University of Miami
Scholarly Repository

Open Access Theses | Electronic Theses and Dissertations

2010-01-01

PrOntoLearn: Unsupervised Lexico-Semantic Ontology Generation using Probabilistic Methods

Saminda Wishwajith Abeyruwan
University of Miami, s.abeyruwan@umiami.edu

Follow this and additional works at: https://scholarlyrepository.miami.edu/oa_theses

Recommended Citation
Abeyruwan, Saminda Wishwajith, "PrOntoLearn: Unsupervised Lexico-Semantic Ontology Generation using Probabilistic Methods" (2010). *Open Access Theses*. 28.
https://scholarlyrepository.miami.edu/oa_theses/28

This Open access is brought to you for free and open access by the Electronic Theses and Dissertations at Scholarly Repository. It has been accepted for inclusion in Open Access Theses by an authorized administrator of Scholarly Repository. For more information, please contact repository.library@miami.edu.

UNIVERSITY OF MIAMI

PRONTOLEARN: UNSUPERVISED LEXICO-SEMANTIC ONTOLOGY GENERATION USING PROBABILISTIC METHODS

By

Saminda Abeyruwan

A THESIS

Submitted to the Faculty
of the University of Miami
in partial fulfillment of the requirements for
the degree of Master of Science

Coral Gables, Florida

May 2010

©2010
Saminda Abeyruwan
All Rights Reserved

UNIVERSITY OF MIAMI

A thesis submitted in partial fulfillment of
the requirements for the degree of
Master of Science

PRONTOLEARN: UNSUPERVISED LEXICO-SEMANTIC ONTOLOGY GENERATION USING PROBABILISTIC METHODS

Saminda Abeyruwan

Approved:

Ubbo Visser, Ph.D. Professor of Computer Science	Terri A. Scandura, Ph.D. Dean of the Graduate School
Geoff Sutcliffe, Ph.D. Professor of Computer Science	Stephan Schuerer, Ph.D. Professor of Molecular and Cellular Pharmocology

ABEYRUWAN, SAMINDA (M.S., Computer Science)
PrOntoLearn: Unsupervised Lexico-Semantic Ontology Generation using Probabilistic Methods (May 2010)

Abstract of a thesis at the University of Miami.

Thesis supervised by Professor Ubbo Visser.
No. of pages in text. (94)

An ontology is a formal, explicit specification of a shared conceptualization [1, 2]. Formalizing an ontology for a domain is a tedious and cumbersome process. It is constrained by the knowledge acquisition bottleneck (KAB). There exists a large number of text corpora that can be used for classification in order to create ontologies with the intention to provide better support for the intended parties. In our research we provide a novel unsupervised bottom-up ontology generation method. This method is based on lexico-semantic structures and Bayesian reasoning to expedite the ontology generation process. This process also provides evidence to domain experts to build ontologies based on top-down approaches.

Acknowledgements

There are may people that I would like to express my sincere thanks, without which this thesis would not be completed. First of all, I would like to start thanking my advisor Prof. Ubbo Vissor for his support, guidance and freedom to work on this thesis. I would also like to thanks my committee, Prof. Geoff Sutcliffe and Prof. Stephan Schuerer for their excellent support, guidance and comments throughout my research. I also wish to thank Department of Computer Science for providing me with graduate assistantship to support my studies, without which this endeavour would not be possible. Finally, I would like to thanks everyone who helped me preparing this thesis and providing me with feedback.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

An ontology is a formal, explicit specification of a shared conceptualization [1, 2]. It is a generally known fact that formalizing an ontology for a given domain with the supervision of domain experts is a tedious and cumbersome process. This process consumes a lot of time and effort. This is formally known as the *knowledge acquisition bottleneck (KAB)* and has been highly investigated by the semantic web communities.

1.1 Goal

There exists a large number of text corpora available for public from different domains such as the BioAssay ontology dataset [4] that needs to be classified into an ontology which would ultimately provide better values for intended parties. A corpus of a domain of disclosure such as [4] contains the following important characteristics.

1. Redundancy

2. Structured and unstructured text

3. Noisy and uncertain data that provide a degree of belief
4. Lexical disambiguity
5. Semantic heterogeneity problems

We discuss in depth the importance of prior characteristics in Chapter 4. Using the features of above the characteristics, our goal in this research is to provide a novel method to construct an ontology from the evidence collected from the corpus. In order to achieve our goal, we have used the lexico-semantic features of the lexicon and probabilistic reasoning to handle the uncertainty of the features. Since our method is applied to build an ontology for a corpus without domain experts, our believe is that this method would be classified as an unsupervised learning technique. Since our method starts from the evidence presented from the corpus, it is generally can be seen as a reverse engineering technique. We have used WordNet [5] to handle lexico-semantic structures, and Bayesian reasoning to handle degree of belief of an uncertain event. Once the conceptualization is learned, we have implemented a Java based application to serialize this to OWL DL [6] format.

1.2 Road map

The rest of this report is organized as follows. Chapter 2 provides background information on OWL and Bayesian reasoning. Chapter 3 provides a broad investigation of the related work. Chapter 4 provides information of our research approach. Chapter 5 provides the details of Java based implementation of the work. Chapter 6 provides a detailed description of the experiments we have conducted on the constructed ontologies in three different data sets and the discussion on what we have learned. Finally, chapter 7 provides the summary and the expected feature work.

Chapter 2

Background Information

In this section we discuss about the fundamental representation of an ontology and the Bayesian reasoning technique.

2.1 Ontology

The word “ontology” has different interpretations in different communities. In our study we are interested in the interpretation given in Computer Science as a special kind of information object or computational artifact. Gruber [1] originally defined the notion of an ontology as a explicit specification of a conceptualization. Borst [7] defined an ontology as a formal specification of a shared conceptualization, i.e., an ontology should have a shared view among several parties. This definition also expresses the fact that an ontology is also machine readable. Finally, Studer et al. [2] merge the prior two definitions stating that an ontology is a formal, explicit specification of a shared conceptualization. All above definitions talk about a notion of a conceptualization. A conceptualization is an abstract, simplified view of the world that we wish to represent for some purpose. This view [8] is based on the objects, concepts and other entities that exist in some domain and relationships that hold among them. According to [8] definition of an ontology is given as,

Definition 2.1 *Let C be a conceptualization and L a logical language with vocabulary V and ontological commitment K. An ontology O for C with vocabulary V and ontological commitment K is a logical theory consisting of a set of formulas of L, designed so that the set of its models approximates to the set of intended models of L according to K.*

Definition 2.1 talks about a notion of a model and an intended model. A model is an interpretation of objects, concepts and other entities in the universe of discourse, and a set of relations amoung them. An intended model is always compatible with ontological commitment K. For more details of a model and an intended model, please refer to the definitions 2.1, 2.4 and 3.3 of [8].

Figure 2.1 shows an example ontology mentioned in [8]. Let the set C_C contain the concepts of the conceptualiaztion C and they are $C_c = \{$Person, Manager, Researcher$\}$. Let the set C_R contain the binary relations of the conceptualization C and they are $C_R = \{$reports-to, cooperates-with$\}$. Lets build the ontology O consisting of a set of logical formulae. We use the Boolean constructors *conjuction* ($\sqcap$), which is interpreted as set intersection, *disjunction* ($\sqcup$), which is interpreted as the set union, is-a ($\Rightarrow$), which which is interpreted as the implication, and *negation* ($\neg$), which is interpreted as the set complement.

- IS-A relations or taxonomic relations,

  $O_1 = \{\text{Researcher(X)} \Rightarrow \text{Person(X)}, \text{Manager(X)} \Rightarrow \text{Person(X)}\ \}$

- Domain and ranges

  $O_2 = O_1 \sqcup \{\text{reports-to(X,Y)} \Rightarrow \text{Person(X)} \sqcap \text{Person(Y)},$

  $\text{cooperates-with(X,Y)} \Rightarrow \text{Person(X)} \sqcap \text{Person(Y)}\}$

- Symmetry

  $O_3 = O_2 \sqcup \{\text{cooperates-with(X,Y)} \Leftrightarrow \text{cooperates-with(Y,X)}\}$

- Transitivity

  $O_4 = O_3 \sqcup \{\text{reports-to(X,Y)} \sqcap \text{reports-to(Y,Z)} \Rightarrow \text{reports-to(X,Z)}\ \}$

- Disjointness

  $O_5 = O_4 \sqcup \{\text{Manager(X)} \Leftrightarrow \neg\ \text{Researcher(X)}\}$

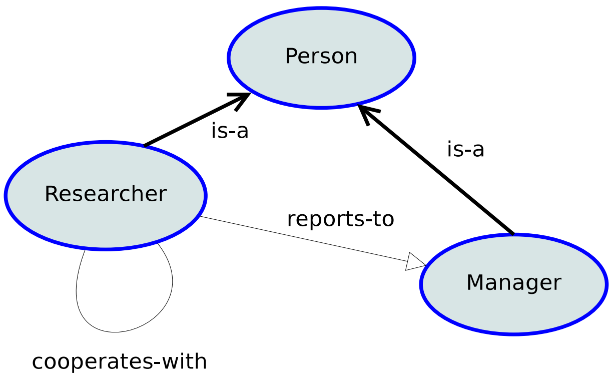


Figure 2.1: Example ontology of three concepts Person, Researcher & Manager and two relations cooperates-with & reports-to

An ontology specifies a domain model and the knowledge base provides assertions about concepts and relations that can be learned. Our approach focuses on learning an ontology using an unsupervised probabilistic approach. In our work, input data is obtained as text documents, and redundancy in the documents is used to capture the concepts and relations. Ontology learning is to some extent considered as a process of reverse engineering. The author of a document has a world or a domain model in mind. Other authors writing the similar text for some degree share the same world or domain model. Hence, the redundancy in authors' worlds or domain models are used to extract the facts that are needed to establish an ontology. Even if there are substantially large amount of data, only partial information about the authors' world or domain model can be extracted. Hence, the learning process is considered to be a challenging task. Thus, according to definition 2.1, in order to learn an ontology, we need to acquire the relevant terminology, identify synonym terms/linguistic variants, concept formation, hierarchical organisation of the concepts, learn relations/attributes/domain/ranges, hierarchical organisation of relations, and axiom induction.

2.1.1 OWL

There are different knowledge representation schemes available to represent an ontology. Web Ontology Language (OWL), which is recommended by the World Wide Web Consortium (W3C), has been adopted as a standard for representing ontologies. We have based our work adhering to this standard and we will discuss our development formalisms in its terms. OWL is defined in three different sub-languages and we are interested in the OWL DL (description logic) sub-language because it is computationally efficient and permits efficient reasoning support. The computational tractability and efficient reasoning support is achieved through vocabulary partitioning, explicit typing, property separation, no transitive cardinality restriction and restricted anonymous classes. OWL DL, which is refered to as OWL hereafter, is based on the description logic $SHOIN(D)$ [9]. The notions of a domain are described by concepts (classes), relations (roles or properties) and individuals (instances). These primitives state facts about the domain in the form of axioms. Terminological (T-box) axioms talk about concepts and relations, and assertional axioms (A-box) talk about the properties of individuals of the domain. OWL uses the atomic constructs in table 2.1 to build complex classes and relations.

Description logic (DL) is a formal logic with well defined semantics. The semantics of DL is specified via model theoretic semantics, which explicates the relationship between the language syntax and the models of a domain.

.

2.2 Statistical Models

Our implementation tries to gain a lot of insight from probability theory and statistical models. The following subsections briefly discuss areas we have touched to build our model.

2.2.1 Uncertainty

Uncertainty in a domain arises from both noise on measurements and the finite size of the data set. Probability theory provides a consistent framework to handle un-

Table 2.1: OWL DL syntax and an abstract syntax of an ontology representation

C	Definition(C)	concept of the ontology
C $\sqcap$ D	intersectionOf(C D)	conjunction of two concepts C and D
C $\sqcup$ D	unionOf(C D)	union of two concepts C or D
$\neg$ C	complementOf(C)	negation of the concept C
$\exists$R.C	R someValuesFrom(C)	existential restriction of an anonymous concept related to a concept C via relation R
$\forall$R.C	R allValuesFrom(C)	value restriction of an anonymous concept related to a concept C via relation R
$\geq$n R	R maxCardinality(n)	number restriction of a concept related to another concepts at least n via relation R
$\leq$n R	minCardinality(n)	number restriction of a concept related to another concepts at most n via relation R
C $\sqsubseteq$ D	SubClassOf(C D)	C is a subconcept of D
C $\sqsubseteq$ $\neg$D	DisjointClasses(C D)	C concepts is disjoint from D concept

certainty. There are mainly two important interpretations of probability, frequentist and Bayesian. In frequentist interpretations, probability is given by frequencies of random, repeatable events. Bayesian interpretation provide a quantification of uncertainty. Consider an uncertain event such as "Mars was a habitable planet 100 millions years ago". This kind of an event cannot be repeated many times in order to define probability. However, using an observation satellite one could gather evidence and take actions or revise opinions. This sort of uncertainty is interpreted thorough the elegant, and very general, Bayesian interpretation of probability [10, 11]. Table 2.2 lists the ontological and epistemological commitments of propositional logic, first order logic and probability.

Text documents contain a lot of sentences. The structure of these sentences such

Table 2.2: The ontological and epistemological commitment

Class	Ontological commitment	Epistemological commitment
Propositional logic	facts	true/false/unknown
First order logic	facts and conceptualization (objects and relations)	true/false/unknown
Probability theory	facts	uncertainty $\in$ [0, 1]

as syntactic pattern (part-of-speech tags), semantic structures can be used to fomulate a knowledge base. The extracted knowledge for certain degree can represent a degree of belief in the relevant sentences. Hence, in order to deal with such degrees of believe, we have to model it with respect to Bayesian probability. It is important to know that the ontological commitment of the probability is the as same as for formal logic, but the epistemological commitment, that is, probability provides uncertainty with a numerical value between 0 (false) and 1(true), will be different. Hence, this infers that probability statements are made with respect to a knowledge state, not with respect to the real world.

