Bachelor's thesis Economics & Informatics

# Taxonomy Learning: a Survey of Approaches

*Author:*
Eveline van den Heuvel
291472

*Supervisor:*
Dr. Flavius Frăsincar

*Co-reader:*
Ludo Waltman

Econometric institute
Erasmus School of Economics
Erasmus University Rotterdam
April 9, 2009

**Abstract**

The last few years the field of taxonomy learning has become more and more important in computer science. However it also has relevance in economics, as it can be used for describing economic domains as finances, macroeconomics etc. There are numerous methods to create a taxonomy from the Web, corpus, or semantic lexicon. This thesis examines the different steps of the investigated methods dealing with the process of taxonomy learning. These methods will be explained and compared throughout this thesis. The main methods that are discussed are term extraction through linguistic, statistic and hybrid filtering, synonym extraction, computing distance between terms, and deriving taxonomic relations through clustering, classification and lexico-syntactic patterns.

# Contents

# Chapter 1

# Introduction, research question and methodology

## 1.1 Introduction

The Semantic Web is an evolving domain with large potential, defining the semantics of information for the purpose of comprehensive and transportable machine understandable information [1]. Machine understandable means that machines are able to understand the semantics of the data, and not just the syntax. Transportable means that interoperability for computers is possible without human intervention. This interoperation is possible, since machine understanding allows computers to perform operations over the information they have. There are many different research fields in the Semantic Web, including the field of taxonomy learning. A taxonomy is an important form of an ontology. Ontologies are schemas containing metadata, including a vocabulary of concepts. Each of the concepts is explicitly defined. An ontology has been defined by [2] and [3] as a formal, explicit specification of a shared conceptualization. "Formal" reflects that an ontology should be machine readable, "explicit" signifies that the concepts in the ontology are explicitly defined, "shared" indicates that there is a consensus about the knowledge and it is accepted by communities, and "conceptualization" implies that it is referring to an abstract model, containing concepts of real-world phenomena.

The main difference between taxonomies and ontologies is that taxonomies only have parent-child relationships (see Figure 1.1), while ontologies have a more extended range of relationships, such as is-part-of. Parent-child relationships are also called hypernymy/hyponymy, subtype/supertype, kind_of, or is-a relationships.

Taxonomy learning in the context of this thesis is concerned with the generation of knowledge from text. Taxonomy learning is the construction of taxonomies going through an automatized process. It facilitates the cre-

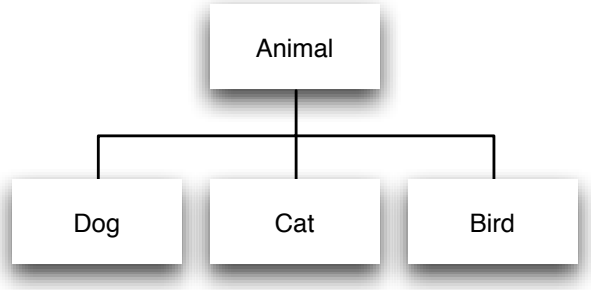


Figure 1.1: Example of a parent-child relationship.

ation of taxonomies, compared to the manual creation, which is very time consuming and needs the availability of a knowledge expert. Taxonomy learning is usually restricted to a given domain of interest and will thus model concepts and relations that are relevant to a particular field.

## 1.2 Motivation

A taxonomy is built in such a way that the taxonomy learning process follows a certain sequence of steps. There are many ways to execute these steps, and in practice these steps are sometimes merged or ignored. In this thesis the steps are classified in chapters, and methods to execute these steps are described in their subsections. A lot of papers describe only a few methods or steps, however there are no complete surveys of taxonomy learning methods. The aim of this thesis is to give a complete overview of the taxonomy learning process.

First we explain what a taxonomy is. There are some papers discussing the usefulness of applying taxonomies and taxonomy learning in economic domains, but not how they could be created in an automatic way. This survey explains why taxonomies and taxonomy learning are useful in economics and which specific things should be taken into account in an economic taxonomy, in contrast to a random taxonomy. In this thesis we explain:

1. What is taxonomy learning?
2. What is the economic relevance of taxonomy learning and how can taxonomy learning be used in the economic domain?
3. What are different methods to create a taxonomy?
4. Which of these methods can be used in economic taxonomy learning?

## 1.3 Research question

Taxonomy learning is important in economics, as it is useful to have a comprehensive hierarchy of all terms in a certain economic domain. This creates the possibility to analyze how a field is evolving or just to understand an existing field in economics.

A lot of research has already been done for parts of the taxonomy learning process, but in order to improve the quality of resulting taxonomies one needs to analyze and fine tune each of the steps. In this thesis we survey different approaches for taxonomy learning. Several methods for each step in the "value chain" of taxonomy learning are investigated and evaluated. The main question in this thesis is:

*How to create an economic taxonomy in an automatic way?*

## 1.4 Methodology

In order to be able to answer the main question in this thesis, first of all, a literature survey is conducted, to explore the field of taxonomy learning. The most important methods are identified. The different methods are evaluated, and compared. Subsequently the pros and cons of each method are discussed. This survey focuses on evaluating different methods that can be used in the process of taxonomy learning in the economic domain.

## 1.5 Outline

Chapter 2 gives an overview of other work in the domain of taxonomy learning. Chapter 3 explains the economic relevance of taxonomy learning and how the economic domain is used in taxonomy learning. Chapter 4 describes the ontology learning layer cake and different properties a taxonomy or a taxonomy learning process can have. Chapters 5 to 8 each explain a different step in the taxonomy learning process, and several methods to perform these steps. Chapter 9 explains how two taxonomies can be compared. Chapter 10 explains the pros and cons of each method, taking into consideration that an economic taxonomy will be the output. Finally, Chapter 11 gives conclusions that can be drawn from this survey and identifies possible future research directions.

# Chapter 2

# Related work

This chapter discusses existing research concerning the field of taxonomy learning. In [4] a taxonomy learning framework proceeds through a corpus, restricted to a certain application domain, and this is processed until a hierarchy with subtype/supertype relationships is formed. This processing is done by in the first place part of speech tagging [5], after this, applying word sense disambiguation [6], and finally calculating distances between concepts [7]. Some of these features are currently independently researched, which combined in an optimized way, could reveal a big benefit for taxonomy learning. This combination creates the possibility to construct better taxonomies, with a high similarity rate when being compared to a golden taxonomy. A golden taxonomy is a taxonomy manually made by some expert depicting the economic domain at a given moment.

The way a taxonomy is globally built, is described in the first four steps of the ontology learning layer cake [8]. This describes gathering of terms, finding their synonyms, and arranging them in a hierarchical way. The process of gathering terms, called terminology extraction [9], creates the possibility of selecting relevant words. There are linguistic, statistical and hybrid methods to do this. Another way of expanding the process of term extraction is adding the feature of named-entity recognition [10], which enables the recognition of names of people and places. A third option is to add a smoothing technique [11] to cope with data sparseness [12]. Data sparseness represents scarcity of data. Some words have relatively rare appearance, while rare occurrence of words can actually mean that they are important in specific domains. This could result in a poor performance of the used methods.

Other surveys in the field of ontology learning, such as [13], describe different methods about the process of learning ontologies from unstructured text. In this thesis ontology learning is categorized in clustering, (subdivided in distributional similarity and extraction patterns,) and classification, which are used for constructing ontologies from scratch and extending

existing ontologies, respectively.

Terms can be recognized as homonyms and synonyms. A homonym is a term with at least two meanings, but written and pronounced equally. A synonym has one meaning, but at least two different terms to express it. Both homonyms and synonyms can be processed by natural language processing (NLP). NLP means that human language is translated into a formal representation, in order to make it computer understandable. Homonyms can be found in different ways, for example in a dictionary or they can be processed through word sense disambiguation (WSD). Synonyms can also be found through a dictionary, but also latent semantic indexing (LSI) helps finding similar terms.

There are different ways of performing WSD [14]:

**Deep approach** tries to understand the structure of the text, such as "For your computer you can use hardware, not a type of rodent" and "rodents have sharp teeth, but are not a type of hardware". However there are not many available texts with this detailed kind of knowledge, so these deep approaches are not very useful in practice.

**Shallow approach** does not try to understand the text, but looks at surrounding words. If mouse has computer related terms nearby, it is probably the computer sense. This can easily be done with a training corpus of words, tagged with their word senses. This approach leads to superior results.

In WSD supervised and unsupervised methods exist. Supervised methods outperform unsupervised methods, but require large amounts of training data in order to yield reliable results. The manual creation of corpora with tagged senses, i.e., corpora in which the particular senses of all words have already been annotated is a labor-intensive task.

LSI [15] is used in information retrieval from text. The idea behind LSI is that it tries to find out which terms have an identical meaning. This is a well known method used for search engines. For example not only a search term is used, but also alternative descriptions of this term are considered. In this case, with a term such as "car" LSI will find that "automobile" is similar.

# Chapter 3

# Economic relevance

As mentioned in the previous chapters, taxonomy learning is important in economics, as it is useful to have a hierarchy of all terms in a certain economic domain. The question is: why is it useful?

In the first place, taxonomies and taxonomy learning can be valuable in economic domains, especially complex or specialized economic domains. When the concepts and the relationships between these concepts are explicitly defined, this will facilitate understanding this field of economics.

