall and F-measure are used to evaluate hypernymy extraction methods by comparing the extracted ho-hr couples to gold-standards—the expected number of ho-hr couples ($AllPos$) is equal to the total number of ho-hr couples exist in the gold-standard.

Let $TP = |\{(x, y)\ is\ labeled\ as\ hypernym, (x, y) \in HC\}|$ the number of extracted couples labeled as hypernym, $FP = |\{(x, y)\ is\ labeled\ as\ not\ hypernym, (x, y) \in HC\}|$ the number of extracted couples labeled as not hypernym, and $AllPos = |\{(x, y) \in GS, \forall\ (x, y) \in C\}|$ the number of all ho-hr couples in the corpus C and found in the given specific-domain gold standard. Usual precision and recall metrics are used, and defined as:

$$Precision = \frac{TP}{TP + FP} \tag{1.21}$$

$$Recall = \frac{TP}{AllPos} \tag{1.22}$$

F-measure (also called F-score) is the harmonic mean of precision and recall, defined as:

$$F - measure = 2 * \frac{Precison * Recall}{Precison + Recall} \tag{1.23}$$

Evaluation process and metrics for the Hypernymy Detection Task

The task of hypernym detection is defined as: Given, as input, both a corpus C and a dataset $D = \{(x, y)\}^n$ of n candidate ho-hr couples, to detect if each $(x, y) \in D$ represents an hypernym relation (predicted labels). It should be noted that several methods for hypernymy detection are supervised. In such methods, the dataset is split into training and testing. The training dataset is used to train a model, while the testing dataset is used to evaluate the method by using the model for detecting hypernymy for all its ho-hr couples.

• **Evaluation.** Hypernymy detection is usually evaluated as a simple binary classification task by using a labeled Dataset D—each $(x, y) \in D$ is labeled as hypernym or not hypernym. Precision, recall, f-measure, and also accuracy are the evaluation metrics. They are calculated by using the following elements: *true positives* (TP), *false positives* (FP), *false negatives* (FN), and *true negatives* (TN). These elements are computed by comparing between the given labels of

(x, y) couples in the Dataset D to the corresponding predicted labels (see table 1.13). Precision, recall, and f-measure are defined as above by 1.21, 1.22, and 1.23 ($AllPos = TP + FN$).

Given \ Predicted	hypernym	not hypernym
hypernym	TP	FN
not hypernym	FP	TN

Table 1.13: TP, FP, FN, and TN computed from actual and predicted labels

Accuracy is defined as follow:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.24}$$

Precision, recall, f-measure, and accuracy are single-value metrics that are computed using the detected labels (hypernymy or non-hypernymy) of the whole ho-hr couples in a given dataset. Alternatively, average precision (AP) is a measure that combines recall and precision for ranked ho-hr couples. This metric is commonly used with approaches providing a score for each ho-hr couples in a dataset, such as unsupervised and supervised distributional approaches. Ho-hr couples are ranked based on their detection score (from the highest score to the lowest one). Average precision is defined as follow:

$$AP = \sum_{i=1}^{n} (R_i - R_{i-1}) P_i \tag{1.25}$$

where P_i and R_i are the precision and recall at the i-th rank of the detected relations; and they are computed as defined before in equations 1.21 and 1.22 by considering the i-th highest ho-hr couples as hypernymy detected. Additionally, average precision at k ($AP@K$) is also used and computed by taking the first k ranks instead of all n extracted relations.

We list below datasets typically used for evaluating the task of hypernymy detection.

— **Bless:** is a dataset introduced by Baroni et al. [8]. It is designed for the purpose of evaluating distributional distributional semantic models. It includes 200 distinct terms. terms are single-word nouns in the singular form. Each term has a set of relatum words. A term and its releta is labeled by one of the following 5 relations: co-hyponym, hypernym, meronym, attribute, and event. An example of a term and its releta with their label, "cat-hyper-animal" where "cat" is the term, "animal" is the relata, and "hyper" is the label.

The labeling of relations is done using semantic and text resources and then validated by the authors.

— **Entailment:** is a dataset introduced by Baroni et al. [9]. They first build a 2.83-billion-token English corpus by concatenating the British National corpus [1], WackyPedia and ukWaC [2]. They then use the corpus to build a dataset by extracting all WordNet nouns and extract their hyponyms and hypernyms. The total number of instances of the dataset are 2770 with equal number of hypernym and non-hypernym instances. The hypernym and non-hypernym instances are manually validated.

— **EVALution:** is a dataset introduced by Santus et al. [85]. It is also designed for the purpose of training and evaluating distributional semantic models. it consists of 7500 tuples of several semantic relations: hypernymy, synonymy, antonymy, and meronymy. The tuples were extracted from ConceptNet 5.0 [59] and WordNet 4.0 [29].

— **Weeds:** is a dataset intorduced by weeds et al. [103]. It consists of 2564 tuples. The tuples are extracted from WordNet [29]. The dataset is balanced where the half of tuples are hypernyms while the other half are non-hypernyms.

Evaluation process and metrics for the Hypernymy Discovery Task

The task of hypernym discovery task is defined as: Given, as input, both a corpus C and a dataset $D = \{x\}^n$ of n candidate hyponym terms, to discover for each $x \in D$ a set of hypernym terms ($\{y_i : i \geq 0\}$) from C.

Here, the evaluation process and related metrics are inspired by the information retrieval task. Each $x \in D$ is considered a query and the discovered hypernym terms ($\{y_i : i \geq 0\}$) are considered the answers to the query. It should be noted that in information retrieval task ranks of answers are significant: thus, for each discovered y upon querying hyponym x, a rank/score value needs to be returned. The following metrics are used, close to information retrieval metrics:

— **Precision at k (P@K):** is the proportion of correctly discovered hypernyms in the top k ranked.

— **Mean Average Precision (MAP):** is the mean of average precision scores calculated for each query. It should estimate the capability of the approach to retrieve k number of

1. http://www.natcorp.ox.ac.uk
2. http://wacky.sslmit.unibo.it

hypernyms from textual data, as well as considering the precision of each of them.

$$MAP = \frac{1}{|Q|} \sum_{q \in Q} AP(q) \tag{1.26}$$

where Q refers to Queries i.e. input hyponym terms in this task (they consider each hyponym term as query).

— **Mean Reciprocal Rank (MRR):** it rewards the position of the first correct result in the ranked list of extracted hypernymy.

$$MPR = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{1.27}$$

where $rank_i$ is the rank position of the first relevant extracted hypernymy for the i-th hyponym term.

In SemEval2018 [18], the organizers of the hypernymy discovery task have made available 5 datasets and corresponding corpora. In each dataset, hyponyms and their corresponding set of ranked hypernyms are provided. The hypernyms for each hyponym have been extracted from multiple resources and manually validated. Table 1.14 shows the 5 datasets and the number of given hyponyms in each dataset.

Dataset	English	Spanish	Italian	Medical	Music
Nb of given hyponyms	3000	2000	2000	1000	1000

Table 1.14: Datasets size for hypernymy discovery evaluation.

1.1.5 Discussion

As stated before, approaches to extract hypernym relations are mainly divided into pattern-based and distributional approaches. Other approaches, which are structural-based, have been also used to extract hypernym relations. Until now, there is no shared viewpoint on which approaches are more effective for hypernym relation extraction. In this section, we provide a general discussion for these approaches and a summary of this discussion is shown in table 1.15 (P stands for precision, R stands for recall, F stands for F-score, and AP stands for average precision).

Pattern-based approaches extract hypernym if the path between a couple of terms (x, y) in a corpus matches a specific pattern. The main limitation of these approaches is that they only

extract explicit relations i.e. relations between terms that occur together in the same sentence. The patterns are either defined manually [40] or extracted automatically [92]. In [40], patterns are defined manually as regular expressions. They are understandable, easy to use and analyze, and their results are easily interpretable. However, they suffer from the following limitations: a limited number of patterns (6 patterns) and it is a hard task to extend them. Several works have used Hearst's patterns to induce taxonomies [49, 25]. In general, we can confirm that Hearst's patterns have good precision, but they suffer from low recall. In [92], authors propose to replace the manually defined patterns by patterns that are extracted automatically using machine learning. Although they solve the main limitation of Hearst's patterns (i.e. few numbers of patterns) by discovering new patterns, their approach suffers from a major limitation which is the sparsity of the feature space. In addition, their approach is a supervised approach; and in supervised approaches, it is a tedious task to build the training data to train the supervised model. In their paper, they compare the performance of their proposed approach to that of Hearst's patterns. For Hearst's patterns, they label a candidate ho-hr couple as hypernymy if at least it is matched one time by any of the Hearst's patterns; otherwise, it is labeled as non-hypernymy. Their approach shows better performance (232% improvement in F-score).

Distributional approaches extract hypernym relations between a couple of terms (x, y) based on their contextual representation. The main advantage of distributional approaches is that they are not restricted to extract hypernym relations between terms that occur together in the same sentence. However, distributional approaches are domain-dependent. For instance, a distributional semantic space created for the *Music* domain does not perform well if it is used for the *Medical* domain. The distributional approaches are either unsupervised [22] or supervised [103]. Unsupervised approaches are based on measures to compute a hypernymy score between a couple of terms. In general, these measures are easy to implement and use. However, their performance is corpus size-dependent. The small corpus is not suitable to build efficient distributional semantic space. In [90], they evaluate and compare several unsupervised distributional approaches to detect hypernymy using a large corpus and four datasets. In general, the results confirm that unsupervised approaches have a low performance. On the other hand, supervised approaches use machine learning to build classifier models that can detect hypernymy. These approaches mostly use word embedding to build distributional semantic space. In general, supervised approaches rely on training data to train the classifier model. However, it is a tedious task to create the training data and the model's performance is dependent on the size and quality of it. In tables 1.11 and 1.12, we show the performance of several supervised approaches. The results confirm that their performance is the best performance in comparison

Aproaches		Supervised	Advantages	Limitations	Results	Conclusion
Pattern-based	Hearst's patterns [40]	No	-Understandable patterns -Easy to use -Easy to analyze -Results are interpretable	-Limited number of patterns -Patterns are hard to extend -Extract only explicit relations	-P=0.98 & R=0.38 in [49] -F=0.15 in [92]	-Good precision -Low recall
	Snow approach [92]	Yes	-Discover new patterns	-Sparsity Feature space -Tedious task for building training data -Extract only explicit relations	-F=0.348 in [92]	-Better performance
Distributional	Unsupervised measures: -Similarity [82, 55] -Inclusion [102, 22, 52] -Informativeness [84]	No	-Easy to implement -Easy to use -Extract implicit relations	-Domain dependent -Corpus size dependent	-AP(invCL)=0.35 in [90]	-Low performance
	Supervised models [9, 81, 103, 108, 60]	Yes	-Extract implicit relations	-Tedious task for building training data -Domain dependent -Training data dependent	-See tables 1.11 & 1.12	-Best performance
Structural-based	Head-modifier principle [46, 51, 76]	No	-Easy to use	-Restricted to extract specific type of hypernym relations where the hypernym term is a part of hyponym term	-P=0.68 & R=0.25 in [51]	-Good precision -Low recall

Table 1.15: Synthesis table for the hypernym relation extraction approaches.

to other types of approaches. However, Levy et al. [53] show recently that the good performance of supervised approaches are due to lexical memorization. They show that supervised approaches learn whether y is a prototypical hypernym, regardless of x, rather than learning the hypernym relation between x and y. For example, let suppose the training set for learning the hypernym classifier consists of the following ho-hr couples: (piano, instrument), (guitar, instrument), and (drum, instrument), then the machine learning method may learn that "instrument" is prototypical hypernym, so the classifier will predict hypernymy for all couples of the form (x, instrument), regardless of the relation between x and instrument.

Structural-based approaches extract hypernym based on the structure of hyponym terms. In general, they are easy to use, but they are restricted to extract only hypernym relations where hypernym term occurs as a part of its hyponym term. In [51], they evaluate their structural-based approach and the results show good precision and low recall.

In general, we can state that supervised approaches are better than unsupervised approaches and the best performance is achieved by supervised distributional approaches. However, these approaches rely on training data where the task to create it is tedious and hard. Consequently, supervised distributional approaches are highly recommended when training data is available. Otherwise, someone must choose between creating the training data or using another approach. The another approach is either a set of manually defined patterns (such as Hearst's patterns),

Approach	Precision	Recall	F-score
Snow	0.843	0.452	0.589
Supervised distributional (concatenation)	0.901	0.637	0.746
Integrated approach	**0.913**	**0.890**	**0.901**

Table 1.16: Comparing integrated approach performance to that of other approaches.

unsupervised distributional approach, or structural-based approach. Approaches using manually defined patterns are recommended in applications where good precision is important regardless of the recall value. In applications where both precision and recall are important, unsupervised distributional approaches are recommended, but it is significant to have enough large corpus to create a distributional semantic space that performs well. Concerning structural-based approaches, they are only recommended to be used in corpus where we know there is a lot of compound terms that are useful to identify hypernym relations.

Recent works to extract hypernym relations propose to integrate two or more approaches together in order to achieve better performance. According to Mirkin et al. [64], the pattern-based and distributional approaches have certain complementary properties. Pattern-based methods are useful to indicate hypernym relations between terms that occur together in the same sentences with good precision, but they lack the ability to extract hypernym relations between terms that do not occur together in the same sentence, limiting their recall. While distributional-based methods are able to detect hypernym relations between terms that do not occur together in the same sentence leading to obtaining a good recall. Consequently, they propose a combined approach by learning a supervised model using a set of features obtained by concatenating pattern-based features and distributional-based features. More recently, Shwartz et al. [89] also propose an approach that integrate pattern-based and distributional methods. Their approach is also a supervised model where they learn a neural network model using a concatenation of three feature vectors. A pattern-based feature vector representing the dependency paths occurrences of a couple of terms (x, y), and the other two feature vectors are distributional-based vectors and they represent the embedding vectors of x and y. In table 1.16, we present the result of the integrated approach, the Snow approach, and the best supervised distributional approach (concatenation) as shown in [89]. The results confirm that the best performance is achieved by the proposed integrated approach.

As a conclusion, we can state that integrating pattern-based and distributional approaches leads to improve the performance. However, supervised distributional approaches already have a good performance and several works [108, 60] have been proposed to improve it. Thus, we find that the best way to improve the performance of integrated approaches is to improve the perfor-

mance of pattern-based approaches. Methods to improve pattern-based approaches mostly rely on extending the existing hypernym patterns by either manually defining or automatically extracting new patterns. The manual definition of new hypernym patterns is a difficult and tedious task, and mostly the defined patterns show a low precision. Thus, it seems more appropriate to consider methods for pattern learning i.e methods that extract patterns automatically. However, most methods to extract new patterns automatically are supervised methods that suffer from the sparsity of the feature space which negatively affects the performance. Consequently, we propose to consider algorithms for sequential pattern mining for the purpose of learning patterns. Sequential pattern mining is a task of mining sequential patterns from domains where ordering is important which is compatible with our purpose of extracting hypernym patterns from textual resources.

1.2 Pattern Learning Using Sequential Pattern Mining

In section 1.1.1, we have mentioned approaches [92, 88, 68] for learning hypernym patterns. In these approaches, machine learning methods have been used to learn hypernym patterns from a large set of dependency paths that fit as features to the methods. Alternatively, several researchers [71, 20, 11] suggest to use sequential pattern mining tools to learn patterns for the purpose of extracting semantic relations. To the best of our knowledge, sequential pattern mining tools have not been used for learning hypernym patterns. In the remainder, we provide a brief introduction about sequential pattern mining, and then we describe some approaches that use sequential pattern mining for learning semantic relations.

1.2.1 Sequential Pattern Mining (SPM)

Pattern mining is a task of mining data to extract association rules between database items, the first algorithm was introduced by Agrawal et al. [1] (Apriori Algorithm). Although, pattern mining is popular due to its applications in many domains, but it is not compatible for domains where ordering of events are important such as the domain of extracting information from texts where the order of words are important in sentences. To address this limitation, Agrawal et al. [2] introduce AprioriAll, a sequential pattern mining algorithm based on Apriori algorithm. Thus, Sequential pattern mining is a process of extracting sequential patterns from a set of sequences called sequence database. A sequence database S_d is a set of tuples (sid, S) where sid is a sequence identifier and S is a sequence. A sequence is a list of transactions

$S =< I_1, I_2, \cdots, I_n >$ (itemsets), and each transaction is a list of literals $I =< i_1, i_2, \cdots, i_m >$ (items). Given a set of sequences and minimum support threshold min_sup as an inputs, sequential patterns are the frequent subsequences in the sequence database where their number of occurrences, also called support $Sup(S)$, are greater than or equal to min_sup. A sequence $S_1 =< I_1, I_2, \cdots, I_n >$ is a subsequence of another sequence $S_2 =< I'_1, I'_2, \cdots, I'_m >$ if there exists integers $1 \leq j_1 \leq j_n \leq m$ such that $I_1 \subset I'_{j_1}, \cdots, I_n \subset I'_{j_n}$. For example sequence X = < {a}, {b,c} > is a subsequence of sequence Y = < {f}, {a,b}, {e}, {b,c,d} >.

For better understanding, consider the table 1.17 that presents a sequence database S_d consists of 4 sequences. For instance, suppose min_sup is equal to 2, then a sequential pattern $X =< \{a\}, \{b, c\} >$ is frequent, since it is included (subsequence) in 3 sequences (1, 2 ,and 4) of S_d. In sequence 3, both itemsets of X, $\{a\}$ and $\{b, c\}$, are included but they do not occur in the same order as they occur in X.

SID	Sequence
1	$< \{a, b\}, \{b, c, d\}, \{d\}, \{e\} >$
2	$< \{a\}, \{b, c, d\}, \{d\}, \{e\} >$
3	$< \{b, c\}, \{a\}, \{f\}, \{e\} >$
4	$< \{a\}, \{d\}, \{b, c\}, \{e, f\}>$

Table 1.17: Sequence Database

AprioriAll suffers from limitations that increase its computation time and complexity. It repeatedly scans the sequential database to calculate the support of candidate patterns i.e. very costly for large databases. And it generates candidate patterns that do not exist in the sequential database. PrefixSpan [77] was then proposed to address these limitations. They used projected databases to avoid multiple scans of sequential databases and generating of candidates that are not exist in the database. The previous mentioned algorithms were proposed to mine all frequent sequential patterns from the sequential database. However, frequent sequential patterns may contain sequential patterns that are subsequences of other sequential patterns and have the same support value. For instance, $X =< \{a\}, \{b, c\} >$ and $Y =< \{a\}, \{b\} >$ are both frequent sequential patterns of support value equal to 3 (from table 1.17), and at the same time, Y is the subsequence of X. Yan et al. [107] consider all sequential patterns that are subsequences of other sequential patterns with the same support as redundant patterns. For this reason, they proposed CloSpan, an algorithm for mining only closed sequential patterns. A sequential pattern is closed if it exists no more super-sequence pattern with the same support—a sequence X is a super-sequence of sequence Y if and only if Y is a subsequence of X. Similarly, Bide

[100] was also proposed to mine only closed sequential patterns. More recent, Bechet at al. [12] proposed CloSPEC, an algorithm for mining closed sequential patterns under constraints. These constraints allow to focus on only interesting sequential patterns. These constraints are:

— **Gap constraint:** refers to the minimum and maximum number of sequence itemsets that can be ignored between two other sequence itemsets.

— **Element number constraint:** refers to the minimum and maximum number of itemsets that can form the sequential patterns.

1.2.2 Approaches for Learning Semantic Relations Using SPM

Sequential pattern mining (SPM) algorithms have been used in several approaches [71, 20, 11] to learn patterns that are capable of extracting specific semantic relations from texts. In [71], SPM have been used to learn patterns for the purpose of extracting semantic relations between named entities from Wikipedia. The entities are: **Person**, **Organization**, **Location**, **Time**, and **Artifact**. "Founder(Bill Gates, Microsoft)" is an example of semantic relations where "Bill Gates" and "Micorsoft" are two entities and "Founder" is the relation indicating that Bill Gates is the founder of Microsoft. Starting from a set of given known semantic relations between entities, they first extract all shortest dependency paths that are connecting the entities in Wikipedia. After that, dependency paths are then transformed into sequences (i.e. called sequence database) to fit as input for SPM to extract frequent sequential patterns. In [20], a method has been proposed to learn sequential patterns using SPM for the purpose of extracting semantic relations that characterize interactions between genes in the biomedical domain. Similarly to [71], they start from given known relations between genes to create the sequence database. After that, they apply SPM on the sequence database to extract frequent sequential patterns. In [11], an extension of the latter method was proposed for the purpose of extracting relations between genes and rare diseases. Rather than using sequences of items, they use sequences of itemsets in order to extract more expressive sequential patterns than those extracted by using sequences of items as in [20].

A key point in these approaches is the representation of sequences that fit as input for SPM to extract frequent sequential patterns. In [71], Nguyen et al. represent each dependency path as a sequence of itemsets. Each itemset is either representing a word or representing a dependency relation connecting two words. In case of representing the word, they consider the combination of the lemma of the word and its POS tag. In case if the word is an entity of a known semantic relation, they only consider the POS tag of the word. For representing the dependency relation,

they consider the combination of the dependency relation and the direction of the relation. In [20], Cellier et al. represent each sentence where at least one known relation occurs inside it as a sequence of items. Each item is the concatenation of the lemma of the word and its POS tag. In [11], instead of representing the sentence as a sequence of items as in [20], Bechet et al. represent the sentence as a sequence of itemsets. Each itemset is representing a word in a sentence by considering the lemma of the word the first item in the itemset and its POS tag is the second item. Figure 1.3 shows an example of representing sequences as proposed in [71, 20, 11].

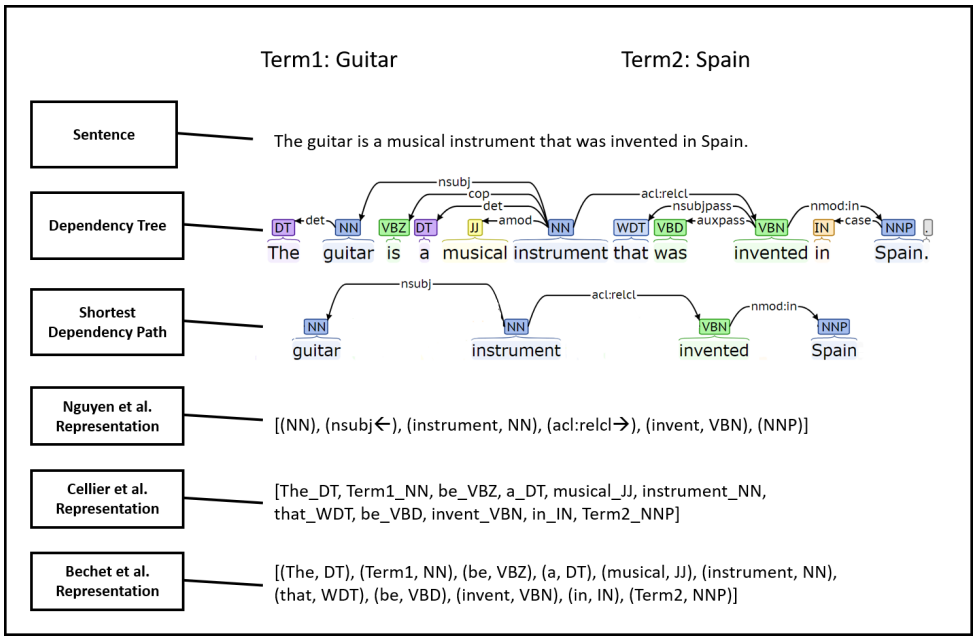


Figure 1.3: An example of sequence representation as proposed in [71, 20, 11]

1.3 Summary

Approaches for extracting hypernym relations from text are divided into three major types: pattern-based, distributional, and structural-based approaches.

Pattern-based approaches predict hypernym relations between terms if they satisfy particular patterns in sentences. These patterns are either defined manually or extracted automatically (pattern learning). The most popular patterns that were defined manually to extract hypernym relations from texts are the Hearst's patterns [40]. They are lexico-syntactic patterns that comprise lexical and syntactic information, and they are defined in the form of regular expressions.

Indeed, Hearst's patterns are only 6 patterns while there are several ways to express hypernym relations in a text. Consequently, they suffer from low recall. To improve their recall, several approaches have been proposed to adapt or extend Hearst's patterns [44, 75]. Another type of approaches to improve recall have been proposed based on hypernym inference [80, 6]. Although, Hearst's patterns have a good precision, few approaches have been proposed to increase their precision [79, 25, 81]. In [79], they use meronym patterns to filter non-hypernym relations identified by the Hearst's patterns. In [25, 81], they use statistical measures to estimate score for the extracted hypernym relations by Hearst's patterns, and then filter relations with score below a given threshold. Rather than defining patterns manually, several approaches have been proposed to extract them automatically [92, 88, 68]. These approaches rely on dependency parser to extract all shortest dependency paths connecting noun couples in a corpus. After that, the paths are used as features to learn hypernym classifier models using machine learning techniques. The models are then used to predict hypernym relations between candidate ho-hr couples. A major limitation of these approaches is the sparsity of the feature space. Instead of using machine learning tools to learn patterns, several approaches have been proposed to learn patterns using sequential patterns mining tools [71, 20, 11]. Note that these approaches have used sequential pattern mining to learn patterns for the purpose of extracting ad-hoc relations and not hypernym relations.

Distributional approaches are statistical methods that predict hypernym relations between terms based on their shared context of words. They are either unsupervised or supervised. Earlier unsupervised approaches were based on similarity measures such as **Cosine** [82] and **Lin** [55]. Similarity measures are symmetric measures that are not able to distinguish between hypernym and hyponym. Consequently, asymmetric measures are proposed based on Distributional Inclusion Hypothesis (DIH) [102, 22, 52]. More recent, Santus et al. [84] suppose that DIH is not correct for all cases, and they propose an entropy-based measure (SLQS). Supervised approaches rely on a training dataset to learn a hypernym classifier model. Most supervised approaches rely on word embedding [63, 78] to represent the feature vector between the couple terms x and y. For representing the ho-hr couple relationship, various vector representations have been used such as concatenation $\vec{x} \oplus \vec{y}$ [9] and difference $\vec{y} - \vec{x}$ [81, 103]. In general, supervised distributional approaches perform better than the unsupervised ones. However, Levy et al. [53] show recently that the good performance of supervised approaches are due to lexical memorization. They show that supervised approaches learn whether y is a prototypical hypernym, regardless of x, rather than learning the hypernym relation between x and y.

Structural-based approaches rely on the structure of terms to infer hypernym relations [46,

51, 76]. They are based on head-modifier principle i.e. the head of the compound noun is considered as the hypernym of the complete noun. In general, they show a good precision, but they are restricted to identify hypernyms that occur as a part of its hyponym term.

CHAPTER 2

DEPENDENCY PATTERNS

This chapter describes our work to study the impact of reformulating Hearst's patterns as dependency patterns instead of lexico-syntactic patterns using dependency parser. In the first section, we provide an introduction to explain the motivation of the work. In the second section, we present our work to formulate dependency patterns. We first describe about dependency parsing and relations. We then describe our proposed process to reformulate Hearst's patterns as dependency patterns. After that, we explain how dependency patterns could be used to extract hypernym relations between noun phrases. We finally present our implemented tool to use the dependency patterns on a corpus and extract hypernym relations. In the third section, we describe our experiment to evaluate dependency patterns. We first present the corpora and datasets used. We then describe our proposed process to label corpus sentences. And finally, we present the proposed evaluation protocol to evaluate and compare patterns. In the fourth section, we provide the results and the analyses of the results. A summary of the chapter is provided in the fifth and last section. It should be noted that this chapter works was first published in KEOD conference [4]. The conference paper was then extended into a chapter book to be published in Springer [43].

2.1 Introduction

Based on the literature review, we can distinguish between two formulations of patterns or paths: lexico-syntactic formulation and dependency relation formulation. It should be noted that patterns are paths that are suggested useful to indicate hypernym relations.