2.2.2 Bayesian approach

Probability is represented through random variables. A collection of multiple random variables gives the full joint probability distribution. If X and Y are random variables, the basic rules of probability, sum rule 2.1 and product rule 2.2 are given as follows,

$$p(X) = \sum_{Y} p(X, Y) \tag{2.1}$$

$$p(X, Y) = p(Y|X)p(X), \tag{2.2}$$

where $p(Y|X)$ is the conditional probability, which is known as probability of Y given X. If X and Y are independent, the joint distribution of the two random variables are factored into product of the marginals 2.3, and conditional probability is given in 2.4:

$$p(X, Y) = p(X)p(Y) \tag{2.3}$$

$$p(Y|X) = p(Y). \tag{2.4}$$

Since $p(X, Y) = p(Y, X)$, Bayes' theorem is given by equation 2.5,

$$p(Y|X) = \frac{p(X|Y)p(Y)}{p(X)} \tag{2.5}$$

where,

$$p(X) = \sum_{Y} p(X|Y)p(Y). \tag{2.6}$$

An independence relationship among variables reduces the number of probabilities that need to be specified in the full joint distribution. A Bayesian network represents such dependencies among variables. If $x_1, \ldots, x_n$ are random variables, the joint distribution is given by 2.7:

$$p(x_1, \ldots, x_n) = p(x_n|x_{n-1}, \ldots, x_1)p(x_{n-1}|x_{n-2}) \ldots p(x_2|x_1) = \prod_{i=1}^{n} p(x_i|x_{i_1}, \ldots, x_1) \tag{2.7}$$

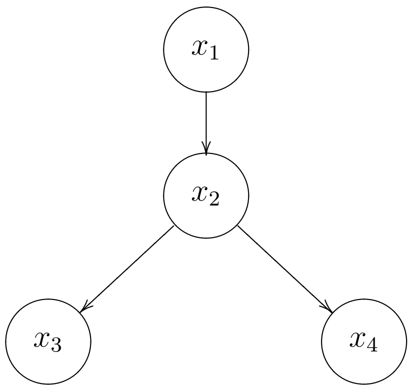


Figure 2.2: Example Bayesian network with four nodes x_1, x_2, x_3 and x_4

Each conditional probability represents a link in the Bayesian network. Each node represents a random variable or collection of random variables. Equation 2.7 represents a fully connected graph in general. The most interesting problems are conveyed from Bayesian networks, when some of the links are absence in the graph. This allows to create very flexible models, which we used in this thesis. For example, consider the Bayesian network in figure 2.2, with four random variables x_1, x_2, x_3 and x_4.

From figure 2.2 and equation 2.7, the joint probability distribution of the four variables x_1, x_2, x_3 and x_4 is given by

$$p(x_1, x_2, x_3, x_4) = p(x_1)p(x_2|x_1)p(x_3|x_2)p(x_4|x_2). \tag{2.8}$$

An important concept for probability distributions over multiple variables is that of conditional independece [10, 11]. If there are three random variables x_1, x_2 and x_3, and conditional distribution of x_1, given x_2 and x_3 is such that x_1 does not depend on the value of x_2 is given by

$$p(x_1|x_2, x_3) = p(x_1|x_3). \tag{2.9}$$

If we consider the joint distribution of x_1 and x_2 condition on x_3, using the product rule 2.2 and equation 2.9, we get:

$$p(x_1, x_2|x_3) = p(x_1|x_2, x_3)p(x_2|x_3) = p(x_1|x_3)p(x_2|x_3). \tag{2.10}$$

Equation 2.10 shows that condition on x_3 the joint distribution of x_1 and x_2 factorizes into the product of the marginal distribution of x_1 and x_2. Thus, the random variables x_1 and x_2 are statistically independent given x_3. The short hand notation for conditional independence is give as in [11],

$$x_1 \perp x_2 | x_3 \tag{2.11}$$

which states that random variable x_1 is conditionally independent ($\perp$) from x_2 given the evidence of random variable x_3. We use this notion throughout our report to represent independence when necessary.

2.2.3 N-gram model

An N-gram is a model for word prediction. An N-gram model uses the previous $n-1$ words to predict the next one. This is a very simple yet powerful model for capturing the meaning of a sentence. N-gram models are interesting if the corpus is sufficiently large enough to capture many recurring sequences. In its original form an N-gram captures all words in the sequence, including the stop words. In general, an N-gram commits in the present, not in the future or past when reasoning. Since stop words do not give us additional information, we remove them before applying an N-gram parser. For an example, lets take the sentence, “Taxonomies or concept hierarchies are crucial for any knowledge-based system”. First, we remove the stop words such as “or”, “for”, “any”, then we run the N-gram generator. Lets assume N is two. Then this will give sequences, “Taxonomies concept”, “concept hierarchies”, “hierarchies crucial”, “crucial knowledge-base” and “knowledge-based system”. This example uses the wordform, which is the inflected form as it appears in the corpus. If the wordform is used, then words such as “concept“ and ”concepts” will be treated as two separate words. This is not a good simplification and we are instead using a single abstract word call a lemma. A Lemma is the set of lexical forms having the same stem and the same word-sense. Hence, word “concepts” is lemmatized to ”concept“ before N-gram generate runs on them.

In our work we are interested in probabilistic models. If bigram models are used, we are interested in the probability of $P(concept \mid taxonomy)$. Generally, in an N-gram, nth word depends on the previous $n-1$ words. This is shown in equation 2.12.

$$P(w_n|w_{n_1},\ldots,w_1) \tag{2.12}$$

We approximate the probability of equation 2.12 by counting the number of occurrences of w_n words given the word sequences $w_{n_1}, \ldots, w_1$:

$$P(w_n|w_{n_1}, \ldots, w_1) = \frac{\#\{w_n\}}{\#\{(n-1)^{th} words\}}. \tag{2.13}$$

$\#\{X\}$ represents the number of X occurrences in the corpus. Equation 2.13 estimates the N-gram probability by dividing the observed frequency of the n^{th} word w_n by observed frequency of prefix $w_{n_1}, \ldots, w_1$. This is the relative frequency and the maximum likelihood estimation (MLE) that maximises the probability of observed data $P(w_n|w_{n_1}, \ldots, w_1)$.

It is assumed that the probability of a future word depends only on one past word given the present word, which is known as the Markov assumption [11]. The Markov assumption states that the probability of a future event is independent of the past events given the present event. A bigram model is known as first-order Markov model because a word looks one word in the past. A trigram model is known as second-order Markov model because a word looks two words in the past. In general N-gram is called $(N-1)^{th}$ order Markov model.

2.2.4 Semantic analysis

All human languages have a form of predicate argument arrangement at the core of their semantic structure [12]. The meaning of a structure is inferred from words, context of words and the grammatical structure. We learn a probabilistic semantic structure from the corpus. The conceptualization of the domain is derived from both IS-A relationships and relationships among the concepts. IS-A relationships are learned according to Bayesian learning. This is discussed in Chapter 4. The relationship among concepts are learned as follows. We assume that a verb of a sentence will provide the required predicate argument arrangement. In order to associate with proper predicate argument, we need to understand the syntactic structure of a sentence. Throughout over work, we have used the following syntactic structure given in Equation 2.14 for sentences:

$$\text{(Subject\_NP+)} \;\; \text{(Verb)} \; \text{(Object\_NP+)} \tag{2.14}$$

A subject noun phrase (Subject_NP+) consists of subject groups formed from N-gram parsing, and an Object noun phrase (Object_NP) consists of object groups formed from N-gram parsing. A Verb provides the predicate argument arrangement. There will exist groups in Subject_NP and Object_NP that will have the highest association for a given criteria. We discuss the criteria in chapter 4. We select the group from each category that will maximise the observed probability. If the group $Subject_1$ is chosen from (Subject_NP) and the group $Object_1$ is chosen from (Object_NP), according to λ notation, the semantics of the sentence will be given in following expression 2.15:

$$\lambda y.\lambda x.Verb(x, y). \tag{2.15}$$

Then, applying β-reduction on 2.15, we obtain

$$((\lambda y.\lambda x.Verb(x, y))(Object_1))(Subject_1)$$

$$Verb(Subject_1, Object_1). \tag{2.16}$$

2.2.5 Extending probability to description logic

As we have seen in subsection 2.1.1, OWL DL is based on the SHOIN(D) description logic. Description logics are formal logics with well defined semantics. Generally, the semantics of a description logic is specified with model theoretic semantics M, which explicates the relationships between the language syntax and the models of a domain [13]. An interpretation $I = (\triangle^I, .^I)$ consists of a set $\triangle^I$, called the domain of I and function $.^I$ maps from individuals, concepts and relations to elements of $\triangle^I$ using binary relations $(\triangle^I \times \triangle^I)$. Description logic knowledge bases consist of a set of axioms, which act as constrains of interpretation I. The meaning of a knowledge base is derived from features and relations that are common in all possible I. An interpretation satisfies a knowledge base, if it satisfies each axiom in the knowledge base. Such an I is called the model of the knowledge base. If there are no models then the knowledge base is inconsistent. There may exists entailed axioms for the knowledge base [9].

Bayesian networks are propositional. The sets of variables are fixed and finite and there is a fixed domain of finite values. If we could combine the probability theory with description logic, this would dramatically increase the range of problems that can be handled. In the propositional logic context, Bayesian networks specify probabilities over atomic events. These atomic events can be considered as models or possible worlds in propositional logic. As we have mentioned before, interpretation or models specify domain of individuals, concepts and binary relations that holds among these entities. Thus, we can consider description logic probabilistic knowledge base specifies all possible description logic models. In our approach, concepts of the interpretation consist of nouns, proper nouns of the lexicon, and binary relations of the interpretation consists of verbs.

Chapter 3

Related Work

The problem of learning a conceptualization from text has been studied in many disciplines such as machine learning, text mining, information retrieval, natural language processing, and Semantic Web (first order logical reasoning). In the following subsections, first we discuss the existing approaches with their pros & cons and second, we show how our research influence in filling the existing gaps, i.e, how unsupervised probabilistically reasoned lexico-semantic ontology generation method brings down the KAB and influence the decision making process of a domain expert.

3.1 Probabilistic reasoning

Knowledge representing languages such as first-order-logic (various other logics), description logic reflect a major trade-off between expressivity and tractability [14]. Even though description logic, the logic that we are most interested in this work, has made compromises that allows to be a more successful one, it is limited in ability to express uncertainty. P-CLASSIC probabilistic version of description logic [14] has used Bayesian networks [15, 11, 16] to express probabilistic subsumption, which computes the probability that a random individual in class C is subsumed by class D. Zig & Ping [17] present an ongoing research on probabilistic extension to OWL using Bayesian networks. Additional cases are used to tag existing classes with prior probabilities. Using these prior probabilities and set of predefined translation rules OWL T-Box is converted into a Bayesian network to do reasoning of the conceptualization covered by the T-Box. Lukasiewicz [18] present expressive probabilistic description logic P-SHIF(D) and P-SHOIN(D) which are probabilistic extensions to description logics SHIF(D) and SHOIN(D) that OWL-Lite and OWL-DL is based on. This allows to express probabilistic knowledge about concepts and roles as well as assertional probabilistic knowledge about instances of concepts and roles. This work has been used as an extension in Pellet to reason an OWL ontology under probabilistic uncertainty [19]. This system processes statements such as “Bird is a subclass-of Flying Objects with probability 90%” or “Tweety is-a Flying Object with probability less than 5%”.

3.2 Never ending language learning (NELL)

Machine learning approaches, especially semi-supervised learning models, have shown a great improvement in extracting information from structured and unstructured text. This is mainly due to the fact that supervised training is very expensive and fully categorised examples are very hard to overcome. Semi-supervised learning methods use a small number of examples to bootstrap the learning process. The most recent work on learning a conceptualization is addressed by never ending language learning (NELL) research that is being done at CMU in the "Read the Web" project [20]. The conceptualization of millions of unlabelled web documents are learned from a coupled semi-supervised learning algorithm. The system is bootstrapped with an initial ontology that consists of concepts such as Company, SportTeam and relations such as CompanyHeadquarteredInCity(Company, City), SportsTeamHomeCity(SportsTeam, City) and a handful set of examples for conceptualization and set of constraints such as Person and Sports are mutually exclusive. Then, it will simultaneously learn classifiers for these concepts and relations using the constrains provided in the initial ontology. The goal of the system is to run 24/7 and to learn good classification of the concepts and relations in next iteration comparied to the current iteration. The project has been focused on several semantic categories and uses 200 millon web pages for the classification [21, 22, 23].

Coupled with this work is the work done on set expansion of named entities in the web [24, 25, 26] . This refers to expanding a given set of objects into a more complete set. The extended version of this called automatic set instance acquirer, which takes a semantic class as input (e.g. fruits) and automatically outputs its probable instances (e.g. apple, orange, banana).

3.3 Knowledge Acquisition

Knowledge acquisition is the transformation of knowledge from the form in which it is available in the world into machine readable forms that can infer useful results. It is not a trivial task to transfer domain knowledge to a machine readable form because of the interpretations may be too disambiguous. Hence, it is required to model the domain in a way that it does not lose the underline interpretation. Due to these intricacies knowledge acquisition and representation is a hard problem and it is known as "the knowledge acquisition bottleneck" [27]. Natural language processing deals with the problem of knowledge acquisition with part-of-speech tagging. This endeavour has been thoroughly investigated in knowledge acquisition from text. Many integrated tasks in natural language processing require a large amount of world knowledge to create expectations, assess plausibility and guide disambiguation. This quest still remains a formidable challenge. Building on ideas by Schubert, a system called DART (Discovery and Aggregation of Relations in Text) has been developed that extracts simple, semi-formal statements of world knowledge (e.g., air-planes can fly, people can drive cars) from text and this has used it to create a database of 23 million propositions of this kind [28]. An evaluation of the DART database on two language processing tasks (parsing and textual entailment) shows that it improves performance, and a human evaluation shows that over half the facts in it are considered true or partially true, rising to 70% for facts seen with high frequency. This work has created a new publicly available knowledge resource for language processing and other data interpretation tasks, and second it provides empirical evidence of the utility of this type of knowledge, going beyond Schubert et al's earlier evaluations which were based solely on human inspection of its contents. KNEXT database [29], TextRunner [30] and ConceptNet [31] similar to DART.