Secondly, it facilitates semantic searching for terms in complex economic domains. If a term does not have enough results, its hypernym is evaluated. These results will still be informative for the term that has initially been searched for. This is called query relaxation. In the case that there is no hypernym, it is easier to find out which other terms are closely related to this term through a taxonomy. As a result, searching systems can give an outcome which is more relevant, than it would be without a taxonomy. For example, when one searches for a Siamese cat, but it cannot find any results, through query relaxation information about cat can be returned, which is more reliable than when it returns nothing.

In the third place it is possible to understand the evolution in a certain economic domain. Taxonomy learning makes it easier to understand economic domains and to visualize occurring changes. When a domain is changing, this is easy to present through a taxonomy that offers a concise, but comprehensive representation of the concepts in the economic domain.

These three reasons mean a more simplified and faster way of researching the economic domain. This is another relevant issue in economics; the use of taxonomies in general makes things faster and simpler, which means less time consuming, and thus more financially attractive.

Real-world application, in which ontologies are used, have demonstrated their usefulness in [16], these fields are for example knowledge management, web commerce, e-business, etc.

In [4] a taxonomy has been built in the macroeconomic domain. It has

been created by going through different steps, and is based on a corpus of publications concerning macroeconomics. In order to measure the reliability of the automatically created taxonomy, a "golden" taxonomy had to be created as a reference. "Golden" means in this case human made. As a computer cannot be expected to give better performance on such a task than a human, the human performance serves as an upper bound. Figure 3.1 shows a part of this golden taxonomy.

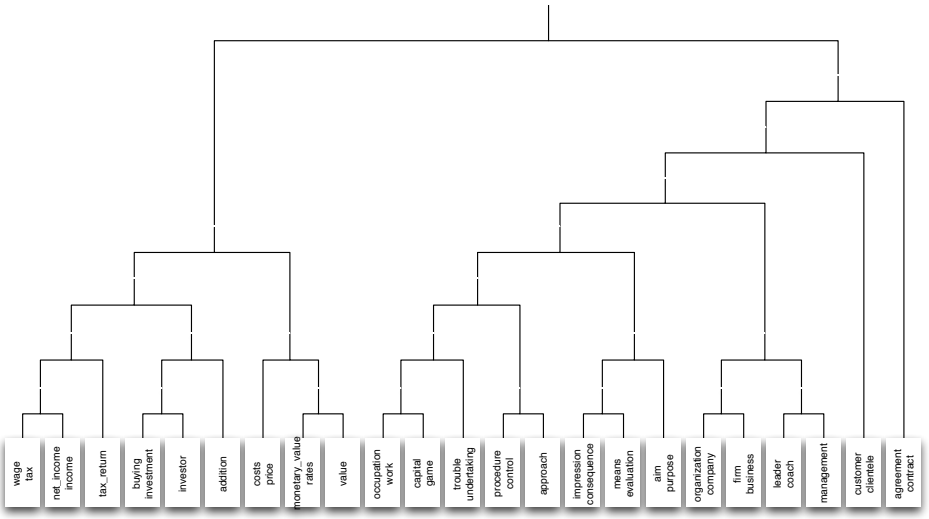


Figure 3.1: An excerpt of a handmade golden taxonomy in the macroeconomic domain

A different issue in taxonomy learning is that creating a taxonomy should not be too time consuming, otherwise it will not be interesting to create it. It is therefore important that a taxonomy should be built in a fast and efficient way.

# Chapter 4

# Taxonomy learning terminology

A taxonomy is a hierarchical structure that holds relevant hyponymy relations between concepts within the scope of a certain domain. This chapter gives a general overview of the taxonomy learning process. Some taxonomies, like WordNet [17], try to cover all words in a language, whereas a domain taxonomy, the taxonomy that is created in the case of this thesis, is more specific. When building an ontology different steps are followed. This is explained in the ontology learning layer cake, [8].

## 4.1 Ontology learning layer cake

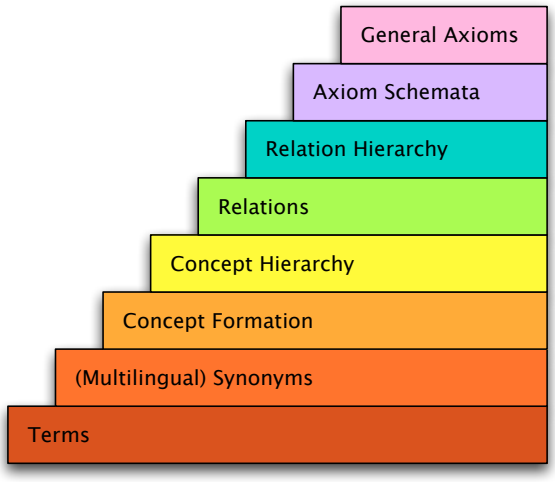


Figure 4.1: The Ontology Learning Layer Cake.

The ontology learning layer cake shows the different steps that have

been established in order to create an ontology according to [8], as given in Figure 4.1. Their description of the terms is as follows:

**Terms** Terms are relevant words or phrases gathered from corpora or the Web. They can be selected through different manners such as linguistic, statistical, or hybrid methods.

**(Multilingual) Synonyms** This is the part where synonyms are defined. In multi-language environments, translations are defined in this layer.

**Concept Formation** In this layer all terms and synonyms forming a concept together, are defined as a concept. A term can indicate a concept if its intent (formal definition) and extent (instances) are defined.

**Concept Hierarchy** The concepts defined in the former step, are organized in a hierarchical way so they will form a taxonomy. Thus the concepts are assigned taxonomic relations.

**Relations** This layer defines different possible relations existing between concepts. Also these can be semantically clustered, because they are actually the same, such as "teach" and "educate".

**Relations Hierarchy** The relations that have been found in the former layer, are arranged in a hierarchical way. For example a relation such as "hasRelative" which is the hypernym of "hasSibling".

**Axiom Schemata** This layer encompasses the axioms that apply in the ontology, such as disjointness between terms, e.g., disjointness between man and woman.

**General Axioms** A detailed description of the axioms is formed in such a way that it is valid in all cases. The axiom is defined by means of a formula, e.g.,
$\forall x(country(x) \rightarrow \exists y\ capital\_of(y,x) \wedge \forall z(capital\_of(z,x) \rightarrow y = z))$

In taxonomy learning, the first four layers are important for creating the taxonomy, the last four are added for creating an ontology. In practice the taxonomy learning methods usually follow these steps, but sometimes in a different sequence, merged or replenished with extra steps.

## 4.2 Knowledge rich vs. knowledge poor

Taxonomies sometimes happen to be based on existing knowledge, this method is called knowledge rich. A common example of this is when it is based on WordNet [17], which holds hyponymy relations between almost every word in the English language, and thus considered a top-level ontology or upper ontology [18]. An upper ontology is an ontology with general

concepts that are the same across each domain, so it does not belong to any specific domain.

When taxonomies are built from scratch, these taxonomies are knowledge poor. These taxonomies have no a priori semantic information. Relations between words will then be based on, e.g., number of times words jointly appear in one sentence.

## 4.3 Corpus-based vs. Web-based

Corpora are created by for example gathering publications within the scope of some field of study, gathering texts from specialized websites or exchanged documents from a virtual community [19]. The advantage of having a corpus with specialized documents is that a specific taxonomy can be created from this corpus. However, using a corpus may lead to data sparseness. Data sparseness means that there is not enough data to be processed in order to obtain reliable outcome. Data sparseness might occur when the corpus is too small, because this could cause that not every term is included in the corpus. A corpus that is too small results in having too less or irrelevant terms. There is however a smoothing technique which addresses the data sparseness problem [20]. Because the Web has billions of webpages, using the Web as one big text corpus can solve the problem of data sparseness. A disadvantage of using the Web is that it cannot be used to create a specialized taxonomy.

## 4.4 Labeled vs. unlabeled

Taxonomies can be labeled or unlabeled. Labeled taxonomies are easier for human understanding, but are difficult to create in an automatic way. A labeled taxonomy has each internal node in the hierarchy named, whereas an unlabeled taxonomy has not. An example of a labeled taxonomy is given in Figure 4.2. An unlabeled taxonomy has no names for internal nodes. An example of an unlabeled taxonomy is given in Figure 4.3.

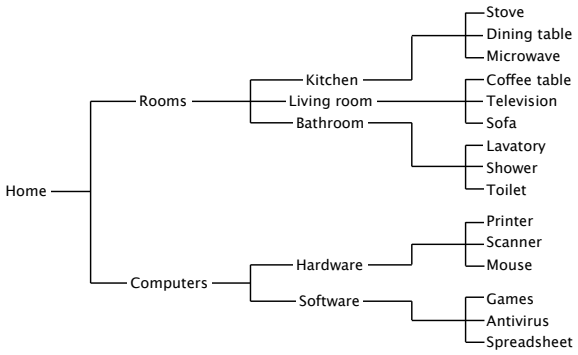


Figure 4.2: An example of a labeled taxonomy.

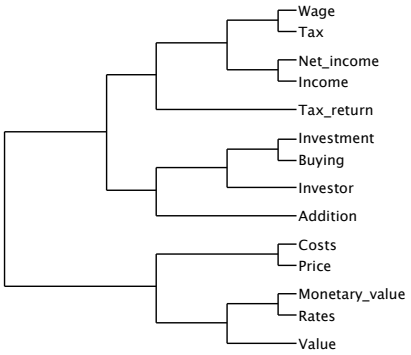


Figure 4.3: An example of an unlabeled taxonomy.