Lexico-syntactic formulation is characterized by lexical and syntactical information that are obtained by performing shallow linguistic techniques such as tokenization and POS tagging. While dependency relation formulation is characterized by dependency (grammatical) relations obtained by performing dependency parsing (more details are provided about dependency parsing and relations in Section 2.2.1). A comparison between both formulations has been done by Sang et al. [96]. For this purpose, they follow the supervised approach of Snow et al. [92] to

learn two classifiers: a lexical-based classifier and a dependency-based classifier. In the lexical-based classifier, lexical paths have been used as a feature space to learn the classifier. While in the dependency-based classifier, dependency paths have been used to learn the classifier. In their analysis results, they evaluate and compare the performance of both classifiers. They conclude that there is no much difference in performance between both formulations, while a high cost of computation time using dependency relation paths should be paid.

Despite the interest of the comparison study done in [96] to compare between both formulations, it is not sufficient to conclude which formulation is better due to the impact of supervised learning on the performance. In addition, it was not possible in their approach to study the impact of using dependency information instead of lexico-syntactic information by each pattern or path. In this work, we focus on understanding to what extent dependencies can improve the performance of lexico-syntactic patterns. In other words, we will study the strong and weak point of dependency patterns over lexico-syntactic patterns by comparing each dependency pattern to its corresponding lexico-syntactic pattern. For this purpose, we propose to manually reformulate lexico-syntactic Hearst's patterns as Dependency Hearst's Patterns (**DHPs**) and compare their performance pattern by pattern. We formulate a dependency pattern as an ordered set of *dependency relations* where each dependency relation is a binary grammatical relation between two words of a sentence.

To appreciate the differences between lexico-syntactic patterns and dependency patterns, let consider the following sentence: "I like musical instruments invented in Spain, such as guitar". Table 2.1 represents the lexico-syntactic information extracted from the sentence by applying shallow linguistics techniques and the dependency information extracted from the sentence by using the enhanced representation of Stanford dependency Parser [1]. In this example, lexico-syntactic information is not sufficient to identify the correct hypernym relation between the hypernym term "instrument" and the hyponym term "guitar'; worse, applying the pattern "NP_{hr} such as NP_{ho}" identifies a wrong hypernym relation between "Spain" and "guitar". On the contrary, dependency information seems to be sufficient to identify the correct hypernym relation between "instrument" and "guitar" thankful to the dependency relation "nmod:such_as(instrument, guitar)". Thus, dependency patterns are expected to be more precise in matching complex sentences since dependency relations provide closer meaning to the semantics of the sentence.

Seitner et al. [87] have collected a large set of lexico-syntactic patterns (59 patterns) from past literature to build a large database of hypernym relations extracted from the web. The ex-

1. https://nlp.stanford.edu/software/lex-parser.html

Lexico-syntactic	Dependency
I/PRP	nsubj(like, I)
like/VBP	root(ROOT, like)
musical/JJ	amod(instrument, musical)
instrument/NNS	dobj(like, instrument)
invented/VBN	acl(instrument, invented)
in/IN	case(Spain, in)
Spain/NNP	nmod:in(invented, Spain)
such/JJ	case(guitar, such)
as/IN	mwe(such, as)
guitar/NN	nmod:such_as(instrument, guitar)

Table 2.1: Lexico-syntactic and dependency relation information.

Patterns
NP_{hr} such as NP_{ho}
NP_{ho} and other NP_{hr}
NP_{ho} is a NP_{hr}
NP_{ho} is example of NP_{hr}
examples of NP_{ho} is NP_{hr}
NP_{ho} and any other NP_{hr}
NP_{ho} and some other NP_{hr}
NP_{hr} which is called NP_{ho}
NP_{hr} like NP_{ho}
NP_{hr} like other NP_{ho}

Table 2.2: Some examples of patterns collected from the past literature.

tended set consists of Hearst's patterns and additional lexico-syntactic patterns. Table 2.2 shows some examples of patterns from the collected patterns (NP_{ho} refers to the hyponym noun phrase and NP_{hr} refers to the hypernym noun phrase). From the set of collected patterns, we notice that some of these patterns are extensions of other patterns by adding one or more words and these additional words do not change the dependency path that may be used to identify hypernymy between "NP_{ho}" and "NP_{hr}". For instance, "NP_{ho} and some other NP_{hr}" is an extension of the Hearst's pattern "NP_{ho} and other NP_{hr}" by adding the word "some" before the word "other" and it has no effect on the dependency path connecting "NP_{ho}" to "NP_{hr}" that may be used to identify hypernymy (see figure 2.1).

Consequently, we expect that a dependency pattern replaces several lexico-syntactic patterns, leading to recall improvement. Table 2.3 shows two of Hearst's patterns and their extended

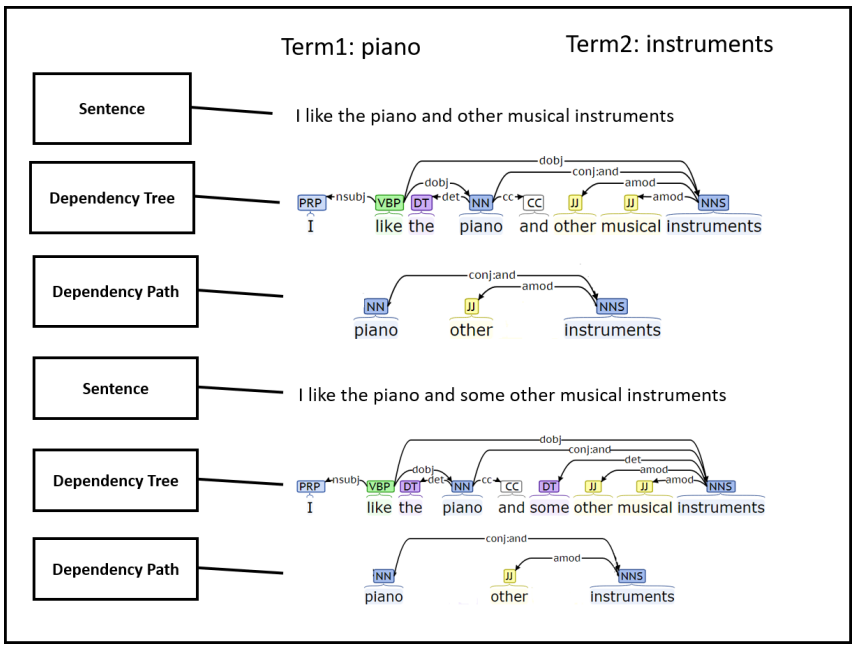


Figure 2.1: An example showing that dependency patterns may replace multiple lexico-syntactic patterns

Hearst's Pattern	Extended Pattern
NP_{ho} and|or other NP_{hr}	NP_{ho} and|or any other NP_{hr}
	NP_{ho} and|or some other NP_{hr}
	NP_{ho} and|or like other NP_{hr}
NP_{ho} is|are|was|were a NP_{hr}	NP_{ho} is|are|was|were example of NP_{hr}
	example of NP_{hr} is|are|was|were NP_{ho}

Table 2.3: Two of Hearst's patterns and their extended patterns that are expected to be replaced by two dependency patterns.

patterns that are expected to be replaced by only two dependency patterns. The examples of extended patterns shown in the table 2.3 are mentioned in the past literature, while our expectation is that dependency patterns are more generic patterns that may include lexico-syntactic patterns that are not extended before. For instance, "NP_{ho}, appositive phrase, is a NP_{hr}" is included in the dependency pattern corresponding to the Hearst's pattern "NP_{ho} is a NP_{hr}", because the appositive phrase does not affect the dependency path between NP_{ho} and NP_{hr} (e.g. "Lion, the king of the forest, is a dangerous animal").

2.2 Dependency Hearst's Patterns (DHPs)

2.2.1 Dependency Parsing and Relations

Dependency parsing is a Natural Language Processing (NLP) technique that provides the syntactic structure of a sentence. The syntactic structure of a sentence is described in terms of directed binary grammatical relations that hold between the words. These grammatical relations are also known as dependency relations each one comprising one governor and one dependent word—$Rel(Governor, Dependent)$. A major advantage of dependencies is the ability to deal with languages that have a relatively free word order. Another advantage of dependencies is that the dependency relations associate distant words in a sentence; in this sense, they are closer to the semantic meaning of a sentence [61].

Recently, additional and enhanced dependency relations have been presented in [86]. Examples of enhanced relations are the augmented modifiers, where all nominal modifiers in enhanced representation comprise the preposition e.g. nmod:such_as. Such relations facilitate the extraction of relationships between words. Figures 2.2 and 2.3 show the enhanced typed dependency tree for sentences "I like musical instruments invented in Spain, such as guitar" and "A march, as a musical genre, is a piece of music with a strong regular rhythm" respectively. Below, some dependency relations used in the remainder of this chapter are explained:

— **nmod:such_as:** nmod refers to nominal modifier, associating a non-head noun that serves as a modifier of a head noun. "such_as" is the preposition name of "nmod". For instance, the dependency relation "nmod:such_as(instrument, guitar)" (see figure 2.2) is useful to indicate the hypernym relation: ho-hr(guitar, instrument).

— **cop:** cop refers to copula, i.e. a relation between a copula verb and its complement (all verbs "to be" are copula verbs). For instance, for a sentence reported above, the dependency parser provides the relation cop(is, piece).

— **nsubj:** nsubj refers to nominal subject, i.e. it represents the subject of a clause. The head in the relation is not always a verb, it can be an adjective or a noun when the verb is copula verb. For instance, for a sentence reported above, the dependency parser provides the relation nsubj(march, piece).

2.2.2 DHPs Formulation Process

To reformulate Hearst's patterns in term of dependency relations (resulting in what we name Dependency Hearst's Patterns), we performed the following steps for each Hearst's pattern:

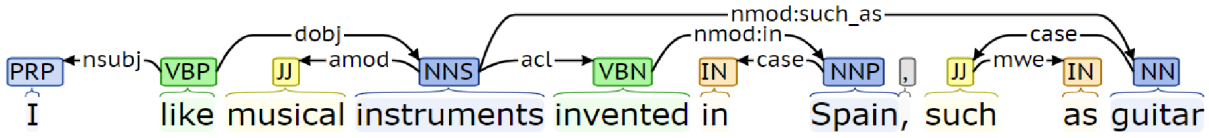


Figure 2.2: An example of enhanced typed dependency tree

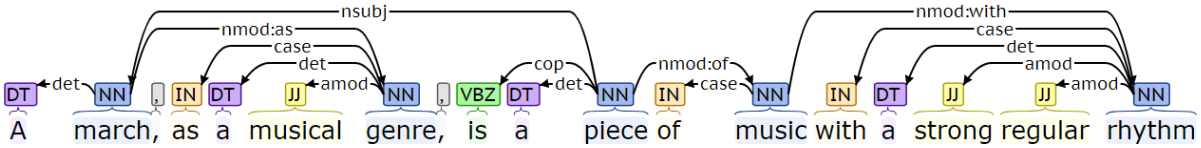


Figure 2.3: An example of enhanced typed dependency tree

i. selecting from a corpus a random set of matching sentences with the lexico-syntactic pattern [2].

ii. applying the dependency parser on each sentence to extract enhanced dependency relations.

iii. analyzing manually the set of parsed sentences of one pattern.

iv. defining a general dependency pattern corresponding to the Hearst's pattern.

Table 2.4 shows the 6 Hearst's patterns, and their corresponding dependency patterns (DHPs). Each dependency pattern is an ordered set of dependency relations. However, dependency relations represent syntactic relations between words, while a sentence expresses semantic relations between noun phrases rather than between words. For instance, the hypernym relation between "instrument" and "guitar" or that between "piece" and "march" are less semantically rich (even not correct) than those respectively between "musical instrument" and "guitar" and "piece of music" and "march". DHPs enable to extract hypernym relations between noun phrases instead of words using the notion of $NPHead$ (headword of a noun phrase). DHPs first suggest hypernym relations between words, then they check if the words are headwords of noun phrases to suggest hypernym relations between the noun phrases. $NP_{ho}Head$ and $NP_{hr}Head$ refer to the headwords of hyponym and hypernym noun phrases respectively.

2. we select 10 sentences

Hearst's Patterns	Dependency Hearst's Patterns
NP$_{hr}$ such as NP$_{ho}$	case($NP_{ho}Head$, such) nmod:such_as($NP_{hr}Head$, $NP_{ho}Head$)
Such NP$_{hr}$ as NP$_{ho}$	amod($NP_{hr}Head$, such) case($NP_{ho}Head$, as) nmod: as($NP_{hr}Head$, $NP_{ho}Head$)
NP$_{hr}$ including NP_{ho}	case($NP_{ho}Head$, including) nmod:including($NP_{hr}Head$, $NP_{ho}Head$)
NP$_{ho}$ and|or other NP_{hr}	cc($NP_{ho}Head$, and|or) amod($NP_{hr}Head$, other) conj($NP_{ho}Head$, $NP_{hr}Head$)
NP$_{hr}$ especially NP$_{ho}$	advmod($NP_{hr}Head$, especially) dep($NP_{hr}Head$, $NP_{ho}Head$)
NP$_{ho}$ was|were|is|are a NP_{hr}	nsubj($NP_{hr}Head$, $NP_{ho}Head$) cop($NP_{hr}Head$, was|were|is|are)

Table 2.4: Hearst's patterns and their corresponding dependency patterns

2.2.3 Hypernymy Extraction by DHPs

Corpus Pre-processing

Each DHP is a set of dependency relations that should match a sentence based on its syntactical structure (its dependency relations) to suggest hypernym relations between noun phrases. Thus, a pre-processing step is necessary to obtain the syntactical structure for each sentence in a corpus. For that purpose, we have implemented a Java process using CoreNLP Library [3]. The process applies on a given corpus the following NLP techniques: Sentence Splitter, Tokenizer, Lemmatizer, POS tagger, and Dependency Parser. Besides that, the process extracts all noun phrases in each sentence and identify the headword of each noun phrase to be used by DHPs to suggest hypernym relations between noun phrases using their headwords. The result of the pre-processing step is saved in a tagged syntax file that comprise all extracted information. We have used three tags: "$< s > ... < /s >$" to indicate the beginning and ending of each sentence, "$< NP > ... < /NP >$" to indicate the beginning and ending of noun phrases, and "$< headword > ... < /headword >$" to indicate the headword of each noun phrase. Figure 2.4 shows an example of tagged sentence after pre-processing.

In order to extract sentence noun phrases that can be involved in a hypernym relation, we use the phrase structure tree of the sentence. Figure 2.5 shows the phrase structure tree of the

3. https://stanfordnlp.github.io/CoreNLP/

Sentence	Tagged sentence after pre-processing							
	<s>							
	<NP>							
	<headword>	word	lemma	POS tag	word index	governor word	governor index	grammatical relation
Tracks		tracks	track	NNS	1	varied	3	nsubjpass-->
	</ headword >							
	</NP>							
are	are	be	VBP	2	varied	3	auxpass-->	
Varied	varied	vary	VBN	3	null	0	root<--	
From	from	from	IN	4	dance	5	case-->	
	<NP>							
	< headword >							
Dance		dance	dance	NN	5	varied	3	nmod:from<--
	</ headword >							
	</NP>							
To	to	to	TO	6	techno	7	case-->	
	<NP>							
	< headword >							
Techno		techno	techno	NN	7	varied	3	nmod:to<--
	</ headword >							
	</NP>							
To	to	to	TO	8	rock	9	case-->	
	<NP>							
	< headword >							
rock		rock	rock	NN	9	varied	3	nmod:to<--
	</ headword >							
	</NP>							
	</s>							

Figure 2.4: An example of tagged sentence after pre-processing

sentence "I like musical instruments invented in Spain, such as guitar". The noun phrases of the sentence are tagged by "NP". In a nested noun phrase where a noun phrase comprises the smallest noun phrases, we select the smallest ones. For instance, "musical instrument invented in Spain" is a noun phrase that comprises the two smallest noun phrases: "musical instrument" and "Spain". On the contrary, in case of nested noun phrases linked by preposition "of", we consider those noun phrases as a unique noun phrase (e.g. "piece of march") based on our assumption that "NP of NP" is more semantically rich than each noun phrase alone.

```
(ROOT
  (S
    (NP (PRP I))
    (VP (VBP like)
      (NP
        (NP (JJ musical) (NNS instruments))
        (VP (VBN invented)
          (PP (IN in)
            (NP (NNP Spain))))
        (, ,)
        (PP (JJ such) (IN as)
          (NP (NN guitar)))))
    (. .)))
```

Figure 2.5: An example of phrase structure tree

Based on our analysis for a set of noun phrases, we notice that a noun phrase headword is one of the noun phrase words that complies the following rules [4]:

— it is a noun.

— it is not a modifier of another noun within the parsed sentence (nmod).

— it is not a compound of another noun within the parsed sentence (compound).

For instance, consider the noun phrase "keyboard instrument", "keyboard" and "instrument" are both nouns, but "keyboard", according to dependency relations, is the "compound" of "instrument", then we consider "instrument" as NPhead.

4. These rules are language dependent, they are defined for English. Thus, they should be adapted to be used for other languages (e.g French).

DHPs Matching

After parsing a sentence and identifying its noun phrases and the headword of each noun phrase, one DHP could match a sentence and suggests hypernym relations between its noun phrases. It matches a sentence if each dependency relation in the pattern also occurs as dependency relation in the syntactical structure of the sentence, and the order of the dependency relations in the pattern is the same to that of the syntactical structure of the sentence. For example, if a DHP dependency relation of index i exists in the sentence set of dependency relations at index j, then the pattern dependency relation of index $i + 1$ should exist in the sentence set of dependency relations at index $j + k$, with i, j, & $k > 0$. After matching, DHP suggests one hypernym relations between pair of noun phrases by associating $NP_{ho}Head$ and $NP_{hr}Head$ of the pattern to the relevant noun phrases of the sentence.

However, lexico-syntactic Hearst's patterns are defined to match a sentence and extract one or more hypernym relations (one or more hyponyms for one hypernym). In order to identify one or more hyponyms for one hypernym when matching one DHP on a sentence, we use the conjunction dependency relation "conj($NP_{ho}Head$, $NP_{ho'}Head$)". Indeed, whenever a hypernym relation (ho, hr) is extracted by a pattern, we look for occurring relations as conj(ho, ho') and state that (ho', hr) is also a hypernym relation extracted by the pattern. Let us consider the following sentence: "I like musical instruments such as piano and guitar". Table 2.5 shows the extracted enhanced dependency relations. A subset of these dependencies matches with the dependency pattern corresponding to "NP such as NP" where "piano" and "instrument" are the headwords of the hyponym and hypernym noun phrases respectively ($NP_{ho}Head$ & $NP_{hr}Head$). Then, a hypernym relation between the hyponym noun phrase "piano" and the hypernym noun phrase "musical instrument" is extracted. And by using the conjunction relation "conj:and(piano, guitar)", hypernym relation between "guitar" and "musical instrument" is also extracted.

nsubj(like, I), root(ROOT, like)
amod(instruments, musical), dobj(like, instruments)
case(piano, such), mwe(such, as)
nmod:such_as(instruments, piano), cc(piano, and)
conj:and(piano, guitar)

Table 2.5: The enhanced dependency relations

For better understanding, figure 2.6 shows an example of matching a DHP that corresponds to "NP such as NP" with the dependency relations of the sentence "I like musical instruments

invented in Spain, such as guitar and violin" and suggesting two hypernym relations (guitar, musical instruments) and (violin, musical instruments). The pattern first matches the dependency relations of the sentences and indicates the hypernym relation between the headword of the noun phrases "guitar" and "instruments". But since "instruments" is the headword of the noun phrase "musical instruments", we then conclude hypernym relation between "guitar" and "musical instruments". In addition, we indicate the hypernym relation (violin, musical instruments) due to the existence of dependency relation "conj:and(guitar, violin)".

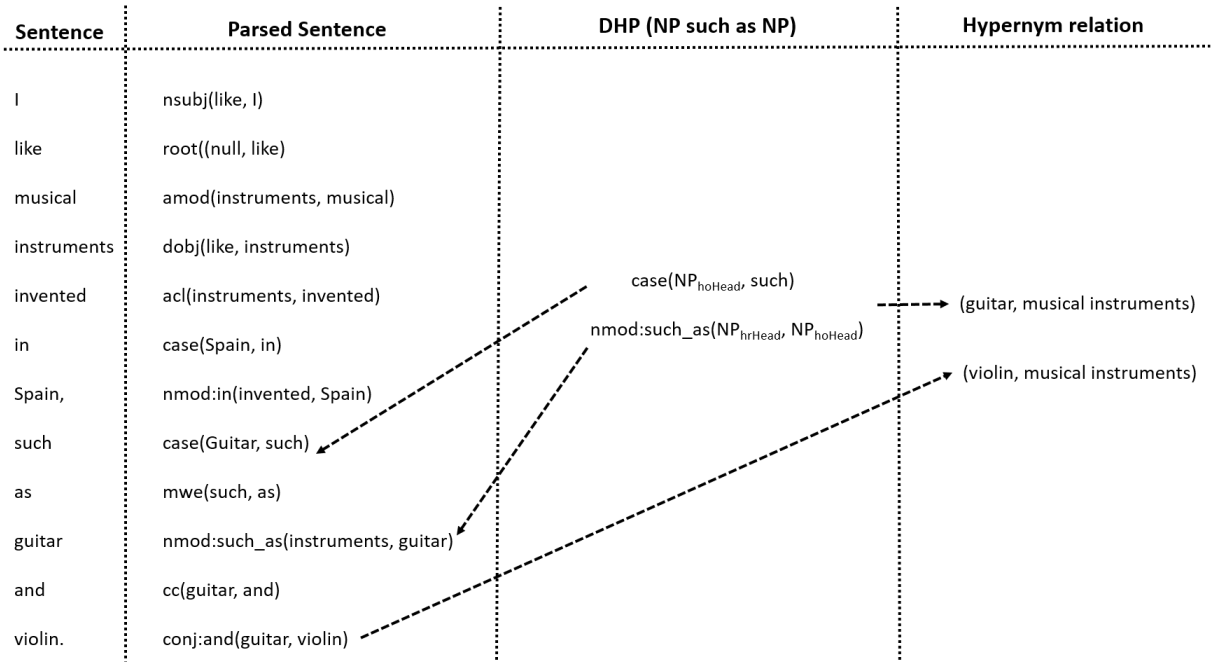


Figure 2.6: An example of matching a DHP with a parsed sentence

2.2.4 Hypernymy Extraction Tool using DHPs

We have implemented a tool to extract hypernym relations from a corpus using DHPs. It is a simple user interface (UI) tool implemented using Python [5]. The tool consists of two dependent models: the first model takes as input a corpus to perform the pre-processing step (described in section 2.2.3) and output the processed corpus file in the same directory of the corpus, the second model takes as input the result of the first model (pre-processed corpus file) and then match the formulated dependency patterns to extract hypernym relations between noun phrases.

5. https://github.com/AhmadIssaAlaa/Dependency-Hearsts-Patterns

The final result of the second model is a list of unique hypernym relations extracted from the corpus and a frequency measure for each relation refers to the number of times the relation is extracted by the patterns. Figure 2.7 shows the interface of the tool. Figure 2.8 shows some hypernym relations with their frequency extracted by the tool from Music corpus where each line represent a hypernym relation in the form "hyponym, hypernym, frequency".

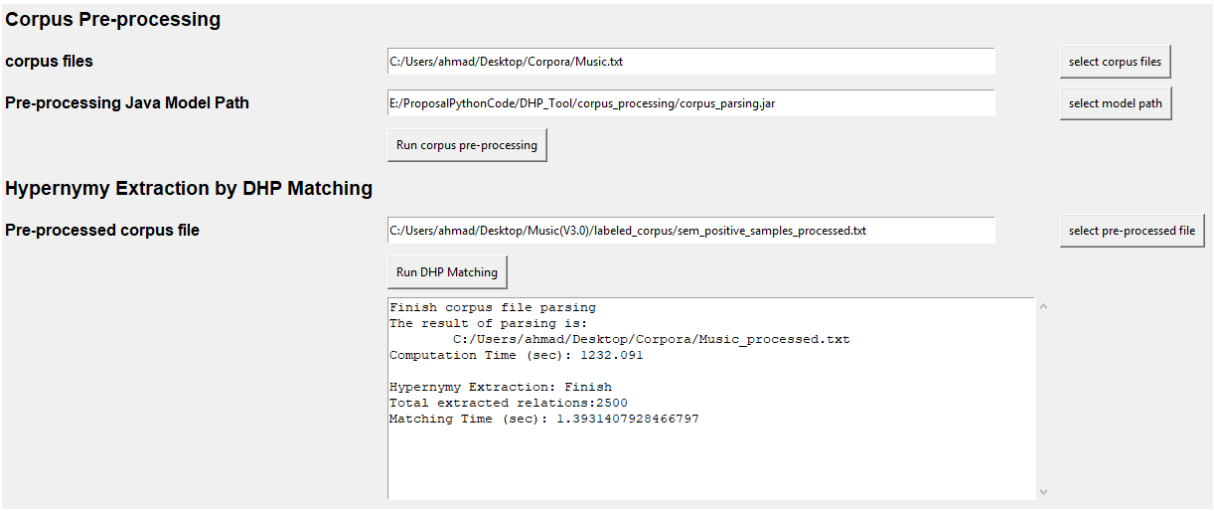


Figure 2.7: UI tool for hypernym relation extraction using DHPs

2.3 Experimental Setup

In this section, we describe our experiment to evaluate the formulated DHPs. We first present the corpora and datasets used in the experiment. We then describe the process of labeling corpus sentences. Finally, we introduce our proposed protocol to evaluate DHPs and other patterns.

2.3.1 Corpora and Datasets

We use three different corpora and two datasets: a domain-specific corpus (Music) and a general-purpose corpus (English-1) that were made available from the organizers of hypernymy discovery task at SemEval2018 [18]. The third corpus is also a general-purpose corpus (English-2) that was used in [90]. It is a concatenation of the following two corpora: a corpus constructed by crawling the .uk domain (*ukWaC*), and a 2009 dump of the English Wikipedia (*WaCkypedia EN*). The two datasets are: Music, a dataset that consists of specific hypernym relations for

```
jazz, genres, 38
voice, instrument, 35
rock, genres, 32
australia, countries, 23
pop, genres, 21
music, rock, 20
blues, genres, 18
united states, countries, 16
france, countries, 16
germany, countries, 16
jazz, styles, 15
rock, styles, 13
spain, countries, 13
hip-hop, genres, 13
folk, genres, 13
metal, genres, 13
guitar, instrument, 13
canada, countries, 13
switzerland, countries, 11
album, work, 11
blues, styles, 10
belgium, countries, 10
punk, genres, 10
japan, countries, 10
korn, band, 10
rock, music, 9
funk, genres, 9
```

Figure 2.8: Extracted hypernym relations by the tool

the Music domain (used for the Music corpus), and English, a dataset that consists of general hypernym relations for the English language (used for both English-1 and English-2 corpora). Both datasets are also provided in the task of hypernym discovery at SemEval2018.

Before launching experiments and establishing an evaluation protocol, we analyze carefully each corpus and its dataset (DS). Indeed, to correctly evaluate precision and recall, each corpus and its dataset need to be compatible in the sense we explain hereinafter. Our analysis reveals that:

1. There are ho-hr couples in the dataset that do not indicate correct hypernym relation i.e:

$$\exists\, (ho, hr) \in DS, ho\ is\ not\ a\ hyponym\ of\ hr$$

2. There are ho-hr couples in the dataset that do not occur in any sentence (patterns suggests couples that occur only in the same sentence) i.e:

$$\exists\, (ho, hr) \in DS, \forall\, S \in Corpus, (ho, hr) \notin\ S$$

3. There are ho-hr couples in the dataset occurring in sentences of the corpus, but the sentences do not convey the expected meaning i.e:

$$\exists\, (ho, hr) \in DS, \exists\, S \in Corpus, (ho, hr) \in\ S$$
$$S\ does\ not\ express\ hypernym\ relation\ between\ ho\ \&\ hr$$

4. There are sentences containing ho-hr couples but the couples are not comprised in the dataset i.e:

$$\exists\, S \in Corpus, \exists\, ho - hr(ho, hr) \in S,\ ho - hr(ho, hr) \notin DS$$

Usually, hypernym extraction performance is evaluated in terms of precision and recall based on the extracted ho-hr couples and the expected ho-hr couples. However, for patterns and due to the problems listed above, we consider that this is partially inaccurate. As a consequence, we have therefore defined an adapted protocol to assess as accurately as possible the quality of patterns. The protocol employes the given dataset for labeling the sentences in the corpus, then adapted precision and recall are evaluated in terms of the number of positive/negative sentences matching/non-matching with the patterns.