Recognising textual entailment (RTE) is the task to find out whether some text $\mathcal{T}$ entails a hypothesis $\mathcal{H}$. The recognition of textual entailment is without doubt one of the ultimate challenges for any NLP system: if it is able to do so with reasonable accuracy, it is clearly an indication that it has some thorough understanding of how language works. Logical inference is the most common method used in recognising textual entailment (RTE). Boss & Markert [32] uses model building techniques borrowed from automated reasoning to approximate entailment. DART uses DIRT [31] database to recognize textual entailment.

A common hypothesis is that there exists a large collection of general knowledge in texts, lying at a level beneath the explicit assertional content. These axioms infer logical consequences that are possible in the world or under certain conditions infers to be normal or commonplace in the world. Marinho et al. [33] had focused on deriving general propositions from noun phrase clauses and then to fortify stronger generalisation based on the nature and statistical distribution of the propositions obtained in the first phase. Cankaya & Moldovan [34] presents a semi-automatic method for generating commonsense axioms. This process first uses commonsense rules as a seed that describe some concept properties. Then the algorithm searches automatically in Extended WordNet [5] for all concepts that have a given property and generates axioms linking those concepts with the seed commonsense rule. The ultimate goal of knowledge capturing is to build systems that automatically construct a knowledge base by reading texts. This requires solving the problem of Natural Language Understanding (NLU) [35]. The main objective of NLU is to read texts to build a formal representation of their content in order to support a variety of tasks such as answering a query or RTE. Ambiguity is largely inherent to any natural language. This will cause a considerable challenge in full NLU understanding. The Learning-by-Reading system [36] focuses to answer the prior problem by integrating snippets of knowledge drawn from multiple texts to build a single coherent knowledge base, which has shown both feasible and promising. Some of the method we have shown so far uses either supervised or unsupervised machine learning algorithms to find a solution. The other aspect of the spectrum is active learning, a promising solution for named entity recognition [37]. Hearst [38] describes six lexico-syntactic patterns for automatic acquisition of the hyponym lexical relations from unrestricted text.

3.4 Ontology learning

Ontology learning from text considers extracting knowledge from textual data. Maedche, Staab & Volz [39, 40, 41] present a state-of-the-art semi-automatic approach to construction and maintenance of ontology from a domain specific text by applying machine learning techniques. This work addresses the usage of data mining techniques to simplify and accelerate ontology construction and ensures that the induced ontology faithfully reflects application data requirements. The implementation of this work is called TEXT-TO-ONTO. It follows the balanced cooperative modeling paradigm, where the interaction between knowledge acquisition and machine learning steps can be done either by a human or by a machine. ONTOEDIT, a stand-alone manual ontology engineering environment is fully embedded in TEXT-TO-ONTO, thus manual ontology construction is possible. The user can also modify propositions from TEXT-TO-ONTO extraction and maintenance components graphically or can reject them. The overall process involves, first, data is pre-processed by a resource processing component and a NLP system. Then, machine learning algorithms are applied to construct modelling primitives. These modelling primitives are integrated in a semi-automatic fashion to learn the overall conceptualization. This application also provides ontology pruning, which refers to the process of removing elements from the ontology that are no more relevant to a given application domain and ontology refinement, which focuses on learning the meaning of unknown words over time. Evaluation of the learned ontology is computed with the similarity between a manually-built reference ontology. It is assumed that a high similarity between the hand-modeled ontology and the ontology learning-based acquired ontology indicates a successful application of a particular ontology learning technique, which is formally known as gold standard as a reference. OntoLearn [42], OntoLT [43] and OntoClean [44] are some of the tools developed to support the user in constructing ontologies from a textual data. The prior mentioned application, OntoLearn, OntoLT and OntoClean, depends either on very specific or on proprietary ontology models, which can not always be translated to other formalisms in a straightforward way. This is certainly undesirable as ontology learning tools should be independent from a certain ontology model in order to be widely applicable and used.

Text2Onto [45] represents the learned ontological structures at a meta-level in form of so called modeling primitives rather than in a concrete knowledge representation language, which gives the flexibility on handling the most prevalent representation languages currently used within the Semantic Web such as RDFS, OWL and F-Logic. Text2Onto is a complete re-design of TEXT-TO-ONTO. Text2Onto uses Probabilistic Ontology Models (POMs) where the results of the system are attached by probabilities. In addition to this, it uses a data-driven change discovery, which is responsible for detecting changes in the corpus, calculating POM deltas with respect to the changes and accordingly modifying the POM without recalculating it for the whole document collection. Many reasoning-based applications in domains such as bioinformatics or medicine rely on much more complex conceptualizations rather bare taxonomies and relationships [32]. Some of these conceptualizations are constructed by pure manual efforts. Hence, methods for the automatic or semi-automatic construction of expressive ontologies could help to overcome the knowledge acquisition bottleneck. The amount of post-processing for complex learned ontologies can be relaxed if proper integration of ontology evaluation and debugging approaches are introduced. Particularly, the treatment of logical inconsistencies, mostly neglected by existing ontology learning frameworks, becomes a great challenge as soon as we start to learn huge and expressive conceptualizations. LexO [32] is such an implementation supporting the automatic generation of complex class descriptions from lexical resources. The learned ontologies may represent uncertain and possibly contradicting knowledge [46]. This is mainly due to imprecision, inconsistency or uncertainty. Imprecision and inconsistency are properties of the knowledge base cause by ambiguous, vague, approximate and contradictory conclusions derived from the information. Uncertainty means that an agent, i.e., a computer or a human, has only partial knowledge about the knowledge base. This has two uncertain categories. An objective uncertainty relates to randomness referring to the propensity or disposition of something to be true. A subjective uncertainty depends on an agents opinion about the truth value of information. In either case, agents can assume knowledge as unreliable or irrelevant. Ontology learning is subjected to imperfection from ambiguity and unreliability, thus, it is subjective uncertainty. Text2Onto [45] generates ontologies based on a Learned Ontology Model (LOM), which is independent of a concrete ontology representation language. LOM represents uncertainty as annotations capturing the confidence about the correctness of the ontology elements. Then the LOM is transformed a standard logic-based ontology language, in order to be able to apply standard reasoning over the learned ontologies. Our research work is also focused on the ontological commitment dealing with uncertainty produced by the text.

Formal Concept Analysis (FCA) [47] is a method that automatically aquisits taxonomies or concept hierarchies from text corpora. In the FCA paradigm, a concept consist of two parts, which is known as extension and intention. Extension covers all objects belonging to this concept and intention comprises all attributes valid for all those objects. These objects and attributes are used in deriving the subconcept-superconcept relations between concepts with respect a formal context. This formal context later translates into a lattice from which the partial order of the concept hierarchy is learned. Since objects and attributes describe the extent and intent, it is important to extract theses from the text using a NLP concept. Cimiano et al.

[47] collects verb/subject, verb/object and verb/propositional phrase dependencies to create the formal context and some of them will be pruned according to a information measure.

3.5 Ontology population

The other side of ontology learning is ontology population. Ontology population means finding instances of the conceptualization [48]. Finding instances of the conceptualization is treated as a difficult problem. For some degree, work that is being done in [21, 22, 23] addresses this issue using coupled co-training. Human-defined concepts are fundamental building-blocks in constructing knowledge bases such as ontologies. Statistical learning techniques provide an alternative automated approach to concept definition, driven by data rather than prior knowledge. Chemudugunta et al. [49] performs automatically taging of Web pages with concepts from a known set of concepts (i.e. an ontology) without any need for labeled documents using latent Dirichlet allocation models.

NLP is used in ontology population, using a combination of rule-based approaches and machine learning [50]. Linguistic and statistical technique methods are used for contextual information to bootstrap learning. Named entities are populated using a weakly supervised automatic approach in [51]. A syntactic model is learned for categories using an ontology. Then, this model is populated using the corpus. Pantel & Pennacchiotti [52] presents algorithms for harvesting semantic relations from text and then automatically linking the knowledge into existing semantic repositories.

Table 3.1: Summary of the related work

Related Word	**Purpose**	**T-Box**	**A-Box**	**Method**
PR	reasoning	available	available	prob. theory
NELL	24×7 learning	fixed	dynamic	ML techniques
POS	type for a word	$\times$	$\times$	prob. & stat.
DART	world knowledge	$\times$	$\times$	semi-automated
RTE	entailment	$\times$	$\times$	ATP
NLU	commonsense rules	$\times$	$\times$	semi-supervised
Text-To-Onto	ontology learning	$\surd$	$\surd$	semi-supervised
Text2Onto	ontology learning	$\surd$	$\surd$	semi-supervised
Lexo	complex classes	$\surd$	$\times$	semi-supervised
FCA	taxonomy	$\surd$	$\times$	FCA
OP	ontology population	available	available	semi-/supervised

3.6 Summary

Tabel 3.1 summarises the attributes of prior mentioned methods. Learning a conceptualization of a domain spans accross different fields. It starts with parsing the structured and unstructured text for syntactic patterns, then from those syntactic patters semantics of the corpus is build. The parsing process is heavily dependent on the accuracy of the natural language processing techniques. After the parsing, it is necessary to build populate the given model. Almost all the methods we have discussed so far are semi-automated. This means that at one stage of the processing chain, a domain expert is needed to formalize the final outcome. Most system uses a domain expert as the authority that handle the uncertainty immerged from the system. Due this reason ontology learning process has become time consuming and cumbersome.

In our work we will formalise a unsupervised method to build an ontology using probabilistic theory. Probabilistic theory quantifies the uncertainty and learn the concepts and relations of the conceptualisation. Since the process is unsupervised, we hypothesis the ontology learning process is much more faster than the prior mentioned method. Chapter 4 provides a detailed discussion of our method through probabilistic reasoning.

Chapter 4

Approach

Chapter 3 has mentioned the pros and cons inherited from different techniques to solve the problem of ontology learning. As shown in table 3.1, each method covers some portion of the problem. Each method learns a conceptualization from terms, and present it as taxonomies and axioms for ontology. On the other hand most of the methods use a top-down approach, i.e., an initial classification of an ontology is given. The uncertainty inherited from the domain is most of the time dealt with by a domain expert, and the conceptualization is defined most of the time using predefined rules or templates. These have shown characteristics of a semi-supervised and semi-automated learning environment.

In our work, we are focusing on an unsupervised method to quantify the degree of belief that a grouping of words in the corpus will provide a substantial conceptualization of the domain of interest. Degree of belief in world states influence the uncertainty of conceptualization. Uncertainty arises from partial observability, non-determinism, laziness and theoretical and practical ignorance [10]. Partial observability arises from the size of the corpus. Even though the corpus is substantially large enough, it might not contain all the necessary evidence of an event of interest. A corpus contains ambiguous statements about an event that will lead to non-determinism of the state of the event. Laziness arises from work that needs to be done in order to learn exceptional rules and it is too hard to learn such rules. Theoretical and practical ignorance arises from lack of complete evidence and it is not possible to conduct all the necessary tests to learn a particular event. Hence, domain knowledge, in our case domain conceptualization, can at best provide only a degree of belief of the relevant groups of words. We use probability theory to deal with the degrees of belief. As mentioned in chapter 2, probability theory has the same ontological commitment as the formal logic, though the epistemological commitment differs.

4.1 Overall process

The overall process of learning and presenting a probabilistic conceptualization is divided into four phases as shown in figure (4.1). They are,

1. Pre-processing
2. Syntactic analysis
3. Semantic analysis
4. Representation

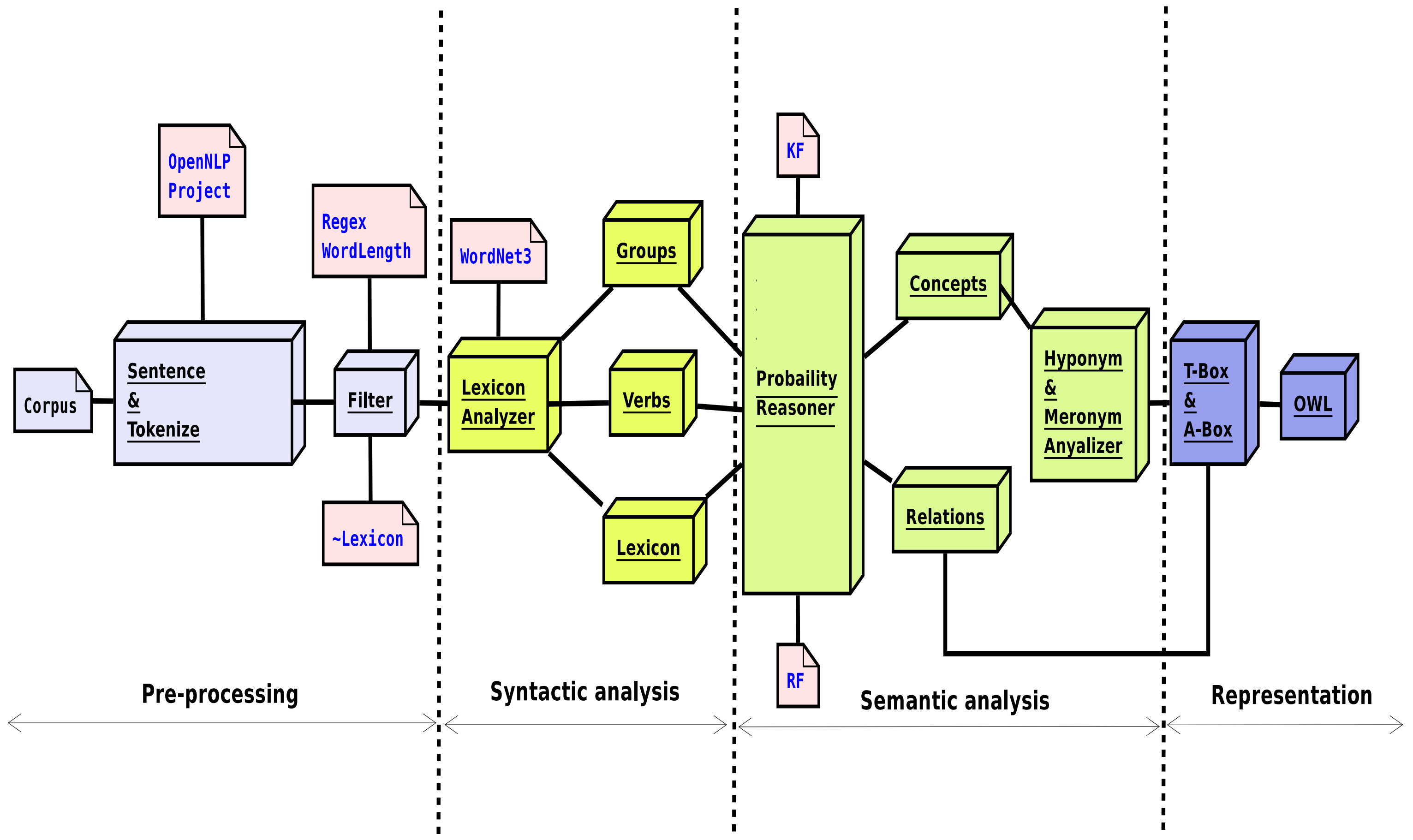


Figure 4.1: Overall process: Process categorizes into four phases; pre-processing, syntactic analysis, semantic analysis & Representation