## 4.5 Taxonomy learning methodology

Different steps are followed when creating a computer–based taxonomy. In the next chapters each step to create a taxonomy are discussed. In taxonomy learning these steps generally are, based on the ontology learning layer cake:

1. Term extraction
2. Synonyms
3. Distance between terms
4. Taxonomic relations

# Chapter 5

# Term extraction

The process of finding relevant terms in a corpus or domain is generally called terminology extraction. According to [8], the process of term extraction can be subdivided in three approaches: the linguistic, statistical and hybrid approach. These approaches are discussed in the following sections.

## 5.1 Linguistic approach

Linguistic analysis can be done with a natural language processing (NLP) system. Such a system translates human language into a formal representation, in order to make it computer understandable.

Examples of NLP techniques are Part of Speech (PoS) tagging or morphological analyses. A PoS tagger is an application which is used to tag parts-of-speech, (e.g., nouns, verbs, adverbs, adjectives) in texts or corpora. One way of linguistic term extraction is when one decides to create a taxonomy which only holds nouns and verbs for example. Morphological analysis analyzes the gender (masculine, feminine or neuter), the number (singular or plural), the case (e.g. subjective, objective, or possessive), etc.

A relevant feature in the linguistic approach of term-extraction is named-entity recognition. Named-entities (NE) are traced from texts and categorized into classes such as names of persons, organizations, locations, dates, percentages, etc. Another feature is the extraction of multi-word noun phrases (NP), such as "swimming pool". The difference between a named-entity and a noun phrase is that named-entities are proper nouns, such as names of people, locations, etc., while noun phrases are common nouns. These NLP techniques could be helpful in performing term extraction.

Another way of linguistic term extraction, is using lexico-syntactic patterns [21], examples of these patterns are:

1. NE is a(n) NP  Keynes is an economist

2. NE, NP Keynes, economist in...

3. NP NE Economist Keynes...

For example, each NE that appears in a lexico-syntactic pattern with the word economist, is extracted.

## 5.2 Statistical approach

Statistical term extraction is a method that extracts its terms only from statistical analysis. This means that there is no lexical processing before or after the statistical analysis.

### 5.2.1 Tf–Idf

One way of gathering terms can be done by calculating the "term frequency – inverse document frequency" (Tf–Idf) [22]. Tf–Idf has many different computation possibilities, but it boils down to the following. The importance of a term is not only dependent on how often it occurs. In this case the weight of a term increases when it frequently appears in one document, but decreases when it frequently appears in more documents. This is to filter out common words such as "and", "the", etc. Equation 5.1 shows the equation used for Tf–Idf:

$$tfidf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \times \log \frac{|D|}{df_i} \tag{5.1}$$

In this equation, $tfidf_{i,j}$ is a statistical measure to measure the importance of a word in a document in the corpus. The higher the value, the more importance a term has. $\frac{n_{i,j}}{\sum_k n_{k,j}}$ is the Term Frequency; $n_{i,j}$ denotes how often the considered term $i$ appears in document $j$, and $\sum_k n_{k,j}$ is the number of occurrences of all terms in document $j$, i.e., the total number of words in $j$. $\log \frac{|D|}{df_i}$ is the Inverse Document Frequency. It indicates the general importance of the term, this is measured by the number of documents that term $i$ appears in. $|D|$ is the total number of documents in the corpus and $df_i$ is the number of documents in which term $i$ appears. Another example of Tf–Idf is leaving out $\sum_k n_{k,j}$, so term frequency only consists of $n_{i,j}$, the number of times the considered term $i$ occurs in document $j$.

It could occur that some important words appear in the corpus more often, which causes them to be filtered out. In order to address this problem, [23] has come up with a so called "stop–list". This means that one takes a sample from the terms that are filtered out, and, based on this list one composes a list with terms that should not be filtered out to make sure that they are still candidate terms, e.g., "economics" in the economic domain.

This stop–list can also be used for filtering out common words, such as "great", "year", "several", etc. We should note that adding this stop–list to the Tf-Idf method, makes this method hybrid, since it is a lexical process.

## 5.3 Hybrid approach

Hybrid filtering is the combination of statistical and linguistic filtering. A corpus is linguistically filtered before or after it has been statistically processed. Two major tasks in hybrid filtering are identifying candidate terms within a document (linguistic) and determining the relative importance of extracted terms (statistical).

### 5.3.1 Filtering method

In [19] a terminology extraction and filtering phase is described. This approach uses a corpus consisting of documents in a certain domain, e.g., economics. Also contrastive corpora are used, containing documents from different domains, e.g., medicine. After that, several filters are applied, and finally, the weight is computed to determine which terms have to be extracted.

This is further described in [24]. In the first place a linguistic processor is used to extract named-entities, e.g., ARIOSTO [25]. This results in a large list of candidate terms. The following five filters are applied in order to yield only the relevant terms.

1. Domain Pertinence: to measure whether a candidate term is approved, the domain pertinence score is calculated. This is to check whether a term $t$ is only relevant for the target corpus or also appears in contrastive corpora. $D_i$ is the target domain. The domain pertinence is measured as stated in Equation 5.2:

$$DP_{D_i}(t) = \frac{freq(t/D_i)}{max_j(freq(t/D_j))} \tag{5.2}$$

   This equation will also be applied to several contrastive corpora $D_j$. $DP_{D_i}(t)$ has a high value if the term frequently appears in the target corpus and less frequently in the contrastive corpora.

2. Domain Consensus: the second filter applies to the fact that a term should appear in several documents in the target corpus. If it appears only in one document it is considered not to be an important term. The importance of the term is computed by the domain consensus. The domain consensus has a high value if the term has an even probability distribution across the documents of the domain. A high value means that a term is selected as a candidate term. See Equation 5.3:

$$DC_{D_i}(t) = -\sum_{d_k \in D_i} norm\_freq(t, d_k) \log(norm\_freq(t, d_k)) \quad (5.3)$$

where $d_k$ is a document in $D_i$ and $norm\_freq$ is normalized frequency. The domain consensus will then be normalized for each term.

3. Lexical Cohesion: this method, introduced in [26], measures the cohesion among words in a term $t$ consisting of at least two words, such as compounds like "swimming pool".

$$LC_{D_i}(t) = \frac{n \cdot freq(t, D_i) \cdot \log(freq(t, D_i))}{\sum_{w_j \in t} freq(w_j, D_i)} \quad (5.4)$$

In Equation 5.4, $n$ is the number of words in the compound, $w_j$ represents each word in the term $t$. The lexical cohesion is high when the compound itself appears more often than the individual words in the compound.

4. Structural Relevance: when a term appears in the title, paragraph title, section title, or a term is highlighted, underlined or bold, this term can be given more significance. The equations from the first three filters are increased by an integer $k$. The size of $k$ depends on how important the user rates the structural relevance, compared to the former three filters.

5. Miscellaneous: in this filter misspellings are detected, by making use for example of WordNet or a dictionary.

The final weight of the term is combined as shown in Equation 5.5:

$$w(t, D_i) = \alpha \cdot DP + \beta \cdot DC + \gamma \cdot LC + k \quad (5.5)$$

where the default is: $\alpha = \beta = \gamma = \frac{1}{3}$, but the coefficients can be changed in order to provide more weight for a filter. The terms with a weight over a certain level, determined by the user, are extracted.

### 5.3.2 Chi-square

In this example by [27], named-entities are extracted from a domain corpus by a named-entity recognition system. After this, the corpus is searched for accompanying terms, called class candidates, through recognizing patterns in the text. This is called pattern extraction and is further described in Section 8.3. The class candidates, acquired through linguistic pattern analysis, subsequently go through a statistical filter, explained hereafter.

| | 1 | 2 |
|---|---|---|
| 1 | occurrence in domain corpus | occurrence in general corpus |
| 2 | # terms in domain corpus | # terms in general corpus |

Table 5.1: $2 \times 2$ contingency table.

Table 5.1 contains the information necessary for statistical analysis. Row 1 contains information about the number of times the considered term appears in the domain corpus, which contains specific texts, and the general corpus, which contains both common and specific texts. Row 2 contains information about the total number of words in these corpora. For this table the chi square statistic is calculated by Equation 5.6. For each term this equation is computed. Using this equation for this purpose is a novel approach. This equation is usually applied on a $2 \times 2$ contingency table.

$$\chi^2 = \frac{N(O_{11}O_{22} - O_{12}O_{21})^2}{(O_{11} + O_{12})(O_{11} + O_{21})(O_{12} + O_{22})(O_{21} + O_{22})} \tag{5.6}$$

The $N$ in this equation is the total number of considered terms, i.e., the sum of $O_{11}, O_{12}, O_{21}$ and $O_{22}$. Only terms with a $\chi^2$ value over a certain threshold are considered as a relevant class candidate. After that will be checked how often they co-occur with the named-entity. If it has a score that passes the threshold, it is established as a relevant class candidate.

### 5.3.3 C–value/NC–value

**C–value** Another method to perform hybrid term extraction is C–value/ NC–value [23]. In this method, the C–value aims to improve the extraction of (multi-word) terms, and the NC–value incorporates context information to the C–value method. First, the corpus needs to be processed by a PoS tagger and a linguistic filter. Since most terms consist of nouns and adjectives [28] and sometimes prepositions [29], other words in the corpus can be filtered out by the linguistic filter. The terms that are left are considered candidate terms. Subsequently Equation 5.7 is applied.

$$C\_value = \log_2 |a| f(a) \tag{5.7}$$

In this equation, $|a|$ is the positive value of the length of the candidate string, and $f(a)$ is the frequency of the occurring term. $|a|$ is moderated by applying the logarithm. The outcome of this function is a list of candidate terms. From this list, a set of top terms $m$ is obtained, namely the set of candidate terms with the highest C–value.