2.3.2 Corpus Labeling

The underlying idea is to label each available sentence in corpus C as positive (PS) if the sentence is likely to express hypernymy relation between at least one ho-hr couple (i.e. the sentence semantically states that a couple of noun phrase occurring in it is related by hypernym relation), and as negative (NS) otherwise. More formally, positive and negative sentences are defined as:

$$PS = \{s \mid s \in C,\ \exists\, ho - hr(ho, hr),\ s\ conveys\ hypernym\ relation\ between\ ho\ \&\ hr\} \tag{2.1}$$

$$NS = C - PS \tag{2.2}$$

Positive sentences are required to estimate recall as better as possible: in other words, positive sentences enable to estimate the number of expected sentences matching patterns as we explain in section 2.3.3. Additionally, it is evident that a pattern, if correct, should not match with a sentence not conveying the expected meaning even if the sentence contains a ho-hr couple found elsewhere. For instance, let us consider this sentence “By the 7th century, the koto (a zither) and the biwa (a lute) had been introduced into Japan from China”; the sentence contains the couple (zither, lute) but the sentence does not convey the meaning of hypernym relation between the couple terms. Indeed, patterns extract what is semantically stated in single sentences.

However, there is the need to automate as much as possible the labeling of corpus sentences. Indeed, we cannot manually check each sentence for verifying the conditions above because this is a time-consuming activity; the availability of a dataset also introduces additional complexity for manual labeling because the dataset may represent a kind of ground truth, specific to a knowledge domain. We have therefore introduced the following heuristics, listed below, leading to automatically check the main condition of labeling sentences as positive:

$$S\ expresses\ the\ hypernym\ relation\ between\ ho\ and\ hr$$

Hereafter are the list of *Heuristics* to check the condition:

(i) at least one ho-hr couple (i.e. found in the given dataset) occurs in the sentence;

(ii) the hyponym and the hypernym occur in the sentence as noun phrases (NP). Generally, the

relationships expressed in a sentence are between noun phrases (NP). For instance, in the sentence “Country music began a slow rise in American main pop charts” the hypernym “pop” occurs as a "modifier" and not as a NP;

(iii) the distance between the hyponym NP and the hypernym NP in a sentence is 10 words maximum (this is similar to what it is suggested in the reviewed approaches for deciding if a path should be a feature). For instance, the sentence “Art rock aspires to elevate rock from teen entertainment to an artistic statement, opting for a more experimental and conceptual outlook on music” is likely to do not express hypernymy between (rock, music) because the distance between "rock" and "music" is more than 10 words;

(iv) the hyponym and the hypernym NPs should not be related by “and” or “or” conjunction. For instance, the sentence “The contradictions may stem from different definitions of the terms ragtime and jazz” is likely to do not express hypernymy between (ragtime, jazz) or (jazz, ragtime) because those terms are related by "and";

(v) the hyponym and the hypernym NPs do not occur in distinct brackets in the sentence. For instance, in the sentence “By the 7th century, the koto (a zither) and the biwa (a short-necked lute) had been introduced into Japan from China”, the terms belonging to the ho-hr couple (zither, short-necked lute) occur in distinct brackets, so that the sentence is likely to do not express hypernymy between those terms.

Therefore, a sentence is labeled positive if at least one dataset ho-hr couple occurs in the sentence being this couple constrained by the heuristic provided above. On the contrary, if none of the dataset ho-hr couples occurs in the sentence, the sentence is labeled negative.

The corpus labeling is achieved using the provided dataset for it. But as mentioned in section 2.3.1, the dataset contains some wrong ho-hr couples. Thus, a preliminary step (Dataset refining) was necessary to refine the dataset from the wrong couples before corpus labeling. Additionally, there are many sentences containing ho-hr couples but the couples are not included in the provided dataset. Thus, we propose another step (Dataset enriching) to enrich the dataset with new couples. The refining step should prevent the wrong labeling of positive sentences. The enriching step should mitigate the wrong labeling of negative sentences.

— **Dataset refining:** manual remove of wrong ho-hr couples from the dataset.

$$\forall (ho, hr) \in DS \; ho \; is \; not \; a \; hyponym \; of \; hr \implies DS - (ho, hr)$$

— **Dataset enriching:** addition of new ho-hr couples automatically extracted by lexico-syntactic Hearst’s patterns (HPs). When one pattern matches one sentence, and at least

one of the extracted ho-hr couples exists in the dataset, we add the other extracted ho-hr couples by the same pattern on the same sentence to the dataset.

$$\begin{aligned} &\forall S \in corpus \;\&\; \exists\, HP \in Hearst's\ patterns\ where \\ &HP\ match\ S\ and\ extracts\ [(ho_1, hr_1), ..., (ho_n, hr_n)] \\ if\ \exists\, i\ (ho_i, hr_i) \in DS \implies &\forall j\ (ho_j, hr_j), DS + (ho_j, hr_j) \end{aligned}$$

2.3.3 Evaluation Protocol

In this section, we target the adapted evaluation protocol for better estimation of patterns performance. Given a labeled corpus C, any set of patterns P, and any dataset of ho-hr couples, sentences in C after being examined by patterns P are divided into 3 partitions: True Matched (**TM**), False Matched (**FM**), and False Not Matched (**FNM**).

$$\begin{aligned} TM \approx \{s \mid s \in PS,\, \exists\, p \in P,\, p\ matches\ s\ \&\ extracts\ (x,y) \in ho-hr\ couples, \\ or\ s \in NS,\, \exists\, p \in P,\, p\ matches\ s\ \&\ extracts\ (x,y), \\ (x,y) \in ho-hr\ couple\ of\ WordNet\ or\ s\ conveys\ hypernym\ relation\ between\ x\ \&\ y\} \end{aligned} \tag{2.3}$$

User intervention is required here to validate if an extracted couple (x, y) from a NS by a pattern p conveys the meaning of hypernymy between x & y.

$$FM \approx \{s \mid s \in C, s \notin TM \;\&\; \exists\, p \in P,\, p\ matches\ s\} \tag{2.4}$$

$$FNM \approx \{s \mid s \in PS,\, \forall\, p \in P,\, p\ does\ not\ match\ s\} \tag{2.5}$$

Consequently, Matching Precision (M-Precision) and Matching Recall (M-Recall), the adapted precision and recall for accurately assess the quality of patterns, are defined based on the above

partitions as follow:

$$M{-}Precision = \frac{|TM|}{|TM| + |FM|} \tag{2.6}$$

$$M{-}Recall = \frac{|TM|}{|TM| + |FNM|} \tag{2.7}$$

2.4 Results and Analysis

In this section, we provide the results of applying Dependency Hearst's Patterns (DHPs), lexico-syntactic Hearst's Patterns (HPs), and the extension of lexico-syntactic Hearst's Patterns (extHPs) [6] on the three corpora (Music, English-1, and English-2) after corpus labeling. Table 3.1 shows for each corpus the number of sentences before labeling and the number of positive and negative sentences after labeling. After that, an analysis of the results is also provided in this section.

Corpora	Before labeling	After labeling	
		Positive	Negative
Music	∼3 Million	6754	6754
English-1	∼55 Million	4848	4848
English-2	∼100 Million	11380	11380

Table 2.6: The number of sentences before and after labeling

2.4.1 Results

Table 2.7, 2.8, and 2.9 show the performance for each lexico-syntactic Hearst's pattern (HP) and its corresponding dependency Hearst's pattern (DHP) when applied on Music, English-1, and English-2 corpora respectively. We can notice the increase of sentences that are true matched (TM) for the most of dependency Hearst's patterns that will lead to increase their M-recall. The results also show that there is no big difference in term of M-precision between HPs and DHPs.

6. The extHP contains all 59 patterns mentioned in the work of Seitner et al. [87]. We provide them in the appendix A

Pattern	HP			DHP		
	TM	FM	Pre (%)	TM	FM	Pre (%)
NP such as NP	480	**118**	**80.3**	**559**	196	74.0
NP including NP	**153**	**120**	**56.0**	137	130	51.3
NP and/or other NP	117	**39**	75.0	**141**	46	**75.4**
NP is a NP	224	**620**	26.5	**486**	1290	**27.4**
NP especially NP	7	**3**	**70.0**	**9**	4	69.2
such NP as NP	32	**9**	**78.0**	**40**	15	72.7

Table 2.7: Pattern by Pattern Comparison on Music corpus.

Pattern	HP			DHP		
	TM	FM	Pre (%)	TM	FM	Pre (%)
NP such as NP	76	**46**	**62.3**	**108**	68	61.4
NP including NP	8	**50**	13.8	**27**	53	**33.8**
NP and/or other NP	180	69	72.3	**220**	**63**	**77.7**
NP is a NP	132	**262**	33.5	**366**	551	**39.9**
NP especially NP	6	0	100	**7**	0	100
such NP as NP	2	**4**	**33.3**	**9**	5	**64.3**

Table 2.8: Pattern by Pattern Comparison on English-1 corpus

Pattern	HP			DHP		
	TM	FM	Pre (%)	TM	FM	Pre (%)
NP such as NP	66	**52.8**	**62.3**	**164**	133	55.2
NP including NP	17	**45**	**27.4**	**28**	138	16.9
NP and/or other NP	88	**79**	**52.7**	**110**	124	47.0
NP is a NP	341	**565**	**37.6**	**998**	1687	37.2
NP especially NP	**6**	**3**	**66.7**	4	11	26.7
such NP as NP	1	**1**	**50.0**	**6**	16	**27.3**

Table 2.9: Pattern by Pattern Comparison on English-2 corpus

Tables 2.10, 2.11, and 2.12 show the M-precision, M-recall, and f-score of HPs, extHPs, and DHPs when applied on Music, English-1, and English-2 corpora respectively. The results in the three corpora confirm that formulating Hearst's patterns as dependency patterns leads to a considerable improvement in terms of M-recall. Even more, it shows better M-recall than extHPs in two corpora (Music and English-2), a set of 59 lexico-syntactic patterns, which confirm that the formulated dependency patterns are generic patterns. Concerning their impact on M-precision, the results in the three corpora show that DHPs leads to a slight reduction.

Patterns Type	M-Precision	M-Recall	F-score
HPs	0.556	0.157	0.245
extHPs	**0.576**	0.192	**0.288**
DHPs	0.464	**0.207**	0.286

Table 2.10: All patterns comparison on Music corpus

Patterns Type	M-Precision	M-Recall	F-score
HPs	0.451	0.073	0.125
extHPs	**0.474**	**0.163**	**0.242**
DHPs	0.428	0.149	0.221

Table 2.11: All patterns comparison on English-1 corpus

Patterns Type	M-Precision	M-Recall	F-score
HPs	**0.41**	0.047	0.084
extHPs	**0.41**	0.102	0.163
DHPs	0.371	**0.123**	**0.185**

Table 2.12: All patterns comparison on English-2 corpus

DHPs match sentences based on dependency relations. Thus, to use such type of patterns, a considerable cost will be paid in term of computation time. Table 2.13 shows the computation time (in seconds) it takes each type of patterns when applied on the three corpora. We can notice from the table results the high cost in computation time we have paid when using DHPs.

2.4.2 Qualitative Analysis

In order to understand the obtained results especially the considerable increase in M-recall when applying DHPs, we select and analyze some positive sentences that are true matched

Patterns Type		HPs	extHPs	DHPs
Computation Time (sec)	Music	230	234	2454
	English-1	164	166	1055
	English-2	358	362	4254

Table 2.13: All patterns computation time on both Music and English corpora.

(TM) with DHPs, while not matching (FNM) or false matched (FM) with HPs. In contrast to HPs, DHPs are capable of matching and identifying correct hypernym relations where non-pattern words occur between hyponym and hypernym noun phrases because DHPs are based on dependency relations and they match sentences without words order restriction. For instance, in these two sentences "A march, as a musical genre, is a piece of music with a strong regular rhythm" and "Piano (pianoforte) is a musical instrument"; the existence of "as a musical genre" and "(pianoforte)" in the two sentences respectively prevents HPs and extHPs to match the sentences, while DHPs correctly match them. Such sentences are frequent and especially to the dependency pattern corresponding to "NP is a NP" which explain its high M-recall. The number of sentences true matched (TM) with DHP corresponding to "NP is a NP" is 466, while it is 214 for HP (see table 2.7). In addition, DHPs perform better than HPs dealing with some ambiguous sentences. For instance, in this sentence "I like musical instruments invented in Spain, such as guitar"; HPs match the sentence and identify wrong hypernym relation between "Spain" and "guitar", while DHPs identify the correct hypernym relation between "instruments" and "guitar" thankful to the dependency relation "nmod:such_as(instruments, guitar)" obtained from the dependency parsing. Furthermore, DHPs are generic patterns that match sentences that can't be matched by HPs. For instance, the formulated dependency pattern corresponding "NP and/or other NP" also matches sentences of the forms "NP and/or many other NP" and "NP and/or any other NP". Below two example of sentences that are matched by the formulated DHP corresponding "NP and/or other NP" and not matched by the HP:

1. Country or any other musical style that drifts outside the commercial margins would be well advised to check out anything by Jimmie Dale Gilmore.
2. Major opera companies have begun presenting their performances in local cinemas throughout the United States and many other countries

2.4.3 Error Analysis

Although, dependency parsing gives a better understanding of the meaning of sentences, it also prone to make errors when applied on complex sentences. These parsing errors may

lead DHPs to either match positive sentences and identify wrong hypernym relations or never match the sentence. Additionally, dependency patterns as defined in this work are more generic patterns and they are prone to match sentences and suggest wrong hypernym relations. For example, the defined dependency pattern corresponding to "NP is a NP" is a generic pattern that includes patterns such "NP is in NP" and "NP is with NP", since the occurrence of "in" and "with" do not affect the dependency path between the two noun phrases (e.g. "the lounge revival was in full swing"). This explains the high number of FM by the dependency pattern corresponding to "NP is a NP".

Moreover, while manually validating the extracted hypernym relations for both DHPs and HPs, we notice many matching errors that are common between both types of patterns. The followings are some of these errors:

— errors in identifying named-entities using noun phrase chunker; e.g. "Alice In Chains were the band that made me discover music". We may mitigate such errors by using Named-Entity recognition tool.

— errors in distinguishing between hyponym and hypernym noun phrases, so many inverted hypernym relations are extracted; e.g. "This instrument was a guitar". Such errors are frequently noticed in the pattern "NP is a NP" and they are difficult to be managed by hypernym patterns.

— errors in matching ambiguous sentences that are very difficult to be managed by both types of patterns (lexico-syntactic and dependency patterns); e.g "I like musicians playing keyboard instruments such as Beethoven".

2.5 Summary

Lexico-syntactic patterns have been extensively used to extract hypernym relations from texts. They are defined as regular expressions based on lexical and syntactic information. In this work, we propose to manually reformulate the most popular lexico-syntactic patterns (Hearst's patterns) as dependency patterns (DHPs) for the purpose of studying the impact of using dependency information instead of lexical and syntactic information to formulate hypernym patterns.

Dependency patterns are formulated as a set of dependency relations that are obtained by performing dependency parser on sentences. Each dependency relation is a binary grammatical relation that hold between two words. A main advantage of dependency relations is their ability to provide closer meaning to the semantic of the sentence. To formulate DHPs based on

dependency relations, we have propose a process of 4 steps that are repeated for each Hearst's pattern. We first select a set of random sentences that are matched one Hearst's pattern. Second, we perform dependency parser on the set of sentence to obtain their dependency relations as a dependency tree. After that, we analyze the obtained dependency trees. Finally, we select the subset of dependency relations from the dependency trees that are useful to replace the corresponding Hearst's pattern.

The formulated dependency patterns are used to match sentences based on their dependency trees to suggest hypernym relations between noun phrases. For this purpose, before matching DHPs to sentences, a pre-processing step is necessary to obtain their dependency trees and to indicate the noun phrases in the sentences. In addition and because dependency relations hold between words and not phrases, we identify and use the headword of the noun phrase to link dependency relations between noun phrases. Furthermore, we have used python to implement a UI tool to pre-process a corpus and extract hypernym relations from the corpus using DHPs.

We have evaluated and compared DHPs to lexico-syntactic Hearst's patterns (HPs) and an extended set of lexico-syntactic patterns (extHPs, 59 patterns) using three corpora. The results in the three corpora show that DHPs lead to a considerable improvement in terms of M-recall with a slight reduction in terms of M-precision. In addition, we compare between both formulation in terms of computation time, the results show that DHPs take a much longer computation than lexico-syntactic patterns due to the usage of dependency parser.

A qualitative analysis has been performed to understand the considerable increase in M-recall. We have analyzed a set of sentences that were truly matched by DHPs and either not matched or falsely matched by HPs. Hereinafter, we present our conclusion:

— In contrast to HPs, DHPs match and identify correct hypernym relations in sentences where non-pattern words occur between hyponym and hypernym noun phrases.

— DHPs truly match some ambiguous sentences that are falsely matched by HPs.

— DHPs are more generic than HPs. For instance, the formulated DHP corresponding to "NP and/or other NP" also match sentences of the forms "NP and/or any other NP" and "NP and/or many other NP".

To understand the slight reduction in M-precision, we have analyzed some sentences that are falsely matched by DHPs and not falsely matched by HPs. We have noticed that the formulated DHPs are generic patterns that may include some specific patterns that are not good to extract hypernym relations. For example, the formulated DHP corresponding to "NP is a NP" includes the non-hypernym patterns "NP is in NP" and "NP is with NP".

As a conclusion, we have reformulated Hearst's patterns as dependency patterns based on dependency relation obtained by applying dependency parser. Dependency patterns are more generic patterns than lexico-syntactic patterns that lead to highly improve M-recall. A slight reduction in M-precision is observed as a drawback of their generality.

CHAPTER 3

SEQUENTIAL HYPERNYM PATTERNS

This chapter describes our work to improve dependency patterns to extract hypernym relations using sequential patterns mining and dependency parsing. In the first section, we provide an introduction including the motivation of the work. In the second section, we present the proposed approach to improve dependency patterns (DHPs). We first introduce the proposed approach of 3-phases to improve dependency patterns precision and recall by: replacing dependency Hearst's patterns by sequential Hearst's patterns (SHPs), discovering new sequential Hypernym patterns (SHyPs), and extending SHPs and SHyPs by anti-hypernym sequential patterns ($SHPs^{-}$ & $SHyPs^{-}$). We then describe the workflow of steps to learn sequential hypernym patterns. After that, we describe our proposed representation of sentences as sequences. We finally explain how sequential patterns could be used to extract hypernym relations between noun phrases. In the third section, we describe the performed experiment to learn and evaluate sequential hypernym patterns. We first present the corpora and datasets used. We then describe the setting up of the approach configuration. And finally, we detail the process to learn SHPs, SHyPs, $SHPs^{-}$, and $SHyPs^{-}$. In the fourth section, we provide the evaluation step results and the analyses of the results. A summary of the chapter is provided in the fifth and last section. A part of this work was first published in a France national workshop [3]. The full work is submitted but not yet accepted in "The Knowledge Engineering Review" Journal.

3.1 Introduction

In the previous chapter, we try to understand the benefits of using grammatical dependencies for specifying patterns. For that purpose, we reformulate manually the original Hearst's patterns as dependency patterns (DHPs) using the Stanford dependency parser dependencies [86]. DHPs show a better (but low) recall than HPs and ExtHPs. Indeed, DHPs generalize HPs and some of ExtHPs. However, they show slightly worse precision. This means that once again despite the richer pattern representation, manually specifying patterns likely leads to decrease precision. As a consequence, extending DHPs as it has been done for HPs, increases recall but likely

decreases precision.

Based on the literature review, just a few approaches have been proposed to improve the precision of hypernym patterns. In [79], *Ponzetto et al* propose to use non-hypernym patterns (i.e. meronym patterns) to identify and filter wrong hypernym relations extracted by the hypernym patterns. Other approaches [25, 81] have been proposed to improve precision by using statistical measures to estimate confidence scores for the extracted hypernym relations by the patterns. Then, hypernym relations that have a score below a given threshold are filtered. However, we consider that the proposed approach using only statistical information is less controllable and constraining (because a couple to be recognized as ho-hr couple needs to frequently occurs in sentences frequently matching with the given set of patterns) that an approach based on anti-patterns, which underlying idea is quite simple and natural. For this reason, we suggest using anti-patterns as the main mechanism for filtering outcomes, without preventing the usage of other mechanisms (anti-patterns are better understandable in Sections 3.2.3 and 3.3.5).

Previous works to improve recall were mostly based on extending existing patterns with new patterns. Patterns are either designed manually [44] or extracted automatically [92, 88]. Most of the existing patterns show quite high precision but recall remains very low because of the large variability in natural language for expressing a given meaning. On the one hand, the manual extension of existing patterns for improving recall while keeping a good precision is quite hard because of the large variability in natural language for expressing a given meaning. On the other hand, to the best of our knowledge, all approaches trying to automatically extract hypernym patterns for getting both good precision and recall, propose to use dependency paths [92, 88] as features to train classifiers. However, using dependency paths as features can lead to a huge and sparse feature space, which decreases precision without considerable gaining in the recall. Additionally, even if patterns (especially patterns, which exclusively comprise generic words and placeholders) are often considered valid across domains, to keep recall high, new patterns need to be discovered whenever new texts become available, and specific patterns can be discovered if those texts belong to specialized domains. Moreover, patterns are necessarily related to a natural language so each language needs specific patterns. So that, relatively simple and understandable mechanisms for continuous discovering of new patterns are needed.

In this work, we propose a new 3-phase (supervised) approach for automatically extracting patterns, from any corpus and any dataset of known ho-hr couples. The main objective of the approach is to systematically increase the recall and precision of a set of seed patterns. The approach is based on sequential pattern mining (using CloSPEC [12]), coupled with both a rich syntactical sentence representation based on grammatical dependencies and pattern selection

strategy for automatically removing the huge amount of raw patterns extracted by the mining algorithm.

The three phases are distinguished in term of the input patterns and target precision and recall to be achieved by the extracted patterns, as listed below:

(i) Phase 1 mines sequential patterns for improving precision or recall of seed patterns while keeping the same level of the non-improved metric;

(i) Phase 2 improves recall by discovering new sequential patterns other than sequential patterns discovered in the first phase, pattern inputs of the second phase, enabling to extract ho-hr couples that are not extracted by inputs patterns—the precision of each extracted pattern is evaluated and should not degrade the precision of input patterns;

(iii) Phase 3 associates, for each sequential pattern extracted in first and second phases, anti-hypernym patterns that should remove wrong ho-hr couples extracted by it.

.

The 3-phases are required because precision and recall move in opposite directions: as a consequence, each phase improves precision (or recall) till recall (or precision) is not degraded, and each phase extracts patterns for matching with distinct sentences in the corpus. The third phase, targeting anti-hypernym patterns, is needed because we noted that if only first and second phases are performed, precision improvement remains limited, despite any modification of parameters and partitions of training and testing sentences. By the way, the usage of anti-hypernym patterns is not new as reported in the state of the art section.

We validate the approach by performing experiments using three distinct corpora (one domain-specific corpus on music, and two generic English corpora) and a specific set of seed patterns. Seed patterns used in the first phase are Dependency Hearst's Patterns (noted DHPs), manually specified in the previous chapter, and showing a similar precision of Hearst's patterns and a much better recall. The first phase results in Sequential Hearst's Patterns (SHPs) as we named them. The second phase then takes as input SHPs and results in new additional patterns (noted SHyPs). Finally, the third phase takes as input SHPs and SHyPs and generates anti-hypernym patterns, respectively noted as SHPs$^-$ and SHyPs$^-$. The first experiment compares the results of extracted patterns to recent mostly complete sets of manually designed patterns taken out from [40, 87] in addition to DHPs. The contribution of anti-hypernym patterns is also evaluated. Further experiments are then performed for showing the replicability of the extracted patterns i.e. to what extent one pattern extracted by using a corpus can be reused to extract ho-hr couples from another corpus. This is a strong advantage of patterns over supervised distributional approaches for which it is difficult (or even impossible) to assess the usability of a model

learned on one corpus to another corpus, thus possibly requiring to learn a new model: explicability and understandability of patterns can then be used for assessing their general applicability.

3.2 The 3-phase Approach

In this section, we describe in detail the 3-phase (supervised) approach that aims to systematically increase the precision and recall of existing seed patterns, using any corpus and any dataset of known ho-hr couples. Specifically, we present each phase in detail, being each phase performed by following the same workflow steps starting from phase-specific inputs, objectives to be achieved, and parameters. The workflow steps are common to all phases, so that they are presented in a dedicated section. The new pattern representation based on lexical and grammatical dependencies is also presented and compared to pattern representations found in the relevant literature. It should be noted that the approach is generic and can be used for any set of seed patterns, independently of their representation, any corpus, and any dataset of expected ho-hr couples. It should also be noted that the workflow is generic too, so that it is suitable to develop an adaptable and modular software for putting in practice the suggested design and configurations. It should be noted that the used concepts and notations in the following have been defined before in section 2.3.3.

3.2.1 Phase 1: Mining Sequential Patterns Associated to Seed Patterns

Phase 1 objective is to extract sequential patterns (noted SP_{p1}) corresponding to a given set of seed patterns such that the former improves either M-precision or M-recall of the latter without degrading the non-improved one. This is done, first, by applying the mining process for extracting sequential patterns from a set of training sentences belonging to **TM** with respect to the seed patterns, and second, selecting the sequential patterns showing M-precision and M-recall evaluated on a validation set of sentences better than M-precision and M-recall of seed patterns.

3.2.2 Phase 2: Discovering New Sequential Patterns

Phase 2 objective is to extend SP_{p1} by discovering new sequential patterns (noted SP_{p2}) for the purpose of increasing M-recall under the constraint of keeping M-precision stable (i.e. close to the M-precision of SP_{p1}). New patterns are discovered by performing the mining process to extract sequential patterns from sentences belonging to **FNM** with respect to SP_{p1} (i.e.

sentences express at least one hypernym relation and not matching to SP_{p1}). Selected patterns are those showing M-precision evaluated on a validation set of sentences, at least equal to M-precision of SP_{p1}.

3.2.3 Phase 3: Anti-hypernym Sequential Patterns

Phase 3 objective is to extract patterns, enabling to identify couples wrongly extracted as hypernymy by using SP_{p1} or SP_{p2}. These patterns are often qualified as anti-patterns, thus globally referred to as anti-hypernym sequential patterns. Using both patterns and anti-patterns in combination contributes to increase M-precision while keeping stable M-recall. Anti-patterns are extracted by applying the mining process for extracting sequential patterns from sentences belonging to **FM** with respect to SP_{p1} and SP_{p2}. Selected sequential patterns are those showing very low M-precision on a validation set of sentences i.e. patterns likely extracting non-hypernym couples.

3.2.4 Sequential Hypernym Pattern Learning Workflow

The steps required to learn sequential patterns are common to the 3 phases. This is an important aspect of the proposed approach. Indeed, some steps are executed only once and other generic steps only need to be configured in terms of inputs and constraint parameter values, according to the objective of each specific phase. Figure 3.1 shows the generic workflow of steps. Hereinafter, we describe in detail the 5 steps, being some of them performed only once (above the dotted line in Figure 3.1), and other ones repeated for each phase (below the dotted line in Figure 3.1): (i) Corpus Labeling, (ii) Sequences Preparation, (iii) SPM, (iv) Relevant Sequential Pattern Selection and (v) Evaluation.

Corpus Labeling Step and Evaluation Step

Corpus labeling objective is to label each available sentence in corpus C as positive (PS) if the sentence is likely to express hypernymy relation between at least one ho-hr couple, and as negative (NS) otherwise. Labeling is done once, in the beginning, and makes it possible to optimize how TM, FN, and FNM are built when needed. The process of corpus labeling is the same process that has been described in the previous chapter (see section 2.3.2).