4.1.1 Pre-processing

A corpus contains plethora of structured and unstructured sentences build from a lexicon. A lexicon of a language is its vocabulary build from lexemes [12, 53, 54]. A lexicon contains words belonging to a language and in our work individual words from the corpus will be treated as the vocabulary, thus, the lexicon of the corpus. In pure form, the lexicon may contain words that appear frequently in the corpus but have little value in formalising a meaningful criterion. These are called stop words or in our terminology negated lexicon will be excluded from the vocabulary. The defintion of the lexicon of our work is given as,

Definition 4.1 *The lexicon $\mathcal{L}$ is the set that contains words belong to the universe of English vocabulary, which is part-of-speech type tagged with the Penn Treebank English POS tag set [55] and the type of the word is given in table 4.1.*

Table 4.1: The definition of the lexicon $\mathcal{L}$ is based on the subset of the Penn treebank tags set

Term	**Description**
NN	Noun, singular or mass
NNP	Proper Noun, singular
NNS	Noun, plural
NNPS	Proper Noun, plural
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present

Definition 4.1 implicitly implies that the negated lexicon $\overline{\mathcal{L}}$ is the set that contains English words that are part-of-speech tagged with the Penn Treebank English POS tag set other than the tags given in definition 4.1. In addition, the word length W_L above some threshold W_{L_T} is also considered when building the lexicon of the corpus. with respect to part-of-speech context, the length of a word is the sequence of characters or symbols that made up the word. i.e., the word ”mika” has a word length of four ($W_L = 4$). By default we have considered a world having a length of more that 2 is sufficiently formalise with some criterion.

Building up the pure lexicon at this stage excluding the negated lexicon of the pre-processing is known as tokenization from sentences. Here, the pure form of the lexicon might contain words that need to be further purified according to some criterion. Words of the corpus contain a lot of standard and constructed words. As mentioned, some words do not provide useful information. In order to filter out these words, in the next phase of the pre-processing phase, each word is processed through a regular expression filter. The regular expression filter is a parameter to the system. The default regular expression is given as $[a-zA-Z]+$ if this parameter is not specified by the user. As an example: a word such as $du-145$ will be filtered out from this regular expression. We also try to do token normalization to some extent. This is the process of canonicalizing the tokens so that matches occur despite superficial differences in the character sequences of the tokens. In the next step, the vocabulary learned from the corpus is subjected to case-folding by reducing all letters to lower case. As an example: *Protocol* will be case-folded to *protocol*. Documents use different forms of a word such as *organize*, *organizes* and *organizing* for grammatical reasons.

In addition to this there are families of derivationally related words with similar meanings. We use stemming and lemmatization to reduce the inflectional forms and derivational forms of a word to a common base form. We achieve this from the aid of WordNet's [5] stemming algorithms, using the first sense of a word.

4.1.2 Syntactic analysis

Once the pre-processing phase eliminates the noise of the corpus, in the second phase, the system tags the word according to its part-of-speech classes. Part-of-speech classes include nouns, verbs, adjective, adverbs, prepositions etc. We are mainly interested in nouns, adjective and verb forms as given in defintion 4.1. We assume that each sentence of the corpus follows the part-of-speech pattern in expression 4.1,

$$(NounPhrase+)(Verb+)(NounPhrase+) \tag{4.1}$$

Using the first word sense of WordNet synsets we derive the tags of each word according to equation 4.1. Up to this point, the process will classify the lexicon according to its part-of-speech. We hypothesis that the lexicon learned from this stage provides the potential candiates for concepts and relations of the ontology. But the lexicon itself does not provide sufficient ontology concepts. We group words that are related to form an OWL concept. This grouping is done using the N-gram model as discussed in the chapter 2. Figures 4.2 and 4.3 provide examples of a 2-gram and a 3-gram model.

According to this figure 4.2 group $w_1|w_2$ forms a potential concept in the concep-

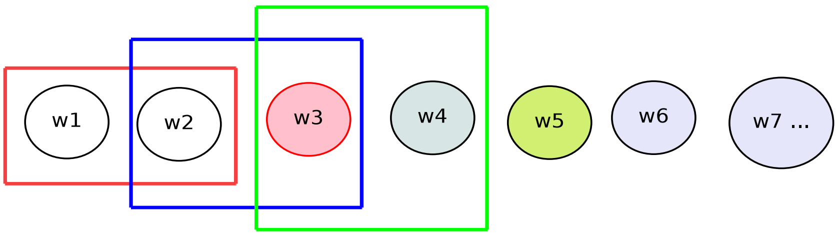


Figure 4.2: 2-gram model example

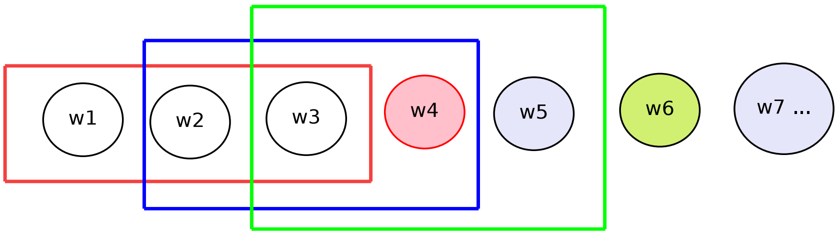


Figure 4.3: 3-gram model example

tualization. Groups $w_2|w_3$, $w_3|w_4$ etc. form other potential concepts in the conceptualization. Word w_3 comes after group $w_1|w_2$. According to Bayes viewpoint, we collect the information to estimate the probability $p(w_3|\{w_1|w_2\})$, which will be used in forming IS-A relationships, $w_1|w_2 \sqsubseteq w_3$ using an independent Bayesian network with conditional probability $p(\{w_1|w_2\}|w_3)$. In this phase, all this information is collected as frequencies and sent to the third phase. In addition to this, we collect the frequencies of patterns discussed in section 4.1, which are used in the third phase to create the relations among concepts.

The bootstrap algorithm 1 gives the main operations conducted by the preprocessing and syntactic analysis phases. $Corpus, Regex$, and $NegatedLexicon$ are inputs to the algorithm 1, and ensure frequencies of concepts G_F, verbs V_F, lexicon L_F, part-of-speech tag POS_F and sets G, V, L, POS and $\{W|G\}$. Variable $Corpus$ contains the set of text documents of the corpus. This is given as $d_i \in Corpus$ where d_i represents the document and $i = 1, \ldots M$. M is the size of the $Corpus$. Each d_i

Algorithm 1 $BOOTSTRAP(Corpus, Regex, NegatedLexicon)$

```
Require: Substantially large text corpus $Corpus$, filtering regular expression $Regex$
    and stop words $NegatedLexicon$
Ensure: Frequencies of concepts $G_F$, verbs $V_F$, lexicon $L_F$, part-of-speech tag $POS_F$
    and sets $G, V, L, POS$, $\{W|G\}$ and $\{G|V\}$
1: $L = \emptyset, G = \emptyset, POS = \emptyset$
2: for $d_i \in Corpus$ do
3:   for $s_j \in d_i$ do
4:     for $w_k \in s_j$ do
5:       if $w_k \in Regex$ && $w_k \notin NegatedLexicon$ then
6:         if $WORDNET\_STEM\_AS\_NOUN\_OR\_ADJ(w_k, w_{k_s})$ then
7:           if $w_{k_s} \notin NegatedLexicon$ then
8:             $L = L \bigcup w_{k_s}$
9:             $L_{w_{k_s}} ++$
10:            $g_l = N\_GRAM\_GENERATOR(w_{k_s})$
11:            $G = G \bigcup g_l$
12:            $G_{g_l} ++$
13:            $\{w_{k_s}|g_l\} ++$
14:          end if
15:        else if $WORDNET\_STEM\_AS\_VERB(w_{k_s}, w_{k_s})$ then
16:          if $w_{k_s} \notin NegatedLexicon$ then
17:            $V = V \bigcup w_{k_s}$
18:            $V_{w_{k_s}} ++$
19:          end if
20:        end if
21:        if $\exists w_{k_s}$ then
22:          $GENERATE\_BINARY\_RELATION(L_{w_{k_s}}, V_{w_{k_s}}, g_l, POS, POS_F, \{G|V\})$
23:        end if
24:      end if
25:    end for
26:  end for
27: end for
```

contains a finite number of sentences s_j. Let $s_j \in d_i$ and let the maximum number of sentences that will be seen by this operation is given by $max(s_j)$. Each sentence consists of finite number of words $w_k \in s_j$. Let the maximum number of words that will be seen in a sentence is given by $max(w_k)$. The word w_k matches against the input regular expression $Regex$ and if it succeeds, then it is checked against the stop word set $NegatedLexicon$. If both operations are succeeded, then w_k is subjected to $WORDNET\_STEM\_AS\_NOUN\_OR\_ADJ$ WordNet first sense check as a noun or an adjective and stemmed using WordNet stemming algorithm to produce w_{k_s}. If w_{k_s} is a noun or an adjective it is added to the lexicon set L and increment the occurrences of w_{k_s} with $L_{w_{k_s}}$. Using w_{k_s} and the N-gram generator $N\_GRAM\_GENERATOR$, group g_l is created for the system. Then the generated g_l is added to the group set G and increment the occurrences of g_l with G_{g_l}. Finally, the algorithm increments the $\{w_{k_s}|g_l\}$ count, which is used by the probabilistic phase. The second option is to check whether w_k is a verb. The sub-routine $WORDNET\_STEM\_AS\_VERB$ is used to check whether w_k is a verb and if it succees, it will produce output w_{k_s}, which will be added to verb set V and increment the occurrences of w_{k_s} with $V_{w_{k_s}}$. Finally, $GENERATE\_BINARY\_RELATION$ uses $L_{w_{k_s}}$ and $V_{w_{k_s}}$ to generate the binary relations POS and associated counts POS_F if-and-only-if w_{k_s} exists. In addition to this, $\{G|V\}$ frequencies will be calculated in order to build the relations probabilities in later sections. The asymptotic running time of the algorithm 1 is,

$$O(M \times max(s_j) \times max(w_k)).$$

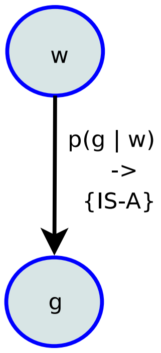


Figure 4.4: Probabilistic IS-A relationship representation of the conceptualization (4.2). w and g are defined as the concepts of the conceptualization.

4.1.3 Semantic analysis

The third phase of the process is semantic analysis with probabilistic reasoning, which constitutes the important operations of our work. This phase determines the conceptualization of the domain calulating probabilities for IS-A relations and relations among the concepts. In addition to this, in order to provide a useful taxonomy we induce concepts from clustered concepts. Our defintion of concept learning is given in 4.2.

Definition 4.2 *The set $W = \{w_1, \ldots, w_n\}$ represents a n-independent lexems of the lexicon L and each w_i has a prior probability θ_i. The set $G = \{g_1, \ldots, g_m\}$ represents m-independent N-gram groups learned from the corpus and each g_j has a prior probability η_j. When $w \in W$ and $g \in G$, $p(w|g)$ is the likelihood probability π learned from the corpus. The entities w and g represent the potential concepts of the conceptualization. Within this environment, an IS-A relationship between w and g is given by the posterior probability $p(g|w)$ and this is represented with a Bayesian network having two nodes w and g as shown in the figure 4.2 and,*

$$p(g|w) = \frac{\pi \times \eta}{\sum_i p(w|g_i) \times p(g_i)}. \tag{4.2}$$

Lets define the knowledge factor lowerbound that select the super-concept of the conceptualization.

Definition 4.3 *$W = \{w_1, \ldots, w_n\}$ represents n-independent lexems of the lexicon L and each w_i has a prior probability θ_i. Lets define $knowledgefactor$ KF as the lowerbound; if $\theta_i \geq \tau$ with $0 \leq \tau \leq 1$ then w_i is considered as a super-concept of the conceptualization.*

Definition 4.3 states that w of the defintion 4.2 is considered as a super-concept of the conceptualization.