**NC–value** NC–values are calculated by adding a "context weighting factor" to the C–value, to consider the context of the candidate term. This is useful, because a context word might be a word that is used to introduce or signify the term. Each word (noun, adjective or preposition) that appears along with a top term in a sentence is considered a context word. Context words $w$ are each assigned a weight, which is computed by dividing the number of times the context word occurs along with a top term $t(w)$ in one sentence, by the total number of top terms $m$, as is stated in Equation 5.8.

$$weight(w) = \frac{t(w)}{m} \tag{5.8}$$

The assumption is that, the higher this number, the higher the chance that this word is actually related to a top term. Therefore, the $m$ has been set as a denominator, to be able to express the percentage of top terms the context word has appeared with.

Subsequently, the context information factor can be computed for each term. It represents additional weight for a candidate term due to the context words it appears with.

$$cif(a) = \sum_{b \in C} f_a(b) weight(b) \tag{5.9}$$

In this equation, $a$ is the candidate term, $C$ is the set of context words, $b$ a context word in $C$, $f_a(b)$ the number of times $b$ appears along with $a$ in a sentence, and $weight(b)$ is the weight of context word $b$.

The last step is to calculate the NC–value. It combines the C–value and the so called context information factor. The NC–value represents the importance of a term within a document.

$$NC\_value(a) = 0.8 C\_value(a) + 0.2 cif(a) \tag{5.10}$$

The values of 0.8 and 0.2 have been chosen after the authors did several experiments and made comparisons of the results, which yield the highest precision on the terms.

# Chapter 6

# Synonyms

The second layer of the ontology learning layer cake is "synonyms". In this layer, it needs to be found out which terms have the same, or nearly the same meaning. These synonyms can be added to terms with similar meaning in the taxonomy in order to extend it. The easiest way to find out about synonyms is searching for the term in WordNet. WordNet has a big collection of synonym sets (synsets) for word senses, however, first we need to determine which particular sense of the word we need. In order to know which synset is actually needed, we need to apply a WordNet related form of word sense disambiguation (WSD), for example structural semantic interconnections (SSI). The following section elaborates on WSD and SSI.

## 6.1 Word sense disambiguation

WSD, sometimes referred to as semantic disambiguation [30], is the process of identifying the meaning of a word from a set of possible meanings (word senses). Most words have more than one sense. For example the word "mouse". Two distinct senses are [31]:

1. "Any of numerous small rodents, such as the common house mouse, characteristically having a pointed snout, small rounded ears, and a long naked or almost hairless tail."
2. "A hand-held, button-activated input device for a computer that, when rolled along a flat surface, directs an indicator to move correspondingly about a computer screen."

It is useful to determine the sense of a term in a sentence. As a result, it will be clear in the taxonomy which specific sense the considered term has, especially for taxonomies with a clarifying function in a certain area of knowledge, e.g. economics. When determining the sense of a term, the text is usually searched for other words in the sentence and subsequently, the decision is made to which of these words it is related. For example:

1. The mouse has a long tail.

2. The computer was not responding to the mouse.

In these sentences, the senses are clearly different. However, it is not always that easy to determine the word sense. For example with the sentence "The dogs bark at the tree". WSD registers the word bark along with "dog", but also with "tree", so this could both be the sound that a dog makes and the outer covering of a tree. This is one of the difficulties that WSD encounters, which could lead to incorrect data. Another problem is the fact that terms and their senses are differently judged by people, which makes it more difficult to verify which algorithm is correct and which one is not.

### 6.1.1 Structural semantic interconnections

One WSD approach involves exploitation of structural semantic interconnections (SSI) [32] discovered in a corpus. In this approach, a set of words is considered. Each word in this set of words is associated with at least one word sense. When these senses are subsequently represented as vertices in a graph, edges in this graph represent (weighted) semantic interconnections between these word senses, determined using a lexical knowledge base such as WordNet. Using the HITS [33] algorithm, the degree of relevance, a so-called confidence factor, is determined for each vertex, hereby taking into account the degree of connectivity conveyed towards these respective vertices. Given this representation of the context, the most appropriate sense is determined for each word in the considered set by selecting the word sense maximizing the confidence factor. The senses that have been determined for each word, belong to a unique synset in WordNet. This synset contains several synonyms. With this method, the synonyms can be determined through WSD.

## 6.2 Latent semantic indexing

LSI [15] is used in information retrieval from text. The idea behind LSI is that it tries to find out which terms have an identical or nearly identical meaning, and which terms are distinct. This is a well known method used for search engines. That is because not only a search term is used, but also alternative descriptions of this term are considered. For example, with a term such as "car", LSI will determine that "automobile" is similar. The implementation of LSI is as follows: a so called "term by document matrix" is created, showing the number of times a word appears in a specific document. Any rectangular matrix can be decomposed into the product of three other matrices as shown in Equation 6.1.

$$X = T_0 S_0 D_0' \tag{6.1}$$

$T_0$ and $D_0$ have orthonormal columns and $S_0$ is diagonal. This is called singular value decomposition (SVD) of X. A property of SVD is that the diagonal elements of $S_0$ all have positive values and are ordered in decreasing magnitude. The dimensions are reduced by, instead of taking the $N$ original index terms, taking the $k < N$ most important values from the diagonal matrix. In case k = N, the original matrix is reconstructed, which is not very useful, because it might contain noise, which we wanted to remove by making use of SVD in the first place. The remaining values, outside the selected ones, are set to zero. In order to simplify the representation, the zero rows and columns of $S_0$ can be removed and the corresponding columns of $T_0$ and rows of $D_0$, which yields new matrices $S$, $T$, and $D$.

The chosen value of $k$ should be large enough to comprise the real data and small enough to lose the noise. The authors chose a value of $k$ yielding a good performance with respect to calculation speeds. The product of the resulting matrices is a matrix $\hat{X}$, approximately equal to $X$, as shown in Equation 6.2.

$$X \approx \hat{X} = TSD' \tag{6.2}$$

The last step is to find out how similar two terms are, which is in this case, comparing two rows in matrix $\hat{X}$. Comparing two rows in matrix theory is usually done through computing the dot product of two rows, or applying the dot product of $\hat{X}$ and $\hat{X}'$. In this matrix a high value between two terms means high similarity and a low value means dissimilarity.

# Chapter 7

# Distance between terms

This step discusses how the lexical distance between terms can be calculated, and is essential for the next step, semantic relations, discussed in Chapter 8. This is not a separate layer in the ontology learning layer cake, but for better understanding and because it is a subject on its own, it has been placed in a separate chapter. One should note that the distance between terms is based on a similarity measure. The smaller the distance, the higher the similarity. These two terms should not be mixed up.

To calculate the distance between terms, several similarity measures have been proposed, such as the path measure, or a variety of similarity measures described in [7]. In these measures, a difference is made between a corpus-based and a Web-based method. Also their methods are divided into knowledge rich, using WordNet, and knowledge poor, not using WordNet. The methods are discussed in the following sections.

## 7.1 Path measure

This knowledge rich method is based on the knowledge of WordNet. It calculates the distance between two considered words, based on their distance in WordNet. Equation 7.1 shows how this is calculated.

$$Sim_{PATH} = -\log(\frac{length(w_1, w_2)}{2D}) \tag{7.1}$$

In this equation, $length(w_1, w_2)$ is the distance between $w_1$ and $w_2$ expressed in total number of words on the shortest hyponym/hypernym WordNet path between them. $D$ is the maximum depth between the common father, and the deepest child of $w_1$ and $w_2$ . This method has been proposed by [34].

## 7.2 Window-based method

The window-based method considers a window of words, i.e., a user defined number of words in sequence. This method checks whether two considered words appear in this window together. The frequencies are entered in Table 7.1. $f_{w_i,w_j}$ means that word $i$ and $j$ appear together in one window, $f_{w_i,\neg w_j}$ means that word $i$ appears in the window and $j$ does not, etc.

| | $w_i$ | $\neg w_i$ |
|---|---|---|
| $w_j$ | $f_{w_i,w_j}$ | $f_{\neg w_i,w_j}$ |
| $\neg w_j$ | $f_{w_i,\neg w_j}$ | $f_{\neg w_i,\neg w_j}$ |

Table 7.1: Frequencies of $w_i$ and $w_j$ in a window.

The PMI (Pointwise Mutual Information) method, as shown in Equation 7.2, calculates the independence of $w_i$ and $w_j$, based on their occurrence.

$$Sim_{PMI}(w_1, w_2) = \log \frac{P(w_1, w_2)}{P(w_1)(w_2)} \tag{7.2}$$

In this method, $P(w_1, w_2)$ is the chance that both $w_1$ and $w_2$ appear in a window. $P(w_1)P(w_2)$ is the chance that only $w_1$ appears, multiplied by the chance of the appearance of only $w_2$ in a window. These chances are calculated by dividing the word frequency by the total number of windows. The total number of windows $W$ is calculated as shown in Equation 7.3.

$$W = n + t - 1 \tag{7.3}$$

In this equation, $n$ is the corpus size and $t$ is the length of the window. This equation can be clarified with an example: the corpus consists of 3 words: A, B and C. The window size is 2. This means that there are 4 windows, namely $.A$, $AB$, $BC$, and $C.$, so $W = 3 + 2 - 1 = 4$.