Once labeling is performed, the set of positive sentences belonging to the labeled corpus is randomly partitioned and 60% of sentences are put in a learning corpus while 40% of sentences

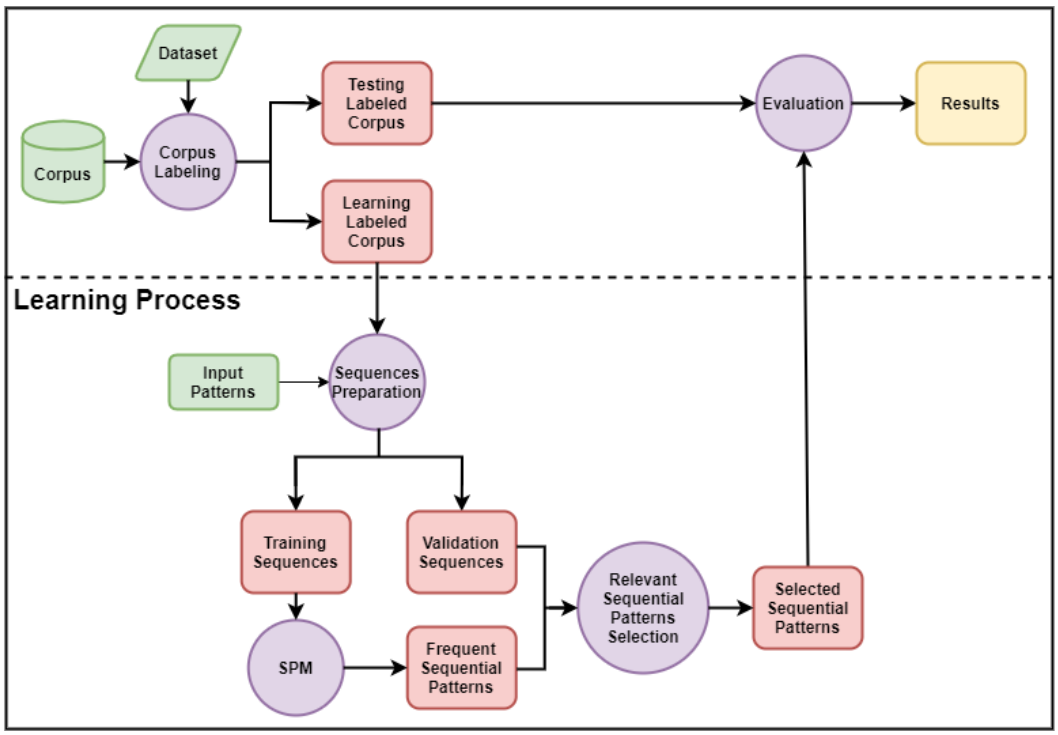


Figure 3.1: The workflow of sequential pattern learning.

are put in a testing corpus [1]. Learning corpus is then completed by adding a number of negative sentences (i.e. a sentence is labeled as negative if it is not labeled as positive) equal to the number of positive sentences already contained in it. Testing corpus is also completed as done for the learning corpus.

Evaluation step is also performed once for a given corpus, whenever all 3 phases have been completed. The objective is to evaluate the final achieved M-precision, M-recall, and f1-score (i.e. the harmonic mean of M-precision and M-recall) by all patterns, possibly in combination with anti-patterns, resulting from all phases. These values are then compared to the metrics values calculated for the seed patterns, original HPs, and ExtHPs—ExtHPs the most extensive set of patterns currently available. All metrics are evaluated by using the testing corpus, built after corpus labeling.

Sequences Preparation Step

The main goal of this step is to prepare the relevant input data (called the sequence database) for the SPM algorithm and the relevant data for validation, according to the phase objective.

1. The partitioning of the labeled corpus into 60% training and 40% testing is an arbitrary choice.

First, sentences belonging to the learning corpus are matched by input patterns and classified, as needed by the phase objective, according to the TM, FM, and FNM definitions.

Second, and depending on the phase objective, classified sentences are partitioned in a set of sentences for mining patterns and a set of sentences for validating extracted patterns in the phase. For instance, in phase 3, all FM sentences by SP_{p1} and SP_{p2} (patterns resulting from phases 1 & 2) are used to extract anti-patterns, and FM and TM sentences are used to validate the extracted patterns.

Finally, all sentences in each partition are represented as sequences, thus respectively leading to a set of Training Sequences (TS) and a set of Validation Sequences (VS) both ones proper to the specific phase (in the remainder, when needed, we can use TS_i, VS_i where index i refers to the phase). The purpose of VS is to automatically perform the pattern selection (as explained in Section 3.2.4 below). The representation of sentences as sequences is described in Section 3.2.5. Being training and validation sets proper to each phase, TS and VS should not be confused with the learning and testing corpora resulting from the corpus labeling step, required both to learn patterns and to evaluate global results once all 3 phases have been completed.

SPM Step

This step aims to extract frequent sequential patterns from the set of training sequences (TS) by applying one SPM algorithm. For experiences (see Section 3.3) targeting the validation of the 3-phase approach, we use CloSPEC [12], an algorithm to extract frequent closed sequential patterns [2] under multiple constraints. The possibility of specifying constraints is important to drive the mining algorithm to extract interesting sequential patterns and to keep under control the complexity, as better explained below. These constraints are considered as parameters, which values must be tuned for the purpose of each phase. These constraints are:

— **Support constraint:** refers to the minimum occurrence frequency to consider a sequence as frequent. In general, high support value is preferred to extract only highly frequent patterns. Unfortunately, interesting hypernym patterns maybe not much frequent in a corpus, and using low support value is necessary. Using a low support value negatively impacts on the mining complexity and a high number of frequent sequential patterns are extracted. For this reason, the other two provided constraints in CloSPEC are very beneficial to reduce the number of extracted frequent sequential patterns and keeping the mining complexity under control.

2. A sequence pattern is closed if no more general sequence pattern exists with the same support.

— **Gap constraint:** refers to the minimum and maximum number of sequence elements that can be ignored between two other sequence elements. The minimum constraint is fixed to zero because we noted that using a minimum gap greater than zero avoid extracting most of the hypernym patterns. Adopting a strictly positive maximum gap is interesting because it enables to ignore words (and dependencies) that are irrelevant for recognizing a hypernym relation. For instance, the word "like jazz" occurring in the following sentence between brackets are irrelevant to be part of the hypernym pattern corresponding to "NP is a NP": "funk (like jazz) is a music that requires the musicians playing it ...". Limiting as much as possible the value of the maximum gap is also beneficial to avoid extracting sequential patterns where their elements occur too far from each other in a sentence and likely to do not express any hypernymy relation.

— **Element number constraint:** refers to the minimum and maximum number of elements (itemsets) composing the sequential patterns. In general, hypernym patterns should comprise at least 2 elements (the hyponym and the hypernym); a minimum number of itemsets fixed to 2 is necessary to extract interesting patterns such as "NP, NP" (corresponding to an appositive phrase). Limiting as much as possible the maximum number of itemsets is beneficial to reduce the number of extracted irrelevant patterns while keeping under control the complexity of mining algorithm. This seems to be feasible because, for instance, we carefully analyzed the ExtHPs (the 59 lexico-syntactic patterns used in [87]) and found that the maximum length of patterns is 6. Similar value can be used as maximum number of itemsets.

Sequential Patterns Selection step

Although the usage of constraints makes it possible to limit the number of extracted patterns, plenty of them remain raw patterns and are not relevant. Earlier works [71, 11] address the selection of relevant patterns from raw patterns by asking experts to perform a manual selection which is a hard and time consuming task. We propose to automatically select relevant sequential patterns by using additional constraints. These constraints depend on validation sequences stated in step "sequences preparation" and the objectives to be achieved by the phase. Patterns selection is performed by following the substeps listed below based on corresponding selection criteria:

(i) Selecting patterns (from the whole set of extracted raw patterns) comprising hypernym-hyponym paths. The existence of one dependency path between hypernym and hyponym

in the pattern increases the guarantee that pattern enables to correctly extract the targeted relation, despite the *distance* occurring between the hypernym and hyponym.

(ii) By using validation sequences, computing M-precision of each sequential pattern selected in the previous substep and the total M-recall. A pattern is kept if it shows M-precision greater (or lesser in the case of anti-patterns) than a given threshold and total M-recall greater than a threshold too. The 2 thresholds are parameters which values are fixed according to the objective of the phase.

(iii) Removing sequential patterns that are super-sequences of another pattern showing the same M-precision. In other words, if two or more patterns have the same M-precision, the one comprising fewer elements is kept (corresponding to the more general pattern).

3.2.5 Sequential Representation of Sentences

The sequence representation of a sentence is one important aspect of the proposed approach. Indeed, pattern representation may or may not enable to extract relevant relations. For instance, a lexico-syntactic pattern representation cannot allow representing the relationship between "musical instrument" and "guitar" in the sentence "I like musical instruments invented in Spain, such as guitar" because of "invented in Spain". However, this is possible by using deep grammatical dependencies because dependencies provide the right context to assign a meaning to a sentence. In the literature review, dependencies have been already used for similar purposes.

We list below the main differences and similarities between our proposed representation of sequences and the ones described in [11, 71] works:

— **Sequence coverage.** In [71], the sequence covers sentence words that are related by the shortest dependency path connecting a couple of terms. In [11], all sentence words are represented in the sequence. In our representation, we suggest covering all sentence words as in [11] because we assume that the shortest dependency paths may not comprise all information that characterizes hypernym patterns.

— **Sequence elements.** In [71], the sequence element is an item that is either representing a word itself or a dependency relation linking two words, and if the word is a noun it is replaced by “NN”. In [11], the sequence element is an itemset (a set of items) including the word itself, its lemma, and its POS tag. In our representation, the sequence element is also an itemset with more item kinds, describing particularly its dependency relation with one another itemset of the sequence (as better explained in the remainder).

— **Taking Noun Phrases into account.** Rather than referring to nouns [71, 11], we refer to Noun Phrases (NPs) for sentence representation as a sequence. Thus, the extracted sequential patterns enable to extract relations between NPs, which is more adapted because hypernym and hyponym are often NPs, not single nouns.

— **Links between sentence elements**. In the sequence representation of [71], sentence elements are linked by the grammatical relation linking them together in the dependency tree. While in [11], sentence elements are linked by their order of occurrence in the sentence. In our sequence representation, sentence elements are linked by their order of occurrence in the sentence and by the grammatical relation linking them together in the dependency tree. The dependency relation of each element and with which it is related are included in its itemset (more details are shown below).

Proposed representation. Let $S =< e_1, e_2, \cdots, e_n >$ denote the sequence of elements that represent a sentence. As also said above, an element of a sentence is either a NP or any other word that does not belong to any NP. And let $I_{e_j} =< i_{(j,1)}, i_{(j,2)}, \cdots, i_{(j,m)} >$ denote a set of items (the itemset) that represent information about the element e_j.

Thus, each element is further represented by a set of information about a NP or a word. For one NP element, most information refers to its head [3]. The headword is the most important word in the NP, conveying the main meaning of the NP. A NP is composed of several words that are mostly either nouns or modifiers, but only one of these nouns is its head (root).

Nature of information associated to each element is lexical, syntactic, and dependency. Information used are:

— **Element label:** the label of the element. If the element is a NP, its label is the concatenation of the words (separated by "_") that compose it.

— **NP:** in case of a NP element, "NP" item is added to the set of information items.

— **Lemma:** the lemma of the element (or of its head if it is a NP element).

— **POS tag:** the POS tag of the element (or of its head if it is a NP element).

— **Relation** & **direction:** the grammatical relation of the element (or of its head if it is a NP element) concatenated by the relation direction. A grammatical relation is a binary relation between two words: the governor and the dependent (the element is the dependent of the relation). The direction is right ($\rightarrow$) if the location of the governor word is after the dependent word in the sentence; otherwise, it is left ($\leftarrow$).

3. For example, for the two noun phrases "musical instrument" and "the title of book", "instrument" and "title" are their headwords respectively.

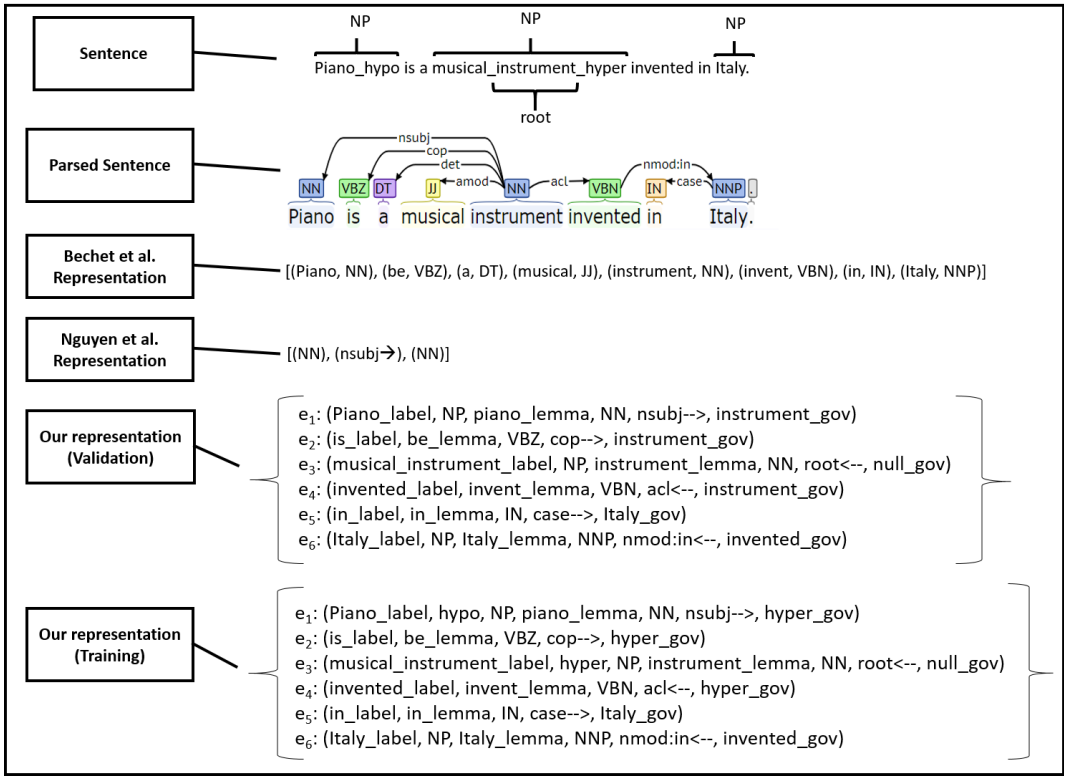


Figure 3.2: Examples of sequence representation of a sentence

— **Governor word:** the word (it can be a headword of a NP) to which the element is grammatically related.

For the purpose of learning sequential hypernym patterns, hyponym and hypernym NPs should also be indicated in the sequences. Thus, we add two items "hypo" and "hyper" to indicate them in the sequences. In addition and for the purpose of representing the dependency path connecting the hyponym to the hypernym, we replace the governor word of a sequence element by “hypo_gov" (respectively “hyper_gov”) if the governor word is the headword of the hyponym (respectively hypernym).

Let’s consider the following sentence “Piano is a musical instrument invented in Italy”. Figure 3.2 shows the representation of the sentence as a sequence according to our representation, representation proposed by Nguyen et al. [71], and representation proposed by Bechet et al. [11]. “_label”, “_lemma”, and “_gov” annotations are used in the sequence representation to discriminate items according to the required representation of **Element label**, **Lemma**, and **Governor Word**.

3.2.6 Hypernymy Extraction by Sequential Hypernym Patterns

Sequential hypernym patterns are used to extract hypernym relations from sentences after representing them as sequences (e.g. the validation sequence representation in figure 3.2). In general, a sequential pattern matches a sentence if it is the subsequence of the sequence representing the sentence. A sequence S_1 is subsequence of another sequence S_2 if for every itemset I_i in S_1 it exists an itemset I_j in S_2 where I_i subset of I_j. The subset of itemsets must satisfy the sequence order.

Based on our assumption that the distance between a couple of terms in a sentence may play a role for identifying hypernym relations, we suggest to match sequential patterns using gap constraint i.e. the minimum and maximum number of sequence elements (itemsets) that can be ignored between the couple of terms. In order to match sequential patterns using gap constraint, the subset of itemsets must satisfy the sequence order and the minimum and maximum gap constraints (min_gap and max_gap). In other words, a sequential pattern $SP = < I_1, ..., I_m >$ match a sentence sequence $SS = < I_1, ..., I_n >$ using gap constraint, if for every itemset I_k of SP it exists an itemset I_i of SS where I_k is subset of I_i and I_{k+1} is subset of I_{i+1+j} with $j = [min\_gap, max\_gap]$, $k = [1, m-1]$ and $i = [1, n]$. Note that "hypo" and "hyper" items in a sequential hypernym pattern are not used to check the matching, they are used to indicate the hyponym and hypernym noun phrases in the matched sentence.

As lexico-syntactic patterns and dependency patterns, sequential hypernym patterns are formulated to match a sentence and extract one or more hypernym relations for the same hypernym noun phrase. This is achieved by finding the co-hyponyms itemsets of the hyponym in the sequence representing the sentence. Co-hyponym itemsets are the sequence itemsets that contain a "conj:and"/"conj:or" relation item and its governor word item is the hyponym.

3.3 Experiments

This section presents the first evaluation performed to validate the proposed approach by comparing the performances (M-precision and M-recall) of extracted patterns to the ones of three sets of patterns: DHPs (proposed original seed patterns), HPs, and ExtHPs. DHPs are included for showing non-regression with respect to the original seed patterns (i.e. the approach leads to patterns which performance cannot be lower than the performance of the seed patterns), while ExtHPs and HPs are included for showing the capability of the proposed approach to extract patterns showing better performance than those of existing manually designed patterns.

Corpora	Initial Size	Learning Labeled Corpus Size		Testing Labeled Corpus Size	
		Positive	Negative	Positive	Negative
Music	∼3 Million	4052	4052	2702	2702
English-1	∼55 Million	2909	2909	1939	1939
English-2	∼100 Million	6828	6828	4552	4552

Table 3.1: Corpus labeling results.

We also show the interest and need of the 3 phases by showing the contribution of each phase to the performance improvement. Moreover, we analyze to what extent the proposed approach tends to learn generic patterns.

3.3.1 Corpora and Datasets

For conducting the experiments, we use three different corpora and two datasets. Two of three corpora are provided for the hypernymy discovery task at SemEval2018 [18]: a domain-specific corpus (Music) and a general-purpose corpus (English-1). The third corpus is also a general-purpose corpus (English-2), used by Shwartz et al. [89], which is a concatenation of the following two corpora: a corpus constructed by crawling the .uk domain (*ukWaC*), and a 2009 dump of the English Wikipedia (*WaCkypedia EN*). The two datasets comprise known ho-hr couples, and are more or less specialized: Music is a dataset consisting of specific hypernym relations for the music domain (used with Music corpus), and English is a dataset consisting of general hypernym relations for the English language (used with both English-1 and English-2 corpora). Both datasets have been provided for the task of hypernym discovery at SemEval2018.

As described above, starting from a given corpus and a dataset of known ho-hr couples, labeling is performed once for each corpus. For all used corpora, table 3.1 shows the results of the corpus labeling step: numbers of sentences, corresponding numbers of positive and negative sentences, and numbers of learning and testing sentences.

3.3.2 Setting up Workflow Configuration

Each learning labeled corpus is used to learn sequential patterns by following the proposed 3-phase approach as shown in figure 3.3. In previous chapter, DHPs have been manually designed from HPs and it has been shown that DHPs lead to better performance than HPs and ExtHPs. For this reason, we propose to use DHPs instead of HPs and ExtHPs as a set of seed patterns. It should be noted that each manually designed DHP has one and only one corresponding HP: in the remainder, we use the HP simple syntax to refer to the corresponding DHP

because DHPs are syntactically complex.

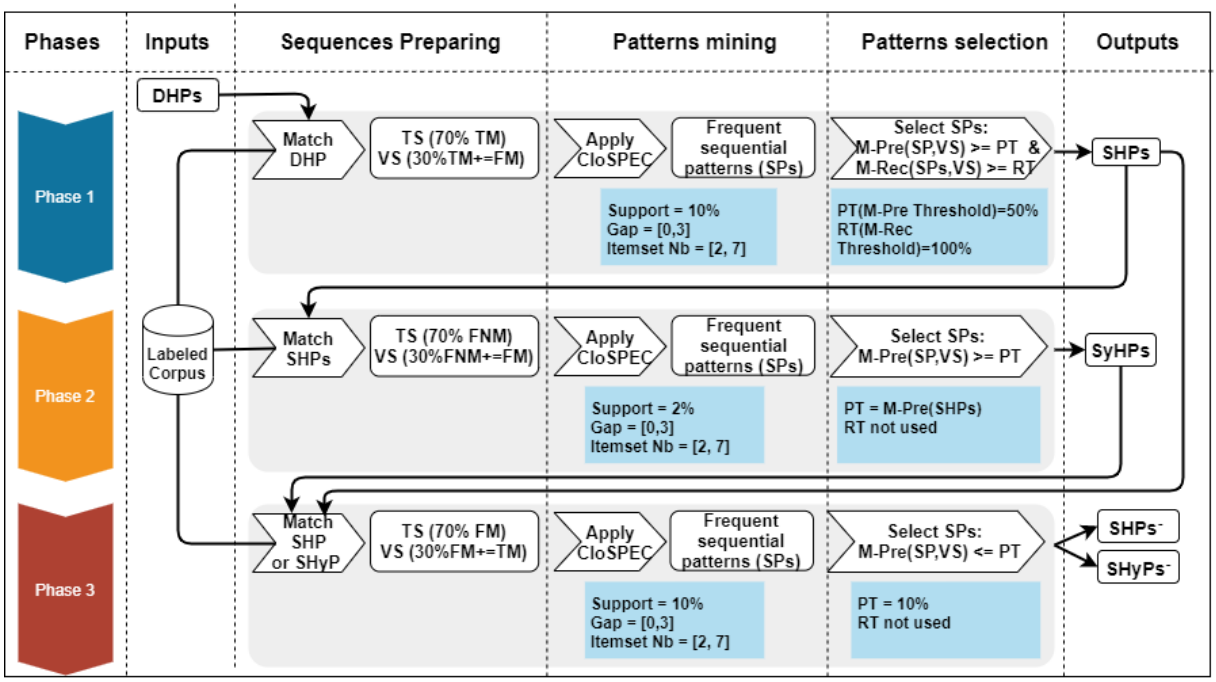


Figure 3.3: Sequential pattern learning using the 3-phase approach.

Starting from DHPs, in phase 1, learning process steps are repeated for each DHP, leading to one extracted set of sequential Hearst patterns ($SHPs_i$) for each DHP (DHP_i). SHPs $= \bigcup_{i=1}^{6} SHPs_i$ is then the union of all extracted sets of sequential patterns for all six DHPs. It should be noted that with this strategy, the usage of CloSPEC can be parallelised, limiting the risk of memory fault due to the mining complexity.

In phase 2, learning process steps are performed once to extract new sequential patterns from FNM sentences with respect to SHPs. SHyPs $= \{SHyPs_j, 1 \leq j \leq m\}$ is the set of extracted sequential patterns in this phase.

In phase 3, learning process steps are repeated for each $SHPs_i$ and for each $SHyPs_j$, leading to a set of sequential anti-hypernym patterns $SHPs^-$ and $SHyPs^-$ respectively. $SHPs^- = \bigcup_{i=1}^{M} SHPs_i^-$ is the union of all extracted sets of sequential anti-hypernym patterns for all set of sequential patterns extracted in phase 1. $SHyPs^- = \bigcup_{i=1}^{K} SHyP_i^-$ is the union of all extracted sets of sequential anti-hypernym patterns for all new sequential patterns extracted in phase 2. Even in this case, the strategy enables to parallelise the CloSPEC usage.

As described in Section 3.2.4, performing the generic workflow requires to set up some configurations. Specifically, the values of constraint parameters used for sequential pattern learning

and selection step should be set. The values of parameters shown in figure 3.3 are the best-fit values after having performed a tuning process. Note that some values are fixed for all phases.

— **Support:** is tuned using the following values {20%, 10%, 5%, & 2%}.

— **Minimum gap:** is fixed to 0.

— **Maximum gap:** has been tuned by using the following values {0, 3, 5, & 10}. As already explained, we found that it is needed to set a maximum gap strictly positive (>0). We also found that using greater values than 3 has no advantage since no new patterns were discovered while, at the same time, falling in high mining complexity.

— **Minimum itemset number:** is fixed to 2.

— **Maximum itemset number:** has been tuned by using the following values {7, 8, 9, & 10}. We found that using 7 is enough to learn hypernym patterns because with greater values extracted patterns were mostly wrong while, at the same time, falling in high mining complexity.

— **PT (M-Precision Threshold):** is set according to the objective of each phase. Phase 1 objective leads to find sequential patterns (SHPs) more precise than DHPs. SHPs will be therefore used for extracting ho-hr couples instead of using DHPs. For this purpose, PT can be set to 0.5 corresponding to the M-precision of any DHP when evaluated on VS1 (by definition VS1 comprises only TM and FM sentences with respect to DHPs, in equal number). Phase 2 objective leads to find patterns other than SHPs and showing M-precision greater than the M-precision of SHPs evaluated on the testing labeled corpus (issued from the corpus labeling step) as the reference threshold PT. These additional patterns will be used in combination with SHPs for potentially extracting additional ho-hr couples. The values for PT for each of the three corpora can be found in tables 3.11, 3.12, and 3.13. In phase 3, we aim to learn anti-patterns for patterns extracted in phases 1 and 2. For this purpose, M-precision threshold is very low (PT= 0.1) and patterns extracted should have even lower M-precision on VS3, because these patterns should highlight counterexamples of hypernym relations, as better explained in the remainder.

— **RT (M-Recall Threshold):** is only used in phase 1 where the learned patterns are constrained to not reduce the M-recall of DHPs. Thus, RT is set to 1 because, by definition of VS1 (which does not comprise any FNM sentence with respect to DHPs), M-recall of DHPs on VS1 is necessarily equal to 1.

DHPs	DHP_{is-a}		$DHP_{such-as}$		$DHP_{including}$		DHP_{other}		$DHP_{especially}$		DHP_{as}	
Music	TM	FM	TM	FM	TM	FM	TM	FM	TM	FM	TM	FM
	283	762	336	114	74	81	78	26	7	1	27	11
	TS	VS	TS	VS	TS	VS	TS	VS	TS	VS	TS	VS
	198	170	235	201	52	44	55	46	5	2	19	16
English-1	TM	FM	TM	FM	TM	FM	TM	FM	TM	FM	TM	FM
	239	328	58	36	15	33	138	35	3	0	4	2
	TS	VS	TS	VS	TS	VS	TS	VS	TS	VS	TS	VS
	167	143	40	36	11	8	97	70	2	0	3	2
English-2	TM	FM	TM	FM	TM	FM	TM	FM	TM	FM	TM	FM
	592	1008	93	81	12	83	60	72	4	5	5	10
	TS	VS	TS	VS	TS	VS	TS	VS	TS	VS	TS	VS
	414	355	65	56	8	8	42	36	3	2	4	2

Table 3.2: Phase 1: Numbers TM, FM, TS, and VS for each DHP

3.3.3 Phase 1: SHPs Learning Results

Phase 1 starts with the sequence preparation step. All DHPs are applied to the learning labeled corpus (thus comprising both sentences labeled as positive and negative) leading to sentence partitioning as TM, FM, and FNM. For each DHP, a set of training sequences (TS) is defined by using 70% of relevant TM sentences; a corresponding set of validation sequences (VS) is also defined by using the remaining 30% of TM sentences and the same number of FM sentences. Table 3.2 shows for each corpus, and for each DHP, the cardinalities of TM, FM, TS, and VS. Some DHPs result in few TM sentences, leading to very limited numbers of TS. We have considered that whenever TS is less than 20, it is unreliable for mining (these TS are shown with a grey background in the table)[4]. As a consequence, no mining has been performed on sentences belonging to any corpus and exclusively matching with DHP patterns (corresponding to HPs) "NP especially NP" and "such NP as NP". The same for sentences belonging to corpora English-1 and English-2 and exclusively matching with DHP pattern (corresponding to HP)"NP including NP".

Table 3.3 shows the number of frequent closed raw sequential patterns (FCSPs) extracted by applying CloSPEC on the appropriate TS and, for each selection criterion, the number of selected patterns. The results show the high selectivity of the first criterion (i.e. hypernym-hyponym path), specifically targeting M-precision. This selectivity is however partially balanced by keeping highest the resulting M-recall. We also report the case of High Precision

4. 20 has been chosen by observing unsuccessful tests to extract good patterns from TS with less than 20, and we believe they can be manually analyzed.