Definition 4.4 *Probabilistic conceptualization of the domain is represented by n-number of indendent Bayesian networks sharing groups.*

Figure 4.5 shows a situation where multiple Bayesian networks share a common group g_2. According to this definition consider a set G contains n-number of finite random variables $\{g_1, \ldots, g_n\}$. There exist a group g_i, which is shared by m words $\{w_1, \ldots, w_m\}$. Then, with respect to Bayesian framework, BN_i of $p(g_i|w_i)$ is calculated and $max(p(g_i|m_i))$ is selected for the construction of the ontology. This means that if there exists two Bayesian networks and Bayesian network one is given by the pair w_1, g_1 and Bayesian network two is given by the pair $\{w_2, g_1\}$ then the Bayesian network that has the most substantial IS-A relationship is obtained through

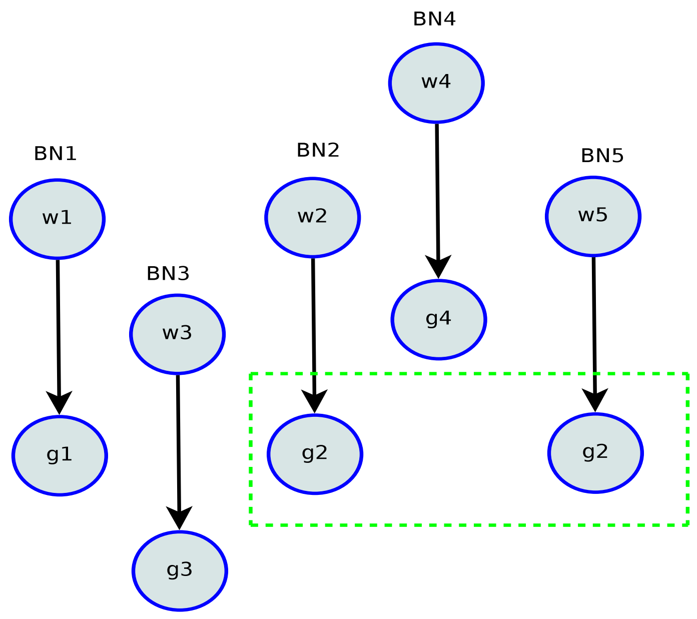


Figure 4.5: w_1, w_2, w_3, w_4 and w_5 are super-concepts. g_1, g_2, g_3 and g_4 are candidate subconcepts. There are 5 independent Bayesian networks. Bayesian networks 2 and 5 share the group g_2 when representing the concepts of the conceptualization

$max_{BN_i}(p(g_1|w_1))$ and this network is retained and other Bayesian networks will be ignored when building the ontolgoy. If all $p(g_1|w_1)$ remains are equal, then the Bayesian network with the highest super-concept probability will be retained. These two conditions will resolve any naming issues. Let's illustrate prior definitions in action with an example.

Example 4.1 *Let's illustrate the concept defintions mentioned in 4.2-4.4. Assume we are using a 2-gram model to gather groups. The set* L_W *and* L_{W_F} *show the lexicon and their relative frequencies. Sets* G_G *and* G_{G_F} *show the groups and their relative frequencies. Sets* $P_{W|G}$ *and* $P_{(W|G)_F}$ *represent the group/word (word* $\in$ *Lexicon) dependencies and their relative frequencies. Table 4.2 shows the sets and prior probabilities. Lets assume that the knowledge factor (KF) for the problem is* 0.5. w_1, w_3, w_4 *and* w_8 *has prior probabilities that is accepted by the system. Then, we need to cal-*

culate $P(G|W)$ in order to obtain posterior probability. Lets calculate the posterior probability of $P(g_1|w_1)$. This will be given by,

$$P(g_1|w_1) = \frac{P(w_1|g_1) \times P(g_1)}{\sum_g P(w_1|g) \times p(g)}$$

$$P(g_1|w_1) = \frac{0.66667 \times 0.103448}{(0.66667 \times 0.103448 + 0.66667 \times 0.310345 + 1.00000 \times 0.034483)}$$

$$P(g_1|w_1) = 0.22222$$

Lets calculate $p(g_1|w_1)$,

$$p(g_1|w_1) = \frac{0.33333 \times 0.103448}{0.33333 \times 0.103448 + 0.16667 \times 0.206897}$$

$$p(g_1|w_1) = 0.5$$

Hence, the calculate posterior probability 0.22222 is the degree of belief that group g_1 or the concept, according to our lexico-semantic definition, has a IS-A relationship with concept w_1. Figure 4.6 shows the ontology for the taxonomy we have just learned.

Figure 4.2 shows that the concept $w_2|w_3$ is a subconcept of w_1 as well as w_8. According to the defintion 4.4 this creates a violation. Since $p(w_8|g_1) > p(w_1|g_1)$ we remove the weaker IS-A relation from the conceptualization.

The next step is to induce the relationships in Figure 4.6 to complete the conceptualization. In order to do this, we need to find semantics associated with each verb.

Definition 4.5 *Given a subset of concepts $G_S = \{g_1, \dots, g_n\}$, $G_S \subset G$, with size*

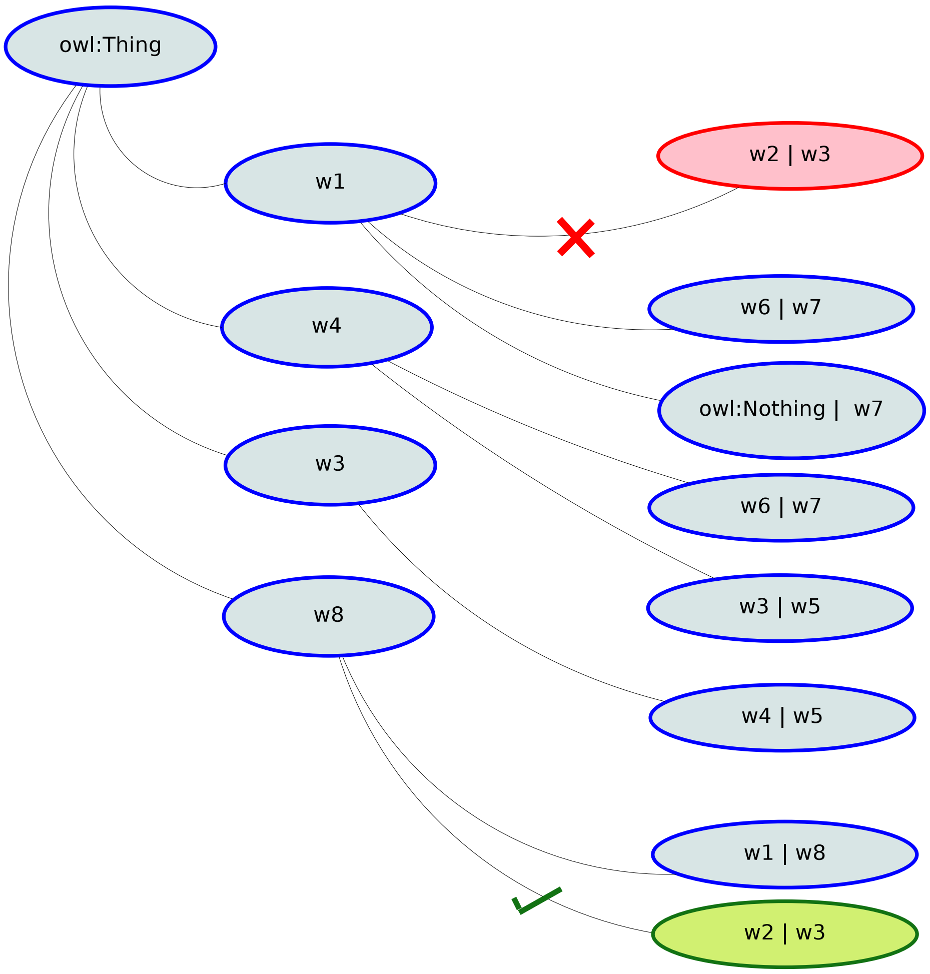


Figure 4.6: Hypothetical example ontology (this shows only the taxonomy)

Table 4.2: Hypothetical data points that will be used to illustrate the example

L_W	L_{W_F}	L_{W_P}	G_G	G_{G_f}	G_{G_P}	$P_{W\|G}$	$P_{(W\|G)_F}$	$P_{(W\|G)_P}$
w_1	10	0.0884						
w_2	9	0.0789						
w_3	20	0.1769						
w_4	30	0.2654						
w_5	5	0.0442						
w_6	8	0.0707						
w_7	9	0.0796						
w_8	13	0.1150						
w_{v_1}	7	0.0619						
w_{v_2}	3	0.0265						
			$g_1 = \{w_2, w_3\}$	3	0.1034			
			$g_2 = \{w_4, w_5\}$	7	0.2413			
			$g_3 = \{w_6, w_7\}$	9	0.3103			
			$g_4 = \{\chi, w_7\}$	1	0.0344			
			$g_5 = \{w_3, w_5\}$	3	0.1034			
			$g_6 = \{w_1, w_8\}$	6	0.2068			
						$P(w_1\|g_1)$	2	0.6667
						$P(w_1\|g_3)$	6	0.6667
						$P(w_1\|g_4)$	1	1.0000
						$P(w_4\|g_3)$	5	0.5556
						$P(w_4\|g_5)$	1	0.3333
						$P(w_3\|g_2)$	6	0.8571
						$P(w_8\|g_1)$	1	0.3334
						$p(w_8\|g_6)$	1	0.1667
						$p(w_i\|g_j)$	0	0

n, for a given super-concpet w, when $p(g_1|w), \ldots, g(g_n|w)$ holds, the prefixes of the concepts are extracted and known as an induced concepts. For a m-gram model, at most up to $m-1$ concepts can be induced. For all induced concepts c, the concepts name collision will be avoided by assigning different namespaces. The induced concept will be given a prior probability of 0.

Definition 4.5 gives an efficient way to represent the taxonomy of the conceptualization. Newly induced concepts contain words up to at most $m-1$. Thus, this concepts induction will lead to concepts collision in the given namespace. This situation is avoided according to Defintion 4.6.

Definition 4.6 *When a concept is induced from a group of concepts, the induced concept is assigned to a different namespace in order to avoid possible concept name conflicts. The namespace assignment is forced, if and only if there exist a concept with the same name in the system, otherwise induced concepts will be subjected to the default namespace of the system.*

Example 4.2 illustrates an instance of concept induction process.

Example 4.2 *Lets assume a learning instance uses a* 3 *gram model. We learn the following concepts from the system. There exist a super-concept W and subconcepts $w_1|w_2|w_3$, $w_1|w_2|w_4$, $w_1|w_5|w_6$, $w_7|w_8|w_9$. For this configuration, the induced concept hierarchy is as shown in Figure 4.8. w_1 and $w_1|w_2$ are induced concepts of the system. They will be assigned to different namespaces if the conditions of definition 4.6 is met.*

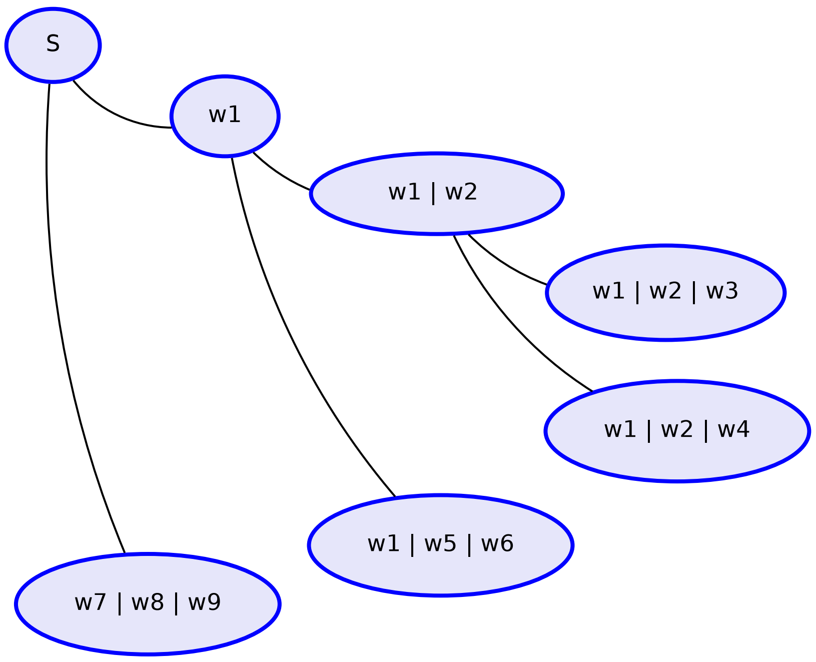


Figure 4.7: Induced concept hierarchy

Relations are as important as concepts in a conceptualization. Relations exists among the concept of the conceptualization. We have hypothesized that relations are generated by the verbs in the corpus.

Definition 4.7 *Relationships of the conceptualization are learned from the syntactic structure model by the equation 2.14 and the semantic structure model by the equation 2.15.*

Definition 4.8 *If there exists a verb V between two groups of concepts C_1 and C_2, the relationship of the triple (V, C_1, C_2) is written as $V(C_1, C_2)$ and model with conditional probability $p(C_1, C_2|V)$. The Bayesian network for relationship is and the model semantic relationship is given by,*

$$p(C_1, C_2|V) = p(C_1|V)p(C_2|V) \rightarrow V(C_1, C_2)$$

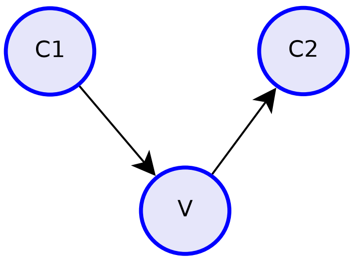


Figure 4.8: Bayesian networks for relations modeling. C_1 and C_2 are groups and V is a verb

Using definitions 4.7 and 4.8, the relationship among multiple concepts are defined in 4.9. We define the relations in terms of groups of words in the lexicon. These groups are clustered around the most probable words found in the corpus.

Definition 4.9 *Let $S_p \subset S$ be a part of co-occurance sentence of the corpus, which can be transformed into $\{G_i \; v_j \; G_k\}$ groups and a verb. The sizes of G_i and G_k are $|G_i|$, $|G_k|$ and $G_i = \{g_1, \ldots, g_m\}$ and $G_k = \{g_{m+1}, \ldots, g_n\}$, $n > m$. Then, the relationships among G_i and G_k are build from the combinations of the elements from G_i and G_k with respect to v_j in accordance with the Bayesian model $p(G_i, G_k|V_j)$. There will be $|G_i| \times |G_k|$ relations,*

$$v_j(g_1, g_{m+1}) \leftarrow p(g_1, g_{m+1}|v_j)$$

$$v_j(g_1, g_{m+2}) \leftarrow p(g_1, g_{m+2}|v_j)$$

$$\ldots$$

$$v_j(g_1, g_n) \leftarrow p(g_1, g_n|v_j)$$

$$v_j(g_2, g_{m+1}) \leftarrow p(g_2, g_{m+1}|v_j)$$

$$\ldots$$

$$v_j(g_m, g_{m+1}) \leftarrow p(g_m, g_{m+1}|v_j)$$

$$\ldots$$

$$v_j(g_m, g_n) \leftarrow p(g_n, g_m|v_j)$$

The relations learned from defintions 4.7 and 4.8 sometimes needs to be subjected to a lower bound. The Knowledge Factor (KF) parameter is used as an input to semantic analysis phase to set this lower bound.