## 7.3 Document co-occurrence similarity

The second knowledge poor method is called document co-occurrence similarity. This method assumes that terms appearing in the same document are similar. Equation 7.4 is used to calculate this:

$$Sim_{COLDOC} = \frac{2df(w_1, w_2)}{df(w_1) + df(w_2)} \tag{7.4}$$

where $df(w_i)$ is the number of documents in which $w_i$ occurs.

## 7.4 Web-based method

In order to address the problem of data sparseness, [7] proposes a Web-based method, that uses the Web, which consists of an enormous amount of webpages. Google has indexed $10^{10}$ webpages, to give an indication of how big the Web is. The method the authors use is almost the same as the equation used in the window-based method. The difference is that it is not applied on a corpus, but on the Web. Words are entered in a search engine, such as Google. The outcome is based on the number of returned pages by entering one or two words. Equation 7.5 describes this method.

$$Sim_{Web\_PMI} = \log \frac{\frac{N(w_1 \cap w_2)}{N_{WEB}}}{\frac{N(w_1)N(w_2)}{N_{WEB}N_{WEB}}} \tag{7.5}$$

where $N_{WEB}$ is the total number of pages indexed by the search engine.

# Chapter 8

# Taxonomic relations

This section describes the way a taxonomic relation between terms is determined. There are various ways to determine the taxonomic relation. In [35], methods for realizing this part of taxonomy learning are classified into three groups.

- Statistics-based extraction using clustering
- Statistics-based extraction using classification
- Relation extraction using lexico-syntactic patterns

## 8.1 Statistics-based extraction using hierarchical clustering

Once the distances between terms are calculated, as explained in Chapter 7, we need to find out how the taxonomic relation can be determined by hierarchical clustering. In the case of statistics-based extraction using hierarchical clustering, there is no existing hierarchy at the start, the taxonomy is built from scratch. The considered terms are placed in clusters in a hierarchical way, based on their similarity.

**Hierarchical linkage** Three methods to create clusters are single, complete, and average linkage. These algorithms determine the distance between two clusters, by using the terms that are associated with the clusters in question. At the beginning, all clusters consist of one term. During the hierarchical linkage process the small clusters are linked until they form one big cluster, the root.

**Single linkage** computes the distance of the two nearest terms for each possible pair of clusters.

**Complete linkage** computes the distance of the two furthest terms for each possible pair of clusters.

**Average linkage** takes the mean distance between all possible pairs of terms for each possible pair of clusters.

The pair of clusters with the shortest distance, i.e., the clusters that are most similar according to this measure, are linked. This process continues until all terms have formed one big cluster altogether.

**Formal concept analysis** Formal concept analysis (FCA) has been introduced by [36]. FCA can be used as a concept clustering method [37]. Objects, attributes and their incidence relation, i.e., "object has attribute" relation, play an important role in FCA. Objects, attributes and their incidence relation combined form a *formal context*. The formal context is the total context that is used for the analysis. The next step is to describe the *formal concept*. To explain this, the example of [38] is used.

In table 8.1 a simple formal context is shown. The objects are lion, finch, eagle, hare and ostrich. The attributes are preying, flying, bird and mammal. The table describes for some objects, in this case animals, what attributes (properties) they have.

| ANIMALS | preying | flying | bird | mammal |
|---|---|---|---|---|
| LION | × | | | × |
| FINCH | | × | × | |
| EAGLE | × | × | × | |
| HARE | | | | × |
| OSTRICH | | | × | |

Table 8.1: Formal context.

To explain what a formal concept is, the following steps have to be followed.

1. Start with an object, e.g., FINCH.
2. Determine what its attributes are, in this case flying and bird.
3. Determine which are the objects that also include the attributes that have been found in the previous step. These are in this case FINCH and EAGLE.

Subsequently there are two different sets. Set A consists of the objects identified in step 3, FINCH and EAGLE. Set B consists of the identified attributes in step 2, flying and bird. A is the set of all objects that have

attributes of set B and B is the set of attributes which are valid for the objects of A. Each pair (A,B) is a formal concept. Set A is called *extent*, set B is called *intent* of the concept.

There is a natural hierarchical order between the concepts, namely the taxonomic relation, e.g, preying flying bird is a child of flying bird. Like in the example of FINCH, for each object $G$ an object concept (A,B) can be constructed and for each attribute $m$ an attribute concept can be constructed. The line diagram in Figure 8.1 can be created with table 8.1.

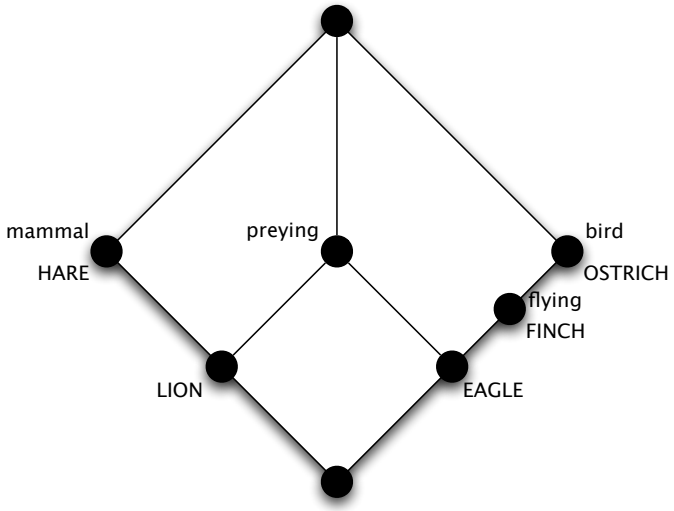


Figure 8.1: Formal concept.

The line diagram consists of lines, circles and all objects and attributes from the formal context. The circles represent the concepts. The following rule applies to all line diagrams. An object has an attribute if and only if there is an upward path from the object to the attribute, e.g., EAGLE has an upward path to attributes preying, flying and bird. This means that for each concept the extent and the intent can be found by finding all objects below and all attributes above the circle respectively.

## 8.2 Statistics-based extraction using classification

In contrast to clustering, which is done in the previous section, the main idea of classification is based on refining existing hierarchies. In this case, new terms are classified and the existing hierarchy is refined by placing the new terms in a specific position in the taxonomy. Relevant terms in this case are for example new words that originate in a language or domain.

### 8.2.1 Classification with corpus and NE tagger

An application of statistics-based extraction using classification, is done in [27]. It starts with a domain corpus, a general corpus and a named-entity tagger to extract named-entities. First, the domain corpus is searched for accompanying noun phrases by looking for linguistic patterns in relevant texts. After comparing the noun phrases to a statistical measure, which decides whether these noun phrases are relevant for the domain, and evaluating the number of times it co-occurs with the named-entity, the decision is made whether the found noun phrase is a relevant class candidate. This has been explained in Section 5.3. In the next step the authors use Wikipedia, Wikitionary, and DWDS (Digitales Woerterbuch der Deutschen Sprache). These sources provide for additional information, such as hierarchical and equivalence relations. With this information, the named-entities and their extracted class candidates can be placed into an existing taxonomy.

### 8.2.2 Tree-descending algorithm

Another classification method is applied in [39]. The authors have created the "tree-descending" and the "tree-ascending" classification algorithm. The tree-descending algorithm starts with descending the tree from the root to a leaf. Each time it encounters a node on its path, a choice is made between at least two nodes to continue its path. The node with the highest similarity, i.e., the node that is most similar to the new term is chosen. Different similarity measures can be used, for example the similarity measures described in Chapter 7. When the complete path from the root to the leaf has been covered, the node in this path with the highest similarity to the new term is determined. The new term is placed below this node.

### 8.2.3 Tree-ascending algorithm

The tree ascending algorithm uses the classification algorithm for a combination of a similarity measure, for example from Chapter 7, and taxonomic similarity, resulting in a number of votes. The target term $t$ is placed below the node with the highest number of votes. When computing the taxonomic similarity between two concepts $a, b$, first the least common superconcept of that pair of concepts $a, b$ is determined as stated in Equation 8.1.

$$lcs(a, b) = \underset{c \in N}{\operatorname{argmin}}(\delta(a, c) + \delta(b, c) + \delta(root, c)) \tag{8.1}$$

In this equation, $\delta(a, c)$ is the number of edges on the shortest hypernym/hyponym path between term $a$ and term $c$. This equation returns $c$, the least common superconcept. Subsequently, the taxonomic similarity between $a$ and $b$ is calculated with Equation 8.2.

$$T(a,b) = \frac{\delta(root,c)}{\delta(a,c) + \delta(b,c) + \delta(root,c)} \tag{8.2}$$

in which $0 \leq T \leq 1$ and 1 means maximum taxonomic similarity. In this algorithm, the taxonomic similarity is used to determine the taxonomic similarity between a hyponym $h$ of the candidate concept $n$ (possible node below which the target term $t$ can be placed) and the candidate concept itself. The considered candidate concepts, can for example be chosen from a list of concepts with the highest similarity to the target term $t$.