DHPs		DHP_{is-a}	$DHP_{such-as}$	$DHP_{including}$	DHP_{other}
Music	FCSPs	73,161	208,038	32,293	74,620
	With Hyper-Hypo path	2024	5104	1329	1803
	M-Pre>= 0.5 & M-Rec= 1	15	16	16	13
	SHPs (only sub-sequnces)	11	14	14	13
	HiPre-SHPs (M-Pre>= 0.8)	2	2	13	7
English-1	FCSPs	51,870	37,251	Irrelevant TS (=11)	45,105
	With Hyper-Hypo path	1424	954		1281
	M-Pre>= 0.5 & M-Rec= 1	7	10		9
	SHPs (only sub-sequnces)	7	8		6
	HiPre-SHPs (M-Pre>= 0.8)	4	4		3
English-2	FCSPs	125,276	89,245	Irrelevant TS (=8)	24,392
	With Hyper-Hypo path	3720	2393		982
	M-Pre>= 0.5 & M-Rec= 1	9	5		7
	SHPs (only sub-sequnces)	7	4		7
	HiPre-SHPs (M-Pre>= 0.8)	4	2		4
Total distinct	SHPs	21	22	14	23
	HiPre-SHPs	10	8	13	12

Table 3.3: Phase 1: Numbers of TS, FCSPs, and selected patterns by each substep of selection.

Sequential Hearst's patterns (HiPre-SHPs) selected by setting $PT = 0.8$ and without setting any RT. This specific experiment shows how to manage thresholds and what can be the impact—as a consequence, the number of patterns selected is dramatically reduced because PT is very high and M-recall may freely decrease.

Table 3.4 shows representative examples of SHPs learned from the Music corpus alongside additional details. For each SHP, the table points the corresponding DHP, one sample of TM sentence is provided pointing the extracted ho-hr couples, one sample of FM sentence matching the corresponding DHP while not matching with the SHP, and the explanation/interpretation of the SHP. The M-precision evaluated on the relevant VS is also shown.

3.3.4 Phase 2: SHyPs Learning Results

Phase 2 starts by applying SHPs, results from phase 1, to the learning corpus, leading to TM, FM, and FNM partitions. A set of training sequences (TS) is then defined as 70% of FNM sentences; and a set of validation sequences (VS) is also defined by including 30% of the remaining FNM sentences and the same number of negative sentences. Table 3.5 shows the cardinality of TS, the number of FCSPs extracted by CloSPEC on TS, and the number of selected patterns in each substep of the selection step; as for phase 1, the high selectivity of the first criterion can be noted.

DHPs	SHPs	M-Pre
DHP_{is-a}	[(NP, hypo, nsubj→ , hyper_gov), (be_lemma, cop→ , hyper_gov), (NP, hyper, band_lemma)]	0.938
	TM sentence: Strangers is a band founded in 2008 ... ; ho-hr(Strangers, band)	
	FM sentence by DHP and not matched by the SHP: Another notable percussionist was Parvinder Bharat (parv) of wolverhampton.	
	Explanation: the lemma of the head of the hypernym NP is "band". This pattern extracts hyponyms for the specific hypernym "band". It increases the M-precision of DHP from 0.5 to 0.938. But, its M-recall will be almost null for a corpus not dealing with "band"!	
DHP_{is-a}	[(NP, hypo, nsubj→ , hyper_gov), (is_word, be_lemma, cop→ , hyper_gov), (NP, NN, hyper, root← , null_gov), (acl:relcl← , hyper_gov)]	0.784
	TM sentence: Jazz is a music genre that originated ... ; ho-hr(Jazz, music genre)	
	FM sentence by DHP and not matched by the SHP: The two founding members Jockel and Fritz are brothers.	
	Explanation: the verb connecting the hyponym NP to hypernym NP is specified to "is", the head of the hypernym NP should be a singular noun (NN) and the root of the dependency tree, and the hypernym NP should be followed by any word having "acl:relcl← " as grammatical relation.	
DHP_{other}	[(NP, NN, hypo, dobj←), (and_word, and_lemma, CC, cc←), (amod→ , other_word, other_lemma, JJ, hyper_gov), (NP, hyper, conj:and← , hypo_gov)]	0.656
	TM sentence: it features piano and other keyboards ... ; ho-hr(piano, keyboards)	
	FM sentence by DHP and not matched by the SHP: you plug one end into your phone socket and the other end into your personal computer ...	
	Explanation: the head of the hyponym NP is specified to occur as object (dobj←).	
$DHP_{such-as}$	[(NP, hyper, NNS), (case→ , such_lemma, JJ, hypo_gov), (as_lemma, IN, mwe← , such_gov), (NP, hypo, nmod:such_as← , hyper_gov)]	0.538
	TM sentence: Unlike genres such as jazz or opera, pop is ... ; ho-hr(jazz, genres) and ho-hr(opera, genres)	
	FM sentence by DHP and not matched by the SHP: ... to lead to a general harmony such as bands made up of instrument known ...	
	Explanation: the head of the hypernym NP is specified to be plural noun (NNS).	

Table 3.4: Phase 1: Representative samples of learned SHPs.

Corpus	Music	English-1	English-2
TS	1129	800	1955
FCSPs	810,423	521,032	913,351
With Hyper-Hypo path	2,362	799	1,137
M-Pre >= M-Pre of SHPs on VS2	15	17	8
SHyPs (only sub-sequences)	3	4	2

Table 3.5: Phase 2 : Numbers of TS, FCSPs, and selected patterns by each substep of selection.

Table 3.6 shows all the (five) new learned SHyPs from the three corpora and samples of TM sentences. To the best of our knowledge, 2 of them ($SHyP_{ranging}$ and $SHyP_{who}$, the first and third SHyP in the table) have not been identified in the literature. New sequential patterns refining the Hearst's pattern "NP and/or other NP" have been discovered. These new patterns do not have any counterpart in both DHPs and SHPs—indeed, dependency paths connecting hyponym NP to hypernym NP in the newly discovered patterns are different from relevant paths occurring in DHPs and SHPs. The limited number of new patterns can be explained by setting their M-precision to be equal or greater than M-precision of SHPs.

Table 3.6: Phase 2: All learned SHyPs from the three corpora

Corpus	**SHyPs**	**M-Pre**
Music	[(NP, hyper), (ranging_word, range_lemma, VBG, acl← , hyper_gov), (from_word, from_lemma, IN, case→ , hypo_gov), (NP, hypo, nmod:from← , range_gov)]	0.903
	TM sentence: ... into various other genres ranging from punk rock to electronic ... ; ho-hr(punk rock, genres)	
	FM sentence: ... many different styles at work here ranging from funk to metal ...; due to dependency parsing error "ranging" is related to "work" and not "styles" leading to identify wrong ho-hr(funk, work)	
	Explanation: A hypernym noun phrase that is followed by "ranging from" and a hyponym noun phrase, and at the same time, both hypernym and hyponym noun phrases are grammatically related to the word "ranging".	
English-1	[(NP, hyper), (other_word, other_lemma, JJ), (than_word, than_lemma, IN, case→ , hypo_gov), (NP, hypo, nmod:than← , hyper_gov)]	1.0

Continued on next page

Table 3.6 – continued from previous page

Corpus	SHyPs	M-Pre
	TM sentence: ... better than any metal other than silver and copper ...; ho-hr(silver, metal) and ho-hr(copper, metal)	
	Explanation: A hypernym noun phrase that is followed by "other than" and a hyponym noun phrase, and at the same time, hypernym and hyponym noun phrases are grammatically related by the grammatical relation "nmod:than← ".	
English-1	[(NP, hyper, NN), (who_word, who_lemma, WP, nsubj→ , hypo_gov), (NP, hypo, NN, acl:relcl← , hyper_gov)]	0.82
	TM sentence: ... a person who has been a patient for more ...; ho-hr(patient, person)	
	FM sentence:a friend who is a teacher spoke to ...	
	Explanation: a hypernym noun phrase that is followed by "who" and a hyponym noun phrase, and at the same time, hyponym and hypernym are grammatically related by the grammatical relation "acl:relcl← ".	
Music English-1 English-2	[(NP, hyper), (like_word, like_lemma, IN, case→ , hypo_gov), (NP, hypo, nmod:like← , hyper_gov)]	0.538
	TM sentence: What's great about living in a city like Paris is that there is literally never a dull moment; ho-hr(Paris, city)	
	FM sentence: ... he had friends like most boys ...	
	Explanation: a hypernym noun phrase that is followed by "like" and a hyponym noun phrase, and at the same time, hyponym and hypernym are grammatically related by the grammatical relation "nmod:like← ".	
Music English-1 English-2	[(NP, hypo, NN, compound→ , hyper_gov), (CC, cc← , hypo_gov), (other_word, other_lemma, JJ), (NP, hyper)]	0.78
	TM sentence: You can check it on youtube or other websites like this; ho-hr(youtube, websites)	
	FM sentence: Their style is a mixture of electro pop, punk, new wave, techno and other trends.	

Continued on next page

Table 3.6 – continued from previous page

Corpus	SHyPs	M-Pre
	Explanation: a hyponym noun phrase that is followed by conjunction word such as "or" and "and" and then followed by "other" word and hypernym noun phrase, and at the same time, hyponym and hypernym are grammatically related by the grammatical relation "compound→ ".	

3.3.5 Phase 3: SHPs$^-$ Learning Results

The objective of phase 3 is to discover anti-patterns for patterns discovered in phases 1 and 2. Hereinafter, we focus on anti-patterns for patterns discovered in phase 1 (i.e. SHPs). The sequence preparation step provides training and validation sequence sets per each set of SHPs patterns corresponding to one DHP. Indeed, for one DHP several SHPs have been discovered. Accordingly, for each set of SHPs patterns corresponding to one DHP, one TS is defined as comprising 70% of FM sentences with respect to SHPs. The same is done for defining validation sets: each validation set VS comprises 30% of remaining FM sentences and the equal number of TM sentences. Indeed, finding extremely low M-precision patterns means that these patterns mostly match with FM sentences (found in TS and VS). FM sentences are by definition sentences (positive or negative) matching with a pattern such that: a couple is extracted but it is not found in the given ho-hr couples or in WordNet, or human experts do not validate the sentences as expressing hypernymy stated by the couple. As a consequence, these sentences are likely to provide counterexamples of hypernymy. Table 3.2 shows all cardinalities of training and validation sequence sets for each set of SHP patterns corresponding to one DHP. TS and VS with few elements are shown with a grey background.

Table 3.8 summarises the pattern selection for this phase. The table shows the number of extracted FCSPs from each TS (except the ones with few elements) and the number of selected patterns in each selection substep.

Table 3.9 shows representative samples of learned SHPs$^-$. Each SHP$^-$ is provided with one sample of FM sentence alongside with the false ho-hr couples extracted from the sentence, and an explanation. We also show the M-precision of each SHP$^-$ calculated on the VS. The very low M-precision of these patterns allows to filter the FM sentences by SHPs but also reduces slightly the M-recall by filtering some TM sentences matching SHPs. For instance, the second learned SHP$^-$ in table 3.9 filters the false matched sentences like "NP is in NP" (for instance,

Corpora	SHPs$_{is-a}$		SHPs$_{such-as}$		SHPs$_{including}$		SHPs$_{other}$	
Music	TM	FM	TM	FM	TM	FM	TM	FM
	214	530	225	60	50	20	62	18
	TS	VS	TS	VS	TS	VS	TS	VS
	371	318	42	36	14	12	13	10
English-1	TM	FM	TM	FM	TM	FM	TM	FM
	135	223	50	32	15	33	79	27
	TS	VS	TS	VS	TS	VS	TS	VS
	156	134	22	20	23	20	19	16
English-2	TM	FM	TM	FM	TM	FM	TM	FM
	426	703	71	49	12	83	39	35
	TS	VS	TS	VS	TS	VS	TS	VS
	492	422	34	30	58	24	25	20

Table 3.7: Phase 3: Numbers of TM and FM sentences, TS and VS for each set of SHPs corresponding to one DHP

SHPs		SHPs$_{is-a}$	SHPs$_{such-as}$	SHPs$_{including}$	SHPs$_{other}$
Music	FCSPs	569,389	284,220	Irrelevant TS (=14)	Irrelevant TS (=13)
	With Hyper-Hypo path	5742	2325		
	M-Pre$<= 0.1$	1	2		
	SHP$^-$ (only sub-sequences)	1	2		
English-1	FCSPs	506,451	132,571	150,258	Irrelevant TS (=19)
	With Hyper-Hypo path	5130	2471	3012	
	M-Pre$<= 0.1$	2	0	0	
	SHP$^-$ (only sub-sequences)	2	0	0	
English-2	FCSPs	1,424,351	201,621	123,771	176,271
	With Hyper-Hypo path	8742	1621	1224	921
	M-Pre$<= 0.1$	2	0	2	0
	SHP$^-$ (only sub-sequences)	2	0	2	0

Table 3.8: Phase 3: Numbers of TS, FCSPs, and selected patterns by each substep of selection.

SHPs	SHPs^{-}	M-Pre
SHPs$_{is-a}$	[(nmod:poss→ , hypo_gov), (NP, NN, hypo, nsubj→ , hyper_gov), (is_word, VBZ, be_lemma, cop→ , hyper_gov), (NP, hyper)]	0.08
	FM sentence: His instrument is the piano ... ; False ho-hr(instrument, piano)	
	Explanation: the verb connecting the hyponym NP to hypernym NP is specified to “is”, the lemma of the head of the hypernym NP is specified to be singular noun (NN), and the hypernym NP is preceded by a possessive pronoun (nmod:poss→).	
SHPs$_{is-a}$	[(NP, hypo, nsubj→ , hyper_gov), (be_lemma, cop→ , hyper_gov), (case→), (NP, hyper)]	0.09
	FM sentence: the situation is under control ... ; False ho-hr(situation, control)	
	Explanation: Existence of a word between the verb (be_lemma) and the hypernym NP whose grammatical relation is “case→ ” .	
SHPs$_{such-as}$	[(NP, hyper), (NP), (case→ , such_lemma, JJ, hypo_gov), (as_lemma, IN, mwe← , such_gov), (NP, hypo, nmod:such_as← , hyper_gov)]	0.05
	FM sentence: Many artists from different styles such as metal, pop ... ; False ho-hr(metal, artists) and False ho-hr(pop, artist)	
	Explanation: A NP occurs between the hypernym NP and the word “such”.	

Table 3.9: Phase 3: Representative samples of learned SHPs^{-}.

"the vehicle is in motion") and “NP is under NP” (for instance "the situation is under control"). However, it also filters some true matched sentences like “NP is among NP” (for instance "the brain is among the organs that$\cdots$").

3.3.6 Phase 3: SHyPs^{-} Learning Results

Phase 3 is also targeting anti-patterns for the patterns discovered in phase 2 (SHyPs). Accordingly, the sequence preparation step leads to appropriate training and validation sequence sets. For each SHyP pattern, one training sequence set is defined as 70% of FM sentences with respect to the pattern. Corresponding validation sequence set is defined by using the remaining 30% of FM sentences and the equal number of TM sentences. Table 3.10 shows the results for each SHyP on each corpus [5]. Only $SHyP_{like}$, learned from Music corpus, has a relevant number of training sequences ($54 >= 20$). From this training sequence set 486,818 FCSPs have been

5. Blank cells are those corresponding to patterns that are not learned by the corpus.

Corpora	SHyP$_{ranging}$		SHyP$_{other-than}$		SHyP$_{who}$		SHyP$_{other}$		SHyP$_{like}$	
Music	TM	FM					TM	FM	TM	FM
	20	7					32	14	152	78
	TS	VS					TS	VS	TS	VS
	4	6					9	10	54	48
English-1			TM	FM	TM	FM	TM	FM	TM	FM
			10	2	18	9	26	20	30	13
			TS	VS	TS	VS	TS	VS	TS	VS
			1	2	6	6	14	12	9	8
English-2							TM	FM	TM	FM
							13	14	23	25
							TS	VS	TS	VS
							9	10	17	16

Table 3.10: Phase 3: Numbers of TM, FM, TS, and VS for each SHyP

extracted and only patterns with M-precision lower than 0.1 are to be kept. Unfortunately, this condition is highly selective and no pattern has been kept (SHyPs$^-$ is void).

3.4 Evaluation step: results and analysis

3.4.1 Evaluation step: quantitative results

As said in the Introduction, the 3-phase approach is validated by performing the first round of comparisons and analysis. For that purpose, we use existing patterns i.e. HPs, ExtHPs, and DHPs, and evaluated M-precision and M-recall for those patterns on each testing corpus established during the corpus labeling step for each one of the used three corpora. Tables 3.11, 3.12, and 3.13 show M-precision, M-recall, and f-score for respectively Music, English-1, and English-2 testing corpora.

The tables also show M-precision and M-recall stated on the testing corpora for the learned patterns resulting from each phase. For the 3 corpora, SHPs show better M-precision than DHPs with a slight reduction in M-recall. When SHPs$^-$ are used to exclude couples found by SHPs (table column header SHPs + SHPs$^-$), M-precision is also improved for the 3 corpora, again with a slight reduction in M-recall. When SHyPs are combined to SHPs and SHPs$^-$, a considerable improvement of M-recall for 3 corpora can be observed while M-precision is almost unchanged. Concerning HiPre-SHPs, as expected, the results for the 3 corpora confirm that learning SHPs with high M-precision is possible but M-recall falls down dramatically. Finally, the best f-score for the 3 corpora and for all patterns compared in the tables, is achieved by the combination of all learned patterns and anti-patterns, issued for the three phases. This confirms

the interest of the approach over the manual design of patterns. The approach can then be practically used by starting from other corpora for getting additional patterns in a systematic way. However, M-precision cannot be globally kept very high if M-recall needs to be improved. As a consequence, only slight improvements to M-precision should be used as thresholds.

Music	HPs	ExtHPs	DHPs	HiPre-SHPs	SHPs	SHPs + SHPs$^-$	SHPs + SHPs$^-$ + SHyPs
M-Precision	0.556	0.576	0.464	**0.667**	0.486	0.518	0.516
M-Recall	0.157	0.192	0.207	0.125	0.203	0.201	**0.24**
F-score	0.245	0.288	0.286	0.210	0.287	0.289	**0.327**

Table 3.11: Evaluation results on Music corpus.

English-1	HPs	ExtHPs	DHPs	HiPre-SHPs	SHPs	SHPs + SHPs$^-$	SHPs + SHPs$^-$ + SHyPs
M-Precision	0.451	0.474	0.428	**0.663**	0.482	0.493	0.499
M-Recall	0.073	0.163	0.149	0.86	0.146	0.145	**0.177**
F-score	0.125	0.242	0.221	0.152	0.224	0.224	**0.261**

Table 3.12: Evaluation results on English-1 corpus.

English-2	HPs	ExtHPs	DHPs	HiPre-SHPs	SHPs	SHPs + SHPs$^-$	SHPs + SHPs$^-$ + SHyPs
M-Precision	0.41	0.41	0.371	**0.493**	0.392	0.424	0.427
M-Recall	0.047	0.102	**0.123**	0.055	0.12	0.113	0.12
F-score	0.084	0.163	0.185	0.099	0.184	0.179	**0.187**

Table 3.13: Evaluation results on English-2 corpus.

3.4.2 Evaluation step: qualitative analysis of learned patterns

Quantitative results in tables 3.11, 3.12, and 3.13 show M-precision of SHPs a little bit higher than the one of DHPs. We conducted a deeper analysis focusing on sentences in the Music corpus being in FM with respect to DHP while non matching with any SHP. These

are 75 sentences out of 653 of FM sentences with respect to DHPs. Let's consider the sentence "Some prominent Swedish bands spawned during this second wave, such as Marduk, Nifelheim and Dark Funeral". The DHP corresponding to HP "NP such as NP" extracts "second wave" as hypernym, which is wrong. This is not the case for SHPs because in all these patterns the headword of the hypernym NP should be plural while "wave" in "second wave" is singular (a representative sample of those SHPs can be found in the last line of table 3.4).

However, constraining hypernym requiring headword to be plural in patterns type "NP such as NP", is not necessarily interesting in all situations. For instance, the sentence "'So before a band such as Tower of Power would reject the style ...' matches with DHPs and is part of TM sentences with respect to DHP but does not match with SHPs because "band" (the hypernym) is a singular noun (indeed the sentence is part of FNM sentences with respect to SHPs). More generally, 8 sentences out 567 sentences being part of TM sentences with respect to DHPs suffer from this "plural/singular" problem, and 9 sentences out 567 sentences being part of TM sentences with respect to DHPs do not match with SHP "NP including NP" pattern. Hence, we may deduce that improving the M-precision of patterns having a quite good M-precision may negatively impact M-recall.

Concerning SHyPs, table 3.11 shows a stable M-precision and a considerable M-recall improvement. To better understand the raising for those quantitative results, we analyze TM and FM sentences with respect to SHyPs while non matching with any SHPs. SHyPs when used with SHPs increase TM by 111. An example of a sentence from the 111 is: "they fuse together sounds from different genres like metal and reggae"—ho-hr couple(metal, genres) and ho-hr couple(reggae, genres) are extracted. SHyPs also increase FM by 105. An example of a sentence from the 105 is: "Her voice is an instrument all it's own, slicing through the air like a weapon sometimes"—wrong ho-hr couple(weapon, air) is extracted. We conclude that since the increase in TM and FM are comparable so that the M-precision remains stable. However, because 111 of FNM with respect to SHPs have been fallen in TM when SHyPs are used, M-recall is importantly increased.

Finally, concerning SHPs$^-$ table 3.11 shows the important impact of these patterns on M-precision. These patterns should mainly impact on FM sentences. Out of 621 sentences found in FM with respect to SHPs, we found 75 sentences have been matched by SHPs$^-$. These sentences should not be taken into account for suggesting hypernym relations, thus automatically increasing M-precision. An example of such matching sentences is: "their music is rock, pop, and alternative" where the occurrence of possessive pronouns "their" leads the sentence to be matched the first SHP$^-$ in table 3.9. To understand the slight reduction in the M-recall caused by

		Music	English-1	English-2	Total common	Total distinct
SHPs	Music	52	5	3	3	80
	English-1	5	21	5		
	English-2	3	5	18		
SHyPs	Music	3	2	2	2	5
	English-1	2	4	2		
	English-2	2	2	2		
$SHPs^{-}$	Music	3	0	0	0	9
	English-1	0	2	0		
	English-2	0	0	4		

Table 3.14: Replicability of the learned sequential patterns

$SHPs^{-}$, we searched for TM sentences with respect to SHPs matching with $SHPs^{-}$. We found only 5 of such sentences out of 557 TM sentences with respect to SHPs. An example of such sentences is: "Berry was among the first musicians to be inducted into the rock and roll hall of fame on its opening in 1986" where the occurrence of "among" whose grammatical relation is "case$\rightarrow$ " leads the sentence to be matched with the second SHP^{-} in table 3.9.

3.4.3 Evaluation step: analysis of learned patterns generality

Table 3.14 shows the number of distinct patterns (SHPs, SHyPs, and $SHPs^{-}$) learned from each corpus of the three corpora [6], the number of common patterns, the number of patterns in common between the three corpora, and the number of distinct patterns learned from the three corpora. We can notice the few numbers of patterns that have been learned from at the same time the three corpora.

We define "pattern generality" as the pattern ability to extract right ho-hr couples across distinct corpora. In figure 3.4, we present f-score across the three corpora when all learned patterns are used. This f-score is then compared to the f-score of DHPs. In the 3 corpora, the results confirm that the learned sequential patterns from one corpus are suitable to be used on the other corpora where they show better f-score than DHPs in all cases. The results also confirm the advantage of the learned sequential patterns from English-1 corpus over the learned sequential patterns from the other two corpora when applied to general-purpose corpora (English-1 and English-2)—ALL SP (English-1) shows the best f-score in both English-1 and English-2 corpora. A deeper analysis of this result we deduce that SHyPs have a larger impact on the f-score especially in general-purpose corpora (English-1 and English-2)—ALL SP (English-1)

6. The value where the column and the row corresponding to the same corpus represents the distinct patterns learned by the corpus.

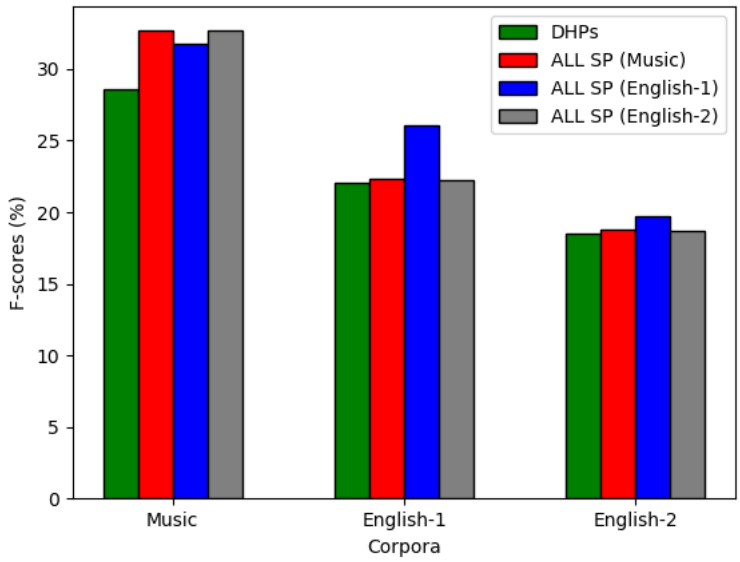


Figure 3.4: Performance of the learned sequential patterns on the three corpora.

contains 2 additional SHyPs (second and third patterns in table 3.6) than those contained in ALL SP (English-2). We also deduce that these additional SHyPs are more related to general-purpose corpora since they have not shown the same impact when applied in the Music corpus (specific-domain).

3.4.4 The Impact of Gap Constraint in Matching Sequential Patterns

We also analyze the impact of using gap constraint in matching sequential patterns on the performance of patterns. For this purpose, we evaluate the performance in terms of M-precision, M-recall, and f-score of the learned sequential patterns with tuning the maximum gap parameter from 0 to 10 [7]. Note that the minimum gap value is fixed to 0. Figures 3.5, 3.6, and 3.7 show the variation of M-precision, M-recall, and f-score by the variation of maximum gap value for Music, English-1, and English-2 corpora respectively. We can notice from the results in the three corpora that M-recall is increasing and M-precision is decreasing by the increase of maximum gap value, especially where maximum gap value ranges between 0 and 5. The M-precision and M-recall approximately remain stable when the maximum gap is greater than 5.

Note that matching sequential patterns with minimum and maximum gaps equal to 0 ($[0, 0]$) is similar to the matching of lexico-syntactic patterns where pattern elements should occur in the sentence without any gap between them. Thus, the results confirm the importance of gap

7. We stop at 10 because it is the maximum distance that we have used to label corpus sentences as positive.

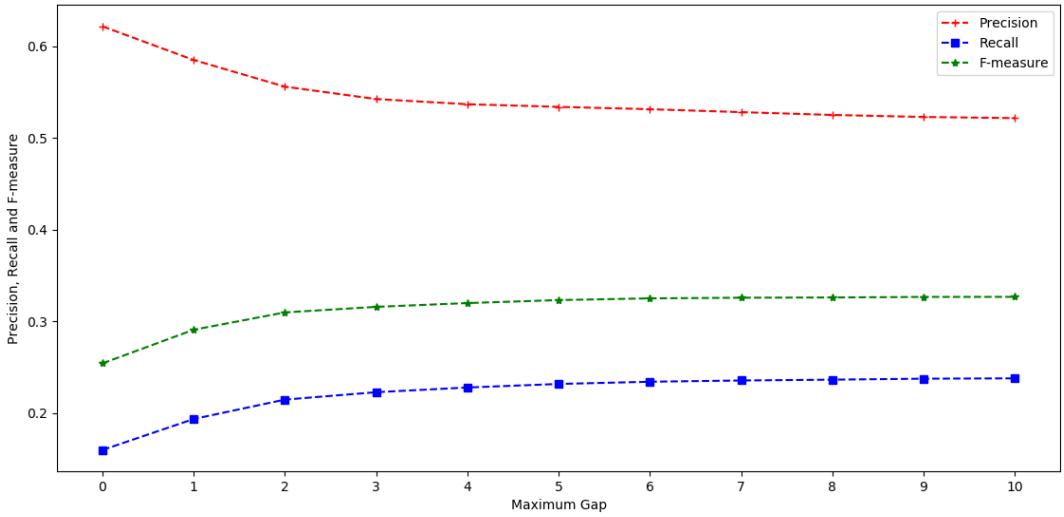


Figure 3.5: M-Precision, M-recall, & f-score variation by the variation of maximum gap value using Music corpus.

constraint and dependency relations that enable us to extract hypernym relations that can not be extracted by lexico-syntactic patterns (recall increases by the increase of maximum gap). Moreover, we may notice the direct relation between M-precision and maximum gap value, so hypernym relations extracted by sequential patterns using a low value of maximum gap are more confident to be true hypernym relations than those extracted by a higher maximum gap. The curve of f-score is proportional to the curve of M-recall. The reason is that M-recall values are very small according to M-precision values, so f-score variation is highly affected by the variation of recall.