Definition 4.10 *Let set $R = \{v_1(C_1, C_2), \ldots, v_m(C_k, C_r)\}$ be the relations that are learned from the corpus. Relations $v_i(C_j, C_k)$ are assigned a probability using a Bayesian model $p(C_j, C_k|V_i)$. When these relations are ordered based on their probability, a threshold φ is defined as the Knowledge Factor (KF) of the system.*

Definition 4.10 allows the user to limit the number of relations learned from the system. When the corpus is substantially large, the number of relations that are learned is proportional to the number of verbs in the lexicon. Not all relations may relevant and the KF is used as the limiting factor.

Definition 4.11 *Let v_i be a verb and v_j is the antonym verb of v_j learned from WordNet ($v_i \bowtie v_j$). Let there be relations $v_i(G_m, G_n)$ and $v_j(G_m, G_n)$ modeled by $p(G_m, G_n|v_r)$ ($v_r = i, j$). Since, $v_i \bowtie v_j$ for G_m and G_n, the relationship with the highest $p(G_m, G_n|v_r)$ value will be selected and the other relationship will be removed.*

We are using verbs of the lexicon as the key elements in forming relationships amoung concepts. Verbs have opposite verbs. Thus, according to definition 4.11, if a verb is associated with some concepts and these concepts happen to be associated with a opposite verb, the verb with the highest Bayesian probability value is selected for the relations map and the other relationship will be removed from the system.

Example 4.3 *Let's extend the example 4.1 with some relations. If the system learns that there is a $v_1(g_1, g_2)$ relation and a $v_2(g_6, g_5)$ relation, then we need to find the probabilities $p(g_1, g_2|v_1)$ and $p(g_6, g_5|v_2)$. The frequencies of these values are found from the booststrap algorithm and they are converted into probabilities later. Lets assume that we found these probabilities, $p(g_1|v_1) = 0.14$, $p(g_2|v_1) = .42$, $p(g_6|v_2) = 1$, and $p(g_5|v_2) = 0.33$. Then according to definitions 4.8-4.11,*

$$v_1(g_1, g_2) \leftarrow p(g_1, g_2|v_1) = 0.06 \text{ and } v_2(g_6, g_5) \leftarrow p(g_6, g_5) = 0.33.$$

This is shown in figure 4.9.

We will now provide the algorithms that are implemented according to the defintions described in this section. Algorithm 2 provides the logic to calculate super-concepts and taxonomy of the corpus subjected to KF. The algorithm 2 requires frequencies of concepts G_F, lexicon L_F, verbs V_F, $\{W|G\}$ and knowledge factor $KF = (0, 1]$ as input, and provides super-concepts and taxonomy IS-A relationship $BN_{SuperConcepts,Groups}$ as the output. First, the algorithm calculates prior probabilities of $p(w_i) = \theta_i$, $p(g_j) = \eta_j$ and likelihood probability $p(w|g)$. Then all $p(w_i)$ are sorted

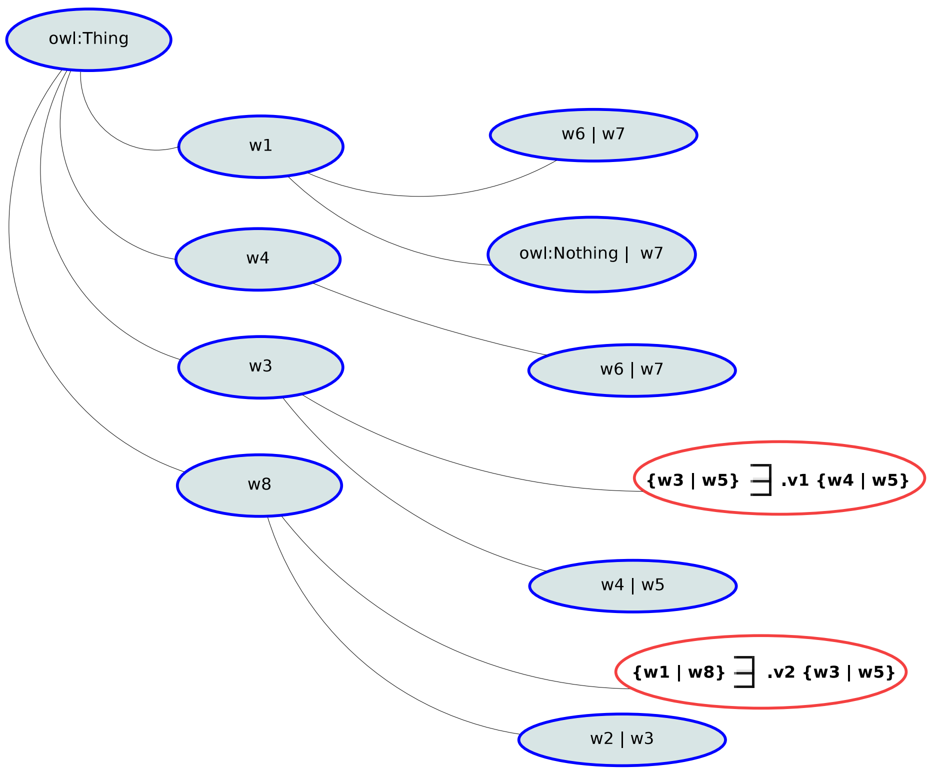


Figure 4.9: Hypothetical example ontology with relations (only two relations are shown)

and only the top $KF \times |BN_{SuperConcepts}|$ will be selected as valid super-concepts. Then, for all valid super-concepts posterior probability of $p(g|w)$ is calculated. If the calculated probability $p(g|w)$ is the highest, the associated super-concept and groups will be added to $BN_{SuperConcepts,Groups}$.

Algorithm 2 $SUPER\_CONCEPTS\_AND\_TAXONOMY(G_F, L_F, V_F, \{W|G\}, KF)$

```
Require: Frequencies of concepts $G_F$, lexicon $L_F$, verbs $V_F$, $\{W|G\}$ and knowledge
    factor $KF$
Ensure: Super-concepts and taxonomy IS-A relationship $BN_{SuperConcepts,Groups}$
 1: $BN_{SuperConcepts} = \emptyset$, $BN_{Groups} = \emptyset$ and $BN_{SuperConcepts|Groups} = \emptyset$
 2: for $w_i \in W$ do
 3:   Calculate prior probability $\theta_i$ for $w_i$ using $L_F$ and $V_F$.
 4:   $BN_{SuperConcepts} = BN_{SuperConcepts} \bigcup w_i$
 5: end for
 6: for $g_i \in W$ do
 7:   Calculate prior probability $\eta_i$ for $g_i$ using $G_F$
 8:   $BN_{Groups} = BN_{Groups} \bigcup g_i$
 9: end for
10: for $\{w|g\}_i \in \{W|G\}$ do
11:   Calculate likelihood probability $\pi_i$ for $\{w|g\}$ using $\{W|G\}$
12:   $BN_{SuperConcepts|Groups} = BN_{SuperConcepts|Groups} \bigcup \{w|g\}_i$
13: end for
14: Sort $BN_{SuperConcepts}$. Keep only top $KF \times |BN_{SuperConcepts}|$ fraction of
    $BN_{SuperConcepts}$ and marked them as valid
15: for $\forall_{valid} w_i \in BN_{SuperConcepts}$ do
16:   Calculate posterior probability $p(g|w_i)$ using $\theta_i$, $\eta_i$ and $BN_{SuperConcepts|Groups}$
      according to definition (4.4)
17:   if $p(g|w_i)$ is largest among $\forall w_i$ then
18:     $BN_{SuperConcepts,Groups} = BN_{SuperConcepts,Groups} \bigcup \{w_i, g\}$
19:   end if
20: end for
```

The output of alogrithm 2 will be used to induce concepts according to algorithms (3) and (4). Algorithm (3) uses $BN_{SuperConcepts,Groups}$ produced from algorithm (2) as input and output a tree $Tree_{SuperConcpet,InducedConcpets,Groups}$ containing all super-concepts, induced concepts and groups. This algorithm simply goes through each element of set $BN_{SuperConcepts,Groups}$ invoking the recursive algorithm 4.

Algorithm 3 $INDUCED\_CONCEPT\_WRAPPER(BN_{SuperConcepts,Groups})$

```
Require: Super-concept and groups relation $BN_{SuperConcepts,Groups}$
Ensure: Tree $Tree_{SuperConcept,InducedConcepts,Groups}$ consists of super-concepts, induced concepts and groups.
1: for $\forall SuperConcept_i, Groups \in BN_{SuperConcepts,Groups}$ do
2:     $INDUCED\_CONCEPT(Tree_{SuperConcept_i,InducedConcepts,Groups}, Groups, 1)$
3: end for
```

Algorithm 4 $INDUCED\_CONCEPT(Tree_{SuperConcpet_i,InducedConcpets,Groups}, \text{Groups, depth})$

```
Require: Tree $Tree_{SuperConcept_i,InducedConcepts,Groups}$ data structure, groups $Groups$ and depth $depth$
Ensure: Induced concepts for $Groups$ with super-concept $SuperConcept_i$.
1: if $depth <= (|n - gram| - 1)$ then
2:     $G \in Groups$ is of the form $w_1|w_2|\dots|w_n$. Create set $Prefix$ using $depth$ $w_i$ words
3:     Cluster all elements in $Groups$ to associated $Prefix$, $(PrefixGroups = \{prefix_i, Groups_k \in Groups\})$
4:     if $|Groups_k| == 1$ then
5:         Add $Groups_k$ to $Tree_{SuperConcpet_i,\_,Groups_k}$
6:     else
7:         if $\exists Prefix_i$ then
8:             Create new namespace $Namespace_j$
9:             Add $Prefix_i$ to $Tree_{SuperConcept_i,Prefix_i,Groups}$ with namespace $Namespace_j$
10:        else
11:            Add $Prefix_i$ to $Tree_{SuperConcpet_i,Prefix_i,Groups}$
12:        end if
13:        $INDUCED\_CONCEPT(Tree_{SuperConcpet_i,Prefix_i,Groups}, Groups_k, (depth + 1))$
14:    end if
15: else
16:     Add $Groups$ to $Tree_{SuperConcept_i,InducedConcept,Groups}$
17: end if
```

$Tree_{SuperConcept_i,InducedConcepts,Groups}$, $Groups$ and $depth$ are the inputs to the algorithm 4 with $SuperConcept$ and $depth$ is initialized to $SuperConcept_i$ and one. This algorithm is applied recursively until $depth <= (|n - gram| - 1)$ according to definition 4.5. $|n - gram|$ is the length of the n-gram. As an example: 2-gram has a length of 2. Each group $g_i \in Groups$ is of the form $w_1| \dots |w_n$. First, take the $depth$ prefix from the groups and cluster them according the prefix. If the cluster contains only one element of the group then add that group into $Tree_{SuperConcpet_i,\_,Groups_k}$. Otherwise, add the prefix to $Tree$. If the prefix already exists in the tree, add this prefix with a different namespace to avoid namespace collisions according to definition 4.5. Then, recursively call algorithm 4 with $Tree_{SuperConcpet_i,Prefix_i,Groups}$, $Groups_k$ and $(depth + 1)$. Finally, the tree $Tree_{SuperConcept,InducedConcepts,Groups}$ represents the probabilistic taxonomy of the corpus.

The next step of the conceptualization is to generate relationships according to definitions 4.7 , 4.8 and 4.9.

Algorithm 5 provides the logic for relations creation. The concept tree $Tree$, verb set V, frequencies of lexicon L_F, frequencies of verbs V_F, frequencies of groups given verbs $\{G|V\}$, part-of-speech POS and relations factor $RF = (0, 1]$ are given as the input. Prior probabilities of $v_i \in V$ are calculated. Then, it is sorted and the top $RF \times |V_{Related}|$ are extracted to set V_{Final}. Then all combinations of groups related to a verb $v_i \in V_{Final}$ are created. If combinations of groups related to a verb $v_i \bowtie v_j \in V_{Final}$ exist, according to definition 4.9, the relation that has the highest Bayesian probability is kept in the tree.

Algorithm 5 $RELATIONS(Tree, V, L_F, V_F, \{G|V\}, POS, RF)$

```
Require: Concept tree $Tree$ from algorithm (4), verbs $V$, frequencies of verbs $V_F$
    and lexicon $L_F$ and part-of-speech $POS$
Ensure: Relations attached to tree $Tree$
 1: $V_{Related}$
 2: for $\forall v_i \in V$ do
 3:     Calculate prior probability $\beta_i$ for each $v_i$ using $L_F$ and $V_F$ and to set $V_{Related}$
 4: end for
 5: Sort $V_{Related}$ and extract top $RF \times |V_{Related}|$ elements to set $V_{Final}$
 6: for $\forall v_i \in V_{Final}$ do
 7:     Find $(g_n)(v_i)(g_m)$ relations from $POS$
 8:     Generate all combinations according to definition (4.9)
 9:     Calculate probability $p(g_k, g_l|v_i)$ for all combinations
10:     Find $g_k$ and $g_l$ concepts from $Tree$ and create the relation $v_i(g_k, g_l)$
11:     if $v_i \bowtie v_j \in V_{Final}$ then
12:         if $g_k$ && $g_l$ related with $v_i$ and $v_j$ then
13:             Keep only the relation associated with $v_i$ or $v_j$ that has the highest prob-
                ability $p(g_k, g_l|v_r)$ $(r = \{i, j\})$
14:         end if
15:     end if
16: end for
```

4.1.4 Representation

The learned $Tree$ from the prior section is serialized as an OWL 1 [6] ontology. $Tree$ is the conceptualization of the system and it is serialized as the $T - Box$. The probabilites associated with the conceptualization are serialized as the $A - Box$. The internal concept representation uses "|" character as the word separator. This is transformed into "_" character when serialized into concepts. We have used this convention because the default regular expression accepts alphabet of the form that has zero or more "_" characters. The simple transformation rule can be defined as $replace('|', '\_')$. As an example, the concept name $a|b|c\_d$ is transformed into $a\_b\_c\_d$.