The combination of the similarity measure and the taxonomic similarity is calculated by multiplying the similarity measure of target term $t$ and hyponym $h$ with the taxonomic similarity of the candidate concept $n$ and its hyponym $h$ and summing that up for each hyponym $h$ of the candidate concept $n$.

$$W(n) = \sum_{h \in I_n} sim(t,h) \cdot T(n,h) \tag{8.3}$$

In this equation, $W(n)$ is the number of votes, $I_n$ is the set of hyponyms below the candidate concept $n$, $sim(t,h)$ is the similarity between hyponym $h$ and target word $t$, and $T(n,h)$ is the taxonomic similarity between $n$ and $h$ as computed in Equation 8.2. The target term $t$ is placed below the candidate concept with the highest number of votes.

## 8.3 Relation extraction using lexico-syntactic patterns

With this linguistic approach to hierarchy learning, introduced by [21], taxonomic relationships are not established through statistical methods, but through recognizing patterns of words in a text. In this approach, the text is scanned for instances of predefined patterns that indicate that there is some relation, e.g., the taxonomic relation. Examples of predefined patterns for taxonomic relations are presented below.

1. such $NP_0$ as $\{NP_i\}^*$ $\{$(or | and)$\}$ $NP_i$
2. $NP_i$ $\{, NP_i\}^*$ $\{,\}$ or other $NP_0$
3. $NP_i$ $\{, NP_i\}^*$ $\{,\}$ and other $NP$
4. $NP_0$ $\{,\}$ including $\{NP_i, \}^*$ $\{$or | and$\}$ $NP_i$
5. $NP_0$ $\{,\}$ especially $\{NP_i, \}^*$ $\{$or | and$\}$ $NP_i$

In this example, $NP$ indicates a noun phrase, which can be compared with a named-entity. An example of 3. is:

*...giraffes, elephants, zebras and other animals...*

$NP_i$ is the child of $NP_0$. The following taxonomic relations can thus be extracted: (Giraffe, Animal), (Elephant, Animal), (Zebra, Animal). The advantage of this method is that taxonomic relations can be established without any foreknowledge.

# Chapter 9

# Comparing taxonomies

This chapter describes how to compare two different taxonomies. This is useful when one wants to evaluate how well a computer-based taxonomy has performed. In order to do this, this has to be compared to a golden taxonomy, made by an expert in this area of knowledge. This golden taxonomy serves as a reference for the computer-based taxonomy. In [40], measures are presented to compare lexical and taxonomic overlap between two ontologies. This is elaborated by [20].

In order to compare two taxonomies, first the "semantic cotopy" (SC) is introduced. The semantic cotopy of a concept $c$ is the set of all its sub- and superconcepts. The formal description of SC is provided in Formula 9.1:

$$SC(c, O_1, O_2) = \{c_j \in C_1 \cap C_2 | c_j <_{C_1} c_i \vee c_i <_{C_1} c_j\} \tag{9.1}$$

This means that the SC is a set of its super- and subconcepts, and the concept itself does not belong to the cotopy. $C_i$ is the set of concepts belonging to ontology $O_i$. Instead of an ontology, one can apply this method to a taxonomy as well. The measure with which is evaluated whether the taxonomies correspond to each other is called the taxonomic overlap (TO). The taxonomic overlap is computed as follows.

$$\overline{TO}(O_1, O_2) = \frac{1}{|C_1|} \sum_{c \in C_1} TO(c, O_1, O_2) \tag{9.2}$$

In Formula 9.2, $\overline{TO}$ is the average taxonomic overlap. For each concept in $O_1$, the TO is computed. The outcomes are summed up and subsequently divided by $|C_1|$, the number of concepts in $C_1$, in order to obtain the average score of the taxonomic overlaps of the concepts in $C_1$.

If the concept $c$ from $C_1$ appears in $C_2$ as well, the $TO'$ is computed. If it does not appear in $C_2$, the $TO''$ is computed, as shown in Formula 9.3.

$$TO(c, O_1, O_2) = \begin{cases} TO'(c, O_1, O_2) & \text{if } c \in C_2 \\ TO''(c, O_1, O_2) & \text{if } c \notin C_2 \end{cases} \tag{9.3}$$

The semantic cotopies of concept $c$ in $C_1$ and in $C_2$ are retrieved. Subsequently the intersection and union of these two cotopies are determined. The number of concepts resulting from the intersection is divided by the number of concepts in the union, as shown in Formula 9.4.

$$TO'(c, O_1, O_2) = \frac{|SC(c, O_1, O_2) \cap SC(c, O_2, O_1)|}{|SC(c, O_1, O_2) \cup SC(c, O_2, O_1)|} \tag{9.4}$$

Formula 9.5 is almost the same as the previous one, but in this case, concept $c$ appears only in $C_1$ and not in $C_2$. That is why a concept $c'$ is searched with a semantic cotopy that has the biggest overlap with c, thus $c'$ is a concept in $C_2$ that maximizes the taxonomic overlap.

$$TO''(c, O_1, O_2) = max_{c' \in C_2} \frac{|SC(c, O_1, O_2) \cap SC(c, O_2, O_1|}{|SC(c, O_1, O_2) \cup SC(c, O_2, O_1)|} \tag{9.5}$$

To measure the accuracy and thoroughness of this method, [20] compute the precision and the recall, because they do not want to compute the taxonomic overlap in one direction. Formula 9.6 shows how the precision is calculated and Formula 9.7 shows how the recall is computed.

$$P(O_1, O_2) = \overline{TO}(O_1, O_2) \tag{9.6}$$

$$R(O_1, O_2) = \overline{TO}(O_2, O_1) \tag{9.7}$$

Subsequently an F–measure [41] is applied to compute the harmonic mean of the precision and the recall, as shown in Formula 9.8.

$$F(O_1, O_2) = \frac{2 \cdot P(O_1, O_2) \cdot R(O_1, O_2)}{P(O_1, O_2) + R(O_1, O_2)} \tag{9.8}$$

The percentage that results from this formula is the measure of similarity between taxonomies.

# Chapter 10

# Discussion

In Chapters 5 to 8, different methods have been discussed that can be used in the taxonomy learning process. In this chapter, the methods for the terminology extraction, synonyms, distance and taxonomic relations phase are discussed, and their pros and cons are described.

## 10.1 Terminology extraction

Different methods can be used to extract terms from text, as shown in Table 10.1. This table gives information about whether the method can produce multi-word terms, can filter irrelevant terms, is knowledge rich or knowledge poor, needs preprocessing, and whether the method uses a contrastive corpus.

| Name | Multi word term | Filters irrelevant terms | Knowledge rich/poor | Contrastive Corpus |
|---|---|---|---|---|
| Linguistic filtering | yes | no | poor | no |
| Tf–Idf | no | yes | poor | no |
| Filtering method | yes | yes | rich | yes |
| Chi square | yes | yes | poor | yes |
| C–/NC–value | yes | yes | poor | no |

Table 10.1: Methods for terminology extraction

For terminology extraction in the process of economic taxonomy learning, it is important that the method can handle multi-word terms, and that it is capable of recognizing names, i.e., names of persons, methods, etc. Also it is important that the method filters irrelevant terms. It should be noticed that some methods use a corpus with multiple documents and some methods have merged all documents into one large corpus. Some of the distance calculating methods, such as the document co-occurrence similarity, cannot cope with one large corpus, as they are based on a corpus with multiple

documents. These methods need multiple documents, because terms from different documents are compared to each other, something that would not be possible if all the documents would be merged into one big corpus. The last useful feature in terminology extraction is that methods sometimes use a contrastive corpus in order to check the occurrence of the word in the corpus in comparison to the occurrence of the term in the contrastive corpus. This is useful, because we are researching a specific domain.

**Linguistic filtering** For terminology extraction, linguistic filtering is not thorough enough. It determines which terms are named-entities, or the parts-of-speech, but it does not give any insight in how often a term occurs or whether it has any relevance regarding the domain. This leads to another disadvantage, namely that it does not filter irrelevant terms. From our investigation, linguistic methods are mostly used as a part of another method and not as a method of their own.

**Tf-Idf** This method has as an advantage and disadvantage at the same time that a term not necessarily needs to appear in different documents to have a high weight. (If it appears frequently in one document and not frequently in other documents it will have more weight.) The advantage can be explained as follows. In a corpus, it might happen that a specific term only appears in one document. The advantage of the Tf-Idf method is that it would give priority to such a term. This means that data sparseness in this case is less likely to occur than when another method is used. However, the disadvantage of this method is that this might cause that also irrelevant terms are included. For example when an irrelevant term that occurs once in ten different documents, with a corpus of 100 documents, the Tf-Idf is relatively high, compared to a random term. A consideration should be made between the importance of specific terms that do not occur often and the chance of having irrelevant terms. Tf-Idf cannot handle multi-word terms, unless the corpus is preprocessed, but that is not the case in statistical methods. Compared to the other methods, Tf-Idf is a rather simple method, as it needs relatively few computations. The corpus should contain different documents, otherwise the inverse document frequency cannot be computed.

**Filtering method** This method is needs linguistic preprocessing to find names and compounds from a specialized corpus. It makes use of different dimensions of terminology extraction, such as the domain relevance, the number of times the considered term occurs in different domain-related documents, the cohesion in a multi-word term, structural relevance, and misspellings. Checking for misspellings makes this method knowledge rich, as it uses WordNet, a dictionary or something similar. However this might be an unnecessary feature, as misspellings will probably not occur so of-

ten that they actually become a considered term. Furthermore the filtering method is reliable, since it checks the considered terms in the contrastive corpora as well. In this method, the corpus consists of different documents for the sake of the domain consensus.