3.5 Summary

Previously, we have reformulated Hearst's patterns as dependency patterns based on dependency parsing, and they are called DHPs. The formulated DHPs show a considerable improvement in terms of recall due to their generality in comparison with lexico-syntactic Hearst's patterns. However, their generality leads to a slight reduction in precision. In this work, we are interested to improve both precision and recall of DHPs. For this purpose, we have proposed a 3-phase approach that integrates sequential pattern mining and syntactical dependencies to systematically improve the performance of pattern-based hypernym extraction approaches.

The first phase aims to extract sequential patterns (SHPs) from a given set of seed patterns in order to improve either precision or recall of the seed patterns without degrading the other

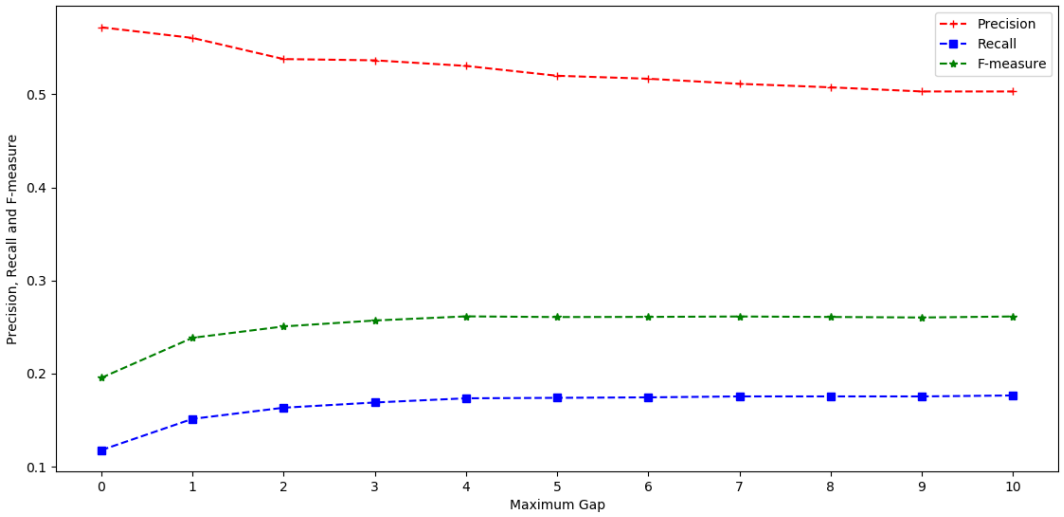


Figure 3.6: M-Precision, M-recall, & f-score variation by the variation of maximum gap value using English-1 corpus.

one. The second phase aims to extend SHPs with new sequential hypernym patterns (SHyPs) for improving the recall. The third phase aims to find anti-patterns to filter wrongly identified hypernym relations by SHPs and SHyPs.

For the purpose of learning and evaluating sequential hypernym patterns, we propose a general workflow of five steps that takes as input a corpus and a dataset of known hypernym relations and gives as output a set of evaluated sequential patterns. The first step is responsible to label corpus sentences as positive and negative sentences and then divide the labeled sentences into learning and testing sets of sentences. In the second step, a relevant sequential database for the purpose of learning is created from the set of learning sentences, it also creates a set of validation sequences to be used later for validation purposes. The third step takes as input the sequential database and performs a sequential pattern mining algorithm to extract the frequent sequential patterns. In the fourth step, the extracted frequent sequential patterns are filtered and then only relevant sequential patterns are selected based on the validation set of sequences. The final step is an evaluation step to evaluate and compare the learned sequential patterns to other types of hypernym patterns.

In order to create the sequential database and the set of validation sequences from sentences, sentences should be represented as sequences. We propose a new representation of sequences from sentences that combines lexical, syntactical, and dependency information about sentence elements. The sentence element in our representation is either a noun phrase or a word. Representing noun phrases in a sequence is significant to learn sequential hypernym patterns that can

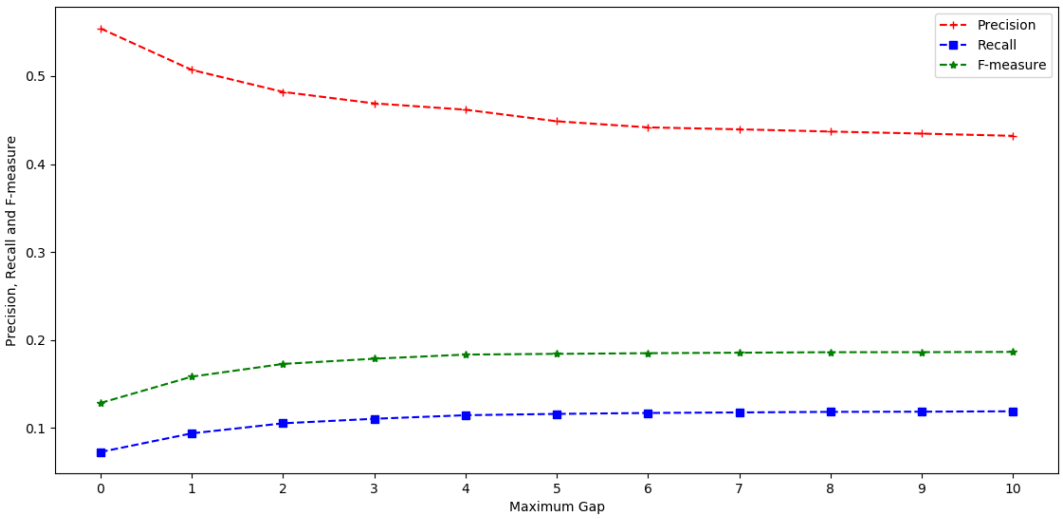


Figure 3.7: M-Precision, M-recall, & f-score variation by the variation of maximum gap value using English-2 corpus.

extract hypernym relations between noun phrases and not between nouns.

To evaluate our proposed approach of 3-phase, we have used 3 corpora and two datasets to learn and evaluate sequential patterns via comparing their performance to that of DHPs, lexico-syntactic Hearst's patterns (HPs), and the extended set of lexico-syntactic patterns (ExtHPs). The results in the three corpora show that the learned sequential patterns from the 3-phases of the proposed approach combined together show the best f-score, even it is better than the f-score of the ExtHPs (59 patterns).

Qualitative analysis is also performed to the learned sequential patterns by the 3 phases to study the impact of each phase in both precision and recall. The learned SHPs are specific patterns of DHPs that improve precision by avoiding the matching of many sentences that are falsely matched by DHPs. But, they also avoid matching some sentences that are truly matched by DHPs which lead to a slight reduction in the recall. The discovered sequential patterns (SHyPs) enable us to match many sentences that are not matched by SHPs, so they lead to a considerable improvement in terms of recall. The learned anti-hypernym patterns perform well to filter many sentences that are falsely matched by SHPs which lead to improve precision. But, they also filter some sentences that are truly matched by SHPs which lead to a slight reduction in the recall. We also perform an analysis to study the generality of the learned sequential patterns. For this purpose, we perform the learned sequential patterns on the corpora other than that was used to learn them. The results confirm that the learned sequential patterns from one corpus are suitable to be used on other corpora.

As a conclusion, we have proposed a 3-phase approach based on sequential pattern mining and dependency parsing to learn sequential patterns that can perform better than DHPs in terms of precision and recall. In the first phase, we learn sequential patterns that can replace DHPs with better precision. In the second phase, we improve recall by discovering new sequential patterns other than that correspond to Hearst's patterns. In the third phase, we learn anti-hypernym patterns that can improve precision by filtering the wrong matching of sequential hypernym patterns. The results show that the learned sequential patterns from the 3 phases combined together outperform other types of patterns such as DHPs in terms of both precision and recall.

CHAPTER 4

METHODS FOR HYPERNYMY DETECTION USING OUR PATTERNS

In this chapter, we describe methods that have been proposed for the purpose of comparing our patterns to unsupervised and supervised distributional methods. In section 4.2, we describe the proposed unsupervised method and the experiment performed to compare our patterns to unsupervised distributional methods. Section 4.3 presents the proposed supervised method and the experiment performed to compare our patterns to supervised distributional methods. Further experiments are also described in section 4.3 to study the complementarity between pattern-based methods and distributional methods and to analyze the generality of pattern-based models and distributional models.

4.1 Introduction

As stated in Chapter 1, a proposed approach dealing with hypernym relations addresses a specific task. This task is either hypernymy extraction, hypernymy detection, or hypernymy discovery. In general, pattern-based methods are usually used for hypernymy extraction by matching hypernym patterns to corpus sentences. However, distributional methods are usually used for hypernymy detection or hypenymy discovery.

In this thesis, we have learned new patterns for hypernymy extraction (i.e. dependency patterns (**DHPs**) in chapter 2 and sequential patterns (**SHPs + SHP$^-$ + SHyPs**) in chapter 3). The new patterns have been evaluated by comparing their performance to other existing patterns for hypernymy extraction. The goal of this chapter is to use the new patterns in the task of hypernymy detection in order to compare their performance to those of unsupervised and supervised distributional methods.

In [81], Roller et al. used Hearst's patterns for hypernymy detection. They calculated a hypernymy score for each candidate ho-hr couple based on how often the couple is extracted by the Hearst's patterns from the corpus. Similarly, we use the new patterns and compute a score

for each candidate ho-hy couple, in order to be able to compare their performance to those of several unsupervised distributional methods.

To compare with supervised distributional methods, we propose a supervised method for hypernymy detection by incorporating the new patterns in a machine learning technique. This method allows us to perform hypernymy detection with the new patterns, to compare them to supervised distributional methods, to study the complementary between pattern-based method and distributional methods, and to analyze the generality of both types of models (i.e. how much each trained model is coupled with the training data).

4.2 Unsupervised Methods for Hypernymy Detection

In this section, we describe our unsupervised method for hypernymy detection using the new patterns (DHPs or sequential patterns). We then present the performed experiment in order to compare the pattern-based method to unsupervised distributional methods.

4.2.1 Our Pattern-based Unsupervised Method for hypernymy detection

Similar to Roller et al. [81], our pattern-based unsupervised method for hypernymy detection also based on a measure to compute a hypernymy score for each candidate ho-hr couple based on its frequency of extraction by patterns.

Let $\chi = \{(x_i, y_i)\}_{i=1}^{n}$ denote the set of n distinct ho-hr couples extracted from a corpus C using our patterns (DHPs or sequential patterns). And let $w(x_i, y_i)$ denote the count of how often the couple $(x_i, y_i) \in \chi$ has been extracted by using the considered patterns. Then, a hypernymy score for each couple (x_k, y_k) is computed as follow:

$$score(x_k, y_k) = \frac{w(x_k, y_k)}{max(\{w(x_i, y_i)\}_{i=1}^{n})} \tag{4.1}$$

with $(x_k, y_k) \in \chi$ and $k \in \{1, ..., n\}$.

Finally, by setting a threshold, the method labels as hypernymy all candidate ho-hr couples having a hypernymy score above the threshold; otherwise couples are considered non-hypernymy.

4.2.2 Experiment and Results

The experiment is performed to first evaluate our patterns for hypernymy detection and then compare their performance to unsupervised distributional methods. We also consider in the comparison the existing hypernym patterns that already used in previous experiments (HPs and extHPs). To avoid setting the threshold, unsupervised methods for hypernymy detection have been usually evaluated using Average Precision (AP) taking into consideration the rank of the ho-hr couples based on their computed score 1.1.4.

Corpora and Datasets

The experiment is performed using four benchmark datasets and the same three corpora (Music, English-1, and English-2) used in Chapters 2 and 3 (see table 3.1). The size of each dataset is shown in table 4.1. Three out of four datasets (BLESS, EVALution, and Weeds) are commonly used to evaluate hypernymy detection methods applied to general-purpose corpora. Music dataset has been already used for learning patterns from the Music corpus. For learning patterns using sequential pattern mining, non-hypernym couples (negative instances) are not needed. However, datasets for hypernymy detection should comprise non-hypernym couples—hypernymy detection methods are evaluated in their ability to classify candidate hypernym relations as hypernymy or not. Additionally, several methods for hypernymy detection are supervised where hypernym (positive) and non-hypernym (negative) couples are necessary for training the model (we use these datasets to train and evaluate supervised hypernymy detection methods in Section 4.3). Consequently, we provide the Music dataset by non-hypernym couples from other datasets (BLESS, EVALution, and Weeds) by selecting non-hypernym couples that either the hypernym or hyponym is a part of Music hypernym couples. For instance, let us suppose (piano, instrument) is a hypernym couple of Music dataset, we provide the dataset by all non-hypernym couples found in the other datasets where “piano” or “instrument” occur as hyponym or hypernym.

Datasets	Music	BLESS	EVALution	Weeds
Hypernym couples	4,478	1,337	3,415	1,469
Non-hypernym couples	14,849	25,217	10260	1,459
Total couples	19,327	26,554	13,675	2,928

Table 4.1: Datasets sizes.

Unsupervised Distributional Methods

Unsupervised distributional methods also use measures to compute a hypernymy score for each candidate ho-hr couple (x_k, y_k). But the score is computed based on the vector representation of both terms x_k and y_k in a distributional semantic space.

According to results presented in [90], distributional semantic spaces built using dependency-based contexts are semantically richer than those using window-based contexts; moreover, the paper states that there is no preference of one weighting feature over another one. Thus, we suggest to build a distributional semantic space for each corpus using dependency-based contexts and occurrence frequency as feature weight (distributional semantic space and feature weighting have been described before in section 1.1.2).

When distributional semantic space is available, measures are used to compute hypernymy score for each candidate ho-hr couple. In this experiment, we suggest to use four measures: one similarity measure (**Cosine**), two inclusion measures (**ClarkeDE** and **invCL**), and one informativeness measure (**SLQS**)—these measures and other measures have been described before in section 1.1.2.

Results

Table 4.2 shows the Average Precision (AP) for sequential patterns, DHPs, HPs, ExtHPs, and the unsupervised distributional methods using the four measures mentioned above. Values are shown for the three corpora and the four datasets. Globally, the results confirm that any set of patterns outperforms any unsupervised distributional approach using one of the four measures. The best results are achieved by the full set of learned sequential patterns (**SHPs + SHPs$^-$ + SHyPs**). However, unsupervised approaches based on inclusion measures (**ClarkeDE** and **invCL**) show better results on Weeds dataset. As stated by *Shwartz et el.* [**shwartz-etal-2017-hypernyms**], the reason is that the Weeds dataset comprises much more specific hypernyms for hyponyms, which is specifically well handled by using inclusion measures.

4.3 Supervised Methods for Hypernymy Detection

In this section, we describe our supervised method for hypernymy detection using our patterns (DHPs or sequential patterns). We then present the performed experiment in order to compare them to supervised distributional methods. Finally, further experiments are described

Corpus	Dataset	Unsupervised Measures				Other patterns			Learned patterns
		Cosine	ClarkeDE	invCL	SLQS	HPs	ExtHPs	DHPs	SHPs+SHP$^-$+SHyPs
Music	Music	0.253	0.375	0.382	0.194	0.374	0.382	0.390	**0.42**
English-1	BLESS	0.093	0.119	0.120	0.047	0.477	0.481	0.493	**0.531**
	EVALution	0.279	0.348	0.353	0.220	0.321	0.322	0.326	**0.372**
	Weeds	0.532	0.685	**0.689**	0.392	0.556	0.556	0.557	0.602
English-2	BLESS	0.090	0.103	0.105	0.059	0.531	0.526	0.525	**0.57**
	EVALution	0.277	0.341	0.347	0.209	0.330	0.329	0.329	**0.42**
	Weeds	0.530	0.683	**0.688**	0.393	0.570	0.571	0.567	0.594

Table 4.2: Results of Average Precision comparing DHPs and the learned sequential patterns to existing patterns and unsupervised methods using the four datasets and the three corpora.

to study the complementary between both types of methods and the generality of each type.

4.3.1 Our Pattern-based Supervised Method for Hypernym detection

Similar to Snow method [92], we suggest to build a classifier model for hypernymy detection. But rather than using the shortest dependency paths as features to train the classifier model, we use the learned sequential patterns from chapter 3. The method uses a set of training couples (labeled couples as hypernymy or not) and the sequential patterns to build a feature space that fit as input to machine learning technique for training the classifier model. After that, the model can be used for detecting hypernymy for new candidates couples.

Table 4.3 shows an example of the feature space that is used to train the model. In the table, rows refer to the set of couples, columns refer to the set of learned sequential patterns (**SHPs + SHP$^-$ + SHyPs**), and the value of a cell refers to the count of how often a couple has been extracted by the sequential pattern using a given corpus. For instance, couple$_1$ has been extracted 4 times by the sequential pattern SHP$_1$ and 2 times by the sequential pattern SHyP$_k$.

Features / Couples	SHP$_1$	...	SHP$_m$	SHP$^-{}_1$	...	SHP$^-{}_n$	SHyP$_1$	...	SHyP$_k$
couple$_1$	4	...	3	1	...	2	0	...	2
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.
couple$_L$	2	...	3	0	...	0	1	...	3

Table 4.3: The feature space of our supervised method based on the learned sequential patterns.

4.3.2 Experiment and Results

The experiment is performed to first evaluate the performance of our patterns for hypernymy detection and then compare it to those of supervised distributional methods. We also consider in the comparison Snow method [92]—a popular pattern-based method. The experiment is performed using the same corpora and datasets used to compare unsupervised methods in the previous section (see section 4.2.2). Using each dataset, 80% of couples are used for training models while the rest of the couples (20%) are used for testing the models.

Snow Method: Dependency Paths as Features

We have re-implemented the method described in [92]. Accordingly, given a dataset of n couples $\{(x_i, y_i)\}_{i=1}^{n}$, we first extract all shortest dependency paths connecting any (x_i, y_i) in a corpus; we consider only dependency paths that occur at least 5 times between any (x_i, y_i); we extend paths with satellite links to cover patterns like "such NP as NP". The paths are then used as features to train the classifier model (Snow method is described with more details in section 1.1.1).

Supervised Distributional Method using Word Embedding

To compare our supervised method with supervised distributional methods, we have re-implemented a distributional supervised method that relies on *word embedding* to represent the feature vector between (x_i, y_i) by *concatenating* their word embedding vector $(\vec{x_i} \oplus \vec{y_i})$ [9]. For this purpose, we use Word2Vec to train three word embedding models, one for each corpus of the three used corpora (more details about word embedding is provided in Section 1.1.2). They are built using Gensim python library [1] with CBOW and dimension equal to 300.

Results

Our supervised method and the other two methods (Snow and Embedding-based method) are then used to train the models. They are trained and evaluated by performing 10-fold cross-validation on each dataset using SVM with RBF kernel. Table 4.4 shows the precision, recall, and f-score across all folds for the three approaches using the three corpora and the four datasets. The results show that using patterns as features outperforms using dependency paths as features. Indeed, using patterns as features highly mitigates the sparsity issue of the Snow method.

1. https://radimrehurek.com/gensim/models/word2vec.html

Corpus	Dataset	Metrics	Pattern-based		Embedding
			Patterns as features	Dep paths as features	Concatenation
Music	Music	Pre	**0.84**	0.77	0.82
		Rec	0.34	0.27	**0.36**
		F1	0.48	0.4	**0.50**
English-1	BLESS	Pre	0.81	0.7	**0.95**
		Rec	**0.54**	0.45	0.44
		F1	**0.65**	0.55	0.60
	EVALution	Pre	0.74	0.5	**0.89**
		Rec	0.3	0.32	**0.67**
		F1	0.43	0.39	**0.77**
	Weeds	Pre	**0.84**	0.72	0.76
		Rec	0.25	0.25	**0.79**
		F1	0.39	0.37	**0.77**
English-2	BLESS	Pre	0.79	0.73	**0.94**
		Rec	0.45	0.4	**0.50**
		F1	0.57	0.52	**0.65**
	EVALution	Pre	0.8	0.41	**0.90**
		Rec	0.32	0.52	**0.65**
		F1	0.46	0.46	**0.75**
	Weeds	Pre	**0.77**	0.71	0.71
		Rec	0.34	0.28	**0.88**
		F1	0.47	0.4	**0.78**

Table 4.4: Results comparing our supervised method based on the extracted sequential patterns to other supervised methods using the four datasets and the three corpora.

However, the best results are mostly achieved by the classifier based on word embedding. Nevertheless, using patterns as features yields a precision close to the precision of the embedding and outperforms embedding on Weeds dataset.

4.3.3 The Complementarity of Pattern-based and Distributional Methods

As stated in the literature review, several works [64, 89] have considered pattern-based and distributional methods complementary and combined methods that integrate both types of methods have been proposed. Hereinafter, we propose to analyze the complementarity by looking to couples rightly classified at the same time by using patterns as features and embedding; and by looking to couples rightly classified by only one of them. For this purpose, we compute the percentage of hypernym couples detected by both approaches and the percentage of couples detected by one of them and not by the other (see Table 4.5). The results confirm that they are complementary i.e. one type of methods may detect distinct hypernym couples that may not be detected by the other type of methods; for instance, in English-1 corpus with BLESS dataset, 24.7% from the total hypernym couples are detected by using patterns as features and not detected by the embedding.

Corpus	Dataset	Exclusively Detected		Commonly Detected
		Patterns as features	Embedding-based	
Music	Music	17.1%	18.87%	17.0%
English-1	BLESS	24.7%	14.69%	29.2%
	EVALution	14.35%	51.82%	15.25%
	Weeds	7.02%	61.07%	17.95%
English-2	BLESS	24.19%	28.93%	20.7%
	EVALution	13.7%	46.46%	18.18%
	Weeds	7.2%	60.84%	26.81%

Table 4.5: Percentages of hypernym couples commonly and exclusively detected by both types of methods

Corpus	Dataset		Our method			Embedding-based		
	Training	Testing	Pre	Rec	F1	Pre	Rec	F1
English-1	BLESS	EVALution	0.60	0.28	**0.38**	0.61	0.02	0.04
	EVALution	BLESS	0.89	0.22	**0.36**	0.31	0.24	0.27
Engish-2	BLESS	EVALution	0.46	0.27	**0.34**	0.75	0.01	0.02
	EVALution	BLESS	0.76	0.18	**0.30**	0.57	0.11	0.18

Table 4.6: Performance across distinct datasets of classifiers using patterns as features and embedding.

4.3.4 The Generality of Pattern-based and Distributional Supervised Models

One of the main issues that should be addressed when training supervised models is the generality of the trained models i.e. the dependency of the model on the training data. In this experiment, we propose to study the generality of a pattern-based model trained based on our sequential patterns and a distributional model trained based on word embedding. In other words, we suggest studying the variation of the performance for both models when the models are used on a dataset otherwise the dataset used for training it.

For this purpose, we train both models on the BLESS dataset, test them on the EVALution dataset, and vice versa using English-1 and English-2 corpora. Table 4.6 shows the performance of all trained models. We can notice that the performance of embedding-based model is much worse in comparison with its performance shown in table 4.4; on the contrary, the performance of using patterns as features also decreases but much less than for embedding-based model. Hence, we can state that using patterns as features is expected to show a better general validity than embedding, probably because less dependent on the training dataset.

Moreover, Levy et al. [53] show recently that the good performance of supervised distributional approaches are because of lexical memorization. They show that mostly supervised dis-

tributional approaches learn whether y is a prototypical hypernym, regardless of x, rather than learning the hypernym relation between x and y. In this experiment, the usage of one dataset for training and another for testing avoids the lexical memorization in the trained model; thus, the results also confirm that the supervised distributional methods are highly affected by lexical memorization while pattern-based methods are only poorly affected.

4.4 Summary

In this chapter, we propose to evaluate and compare our formulated dependency patterns (**DHPs**) and our leaned sequential patterns (**SHPs + SHP$^-$ + SHyPs**) to unsupervised and supervised distributional methods in the task of hypernymy detection. For this purpose, we propose two pattern-based methods using our patterns: unsupervised and supervised.

The unsupervised pattern-based method uses a measure to compute a hypernymy score for a candidate ho-hr couple based on how often the couple has been extracted by the patterns. The method is then used to evaluate our patterns for hypernymy detection. It is also used to evaluate existing patterns: HPs and extHPs. Finally, unsupervised distributional methods based on Cosine, ClarkeDE, invCL, and SLQS measures are evaluated for hypernym detection. The results show that patterns perform better than unsupervised measures in most cases, and the learned sequential patterns perform better than the other types of patterns.

The supervised pattern-based method uses the sequential patterns as features to train a classifier model. The classifier model is evaluated and compared to Snow model (pattern-based) that is trained by following the original paper of *Snow et al.* [92], and to embedding-based model (distributional) that is trained based on word embedding. The results confirm that the pattern-based model trained using the sequential patterns performs better than the pattern-based model of Snow, but the best results are achieved by models that are based on word embedding.

In addition, we study the complementarity between the pattern-based model and the distributional model. The results confirm that combining these two kinds of approaches should lead to a considerable increase in the recall due to the ability of each of them to detect a large set of hypernymy couples that cannot be detected by the other one. Finally, We have also shown that using patterns as features leads to a classifier much less dependent on the training dataset than classifiers based on embedding.

CONCLUSIONS AND PERSPECTIVES

In this thesis, we provided new contributions to the task of hypernym relations extraction from texts. More specifically, our contributions are basically guided to improve hypernym patterns i.e. patterns to extract hypernym relations. Hypernym relations are semantic relationships useful to induce taxonomies considered the backbone of building ontologies.

In chapter 1, we first provided a detailed description of several approaches from the literature review to extract hypernym relations. These approaches are commonly categorized into two main types of approaches: pattern-based and distributional. The advantages and limitations of these approaches were also discussed in the chapter. Finally, we described approaches from the literature review to learn patterns using sequential pattern mining.

In chapter 2, we presented our first contribution to hypernym patterns. We studied the impact of reformulating Hearst's patterns, popular hypernym patterns, as dependency patterns instead of lexico-syntactic patterns. We first manually redefined Hearst's patterns as a set of dependency relations that are obtained by performing dependency parser on sentences. Dependency Hearst's patterns (DHPs) were then used to match sentences based on their dependency trees to extract hypernym relations. Finally, the result of their matching is compared to the result of matching traditional Hearst's patterns defined (HPs) to conclude the following:

— DHPs are more generic patterns than HPs which lead to a considerable improvement in terms of recall.

— In contrast to HPs, DHPs can extract hypernym relations from sentences where non-pattern words (words not included in the pattern) occur between the hyponym and hypernym.

— DHPs are better than HPs in matching complex and/or ambiguous sentences.

— DHPs are prone to make more errors than HPs due to parsing errors.

Chapter 3 introduced our second contribution to hypernym patterns. We proposed an approach to learn hypernym patterns from texts by integrating sequential pattern mining and dependency parsing. The approach consists of 3 phases:

1. learn sequential patterns that replace a given set of seed patterns to improve either their precision or recall without degrading the other one.

2. extend the seed pattern by learning new sequential hypernym patterns to improve their recall.
3. learn anti-hypernym patterns that should filter wrongly identified hypernym relations by the sequential patterns learned in phase 1 and 2.

Given DHPs as a given set of seed patterns, the approach was then used: (1) to learn sequential Hearst's patterns (SHPs) that replace DHPs, (2) to extend SHPs by learning new sequential hypernym patterns (SHyPs) that do not correspond to Hearst's patterns, and (3) to learn anti-hypernym sequential patterns that should filter wrongly extracted hypernym relations by SHPs and SHyPs. After that, all learned sequential patterns are combined together and used to extract hypernym relations from sentences. Finally, analyses were performed to study the impact of each phase on the performance by comparing the result of their extraction of hypernym patterns to the result of DHPs extraction. Based on the analyses, we conclude the following:

— SHPs are specific patterns of DHPs that improve precision by avoiding the matching of some sentences that were wrongly macthed by DHPs.