Chapter 5

Implementation

The implementation of our work use several open source projects to populate the required contexts at different phases that we introduced in chapter 4. The bootstrapping process of our project requires tokenizing sentences and stemming/ lemmatizing of tokens to produced the lexicon of the corpus. In addition to this, we also require to know the type of a word. We used the Stanford log-linear part-of-speech tagger [56, 57, 58] to parse the syntactic structure of a sentence and also we use the OpenNLP [59] project to produce sentences and tokens and the WordNet [5] project to lookup for the type, stem and lemma of a word. In order to access the WordNet electronic library, we use the JWI [60] project.

The BioAssayOntology [4, 61] corpus contains xHtml-documents. We use the html parser [62] library to extract text from these documents. One of our other corpora contains pdf-documents. We use the Apache PDFBox [63] library to extract the contents from the pdf-documents. According to defintion 4.1, the lexicon is defined based on the Penn Treebank English POS tag set. We use the Stanford log-linear

POS tagger [57, 58], which uses the standard Penn Treebank tag set. Finally, we used Jena API [64] to serialize our probabilistic model into OWL DL [6].

We use several important third party open source projects in our implementation. Each third party project primarily specializes in different discipline of computer science. We use the Stanford log-linear POS tagger to type tag words in a sentences. This is a very efficient statistical parser develop and maintain for the last 8 years. The accuracy of the syntactic phase primarily depends on the efficiency of the part-of-speech tagger use. Since, we need higher accuracy, we believe this is a good research decision. WordNet is lexical database. We use the base form of an inflected word in the process. An efficient access to an off-line lexical database is paramount in order to speed up the ontology construction process. One of the important problems in natural language processing and machine learning is to detect the sentence boundary. This is not a easy task with a collection of documents. We need approximate sentence detection in our work to generate sentences. The OpenNLP projects hosts several efficient sentence detectors and we use it in our work to achieve maximum efficiency. Finally, the Jena API is a Java based OWL object model. Once the conceptualization is learned, it is needed to serialize to a file. We could have written our own OWL object model, but this is not our purpose in this research. It is important that the implementation to be efficient, less error prone and easily extendible and this is the reason we have use these open source projects in our implementation.

The first section provides a brief description of the third party projects. The second section provides the implementation details of our work named as "*PrOntoLearn*" (Probabilistic Ontology Learning) and the associations with section 5.1.

5.1 Third party projects

Our implementation is based on Java 1.6 specification [65]. Due to our implementation decision, we have the opportunity of using number of open source projects implemented in Java in different phases of the project. In the pre-processing phase 4.1.1, our system is required to read the sentences from the corpus, tokenize the sentences to words and obtain the normalized form of the word such as stem or lemma of the word to produce the lexicon of the corpus.

First, the sentences of the corpus are read using OpenNLP/Maxent project [59]. Then, the sentence is tagged with its part-of-speech tags using Stanford log-linear part-of-speech tagger [56, 57, 58]. This tagger uses the Penn Treebank project tagset [55]. From these tags, we have extracted all words related to nouns (NN, NNP & NNS), adjectives (JJ, JJR & JJS) and verbs (VB, VBD, VBG, VBN, VBP & VBZ) preserving the word sequence. These words are the candidate lexicons of the corpus. These candidate (potential) words are subjected to filtering as discussed in the section 4.1.1. Default filtering rules include that any word of the candidate lexicon set should have a word length > 2 and the word should be an element of the alphabet of the regular expression [a-zA-Z]+[-_]?\w*. Then, use the WordNet [5] project to normalize the candidate lexicon to their stem or lemma using WordNet stemming and lemmatizing algorithms.

Second, the learned probabilistic conceptualization model is serialzed into OWL DL sub-language using Jena API [64] in the representation phase of the section 4.1.4. We provided a detail description in chapter 2 why OWL DL sub-language is considered in our work. We also provide a debug representation of the conceptualization using Java Swing API [65]. We have only provided the conceptualization observation facility in the debug tool implemented in Java Swing API. In order to construct complex tasks on the serialized ontology, we recommend using Protégé ontology editor and knowledge-base framework [66].

5.1.1 WordNet

The WordNet [5] is a large lexical database for the English language developed and maintained at the Cognitive Science Laboratory of Princeton University. In addition to being a standard dictionary and thesaurus, the WordNet distinguishes between lexical and semantic relations of a word. The lexical relations hold between semantically related word forms, and semantic relations hold between word meanings. In order to represent these concepts, WordNet uses a logical grouping called synset, which is a list of synonymous words or collections that share a common meaning in some context. A word or a collection of words may appear in multiple synsets based on the meaning they hold considering the global context. WordNet has taken a great depth in describing relations among synsets based on the different gramatical rules. WordNet provides information for nouns, verbs, adjectives and adverbs. This is also the part-of-speech that WordNet recognizes. There are lexical and semantic relations among words and synsets and it is based on the type of the word in question.

1. Nouns are organized into hierarchies based on hypernymy/hyponymy, holonomy, meronym, antonymy and coordinated terms relations among synsets.

- Hypernymy : Synset S_1 is a hypernymy of synset S_2, if the meaning of S_1 subsumes the meaning of S_2.
- Hyponymy : The inverse relation of hypernymy.
- Meronymy : Synset S_1 is a meronymy of of synset S_2, if S_1 denotes a part of or member of S_2.
- Holonymy : The inverse relation of meronymy.
- Antonymy : When the relations between synsets hold the opposite meaning.
- Coordinated terms : Synset S_1 is a coordinated term of synset S_2, if S_1 and S_2 share a common hypernymy.

2. Verbs are organized into hierarchies based on hypernymy/hyponymy, troponym, entailment and coordinate terms relations among synsets.

   - Troponym : Synset S_1 is a troponym of synset S_2, if the activity S_2 is doing is same as S_1.

3. Adjectives are arranged in clusters containing head synsets and satellite synsets. Each cluster is organized around antonymous pairs.

4. Adverbs are often derived from adjectives, and occasionally have antonyms. Hence, from each adverb, there exist an lexical pointer to the adjective which it is derived from.

According to chapter 4, we forcus on nonus, adjectives and verbs to learn the underlying conceptualization, where collections of nouns provide a concept and verbs provide the relations among concepts. Table 5.1 shows some examples to understand the prior mentioned relations in WordNet.

WordNet is organized around a logical grouping called synset. Each synset has a meaning and this meaning is given by a gloss. Though, these glosses can be used as examples for a machine learning algorithm to populate or extract a useful structure, it is beyond the scope of our research. For our research we use WordNet 3.0 distribution. It has 117,798 nonus, 11,529 verbs and 21,479 adjectives. The average polysemy for nouns is 1.24, i.e., a noun has 1.24 different meanings in average. This is one of the main factors that we used the first sense of the word when classifying whether the word is a noun, a verb or an adjective. Our dataset contains word that are not recognized by WordNet. This is mainly true for nouns and proper nouns. The BioAssay ontology dataset is a specialized molecule biological screening documents. These documents use a domain specific jargon, which may not be available in WordNet. We experimented with Porter stemming algorithm [67] to stem the inflected word forms, which in result a ad-hoc outputs. Therefore, we use the inflected word form itself as a potential candidate of the lexicon and this word form uses in generating the N-gram groups, in the case WordNet does not recognize the word. These words are treated as nouns and conducted our experiements.

5.1.2 MIT Java WordNet Interface

WordNet provides different interfaces that are build in many languages to interact. Since our implementation is based on Java, out of several Java interfacing projects listed in WordNet site, we choose MIT Java Interface to Wordnet (JWI), developed at MIT [60]. JWI is written for Java version 1.5 and uses Java NIO packages for file access, with significant increase in speed. We have used JWI version 2.1.5 in our implementation. In order to complete this section, we presente a sample Java code 5.1 to connect to WordNet database using JWI. As lines (3 - 4) creates the URL where the WordNet database is available. Lines (6 - 7) create and open the connection to the database. Lines (9 - 11), look up for the first sense of the noun "man". As shown in the output, each word belongs to a synset and each synset has an id. Each word has an id. Each word comes with a gloss, which can be used in other purposes.

5.1.3 OpenNLP

OpenNLP [59] is an umbrella project that hosts open source projects related to natural language processing (NLP). It facilitates and encourages researches to develop such projects. Out of the listed projects, we use Maxent and "OpenNLP tools" Java package to generate sentences. We use Maxent version 2.4 and OpenNLP tools 1.3.0 in our implementation.

5.1.4 Stanford Log-linear Part-Of-Speech Tagger

The Stanford log-linear part-of-speech tagger reads text in the english language and assigns part-of-speech to each word such as nouns, verbs, adjectives etc. [56]. This tagger uses the Penn Treebank tag set to tag English words. We use Java 1.5+ implementation of this tagger in our implementation. This tagger has the current

Table 5.1: WordNet: lexical relations example

Class	**Meaning**	**Example**
Synonym	__is_the_same_as__	$synonym(theme, topic)$
Hypernym	__is_the_general_form_for__	$hypernym(furniture, chair)$
Hyponym	__is_kind_of__	$hyponym(chair, furniture)$
Meronym	__is_part__ __is_substance_of__ __is_member_of__	$meronym(branch, tree)$ $meronym(wood, tree)$ $meronym(person, group)$
Holonym	__has_part__ __as_substance__ __as_member__	$holonym(car, tire)$ $holonym(tree, wood)$ $holonym(group, person)$
Anonym	__is_the_opposite_of__	$antonym(up, down)$

best publish performance among taggers. In-depth details are available in [57, 58].

5.1.5 Jena API

One goal of our work is to create the OWL [6] environment of the learned ontology. Jena [64] is a Java framework for building Semantic Web applications, where it provides an environment for RDF, RDFS, OWL and SPARQL [68] models. In addition to this, it provides a very powerful rule-based inference engine.

Listing 5.1: JWI sample code & output

```
public void sampleCode() throws IOException {
  //construct the URL to the Wordnet dictionary directory
  String path = "...";
  URL url = new URL(path);
  //construct the dictionary object and open it
  IDictionary dict = new Dictionary(url);
  dict.open();
  //look up first sense of the word "man"
  IIndexWord idxWord = dict.getIndexWord("man", POS.NOUN);
  IWordID wordID = idxWord.getWordIDs().get(0);
  IWord word = dict.getWord(wordID);
  System.out.println("Id=" + wordID);
  System.out.println("Lemma=" + word.getLemma());
  System.out.println("Gloss=" + word.getSynset().getGloss());
}
Synset = SYNSET{SID-10287213-N : Words[W-10287213-N-1-man,
                    W-10287213-N-2-adult_male]}
Id = WID-10287213-N-??-man
Lemma = man
Gloss = an adult person who is male (as opposed to a woman);
          "there␣were␣two␣women␣and␣six␣men␣on␣the␣bus"
```

5.1.6 HTML parser & PDFBox

HTML parser [62] is an efficient open source library hosted in Sourceforge to extract text from Html documents. We use this library to extract all text fragments from the available tags. PDFBox [63] is an open source library hosted in the Apache Software Foundation to extract text from pdf documents. We use version 1.0 of this library, which is the latest release. It has been observed that the extracted text has a high level of noise associated with it. Even though we use normalizing algorithms from Information Retrival in the pre-processing phase, we are not able to eleminate all of the noise that come with the extraction process. As a future feature, we would like to use Google SOAP API estimating the confidence level of a word given its context and words inside the Markov blanket [11].

5.2 PrOntoLearn Implementation

In this section we discuss about the implementation details of our work. Figure 5.1 shows the high level class diagram of our implemenation created by the IntelliJ IDEA 9.0 UML plugin [69].

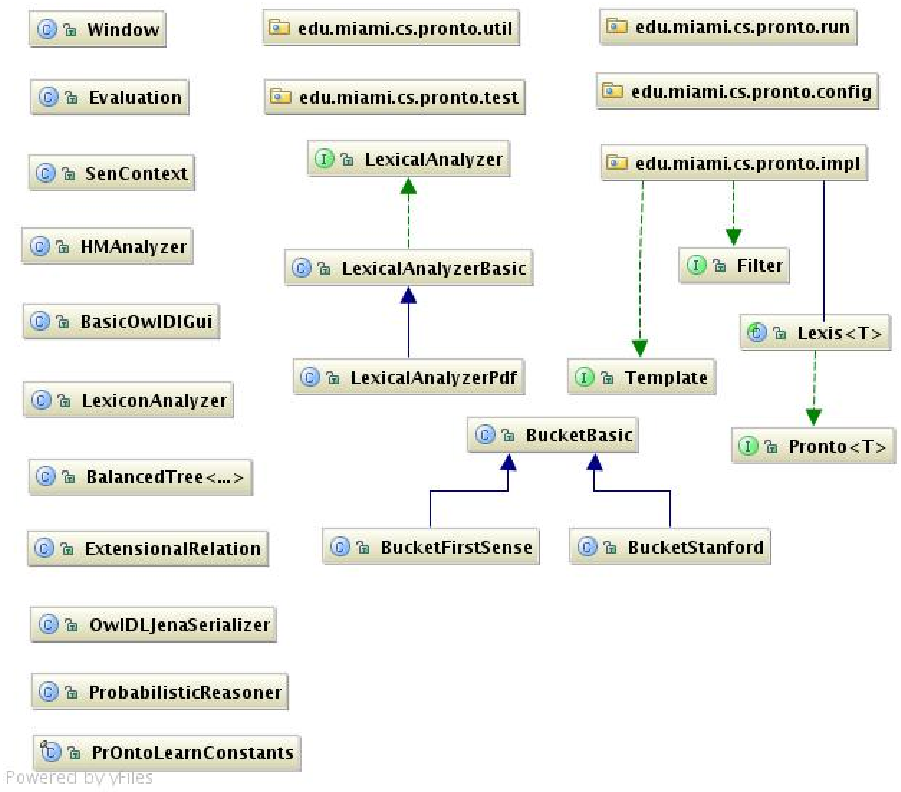


Figure 5.1: PrOntoLearn high level class diagram of package *edu.miami.cs.pronto*

We use the code name **PrOntoLearn** for our project. We use Java 1.6 version in our implementation. Main ProOntoLearn API contains classes belong to the package *edu.miami.cs.pronto.* The main interface of our implementation is $edu.miami.cs.pronto.Pronto < T >$. Implementations of this interface are available in the package *edu.miami.cs.pronto.impl* as shown in figure 5.2. along the line of pre-processing and syntactic phases, the lexion of the corpus is learned and they are typed-tag to nouns, verbs, adverbs and adjectives. Representations of each of lexicon is mapped to an instance of the type $edu.miami.cs.pronto.Pronto < T >$. The main-sub types of this interface are $edu.miami.cs.pronto.impl.Noun$, $edu.miami.cs.pronto.impl.Verb$, $edu.miami.cs.pronto.impl.Adverb$ and $edu.miami.cs.pronto.impl.Adjective$ repectively. Candidate groups learned from N-gram model is mapped to objects of the instance type *edu.miami.cs.pronto.impl.Group.* Part-of-speech with respect to a verb is mapped to instances of the type *edu.miami.cs.pronto.impl.PosGroup.* All the instances from prior mentioned classes are created in the pre-processing and syntactic analysis phases.