**Chi square method** The property of the $\chi^2$ method is that its results are not only based on one, but on two factors, namely both the chi square and the co-occurrence with a named-entity. Furthermore it is based on both relation extraction using lexico-syntactic patterns and statistical analysis. Moreover it considers candidate terms through preprocessing. The unique feature of this method is that it applies a chi square method for a $2 \times 2$ contingency table. However, it does not have any special features that are useful for economic taxonomies. This method makes use of a domain corpus and a general corpus. Words that do not occur often are omitted in advance, so the method suffers from the data sparseness problem.

**C–value/NC–value method** This method is based on two values: the "C–value", which aims to improve extraction of the multi-word terms from a text, and the "NC–value", which incorporates context information to the C–value, in order to improve term extraction. This method does not check the frequency of a term in other documents and thus does not consider contrastive corpora.

## 10.2 Synonyms

**Structural semantic interconnections** WSD tries to find a precise definition of terms in a corpus. SSI does that by searching WordNet for different senses, considering every sense of the words in a sentence and comparing it to the other senses. This way it determines the right sense for each term. Subsequently, the corresponding synonym set can be found in WordNet. SSI preferably needs to be applied before the terminology extraction, otherwise the word senses will not be processed individually. For example, mouse#1 (WordNet's first word sense for mouse) and mouse#2 are processed as if they are different when SSI is applied before the terminology extraction, but if SSI is applied after terminology extraction, mouse has been treated as one word, regardless of which particular meaning it has. Using SSI in the taxonomy learning process is in the first place useful for finding synonyms, and in the second place useful because it incurs WSD. One disadvantage is that it cannot process every single term, since it is restricted to the terms that are known to WordNet. However that depends on how specific the economic domain in question is. If it is not very specific, WordNet is able to process it. One can process a sample of the corpus with SSI in order to

determine whether SSI links the right sense to the term or whether WordNet knows the term.

**Latent semantic indexing** LSI tries to find similar terms by using SVD. The advantage of this method, compared to SSI, is that it can compare any two terms, the only condition is that the terms have to occur at least once in the corpus. The context is not important for comparing two terms, while SSI at least needs some context words, to determine the sense. It requires a corpus with several documents. Then it puts the information the occurrence of every considered term in every document in the corpus in one big matrix. The output of this method is a matrix which gives a high value to similar words and a low value to dissimilar words. This method is knowledge poor and it can therefore process specific terms in a specific domain, that WordNet does not know.

## 10.3 Distance between terms

The different methods to calculate the distance between terms that have been discussed in this thesis are outlined in Table 10.2.

| Name | Corpus | Web | Knowledge rich/poor | Data sparseness |
|---|---|---|---|---|
| Path measure | × | | rich | |
| Window-based | × | | poor | |
| Document co-occurrence | × | | poor | |
| Web-based | | × | semi | × |

Table 10.2: Methods for calculating distance between terms.

For calculating the distance between terms in the process of automatically creating an economic taxonomy, it is important that there is place for new words, as the economic domain is a rapidly changing domain. In a detailed economic domain, many new terms can easily appear in a relatively short period of time. The method should have the ability to calculate the distance between multi-word terms. It depends on one's preferences whether it is important to include named-entities. When a method is knowledge rich, named-entities are not allowed, since these usually do not appear in dictionaries or WordNet.

**Path measure** The path measure is based on the use of WordNet. Thus named-entities will not be recognized. Named-entities can be important, since sometimes it is useful to have names of economists, systems or methods in the taxonomy as well. Therefore this method is not suitable for creating

a taxonomy in an economic domain. Multi-word terms cannot always be processed, because WordNet does not know all the multi-word terms.

**Document co-occurrence similarity** This method can only be used when the corpus consists of different documents. The document co-occurrence similarity method considers terms as if they do not have multiple meanings, unless the terms have already been identified by WSD. Multi-word terms can be used in this method.

**Window-based method** The corpus does not necessarily have to consist of different documents. Furthermore, the window-based method works more or less the same as the method above, but has smaller intervals than the document co-occurrence method, namely the interval has the size of the window instead of the size of the document. From our investigation, smaller intervals are better because similar words are more likely to appear together in one window than in one document.

**Web-based method** The Web-based method considers the number of results after entering a term in a search engine. It is very useful that it addresses the problem of data sparseness, something that happens especially in specific domains. However, this method just searches for words and does not consider their meaning, which could influence the similarity in a negative way. For example when looking for the terms "mouse" and "device". Because mouse has at least two senses, namely mouse as an animal and mouse as a computer device, $\frac{N(w_1)}{N_{WEB}}$ is a much bigger number than if mouse would have had only one sense. This causes that the similarity is smaller because mouse has two senses, while this should not have influence on the similarity between the words mouse and device. This makes this method less appropriate for researching economic domains.

## 10.4 Taxonomic relations

In the first place one should consider whether the taxonomy is based on an existing taxonomy. If this is the case, classification should be chosen, otherwise one needs to use clustering. A lexico-syntactic approach is the non statistic approach for finding the taxonomic relation.

### 10.4.1 Clustering

**Hierarchical linkage** This is one of the methods to be used when building a taxonomy from scratch. Single linkage tends to have a chaining effect [42], which means that small new clusters connect to a big cluster, resulting in a "chain" of concepts. Usually, the problem with complete linkage is that it

is sensitive to outliers, which could cause results being negatively influenced by outliers. Average linkage avoids the disadvantages of single and complete linkage by evaluating the clusters on the similarities between all terms.

**Formal concept analysis** Formal concept analysis is a good way to cluster terms, but it is actually a different form of taxonomy learning, as it exclusively contains objects and attributes. It has a good visualization of the object and their attributes that appear in one taxonomy. However for making an economic taxonomy it is not suitable, because the objects should match with at least a few attributes, otherwise the resulting taxonomy is useless. Finding matching objects and their attributes is beyond the scope of this thesis.

### 10.4.2 Classification

**Classification with corpus and NE tagger** The purpose of this method is finding words over a certain threshold ($\chi^2$ needs to be high enough and number of co-occurrences with NE should be above a certain level), but the essence is that the approved terms are looked up in the dictionary, Wikipedia, or Wikitionary in order to find taxonomic relations. This makes the method partially lexico-syntactic. This method requires a lot of calculations while the most important part of this method is the part that needs the least statistical effort; searching for accompanying terms in dictionaries.

**Tree-descending algorithm** The advantage of this method is that it needs few computations, as it uses only a small part of the search space in the tree. The second advantage is that the most relevant path is considered, so the new term is subsumed in the most relevant path in the tree. A disadvantage is that when it goes wrong somewhere up in the tree, that it immediately takes the wrong path. This chance turns out to be fairly high, since a taxonomy can be too general near the root, so a wrong choice could easily be made.

**Tree-ascending algorithm** Compared to the tree-descending algorithm, this algorithm is more comprehensive as it considers the whole taxonomy, so fewer mistakes are made than in the previous approach. The complete structure of the existing taxonomy is combined with the taxonomic distance and incorporated into the decision where to place the new term.

### 10.4.3 Relation extraction using lexico-syntactic patterns

Extracting patterns from text can be a good way to find hypernyms and hyponyms. This method does not need a similarity measure, nor terminology extraction. However when sentences are formed like “such things as...” the

number of “things” will be very big, so general concepts will probably have too many children. Also there might not be enough economical information, because all economic documents do not contain enough information, i.e., sentences that match the lexico-syntactic patterns, to create a taxonomy that encompasses the economic domain. This method might however be a nice addition to the clustering or the classification method, e.g., for finding additional taxonomic relations for in a taxonomy.

# Chapter 11

# Concluding remarks

## 11.1 Conclusion

In this thesis we have answered the four questions set at the beginning. The questions are “what is taxonomy learning?”, “what is the economic relevance of taxonomy learning and how can it be used in the economic domain?”, “what are different methods to create a taxonomy?” and ”which of these methods can be used in economic taxonomy learning?”

Taxonomy learning is the process of automatic creation of a taxonomy. The goal is to automatically extract relevant terms and their taxonomic relations. The input is a corpus, which can be a general corpus, or a domain specific corpus. This corpus is processed by several NLP processes. The output is a taxonomy, containing the relevant terms from the corpus.

Taxonomy learning can be used in economics for several reasons, namely to facilitate understanding in complex economic domains, query relaxation when searching for economic terms, and understanding the evolution in the economic domain. This can be visualized through taxonomy learning, which makes it easier to interpret the relations between the economic terms.

Different methods are described in this thesis, the main steps in taxonomy learning are terminology extraction, finding synonyms, distance computation between terms, and taxonomic relations. In this thesis we have tried to structure the different steps of the taxonomy learning process, based on the ontology learning layer cake, but there are many ways to perform taxonomy learning, so the actual sequence of steps may vary.

The methods that can be used in economic taxonomy learning are chosen, based on their pros and cons, which have been evaluated in the discussion. The answers to the questions posed in the first chapters of this thesis help to investigate the main question: ”How to create an economic taxonomy in an automatic way?”.

In the first place, a corpus is needed, which contains texts, such as scientific papers, in the economic domain. From our investigation, we can

conclude the following methods can be used to build a taxonomy in an automatic way.