— SHyPs lead to a considerable improvement in terms of recall by matching correctly several sentences that were not matched by SHPs.

— Anti-hypernym patterns improve precision by filtering some sentences that were wrongly matched by SHPs and SHyPs.

In chapter 4, we provided further experiments and analyses to confirm the efficiency of our contributions. Hereafter, we listed these experiments and analyses:

— We first performed an experiment to compare the performance of our hypernym patterns (DHPs and sequential patterns) to the performance of existing lexico-syntactic patterns and unsupervised distributional methods in the task of hypernym detection.

— In the second experiment, we proposed a supervised method to detect hypernymy by using sequential patterns as features to train a hypernymy classifier model, and then its performance was compared to the performance of existing supervised methods: pattern-based and distributional.

— We performed an analysis based on the result of the previous experiment to study the complementary between our sequential pattern-based model and the distributional models.

— Another analysis was also performed to study the generality of our sequential pattern-based model in comparison to the generality of the distributional model.

— We also carried our a third experiment to study the generality of each sequential pattern across multiple corpora based on the weights associated to sequential patterns in a supervised linear model after training. Sequential patterns were used as features to train the model.

In the following, we listed our conclusions based on the performed experiments and analyses:

— In general, patterns perform better than unsupervised distributional approaches. And our hypernym patterns (DHPs and sequential patterns) outperform other existing patterns.

— Our supervised sequential pattern-based model performs better than the other pattern-based model, but the best results were achieved by the supervised distributional model.

— Pattern-based and distributional methods are complementary, and combining them should lead to a considerable increase the recall.

— The sequential pattern-based supervised model is much less dependent on the training dataset than the distributional supervised model.

— Although most of the learned sequential patterns in chapter 3 using the proposed approach are generic across multiple corpora, it exists several sequential patterns that are not generic—they are corpus dependent.

For future works, it would be interesting to perform further analysis for DHPs in order to better understand their behavior in complex and long sentences. In addition, it would be interesting to compare various dependency parsers in the process of formulation DHPs. I recommend using Spacy parser which is very fast in comparison to the Stanford parser used in our thesis.

Moreover, it would be interesting to work on combining pattern-based and distributional approaches in the task of hypernymy extraction, taking into account the strong dependency of the distributional approach on the training dataset. It should be noted here that these two types of approaches were already combined in several works for hypernymy detection; but to the best of our knowledge, they have not been combined for hypernymy extraction which is a more difficult and challenging task. We also intend to further improve the pattern-based hypernym extraction by applying the 3-phase approach to other sets of corpora and using richer seed patterns than Hearst's patterns. Because our SPM-based approach is quite generic, we expect its applicability to discover other types of relations (for instance, meronymy). We, therefore, plan to prove its applicability by conducting practical experiences as done for hypernym relations. We also think it would be interesting if we apply our SPM-based approach in other NLP tasks where pattern-based methods are applicable such as recognition of exchanges in social media and sentiment analysis.

Finally, we believe that our contributions in this thesis to pattern-based methods should open many research works in several applications and fields such as ontology learning, semantic recognition, and sentiment analysis. For instance, we believe it would be interesting if our pattern-based methods to extract hypernym relations are integrated into tools for ontology learning.

BIBLIOGRAPHY

[1] Rakesh Agrawal and Ramakrishnan Srikant, « Fast Algorithms for Mining Association Rules in Large Databases », *in*: *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1994, pp. 487–499, ISBN: 1-55860-153-8, URL: http://dl.acm.org/citation.cfm?id=645920.672836.

[2] Rakesh Agrawal and Ramakrishnan Srikant, « Mining Sequential Patterns », *in*: *Proceedings of the Eleventh International Conference on Data Engineering*, ICDE '95, Washington, DC, USA: IEEE Computer Society, 1995, pp. 3–14, ISBN: 0-8186-6910-1, URL: http://dl.acm.org/citation.cfm?id=645480.655281.

[3] Ahmad Issa Alaa Aldine et al., « Mining Sequential Patterns for Hypernym Relation Extraction », *in*: *Proceedings of the TextMine*, vol. 19.

[4] Ahmad Issa Alaa Aldine et al., « Redefining Hearst Patterns by using Dependency Relations », *in*: *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - Volume 2: KEOD,* INSTICC, SciTePress, 2018, pp. 148–155, ISBN: 978-989-758-330-8, DOI: 10.5220/0006962201480155.

[5] Maya Ando, Satoshi Sekine, and Shun Ishizaki, « Automatic Extraction of Hyponyms from Japanese Newspapers. Using Lexico-syntactic Patterns », *in*: *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal: European Language Resources Association (ELRA), May 2004, URL: http://www.lrec-conf.org/proceedings/lrec2004/pdf/80.pdf.

[6] Tuan Anh, Jung-Jae Kim, and See Ng, « Taxonomy Construction Using Syntactic Contextual Evidence », *in*: *EMNLP*, Jan. 2014, pp. 810–819, DOI: 10.3115/v1/D14-1088.

[7] Muhammad Asim et al., « A survey of ontology learning techniques and applications », *in*: *Database The Journal of Biological Databases and Curation* 2018 (Oct. 2018), pp. 1–24, DOI: 10.1093/database/bay101.

[8] Marco Baroni and Alessandro Lenci, « How We BLESSed Distributional Semantic Evaluation », *in*: *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, GEMS '11, Edinburgh, Scotland: Association for Computational Linguistics, 2011, pp. 1–10, ISBN: 978-1-937284-16-9, URL: http://dl.acm.org/citation.cfm?id=2140490.2140491.

[9] Marco Baroni et al., « Entailment above the word level in distributional semantics », *in*: *In EACL* (2012), pp. 23–32.

[10] Marco Baroni et al., « The WaCky wide web: a collection of very large linguistically processed web-crawled corpora », *in*: *Language Resources and Evaluation* 43.*3* (Sept. 2009), pp. 209–226, ISSN: 1574-0218, DOI: 10.1007/s10579-009-9081-4, URL: https://doi.org/10.1007/s10579-009-9081-4.

[11] N. Bechet et al., « Sequential pattern mining to discover relations between genes and rare diseases », *in*: *In IEEE Int. Symp. on Computer-Based Medical Systems (CBMS)* (2012), pp. 1–6.

[12] Nicolas Béchet et al., « Sequence Mining Under Multiple Constraints », *in*: *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, SAC '15, Salamanca, Spain: ACM, 2015, pp. 908–914, ISBN: 978-1-4503-3196-8, DOI: 10.1145/2695664.2695889, URL: http://doi.acm.org/10.1145/2695664.2695889.

[13] Chris Biemann, « Ontology Learning from Text: A Survey of Methods », *in*: *LDV Forum* 20 (2005), pp. 75–93.

[14] Eric Brill, « A Simple Rule-based Part of Speech Tagger », *in*: *Proceedings of the Third Conference on Applied Natural Language Processing*, ANLC '92, Trento, Italy: Association for Computational Linguistics, 1992, pp. 152–155, DOI: 10.3115/974499.974526, URL: https://doi.org/10.3115/974499.974526.

[15] Alexander Budanitsky, *Lexical Semantic Relatedness and Its Application in Natural Language Processing*, tech. rep., Technical Report CSRG-390, Computer SystemsResearch Group, University of Toronto, August, 1999.

[16] Paul Buitelaar, Philipp Cimiano, and Bernardo Magnini, « Ontology learning from text: An overview », *in*: *In Ontology Learning from Text: Methods, Applications and Evaluation* (2005), pp. 3–12.

[17] John A. Bullinaria and Joseph P. Levy, « Extracting semantic representations from word co-occurrence statistics: A computational study », *in*: *Behavior Research Methods* 39.*3* (Aug. 2007), pp. 510–526, ISSN: 1554-3528, DOI: 10.3758/BF03193020, URL: https://doi.org/10.3758/BF03193020.

[18] Jose Camacho-Collados et al., « SemEval-2018 Task 9: Hypernym Discovery », *in*: *Proceedings of the 12th International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, United States: Association for Computational Linguistics, 2018.

[19] Cristian Cardellino, *Spanish Billion Words Corpus and Embeddings*, Aug. 2019, URL: https://crscardellino.github.io/SBWCE/.

[20] P. Cellier, T. Charnois, and M. Plantevit, « Sequential patterns to discover and characterise biological relations », *in*: *In Computational Linguistics and Intelligent Text Processing, LNCS* (Springer, 2010), pp. 537–548.

[21] Philipp Cimiano and Johanna Völker, « Text2onto », *in*: vol. 3513, June 2005, pp. 227–238, DOI: 10.1007/11428817_21.

[22] Daoud Clarke, « Context-Theoretic Semantics for Natural Language: An Overview », *in*: *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, GEMS '09, Athens, Greece: Association for Computational Linguistics, 2009, pp. 112–119.

[23] Inderjit S. Dhillon, Subramanyam Mallela, and Rahul Kumar, « A Divisive Information Theoretic Feature Clustering Algorithm for Text Classification », *in*: *J. Mach. Learn. Res.* 3 (Mar. 2003), pp. 1265–1287, ISSN: 1532-4435, URL: http://dl.acm.org/citation.cfm?id=944919.944973.

[24] Euthymios Drymonas, Kalliopi Zervanou, and Euripides G. M. Petrakis, « Unsupervised Ontology Acquisition from Plain Texts: The OntoGain System », *in*: *Natural Language Processing and Information Systems*, ed. by Christina J. Hopfe et al., Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 277–287, ISBN: 978-3-642-13881-2.

[25] Oren Etzioni et al., « Web-scale Information Extraction in Knowitall: (Preliminary Results) », *in*: *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, New York, NY, USA: ACM, 2004, pp. 100–110, ISBN: 1-58113-844-X, DOI: 10.1145/988672.988687, URL: http://doi.acm.org/10.1145/988672.988687.

[26] Stefan Evert, Stefan Evert@uos, and De, « Corpora and collocations », *in*: *Corpus Linguistics: An International Handbook* (Mar. 2009), DOI: 10.1515/9783110213881.2.1212.

[27] David Faure and Claire Nédellec, « ASIUM Learning subcategorization frames and restrictions of selection », *in*: *10th European Conference on Machine Learning (ECML 98) – Workshop on Text Mining*, ed. by Yves Kodratoff, Chemnitz Allemagne, Avril 1998.

[28] David Faure and Thierry Poibeau, « First Experiments of Using Semantic Knowledge Learned by ASIUM for Information Extraction Task Using INTEX », *in*: *Proceedings of the First International Conference on Ontology Learning - Volume 31*, OL'00, Berlin, Germany: CEUR-WS.org, 2000, pp. 7–12, URL: http://dl.acm.org/citation.cfm?id=3053703.3053706.

[29] C. Fellbaum, « WordNet: An Electronic Lexical Database », *in*: *Cambridge, MA: MIT Press* (1998).

[30] Adriano Ferraresi et al., « Introducing and evaluating ukwac, a very large web-derived corpus of english », *in*: *In Proceedings of the 4th Web as Corpus Workshop (WAC-4*, 2008.

[31] Hermine Njike Fotzo and Patrick Gallinari, « Learning « Generalization/Specialization » Relations between Concepts: Application for Automatically Building Thematic Document Hierarchies », *in*: *Coupling Approaches, Coupling Media and Coupling Languages for Information Retrieval*, RIAO '04, Vaucluse, France: LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE, 2004, pp. 143–155, ISBN: 905450096.

[32] K. T. Frantzi, S. Ananiadou, and Hideki Mima, « Automatic recognition of multi-word terms:. the C-value/NC-value method », *in*: *International Journal on Digital Libraries* 3 (2000), pp. 115–130.

[33] Pablo Gamallo, Alexandre Agustini, and Gabriel Lopes, « Learning Subcategorisation Information to Model a Grammar with "Co-restrictions », *in*: (Jan. 2003).

[34] David Graff and Christopher Cieri, *English Gigaword LDC2003T05*, Philadelphia: Linguistic Data Consortium, 2003.

[35] Grolier, *Academic American Encyclopedia*, Danbury, Conneeticut: Grolier Electronic Publishing, 1990.

[36] Thomas R. Gruber, « A translation approach to portable ontology specifications », *in*: *Knowledge Acquisition* 5.*2* (1993), pp. 199–220, ISSN: 1042-8143, DOI: https://doi.org/10.1006/knac.1993.1008, URL: http://www.sciencedirect.com/science/article/pii/S1042814383710083.

[37] Lushan Han et al., « UMBC_EBIQUITY-CORE: Semantic Textual Similarity Systems », *in*: *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, Atlanta, Georgia, USA: Association for Computational Linguistics, June 2013, pp. 44–52, URL: https://www.aclweb.org/anthology/S13-1005.

[38] Z.S. Harris, « Mathematical Structures of Language », *in*: *Wiley (Interscience), New York* (1968), pp. 613–662.

[39] M. A. Hearst, « Automated discovery of wordnet relations », *in*: *WordNet: an electronic lexical database* (1998), pp. 131–152.

[40] M. A. Hearst, « Automatic acquisition of hyponyms from large text corpora », *in*: *In Proceedings of the 14th International Conference on Computational Linguistics* (1992), pp. 539–545.

[41] Andrew Hippisley, David Cheng, and Khurshid Ahmad, « The Head-modifier Principle and Multilingual Term Extraction », *in*: *Nat. Lang. Eng.* 11.*2* (June 2005), pp. 129–157, ISSN: 1351-3249, DOI: 10.1017/S1351324904003535, URL: https://doi.org/10.1017/S1351324904003535.

[42] Chung Hee Hwang, « Incompletely and Imprecisely Speaking: Using Dynamic Ontologies for Representing and Retrieving Information », *in*: *Technical, Microelectronics and Computer Technology Corporation (MCC*, 1999, pp. 29–30.

[43] Ahmad Issa Alaa Aldine et al., « DHPs: Dependency Hearst's Patterns for Hypernym Relation Extraction », *in*: Springer, Cham, Sept. 2018, URL: https://link.springer.com/chapter/10.1007/978-3-030-49559-6_11.

[44] Marie-Paule Jacques and Nathalie Aussenac-Gilles, « Variabilité des performances des outils de TAL et genre textuel. Cas des patrons lexico-syntaxiques », *in*: *T.A.L.* 47 (Jan. 2006), pp. 11–32.

[45] Xing Jiang and Ah-Hwee Tan, « CRCTOL: A Semantic-based Domain Ontology Learning System », *in*: *J. Am. Soc. Inf. Sci. Technol.* 61.*1* (Jan. 2010), pp. 150–168, ISSN: 1532-2882, DOI: 10.1002/asi.v61:1, URL: https://doi.org/10.1002/asi.v61:1.

[46] K.Sparck Jones, « Experiments in relevance weighting of search terms », *in*: *Information Processing Management* 15.*3* (1979), pp. 133–144, ISSN: 0306-4573, DOI: https://doi.org/10.1016/0306-4573(79)90060-8, URL: http://www.sciencedirect.com/science/article/pii/0306457379900608.

[47] L. Karoui, M. Aufaure, and N. Bennacer, « Contextual Concept Discovery Algorithm », *in*: *FLAIRS Conference*, 2007, pp. 460–465.

[48] Dan Klein and Christopher D. Manning, « Accurate Unlexicalized Parsing », *in*: *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, Sapporo, Japan: Association for Computational Linguistics, 2003, pp. 423–430, DOI: 10.3115/1075096.1075150, URL: https://doi.org/10.3115/1075096.1075150.

[49] Zornitsa Kozareva and Eduard Hovy, « A Semi-supervised Method to Learn and Construct Taxonomies Using the Web », *in*: *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, Cambridge, Massachusetts: Association for Computational Linguistics, 2010, pp. 1110–1118, URL: http://dl.acm.org/citation.cfm?id=1870658.1870766.

[50] Thomas K Landauer, Peter W. Foltz, and Darrell Laham, « An introduction to latent semantic analysis », *in*: *Discourse Processes* 25.*2-3* (1998), pp. 259–284, DOI: 10.1080/01638539809545028, eprint: https://doi.org/10.1080/01638539809545028, URL: https://doi.org/10.1080/01638539809545028.

[51] Els Lefever, « LT3: A Multi-modular Approach to Automatic Taxonomy Construction », *in*: *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, Denver, Colorado: Association for Computational Linguistics, June 2015, pp. 944–948, DOI: 10.18653/v1/S15-2157, URL: https://www.aclweb.org/anthology/S15-2157.

[52] Alessandro Lenci and Giulia Benotto, « Identifying Hypernyms in Distributional Semantic Spaces », *in*: *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task,*

and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation, SemEval '12, Montral, Canada: Association for Computational Linguistics, 2012, pp. 75–79, URL: http://dl.acm.org/citation.cfm?id=2387636.2387650.

[53] Omer Levy et al., « Do Supervised Distributional Methods Really Learn Lexical Inference Relations? », *in*: *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Denver, Colorado: Association for Computational Linguistics, May 2015, pp. 970–976, DOI: 10.3115/v1/N15-1098, URL: https://www.aclweb.org/anthology/N15-1098.

[54] Rinaldo Lima et al., « An Inductive Logic Programming-Based Approach for Ontology Population from the Web », *in*: *Database and Expert Systems Applications*, ed. by Hendrik Decker et al., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 319–326, ISBN: 978-3-642-40285-2.

[55] Dekang Lin, « An information-theoretic definition of similarity », *in*: *In ICML* (1998), pp. 296–304.

[56] Dekang Lin, « Dependency-Based Evaluation of Minipar », *in*: *Treebanks: Building and Using Parsed Corpora*, ed. by Anne Abeillé, Dordrecht: Springer Netherlands, 2003, pp. 317–329, ISBN: 978-94-010-0201-1, DOI: 10.1007/978-94-010-0201-1_18, URL: https://doi.org/10.1007/978-94-010-0201-1_18.

[57] Dekang Lin, « PRINCIPAR: An Efficient, Broad-coverage, Principle-based Parser », *in*: *Proceedings of the 15th Conference on Computational Linguistics - Volume 1*, COLING '94, Kyoto, Japan: Association for Computational Linguistics, 1994, pp. 482–488, DOI: 10.3115/991886.991970, URL: https://doi.org/10.3115/991886.991970.

[58] Francesca A. Lisi and Umberto Straccia, « A Logic-based Computational Method for the Automated Induction of Fuzzy Ontology Axioms », *in*: *Fundam. Inf.* 124.*4* (Oct. 2013), pp. 503–519, ISSN: 0169-2968, DOI: 10.3233/FI-2013-846, URL: http://dx.doi.org/10.3233/FI-2013-846.

[59] H. Liu and P. Singh, « ConceptNet — A Practical Commonsense Reasoning Tool-Kit », *in*: *BT Technology Journal* 22.*4* (Oct. 2004), pp. 211–226, ISSN: 1573-1995, DOI: 10.1023/B:BTTJ.0000047600.45421.6d, URL: https://doi.org/10.1023/B:BTTJ.0000047600.45421.6d.

[60] Anh Tuan Luu et al., « Learning Term Embeddings for Taxonomic Relation Identification Using Dynamic Weighting Neural Network », *in*: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, 2016, pp. 403–413, URL: http://aclweb.org/anthology/D/D16/D16-1039.pdf.

[61] Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D. Manning, « Generating typed dependency parses from phrase structure parses », *in*: *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)* (2006), pp. 449–454.

[62] Lluís Màrquez and Horacio Rodríguez, « Part-of-speech tagging using decision trees », *in*: *Machine Learning: ECML-98*, ed. by Claire Nédellec and Céline Rouveirol, Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 25–36, ISBN: 978-3-540-69781-7.

[63] Tomas Mikolov et al., « Distributed Representations of Words and Phrases and their Compositionality », *in*: *CoRR* abs/1310.4546 (2013), arXiv: 1310.4546, URL: http://arxiv.org/abs/1310.4546.

[64] Shachar Mirkin, Ido Dagan, and Maayan Geffet, « Integrating pattern-based and distributional similarity methods for lexical entailment acquisition », *in*: *In COLING and ACL* (2006), pp. 579–586.

[65] Michele Missikoff, Roberto Navigli, and Paola Velardi, « Integrated approach to Web ontology learning and engineering », *in*: *Computer* 35 (Dec. 2002), pp. 60–63, DOI: 10.1109/MC.2002.1046976.

[66] Emmanuel Morin, « Projecting Corpus-Based Semantic Links on a Thesaurus », *in*: *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, College Park, Maryland, USA: Association for Computational Linguistics, June 1999, pp. 389–396, DOI: 10.3115/1034678.1034739, URL: https://www.aclweb.org/anthology/P99-1050.

[67] Borst W N, « Construction of Engineering Ontologies », Thèse de doctorat, Institute for Telematica and Information Technology, University of Twente, Enschede, The Netherlands, 1997.

[68] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek, « PATTY: A Taxonomy of Relational Patterns with Semantic Types », *in*: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational*

Natural Language Learning, EMNLP-CoNLL '12, Jeju Island, Korea: Association for Computational Linguistics, 2012, pp. 1135–1145, URL: http://dl.acm.org/citation.cfm?id=2390948.2391076.

[69] Roberto Navigli and Paola Velardi, « Learning Word-class Lattices for Definition and Hypernym Extraction », *in*: *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, Uppsala, Sweden: Association for Computational Linguistics, 2010, pp. 1318–1327, URL: http://dl.acm.org/citation.cfm?id=1858681.1858815.

[70] Roberto Navigli and Paola Velardi, « Semantic interpretation of terminological strings », *in*: *Proc. 6th Int'l Conf. on Terminology and Knowledge Engineering (TKE 2002*, Springer-Verlag, 2002, pp. 95–100.

[71] Dat P. T Nguyen, Yutaka Matsuo, and Mitsuru Ishizuka, « Exploiting syntactic and semantic information for relation extraction from wikipedia », *in*: *In IJCAI07-TextLinkWS*, 2007.

[72] Ana Cristina Mendes Oliveira, Francisco Câmara Pereira, and Amílcar Cardoso, « Automatic Reading and Learning from Text », *in*: *International Symposium on Artificial Intelligence (ISAI)*, 2001.

[73] Sergio Oramas et al., « ELMD: An Automatically Generated Entity Linking Gold Standard Dataset in the Music Domain », *in*: *Language Resources and Evaluation Conference (LREC 2016)*, Portorož (Eslovenia), 23/05/2016 2016, pp. 3312–3317, URL: https://repositori.upf.edu/handle/10230/27835.

[74] Sergio Oramas et al., « Multi-label Music Genre Classification from Audio, Text, and Images Using Deep Features », *in*: *CoRR* abs/1707.04916 (2017), arXiv: 1707.04916, URL: http://arxiv.org/abs/1707.04916.

[75] Concepción Orna-Montesinos, « Words and patterns: lexico-grammatical patterns and semantic relations in domain-specific discourses », *in*: *Alicante Journal of English Studies / Revista Alicantina de Estudios Ingleses* 0.*24* (2011), pp. 213–233, ISSN: 2171-861X, DOI: 10.14198/raei.2011.24.09, URL: https://raei.ua.es/article/view/2011-n24-words-and-patterns-lexico-grammatical-patterns-and-semantic-relations-in-domain-specific-discourses.

[76] Alexander Panchenko et al., « TAXI at SemEval-2016 Task 13: a Taxonomy Induction Method based on Lexico-Syntactic Patterns, Substrings and Focused Crawling », *in*: *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, San Diego, California: Association for Computational Linguistics, June 2016, pp. 1320–1327, DOI: 10.18653/v1/S16-1206, URL: https://www.aclweb.org/anthology/S16-1206.

[77] Jian Pei et al., « PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth », *in*: *International Conference on Data Engineering*, Feb. 2001, pp. 215–224, ISBN: 0-7695-1001-9.

[78] Jeffrey Pennington, Richard Socher, and Christopher D. Manning, « Glove: Global vectors for word representation », *in*: *In EMNLP* (2014), pp. 1532–1543.

[79] Simone Paolo Ponzetto and Michael Strube, « Taxonomy induction based on a collaboratively built knowledge repository », *in*: *Artificial Intelligence* 175.*9* (2011), pp. 1737–1756, ISSN: 0004-3702, DOI: https://doi.org/10.1016/j.artint.2011.01.003, URL: http://www.sciencedirect.com/science/article/pii/S000437021100004X.

[80] Alan Ritter, Stephen Soderland, and Oren Etzioni, « What Is This, Anyway: Automatic Hypernym Discovery. », *in*: *AAAI Spring Symposium - Technical Report*, Jan. 2009, pp. 88–93.

[81] Stephen Roller, Douwe Kiela, and Maximilian Nickel, « Hearst Patterns Revisited: Automatic Hypernym Detection from Large Text Corpora », *in*: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 358–363, DOI: 10.18653/v1/P18-2057, URL: https://www.aclweb.org/anthology/P18-2057.

[82] Gerard Salton and Michael J. McGill, *Introduction to Modern Information Retrieval*, New York, NY, USA: McGraw-Hill, Inc., 1986, ISBN: 0070544840.

[83] E. Sang and Katja Hofmann, « Automatic Extraction of Dutch Hypernym-Hyponym Pairs », *in*: *LOT Occasional Series*, vol. 7, Jan. 2007, pp. 163–174.

[84] Enrico Santus et al., « Chasing hypernyms in vector spaces with entropy », *in*: *In EACL* (2014), pp. 38–42.

[85] Enrico Santus et al., « EVALution 1.0: an Evolving Semantic Dataset for Training and Evaluation of Distributional Semantic Models », *in*: *Proceedings of the 4th Workshop on Linked Data in Linguistics: Resources and Applications*, Beijing, China: Association for Computational Linguistics, July 2015, pp. 64–69, DOI: 10.18653/v1/W15-4208, URL: https://www.aclweb.org/anthology/W15-4208.

[86] Sebastian Schuster and Christopher D. Manning, « Enhanced English Universal Dependencies: An Improved Representation for Natural Language Understanding Tasks », *in*: *LREC*, 2016.

[87] Julian Seitner et al., « A Large DataBase of Hypernymy Relations Extracted from the Web », *in*: *LREC*, 2016.

[88] N. Sheena, Smitha M. Jasmine, and Shelbi Joseph, « Automatic Extraction of Hypernym and Meronym Relations in English Sentences Using Dependency Parser », *in*: *In Procedia Computer Science* (2016), pp. 539–546.

[89] Vered Shwartz, Yoav Goldberg, and Ido Dagan, « Improving Hypernymy Detection with an Integrated Path-based and Distributional Method », *in*: *CoRR* abs/1603.06076 (2016), arXiv: 1603.06076, URL: http://arxiv.org/abs/1603.06076.

[90] Vered Shwartz, Enrico Santus, and Dominik Schlechtweg, « Hypernyms under Siege: Linguistically-motivated Artillery for Hypernymy Detection », *in*: *CoRR* abs/1612.04460 (2016), arXiv: 1612.04460, URL: http://arxiv.org/abs/1612.04460.

[91] Daniel Dominic Sleator and David Temperley, « Parsing English with a Link Grammar », *in*: *CoRR* abs/cmp-lg/9508004 (1995), arXiv: cmp-lg/9508004, URL: http://arxiv.org/abs/cmp-lg/9508004.

[92] Rion Snow, Daniel Jurafsky, and Andrew Ng, « Learning syntactic patterns for automatic hypernym discovery », *in*: *MIT Press* (2005), pp. 1297–1304.

[93] Mohamed Sordo, Sergio Oramas, and Luis Espinosa-Anke, « Extracting Relations from Unstructured Text Sources for Music Recommendation », *in*: *Natural Language Processing and Information Systems*, ed. by Chris Biemann et al., Cham: Springer International Publishing, 2015, pp. 369–382, ISBN: 978-3-319-19581-0.

[94] Ramakrishnan Srikant and Rakesh Agrawal, « Mining generalized association rules », *in*: *Future Generation Computer Systems* 13.2 (1997), Data Mining, pp. 161–180, ISSN: 0167-739X, DOI: https://doi.org/10.1016/S0167-739X(97)00019-

8, URL: http://www.sciencedirect.com/science/article/pii/S0167739X97000198.

[95] Rudi Studer, V.Richard Benjamins, and Dieter Fensel, « Knowledge engineering: Principles and methods », *in*: *Data Knowledge Engineering* 25.*1* (1998), pp. 161–197, ISSN: 0169-023X, DOI: https://doi.org/10.1016/S0169-023X(97)00056-6, URL: http://www.sciencedirect.com/science/article/pii/S0169023X97000566.