For terminology extraction, the part where the terms are extracted from the corpus, it is preferred to use one of the hybrid methods, such as the filtering method. This method can be used in specific domains, checks contrastive corpora and processes multi-word terms. The best method to find synonyms depends on whether the domain is very specific. If it is not too specific, i.e., WordNet is able to recognize the terms, the best method for finding synonyms is structural semantic interpretation. If that is not the case and WordNet does not know the terms, it is better to use latent semantic indexing for finding synonyms. For computing the distance between terms, the window-based similarity method is the best method to be used of the methods discussed in this thesis, because it considers windows of words. The chance that terms are similar when they appear together in a window is bigger than the chance that they are similar when they appear together in a whole document. For taxonomic relations we conclude that the best method is average linkage clustering, since it does not suffer from outliers or the chaining effect.

When one wants to expand a taxonomy with new terms, we advise to use tree-ascending classification, since it considers the candidate concepts that are most similar to the new term and subsequently the hyponyms of the candidate terms before it determines where the new term is placed.

## 11.2 Future work

It is difficult to state whether a method to create a taxonomy is right or wrong. It is not clear what a taxonomy should look like before it has been created. Taxonomies can have different possibilities of conceptualizations but this does not mean they are incorrect, because they have different target groups. Manual creation of a taxonomy by some expert could be an option, but it is slow and expensive, and created to the point of view of the person that creates it. Future work allows research for a "right" or "wrong" taxonomy.

Another example of this is word sense disambiguation. Terms and their senses can be judged differently by people, even when the rest of the sentence is known. Also computer systems have difficulties with detecting word senses sometimes.

Overcoming the problem of data sparseness is an issue in natural language processing. For a computer system it is difficult to determine which of the words that do not occur often are important. At the same time, human language is inconsistent and contains a lot of noise. This problem could be addressed by finding a balance between data sparseness and noise, which yields the most important terms and disposes of unnecessary terms.

Future work can be aimed towards addressing these problems. This should help in improving the process of taxonomy learning. This thesis encompasses the different methods that one can use for taxonomy creation in an economic domain. However it has not been tested in an application, so possible future work is testing these methods by means of an application. This would also allow other methods to be tested on and compared with the same application.

# Appendices

# Bibliography


[1] Maedche, A., Staab, S.: Ontology learning for the Semantic Web. IEEE Intelligent Systems **16** (2001) 72–79

[2] Gruber, T.: A translation approach to portable ontology specifications. Knowledge Acquisition **5** (1993) 199–220

[3] Borst, W.: Construction of engineering ontologies for knowledge sharing and reuse. PhD thesis, University of Twente (1997)

[4] Dietz, E., van den Heuvel, E.: Construction of taxonomies based on three different measurement approaches. Unpublished (2008)

[5] Charniak, E.: Statistical techniques for natural language parsing. AI Magazine **18** (1997) 33–44

[6] Li, X., Szpakowicz, S., Matwin, S.: A wordnet-based algorithm for word sense disambiguation. In: International Joint Conferences on Artificial Intelligence (IJCAI 1995), Montreal, Quebeq, Canada. (1995) 1368–1374

[7] Neshati, M., Abolhassani, H., Rahimi, A.: Taxonomy learning using compound similarity measure. In: The 2007 IEEE/WIC/ACM International Joint Conference on Web Intelligence (WI 2007), Silicon Valley, California, USA. (2007) 487–490

[8] Buitelaar, P., Cimiano, C.: Ontology learning from text. In: Tutorial at the 11th Conference of the European Association for Computational Linguistics (EACL 2006), Trento, Italy. (2006) http://www.aifb.uni-karlsruhe.de/WBS/pci/EACL_OL_Tutorial_06/EACL06_OLTutorial.pdf.

[9] Bourigault, D.: Surface grammatical analysis for the extraction of noun phrases. In: 14th International Conference on Computational Linguistics (Coling 1992), Nantes, France. (1992) 977–981

[10] Chinchor, N., Brown, E., Ferro, L., Robinson, P.: Named entity recognition task definition. Technical report, MITRE and SAIC (1999)

[11] Chen, S., Goodman, J.: An empirical study of smoothing techniques for language modeling. In: 34th Annual Meeting of the Association for Computational Linguistics (ACL 1996), Santa Cruz, California, USA. (1996) 310–318

[12] Dagan, I., Marcus, S., Markovitch, S.: Contextual word similarity and estimation from sparse data. In: 31th Annual Meeting of the Association for Computational Linguistics (ACL 1993), Columbus, Ohio, USA. (1993) 164–171

[13] Biemann, C.: Ontology learning from text: a survey of methods. LDV Forum **20** (2005) 75–93

[14] Bos, J., Markert, K.: Recognising textual entailment with logical inference. In: Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005). (2005) 628–635

[15] Deerwester, S., Dumais, S., Furnas, G., Landauer, T., Harshman, R.: Indexing by latent semantic analysis. Journal of the American Society for Information Science **41** (1990) 391–407

[16] Fensel, D., van Harmelen, F., Horrocks, I., McGuinness, D., Patel-Schneider, P.: OIL: An ontology infrastructure for the Semantic Web. IEEE Intelligent Systems **16** (2001) 38–45

[17] Fellbaum, C.: Wordnet: An Electronic Lexical Database. Bradford Books (1998)

[18] Guarino, N., Poli, R.: Formal ontology, conceptual analysis and knowledge representation. International Journal of Human and Computer Studies **43** (1995) 625–640

[19] Missikoff, M., Navigli, R., Velardi, P.: Integrated approach to Web ontology learning and engineering. Computer **35** (2002) 54–57

[20] Cimiano, P., Hotho, A., Staab, S.: Learning concept hierarchies from text corpora using formal concept analysis. Journal of Artificial Intelligence Research **24** (2005) 305–339

[21] Hearst, M.: Automatic acquisition of hyponyms from large text corpora. In: 14th International Conference on Computational Linguistics (Coling 1992), Nantes, France. (1992) 539–545

[22] Salton, G., McGill, M.: Introduction to modern information retrieval. McGraw-Hill (1983)

[23] Frantzi, K., Ananiadou, S., Mima, H.: Automatic recognition of multi-word terms: the c-value/nc-value method. International Journal on Digital Libraries **3** (2000) 115–130

[24] Sclano, F., Velardi, P.: TermExtractor: a Web application to learn the shared terminology of emergent Web communities. In: 9th Conference on Terminology and Artificial Intelligence (TIA 2007), Nice, France. (2007)

[25] Basili, R., Pazienza, M., Velardi, P.: An empyrical symbolic approach to natural language processing. Artificial Intelligence **84** (1996) 59–99

[26] Park, Y., Byrd, R., Bouraev, B.: Automatic glossary extraction: beyond terminology identification. In: 19th International Conference on Computational Linguistics (Coling 2002), Taipei, Taiwan. (2002)

[27] Weber, N., Buitelaar, P.: Web-based ontology learning with ISOLDE. In: Workshop on Web Content Mining with Human Language at the International Semantic Web conference (ISWC 2006), Athens, Georgia, USA. (2006)

[28] Sager, J.: A practical course in terminology processing. John Benjamins Publishing Company (1990)

[29] Justeson, J., Katz, S.: Technical terminology: some linguistic properties and an algorithm for identification in text. Natural Language Engineering **1** (1995) 9–27

[30] Navigli, R., Velardi, P., Gangemi, A.: Ontology learning and its application to automated terminology translation. IEEE Intelligent Systems **18** (2003) 22–31

[31] The Free Dictionary: Online dictionary, encyclopedia, and thesaurus (2008) Accessed July 27th, 2008. http://www.thefreedictionary.com.

[32] Navigli, R.: Online word sense disambiguation with structural semantic interconnections. In: 11th Conference of the European Association for Computational Linguistics (EACL 2006), Trento, Italy. (2006) 107–110

[33] Kleinberg, J.: Authorative sources in a hyperlinked environment. Journal of the ACM (JACM) **46** (1999) 604–632

[34] Leacock, C., Chodorow, M.: Combining local context and wordnet sense similarity for word sense disambiguation. In: WordNet: An Electronic Lexical Database. Bradford Books (1998) 266–283

[35] Maedche, A., Staab, S.: Ontology learning. In: Handbook on Ontologies. Springer-Verlag (2004) 173–190

[36] Wille, R.: Restructuring lattice theory: an approach based on hierarchies of concepts. In: Ordered Sets. Dordrecht-Boston (1982) 445–470

[37] Stumme, G., Taouil, R., Bastide, Y., Lakhal, L.: Conceptual clustering with iceberg concept lattices. In: GI-Fachgruppentreffen Maschinelles Lernen (FGML 2001), Dortmund, Germany. (2001) 106–114

[38] Wolff, K.: A first course in formal concept analysis: how to understand line diagrams. Advances in statistical software **4** (1993) 429–438

[39] Pekar, V., Staab, S.: Taxonomy learning – factoring the structure of a taxonomy into a semantic classification decision. In: 19th International Conference on Computational Linguistics (Coling 2002), Taipei, Taiwan. (2002) 786–792

[40] Maedche, A., Staab, S.: Measuring similarity between ontologies. In: European Conference on Knowledge Engineering and Knowledge Management (EKAW 2002), Siguenza, Spain. (2002) 251–263

[41] van Rijsbergen, C.: Information Retrieval. London Butterworth (1979)

[42] Manning, C., Raghavan, P., Schuetze, H.: Optimality of HAC. In: Introduction to information retrieval. Cambridge University Press (2008) 360–362