[96] Erik Tjong Kim Sang and Katja Hofmann, « Lexical Patterns or Dependency Patterns: Which Is Better for Hypernym Extraction? », *in*: *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, Boulder, Colorado: Association for Computational Linguistics, June 2009, pp. 174–182, URL: https://www.aclweb.org/anthology/W09-1122.

[97] Peter D. Turney, « Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL », *in*: *Machine Learning: ECML 2001*, ed. by Luc De Raedt and Peter Flach, Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 491–502, ISBN: 978-3-540-44795-5.

[98] Paola Velardi, Paolo Fabriani, and Michele Missikoff, « Using Text Processing Techniques to Automatically Enrich a Domain Ontology », *in*: *Proceedings of the International Conference on Formal Ontology in Information Systems - Volume 2001*, FOIS '01, Ogunquit, Maine, USA: Association for Computing Machinery, 2001, pp. 270–284, ISBN: 1581133774, DOI: 10.1145/505168.505194, URL: https://doi.org/10.1145/505168.505194.

[99] Paola Velardi et al., « Evaluation of ontolearn, a methodology for automatic population of domain ontologies », *in*: *Ontology Learning from Text: Methods, Evaluation and Applications* (Sept. 2005).

[100] Jianyong Wang and Jiawei Han, « BIDE: Efficient Mining of Frequent Closed Sequences », *in*: *Proceedings of the 20th International Conference on Data Engineering*, ICDE '04, Washington, DC, USA: IEEE Computer Society, 2004, pp. 79–, ISBN: 0-7695-2065-0, URL: http://dl.acm.org/citation.cfm?id=977401.978142.

[101] Julie Weeds and David Weir, « A general framework for distributional similarity », *in*: *In EMLP* (2003), pp. 81–88.

[102] Julie Weeds, David Weir, and Diana McCarthy, « Characterising Measures of Lexical Distributional Similarity », *in*: *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, Geneva, Switzerland: Association for Computational Linguistics, 2004, DOI: 10.3115/1220355.1220501, URL: https://doi.org/10.3115/1220355.1220501.

[103] Julie Weeds et al., « Learning to distinguish hypernyms and co-hyponyms », *in*: *In COLING* (2014), pp. 2249–2259.

[104] Wilson Wong, Wei Liu, and Mohammed Bennamoun, « Ontology Learning from Text: A Look Back and into the Future », *in*: *ACM Comput. Surv.* 44.*4* (Sept. 2012), ISSN: 0360-0300, DOI: 10.1145/2333112.2333115, URL: https://doi.org/10.1145/2333112.2333115.

[105] Wilson Wong, Wei Liu, and Mohammed Bennamoun, « Tree-Traversing Ant Algorithm for term clustering based on featureless similarities », *in*: *Data Min. Knowl. Discov.* 15 (Oct. 2007), pp. 349–381, DOI: 10.1007/s10618-007-0073-y.

[106] Wentao Wu et al., « Probase: A Probabilistic Taxonomy for Text Understanding », *in*: *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, Scottsdale, Arizona, USA: ACM, 2012, pp. 481–492, ISBN: 978-1-4503-1247-9, DOI: 10.1145/2213836.2213891, URL: http://doi.acm.org/10.1145/2213836.2213891.

[107] Xifeng Yan, Jiawei Han, and Ramin Afshar, « CloSpan: Mining Closed Sequential Patterns in Large Datasets », *in*: *In SDM*, 2003, pp. 166–177.

[108] Zheng Yu et al., « Learning Term Embeddings for Hypernymy Identification », *in*: *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, Buenos Aires, Argentina: AAAI Press, 2015, pp. 1390–1397, ISBN: 978-1-57735-738-4, URL: http://dl.acm.org/citation.cfm?id=2832415.2832443.

[109] Marie Lisandra Zepeda-Mendoza and Osbaldo Resendis-Antonio, « Hierarchical Agglomerative Clustering », *in*: *Encyclopedia of Systems Biology*, ed. by Werner Dubitzky et al., New York, NY: Springer New York, 2013, pp. 886–887, ISBN: 978-1-4419-9863-7, DOI: 10.1007/978-1-4419-9863-7_1371, URL: https://doi.org/10.1007/978-1-4419-9863-7_1371.

APPENDIX A

EXTENDED SET OF HEARST'S PATTERNS (EXTHPS)

This appendix provides the extended set of Hearst's patterns used for comparing them to our patterns provided in Chapters 2 and 3. The extended set consists of 59 lexico-syntactic patterns collected from the literature review by Seitner et al. [87]. Table A.1 shows the 59 patterns and their precision as presented in the original paper.

Table A.1: The 59 lexico-syntactic patterns as presented in [87]

Nb	Pattern	Precision
1	NP_{ho} and other NP_{hr}	0.70
2	NP_{hr} especially NP_{ho}	0.19
3	NP_{hr} including NP_{ho}	0.44
4	NP_{ho} or other NP_{hr}	0.70
5	NP_{hr} such as NP_{ho}	0.58
6	NP_{ho} and any other NP_{hr}	0.76
7	NP_{ho} and some other NP_{hr}	0.54
8	NP_{ho} is a NP_{hr}	0.44
9	NP_{ho} was a NP_{hr}	0.39
	Continued on next page	

Table A.1 – continued from previous page

Nb	Pattern	Precision
10	NP_{ho} are a NP_{hr}	0.57
11	NP_{ho} were a NP_{hr}	0.42
12	NP_{hr} like NP_{ho}	0.17
13	such NP_{hr} as NP_{ho}	0.58
14	NP_{ho} like other NP_{hr}	0.31
15	NP_{ho}, one of the NP_{hr}	0.38
16	NP_{ho}, one of these NP_{hr}	0.13
17	NP_{ho}, one of those NP_{hr}	0.15
18	examples of NP_{hr} is NP_{ho}	0.33
19	examples of NP_{hr} are NP_{ho}	0.45
20	NP_{ho} are examples of NP_{hr}	0.20
21	NP_{ho} is example of NP_{hr}	0.36
22	NP_{hr} for example NP_{ho}	0.31
23	NP_{ho} is adjsup NP_{hr}	0.63
24	NP_{ho} are adjsup NP_{hr}	0.41
25	NP_{ho} is adjsup most NP_{hr}	0.63
26	NP_{ho} are adjsup most NP_{hr}	0.49
27	NP_{ho} which is called NP_{hr}	0.50
28	NP_{ho} which is named NP_{hr}	0.26
29	NP_{hr} mainly NP_{ho}	0.22
30	NP_{hr} mostly NP_{ho}	0.16
31	NP_{hr} notably NP_{ho}	0.28

Continued on next page

Table A.1 – continued from previous page

Nb	Pattern	Precision
32	NP_{hr} particularly NP_{ho}	0.19
33	NP_{hr} principally NP_{ho}	0.26
34	NP_{hr} in particular NP_{ho}	0.25
35	NP_{hr} except NP_{ho}	0.22
36	NP_{hr} other than NP_{ho}	0.44
37	NP_{hr} e.g. NP_{ho}	0.33
38	NP_{hr} i.e. NP_{ho}	0.29
39	NP_{ho}, a kind of NP_{hr}	0.18
40	NP_{ho}, kinds of NP_{hr}	0.45
41	NP_{ho}, a form of NP_{hr}	0.18
42	NP_{ho}, forms of NP_{hr}	0.33
43	NP_{ho} which look like NP_{hr}	0.13
44	NP_{ho} which sound like NP_{hr}	0.18
45	NP_{hr} which are similar to NP_{ho}	0.28
46	NP_{hr} which is similar to NP_{ho}	0.29
47	NP_{hr} example of this is NP_{ho}	0.25
48	NP_{hr} examples of this are NP_{ho}	0.18
49	NP_{hr} types NP_{ho}	0.17
50	NP_{ho} NP_{hr} types	0.12
51	NP_{hr} whether NP_{ho} or	0.13
52	compare NP_{ho} with NP_{hr}	0.16
53	NP_{hr} compared to NP_{ho}	0.17

Continued on next page

Table A.1 – continued from previous page

Nb	Pattern	Precision
54	NP_{hr} among them NP_{ho}	0.23
55	NP_{ho} as NP_{hr}	0.17
56	NP_{hr} NP_{ho} for instance	0.13
57	NP_{ho} or the many NP_{hr}	0.31
58	NP_{ho} sort of NP_{hr}	0.18
59	NP_{ho} sort of NP_{hr}	0.14

APPENDIX B

LEARNED SEQUENTIAL PATTERNS

In this appendix, we provide all learned sequential patterns in Chapter 3 using the three corpora: Music, English-1, and English-2. Table B.1 shows the all distinct learned sequential patterns corresponding Hearst's patterns (SHPs). In table B.2, we present all distinct learned sequential hypernym patterns (SHyPs) other than that correspond to Hearst's patterns. And finally, table B.3 shows the learned anti-hypernym sequential patterns (SHPs^{-}).

Table B.1: All distinct learned SHPs from the three corpora.

Nb	Sequential Pattern	Corpora	Type
1	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper)(NP, hyper_dep)]	Music	SHP$_{is-a}$
2	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, hyper, root<–, null_dep)(case–>)]	Music	SHP$_{is-a}$
3	[(NP, hypo, NN, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>, VBD)(NP, hyper)]	Music	SHP$_{is-a}$
4	[(NP, hypo, NN, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper)]	Music	SHP$_{is-a}$
5	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper)(case–>)(NP)]	Music	SHP$_{is-a}$

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
6	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper, root<–, null_dep)(NP)]	Music	SHP$_{is-a}$
7	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, hyper)(hyper_dep, acl:relcl<–)(NP, NN)]	Music	SHP$_{is-a}$
8	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, is_label, be_lemma, VBZ, cop–>)(NP, NN, hyper, root<–, null_dep)(hyper_dep, acl:relcl<–)]	Music	SHP$_{is-a}$
9	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper, band_lemma)]	Music	SHP$_{is-a}$
10	[(NP, hyper, genre_lemma, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)]	Music	SHP$_{such-as}$
11	[(NP)(IN, case–>, hyper_dep)(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)]	Music	SHP$_{such-as}$
12	[(case–>, hyper_dep)(NP, hyper)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, NN, hyper_dep, hypo, nmod:such_as<–)(NP)(NP)]	Music	SHP$_{such-as}$

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
13	[(NP, hyper)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)(hypo_dep, and_label, and_lemma, CC, cc<–)(NP)]	Music	$SHP_{such-as}$
14	[(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)(NP)(cc<–)]	Music	$SHP_{such-as}$
15	[(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, NN, hyper_dep, hypo, nmod:such_as<–)(NP)(NP)]	Music	$SHP_{such-as}$
16	[(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)(hypo_dep, and_label, and_lemma, CC)(NP)]	Music	$SHP_{such-as}$
17	[(case–>, hyper_dep)(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)(cc<–)]	Music	$SHP_{such-as}$
18	[(case–>, hyper_dep)(NP, hyper)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)(hypo_dep, and_label, and_lemma, CC, cc<–)]	Music	$SHP_{such-as}$

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
19	[(root<–, null_dep)(NP)(IN)(NP, hyper)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)]	Music	$SHP_{such-as}$
20	[(NP, NN)(case–>, hyper_dep)(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, NN, hyper_dep, hypo, nmod:such_as<–)]	Music	$SHP_{such-as}$
21	[(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)(NP)(CC)(NP)]	Music	$SHP_{such-as}$
22	[(case–>, hyper_dep)(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)(and_label, and_lemma, CC)]	Music	$SHP_{such-as}$
23	[(NP, NNS)(NP, hyper, NNS)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)]	Music	$SHP_{such-as}$

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
24	[(VBZ)(IN, case–>, hyper_dep)(NP, hyper, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)(and_label, and_lemma, CC, cc<–)]	Music	$SHP_{including}$
25	[(root<–, null_dep)(NP)(hyper_dep, amod–>)(NP, hyper, genre_lemma, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)(and_label, and_lemma, CC, cc<–)]	Music	$SHP_{including}$
26	[(NP)(NP, NN)(IN)(NP, hyper, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)(hypo_dep, and_label, and_lemma, CC, cc<–)]	Music	$SHP_{including}$
27	[(NP)(NP, NN)(hyper_dep, JJ, amod–>)(NP, hyper, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)]	Music	$SHP_{including}$
28	[(NP, NN)(IN, case–>, hyper_dep)(NP, hyper, genre_lemma, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)(and_label, and_lemma, CC, cc<–)]	Music	$SHP_{including}$

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
29	[(IN, case–>, hyper_dep, in_label, in_lemma)(NP, hyper, nmod:in<–)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)]	Music	$SHP_{including}$
30	[(VBZ, root<–, null_dep)(NP, hyper)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)]	Music	$SHP_{including}$
31	[(NP, NN)(VBZ)(hyper_dep, JJ, amod–>)(NP, hyper, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)]	Music	$SHP_{including}$
32	[(nsubj–>)(VBZ)(NP)(NP, hyper, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, nmod:including<–)(and_label, and_lemma, CC, cc<–)]	Music	$SHP_{including}$
33	[(NP, hyper)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, nmod:including<–)(hypo_dep, and_label, and_lemma, CC, cc<–)(conj:and<–)]	Music	$SHP_{including}$
34	[(several_label, several_lemma, JJ, amod–>)(NP, hyper, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)]	Music	$SHP_{including}$

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
35	[(VBZ)(NP, hyper)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)(hypo_dep, and_label, and_lemma, CC, cc<–)]	Music	$SHP_{including}$
36	[(VBN)(JJ, amod–>)(NP, hyper, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, nmod:including<–)]	Music	$SHP_{including}$
37	[(VBN)(IN)(NP, hyper, NNS)(case–>, including_label, include_lemma, VBG, hypo_dep)(hyper_dep, NP, hypo, NN, nmod:including<–)(hypo_dep, and_label, and_lemma, CC, cc<–)]	Music	$SHP_{including}$
38	[(NP, NN, hypo, dobj<–)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, hypo_dep, hyper, conj:and<–)]	Music	SHP_{other}
39	[(case–>, hypo_dep)(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, JJ, hyper_dep)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–)]	Music	SHP_{other}

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
40	[(root<–, null_dep)(NP, NN)(NP, NN)(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–)]	Music	SHP_{other}
41	[(NP)(case–>, hypo_dep)(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, hypo_dep, hyper, conj:and<–)]	Music English-1	SHP_{other}
42	[(NP, NN, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–, country_lemma)]	Music	SHP_{other}
43	[(NP, hypo, dobj<–)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–)]	Music	SHP_{other}
44	[(NP)(in_label, in_lemma, IN, case–>)(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–)]	Music	SHP_{other}

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
45	[(in_label, in_lemma, IN, case–>)(NP)(case–>)(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–)]	Music	SHP_{other}
46	[(in_label, in_lemma, IN, case–>)(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–, country_lemma)]	Music	SHP_{other}
47	[(NP)(case–>, hypo_dep)(NP, hypo)(and_label, and_lemma, CC, cc<–, hypo_dep)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, conj:and<–, hypo_dep, hyper)]	Music English-1	SHP_{other}
48	[(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, hypo_dep, hyper, conj:and<–)]	Music English-1	SHP_{other}
49	[(NP, hypo)(or_label, or_lemma, CC, cc<–, hypo_dep)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, hypo_dep, hyper, conj:or<–)]	Music English-1 English-2	SHP_{other}

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
50	[(NP, hypo)(CC, cc<–, hypo_dep, or_label, or_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, hypo_dep, hyper, conj:or<–)]	Music English-1 English-2	SHP_{other}
51	[(case–>, hypo_dep)(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–, country_lemma)]	Music	SHP_{other}
52	[(NP, hypo)(hypo_dep, and_label, and_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, NNS, hypo_dep, hyper, conj:and<–, genre_lemma)]	Music	SHP_{other}
53	[(NP, hypo)(and_label, and_lemma, CC, cc<–, hypo_dep)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, conj:and<–, hypo_dep, hyper)]	Music English-1	SHP_{other}
54	[(NP, hypo)(hypo_dep, or_label, or_lemma, CC, cc<–)(amod–>, other_label, other_lemma, JJ, hyper_dep)(NP, hypo_dep, hyper, conj:or<–)]	Music English-1 English-2	SHP_{other}
55	[(hyper, NP)(especially_lemma, advmod<–, hyper_dep)(dep<–, hypo, hyper_dep, NP)]	Music English-1 English-2	$SHP_{especially}$

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
56	[(such_label, such_lemma, JJ, amod–>, hyper_dep)(NP, hyper)(as_label, as_lemma, IN, case–>, hypo_dep)(NP, hypo, nmod:as<–)]	Music English-1 English-2	SHP$_{as}$
57	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper, root<–, null_dep)(nsubj–>)]	English-1	SHP$_{is-a}$
58	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper, country_lemma, root<–, null_dep)]	English-1	SHP$_{is-a}$
59	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, is_label, be_lemma, VBZ, cop–>)(NP, NN, hyper, country_lemma)]	English-1	SHP$_{is-a}$
60	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper)(acl:relcl<–)]	English-1	SHP$_{is-a}$
61	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(hyper_dep, amod–>)(NP, NN, hyper)]	English-1	SHP$_{is-a}$
62	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper)(that_label, that_lemma)]	English-1	SHP$_{is-a}$
63	[(NP, hypo, NN, nsubj–>, hyper_dep)(hyper_dep, is_label, be_lemma, VBZ, cop–>)(NP, NN, hyper)]	English-1	SHP$_{is-a}$
64	[(NP, hypo, NN, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, hyper, root<–, null_dep)]	English-1	SHP$_{is-a}$

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
65	[(NP, hyper)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)]	English-1 English-2	$SHP_{such-as}$
66	[(NP, hyper)(case–>, such_label, such_lemma, JJ, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hypo, hyper_dep, nmod:such_as<–)]	English-1 English-2	$SHP_{such-as}$
67	[(NP, hyper)(including_label, include_lemma, VBG, case–>, hypo_dep)(NP, hypo, nmod:including<–, hyper_dep)]	English-1 English-2	$SHP_{including}$
68	[(NP, hypo)(CC, cc<–, hypo_dep, or_label, or_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, hypo_dep, hyper, conj:or<–)(NP, NN)]	English-1	SHP_{other}
69	[(NP, NN, hypo)(CC, cc<–, hypo_dep, or_label, or_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, hypo_dep, hyper, conj:or<–)(NP)]	English-1	SHP_{other}
70	[(NP, NN, hypo)(CC, cc<–, hypo_dep, or_label, or_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, NN, hypo_dep, hyper, conj:or<–)(IN)]	English-1	SHP_{other}
71	[(NP)(NP, NN, hypo)(and_label, and_lemma, CC, cc<–, hypo_dep)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, NNS, conj:and<–, hypo_dep, hyper, organ_lemma)]	English-1	SHP_{other}

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
72	[(NP, NN)(NP, hypo)(and_label, and_lemma, CC, cc<–, hypo_dep)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, NNS, conj:and<–, hypo_dep, hyper, organ_lemma)]	English-1	SHP_{other}
73	[(NP, NN, hypo)(CC, cc<–, hypo_dep, or_label, or_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, NN, hypo_dep, hyper, conj:or<–, person_lemma)]	English-1	SHP_{other}
74	[(NP, hypo, NN, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, hyper)(NP)]	English-2	SHP_{is-a}
75	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, hyper, root<–, null_dep)(NP)]	English-2	SHP_{is-a}
76	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper)(IN)(NP, appos<–)]	English-2	SHP_{is-a}
77	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, hyper)(NP, NN, hyper_dep)]	English-2	SHP_{is-a}
78	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper, root<–, null_dep)(IN)]	English-2	SHP_{is-a}
79	[(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, is_label, be_lemma, VBZ, cop–>)(NP, NN, hyper)]	English-2	SHP_{is-a}
80	[(NP, hypo, NN, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>)(NP, NN, hyper)(IN)]	English-2	SHP_{is-a}

Continued on next page

Table B.1 – continued from previous page

Nb	Sequential Pattern (SHP)	Corpora	Type
81	[(NP, hypo)(CC, cc<–, hypo_dep, and_label, and_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, hypo_dep, hyper, NNS, conj:and<–)(NP, NN)]	English-2	SHP_{other}
82	[(NP, NN)(NP, hypo)(or_label, or_lemma, CC, cc<–, hypo_dep)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, NN, hypo_dep, hyper, conj:or<–)]	English-2	SHP_{other}
83	[(NP)(NP, hypo, NN)(CC, cc<–, hypo_dep, and_label, and_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, hypo_dep, hyper, NNS, conj:and<–)]	English-2	SHP_{other}
84	[(NP)(case–>)(NP, hypo, NN)(or_label, or_lemma, CC, cc<–, hypo_dep)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, NN, hypo_dep, hyper, conj:or<–)]	English-2	SHP_{other}
85	[(NP)(hypo_dep, case–>)(NP, hypo)(CC, cc<–, hypo_dep, and_label, and_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, hypo_dep, hyper, NNS, conj:and<–)]	English-2	SHP_{other}
86	[(NP, hypo, NN, brain_lemma)(CC, cc<–, hypo_dep, and_label, and_lemma)(other_label, other_lemma, JJ, amod–>, hyper_dep)(NP, hypo_dep, hyper, NNS, organ_lemma, conj:and<–)]	English-2	SHP_{other}

Table B.2: All distinct learned SHyPs from the three corpora.

Nb	Sequential Pattern (SHyP)	Corpora	Type
1	[(NP, hyper)(case–>, hypo_dep, IN, like_label, like_lemma)(NP, hyper_dep, hypo, nmod:like<–)]	Music English-1 English-2	SHyP$_{like}$
2	[(NP, hyper)(hyper_dep, VBG, acl<–, ranging_label, range_lemma)(case–>, hypo_dep, IN, from_label, from_lemma)(NP, hypo, nmod:from<–, range_dep)]	Music	SHyP$_{ranging}$
3	[(NP, hypo, NN, hyper_dep, compound–>)(hypo_dep, CC, cc<–)(JJ, other_label, other_lemma)(NP, hyper, NNS)]	Music	SHyP$_{other}$
4	[(NP, hyper, NN)(nsubj–>, hypo_dep, who_label, who_lemma, WP)(NP, hyper_dep, hypo, NN, acl:relcl<–)]	English-1	SHyP$_{who}$
5	[(NP, hyper, NN)(case–>, JJ, hypo_dep, other_label, other_lemma)(case–>, hypo_dep, than_label, than_lemma, IN)(NP, hyper_dep, hypo, nmod:than<–)]	English-1	SHyP$_{other-than}$

Table B.3: All distinct learned SHPs^{-} from the three corpora.

Nb	Sequential Pattern (SHP^{-})	Corpora	Type
1	[(nmod:poss–>, hypo_dep)(NP, NN, nsubj–>, hypo, hyper_dep)(hyper_dep, be_lemma, cop–>, is_label, VBZ)(NP, hyper)]	Music	SHP^{-}_{is-a}
2	[(NP, hyper)(NP)(case–>, JJ, such_label, such_lemma, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, hyper_dep, hypo, nmod:such_as<–)]	Music	$SHP^{-}_{such-as}$
3	[(NP, hyper)(case–>, JJ, such_label, such_lemma, hypo_dep)(IN, as_label, as_lemma, mwe<–, such_dep)(NP, NNS, hyper_dep, hypo, nmod:such_as<–)(IN, case–>)(NP)]	Music	$SHP^{-}_{such-as}$
4	[(NP)(NP)(NP, hypo, nsubj–>, hyper_dep, NNS)(hyper_dep, be_lemma, cop–>)(NP, hyper)]	English-1	SHP^{-}_{is-a}
5	[(compound–>)(NP, hypo, nsubj–>, hyper_dep)(hyper_dep, be_lemma, cop–>, VBZ)(NP, hyper)]	English-1	SHP^{-}_{is-a}
6	[(NP)(VB)(IN, case–>)(NP, hyper)(including_label, include_lemma, VBG, case–>, hypo_dep)(NP, hyper_dep, hypo, nmod:including<–)]	English-1	$SHP^{-}_{including}$
7	[(NP, hyper)(NP, NNS)(including_label, include_lemma, VBG, case–>, hypo_dep)(NP, hyper_dep, hypo, nmod:including<–)]	English-1	$SHP^{-}_{including}$
8	[(NP, nsubj–>, hypo, hyper_dep)(be_lemma, cop–>, hyper_dep)(IN)(NP, hyper)]	English-2	SHP^{-}_{is-a}

Continued on next page

Table B.3 – continued from previous page

Nb	Sequential Pattern	Corpora	Type
9	[(NP, nsubj–>, hypo, hyper_dep)(be_lemma, cop–>, hyper_dep)(case–>)(NP, hyper)]	English-2	SHP^{-}_{is-a}
10	[(NP, hyper, NN)(case–>, including_label, include_lemma, VBG, hypo_dep)(NP, NN, hyper_dep, hypo, nmod:including<–)]	English-2	$SHP^{-}_{including}$
11	[(NP, hyper, NN)(case–>, including_label, include_lemma, VBG, hypo_dep)(NP, hyper_dep, hypo, nmod:including<–)(NP)]	English-2	$SHP^{-}_{including}$

DOCTORAT BRETAGNE LOIRE / MATHSTIC

Université Bretagne Sud

Université Libanaise
École Doctorale Sciences et Technologies
Doyen

Titre : Contributions A La Représentation Et A L'apprentissage De Patrons D'hyperonymie Basées Sur Les Dépendances Syntaxiques Et La Fouille De Motifs Séquentiel

Mots clés : Patrons D'hyperonymies, Motifs Séquentiels, Fouille De Textes, Dépendances Syntaxiques, Apprentissage Supervisé, Patrons De Hearst

Résumé : L'hyperonymie est une relation sémantique entre un terme général et un terme spécifique. Les approches pour extraire de telles relations à partir des textes ont connu un regain d'intérêt en raison de la disponibilité d'une gamme de ressources textuelles ainsi que de leur rôle clé dans la construction de l'ontologie. Généralement, il existe deux types de ces approches: celles basées sur des patrons et celles distributionnelles. L'approche basée sur des patrons s'avère être plus intéressante que l'approche distributionnelle pour la construction de l'ontologie, ce qui est dû au fait que la première permet d'extraire de relations explicites à partir des textes avec une grande précision. Dans cette thèse, nous nous intéressons sur les patrons et nous introduisons une approche pour améliorer systématiquement les performances des patrons. Notre approche est basée sur le couplage d'une fouille des patrons séquentiels avec une représentation de patrons spécifique en utilisant les dépendances grammaticales pour l'apprentissage des patrons d'hyperonymie et d'anti-hyperonymie. Les résultats confirment que notre approche peut apprendre les patrons qui surpassent les approches non supervisées et une approche supervisée basée sur des patrons. Cependant, bien que les meilleures performances sont obtenues par certaines approches distributionnelles supervisées utilisant l'incorporation de mots, les patrons peuvent extraire de relations d'hyperonymie distinctes. Cela confirme que les deux types d'approches sont complémentaires. De plus, d'autres expériences confirment également que l'approche proposée tend à apprendre des patrons génériquement valides à travers divers corpora.

Title : Contributions to Hypernym Patterns Representation and Learning based on Dependency Parsing and Sequential Pattern Mining

Keywords : Hypernym Patterns, Sequential Patterns, Text Mining, Syntactical Dependencies, Supervised Learning, Hearst Patterns

Abstract : Hypernym relation is a semantic relationship between a specific term and a generic term of it. Approaches to extract such relations from texts have gained a large interest due to the huge availability of textual resources and their key role in ontology building. These approaches are commonly divided into two types: pattern-based and distributional. Patterns seem quite more interesting than distributional for building ontology due to their ability in extracting explicit relations from texts and their good precision. Thus, we focus on patterns and we describe an approach for systematically improving pattern performances. The approach is based on the coupled usage of sequential pattern mining and a specific pattern representation using grammatical dependencies to learn hypernym and anti-hypernym patterns. The results confirm that our approach can learn patterns that outperform unsupervised approaches and a supervised pattern-based approach. However, while the best performances are achieved by some supervised distributional approaches using word embedding, patterns can extract distinct hypernym relationships. This confirms that both types of approaches are complementary. Additionally, further experiments also confirm that the proposed approach tends to learn generically valid patterns across various corpora.