The most important implementation of our work is available in the classes of $edu.miami.cs.pronto.ProbabilisticReasoner$ and $edu.miami.cs.pronto.HMAnalyzer$. Combination of the algorithms in these classes will create the probabilistic conceptualization and the object model of PrOntoLearn. Once the object model is created, it is serialized as an OWL DL document using $edu.miami.cs.pronto.OwlDLJenaSerializer$ class. We also provide a simple yet powerful Java Swing GUI class $edu.miami.cs.pronto.BasicOwlDlGui$ as a debug tool and quick visualization of the ontology. Once the OWL DL file is created, it is recommended that an open source application such as Protégé 4.X is used for visualization, reasoning, SPARQL querrying etc.

Configuration parameters of the system are populated to an instance of the class *edu.miami.cs.pronto.config.ProntoConfig*. The configuration parameters include the location of WordNet database, OpenNLP training data, etc. Listing 5.2 shows how one can use PrOntoLearn API to create an ontology from a corpus, visualize it and searialize it as an OWL DL document. This example assumes that the documents of the corpus consists of Html files. It should be noted that the training files for different open source projects that are used in our work needs to be downloaded separately from the respective web sites. Throughout our implementation we use open source libraries licenced under GPL, LGPL and ASF 2.0. Therefore, we have included them in our final distribution.

Listing 5.2: Example of PrOntoLearn API

```java
public void run() throws IOException {
  // Location to corpus
  File base = new File("<location-to-corpus>");
  // Wordnet base
  File wordnetBase = new File("<location-to-WordNet-3.0>");
  // OpenNLP training data
```

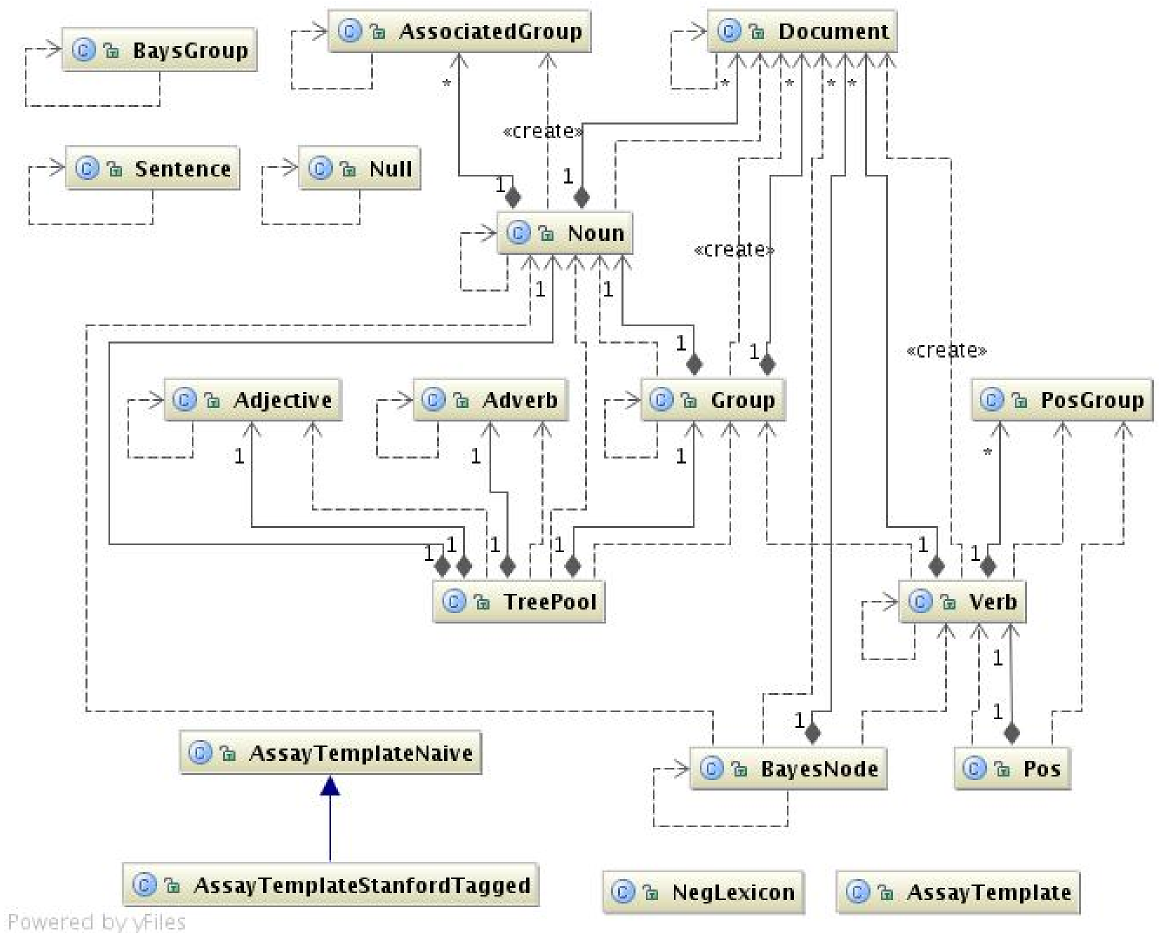


Figure 5.2: PrOntoLearn *edu.miami.cs.pronto.impl* class diagram

```
File openNlpTrainingDataBase =
      new File("<location-to-OpenNLP-training-data>");
// Stanford tagger model
File taggerModel = new File("<location-to-stanford-tagger>");
// Create PrOntoLearn configuration
ProntoConfig config = ProntoConfig.createProntoConfig(base,
                                wordnetBase,
                                openNlpTrainingDataBase,
                                taggerModel);
// N-gram model needs to be used
int nGram = 3;
Template assayTemplate =
      new AssayTemplateStanfordTagged(config, null, nGram);
// Create the lexical analyzer
LexicalAnalyzer2 la2 =
      new LexicalAnalyzerBasic(config, assayTemplate);
la2.analyze();
// Knowledge factor of the system
double kf = 0.5;
// Relations factor of the system
double rf = 0.9;
// Create the probabilistic reasoner
final ProbabilisticReasoner pr =
      new ProbabilisticReasoner(assayTemplate, kf, rf);
```

```
    pr.calculate(null);
    // Create the hypernomy and meronomy analyzer
    final HMAnalyzer hmAnalyzer = new HMAnalyzer(pr, nGram);
    hmAnalyzer.hm(null);
    // OWL DL output location
    File opf = new File("<location-to-owl-dl-output");
    OutputStream ops = new FileOutputStream(opf);
    // Write the OWL DL
    OwlDLJenaSerializer owl =
            new OwlDLJenaSerializer(hmAnalyzer, pr);
    owl.tBox(ops);
    // Java Swing GUI
    new BasicOwlDlGui(hmAnalyzer, pr).show();
}
```

Since our approach is unsupervised, the user has to provide configuration parameters, such as knowledge factor (KF), relations factor (RF) and lexicon filtering pattern. These parameters are analogous to clustering parameters K of the K-means algorithm [11].

Chapter 6

Experiments & Results

We have conducted our experiments on three main data corpora,

1. The PCAssay, of the BioAssay Ontology (BAO) project, Department of Molecular and Cellular Pharmacology University of Miami, School of Medicine [61, 4].

2. A sample collection of 38 pdf files from ISWC 2009 proceedings.

3. A substantial portion of the web pages extracted from the University of Miami, Department of Computer Science (*www.cs.miami.edu*) domain.

We have constructed ontologies for all three corpora with different parameter settings. One of the key problems we have is ontology evaluation. The BioAssay ontology dataset and the pdf dataset was impossible to evaluate as there are no existing reference ontologies or no ground truth that we could find. Therefore, we use the third dataset from the University of Miami, Department of Computer Science domain and conducted recall and precision on a reference ontology. The following section provides experimental results and we discuss the significance of these results.

6.1 Experiments

The first corpus, which is the primary data corpus of our experiment, contains molecule biological assays performed on various screening centres. The PCAssay dataset has an exponential growth rate. We specifically limited our dataset to assays available on the 1^{st} of January 2010. Table 6.2 provides the statistics of the corpus. We extracted the vocabulary generated from [a-zA-Z]+[-_]?\w* regular expression, and normalized them to create the lexicon of the corpus. One of the important aspects that shows in the corpus is that we will get exactly the same number of groups for every N-gram model. We explain this phenomenon with an example sentence "University of Miami, Department of Computer Science". The filtered sentence of this example is "University Miami Department Computer Science". Table 6.1 shows the outcome of 2-gram and 3-gram generator. According to table 6.1 both model generate 6 groups each.

Table 6.1: 2-gram & 3-gram generator results of a filtered sentence "University Miami Department Computer Science"

2-gram	**3-gram**
$< s >$ University	$< s_1 > < s_2 >$ University
University Miami	$< s_2 >$ University Miami
Miami Department	Miami Department Computer
Department Computer	Department Computer Science
Computer Science	Computer Science $< /s_2 >$
Science $< /s >$	Science $< /s_2 > < s_1 >$

The elements of the set $\{< s >, < s_1 > < s_2 >\}$ are empty starting elements and the elements of the set $\{< /s >,< /s_2 > < /s_1 >\}$ are empty ending elements. We need these extra elements because we calculate the probabilities such as

$p(University| < s >)$ (The n^{th} word given the previous $(n-1)$ words in the Markov blanket). This is true for all values of N-gram.

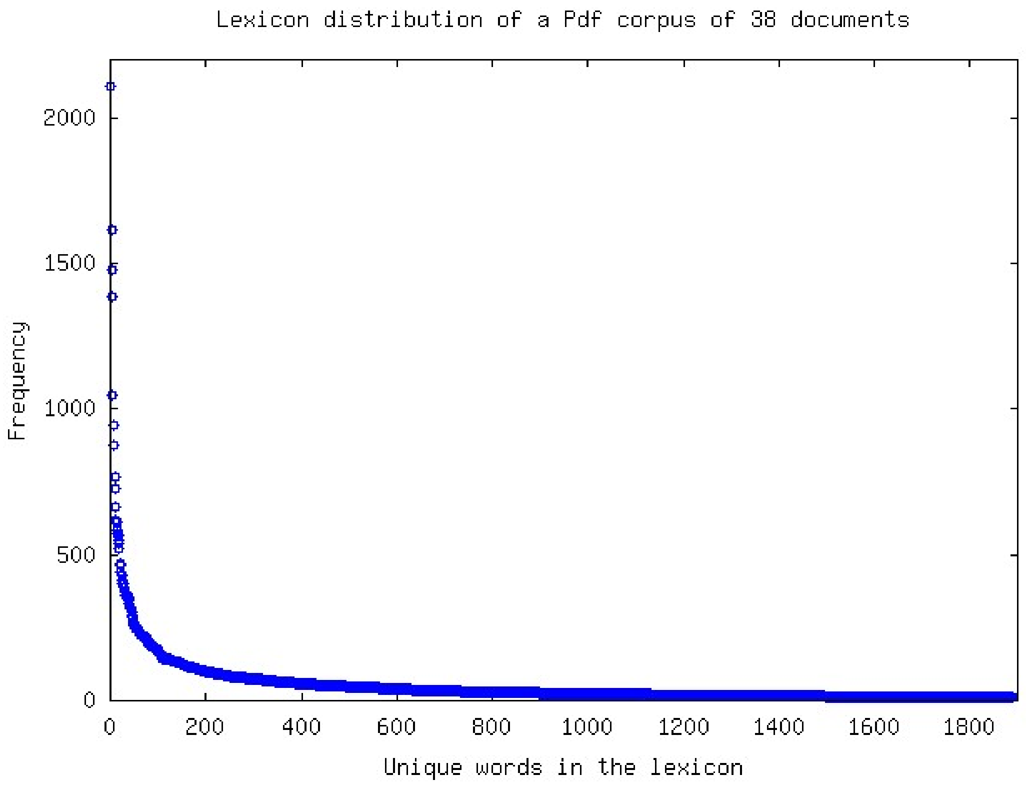
Table 6.2: The PCAssay (the BioAssay Ontology project) corpus statistics

Title	**Statistics**	**Description**
Documents	1,759	All documents are xhtml formated with a given template
Unique *ConceptWords*	13,017	Normalized candidate concept words from NN, NNP, NNS, JJ, JJR & JJS using [a-zA-Z]+[-_]?\w*
Unique *Verbs*	1,337	Normalized verbs from VB, VBD, VBG, VBN, VBP & VBZ using [a-zA-Z]+[-_]?\w*
Total *ConceptWords*	631,623	
Total *Verbs*	109,421	
Total Lexicon	741,044	$Lexicon = ConceptWords \bigcap Verbs$
Total *Groups*	631,623	

Figure 6.1 shows the frequency distribution of the lexicon of the pdf corpus. We also observe the similar distributions in other two corpora. According to Figure 6.1, it shows that the distributions of the three corpora can be approximated to an exponential family of graph $\lambda_1 e^{-\lambda_2 x}$. Thus, the probability distribution of this distribution is proportional to exponential distribution. The knowledge factor limits the number of words that we extract as super-concepts. A lot of words in the lexicon occur only once. Using KF, we are pruning the number of super-concepts needed in the representation.

Figure 6.2 shows the time in hours that it took our implementation to build the probabilistic conceptualization model. It is found from the experiments that the part-of-speech tagger requires approximately 2,600 ms to train itself. The average file size

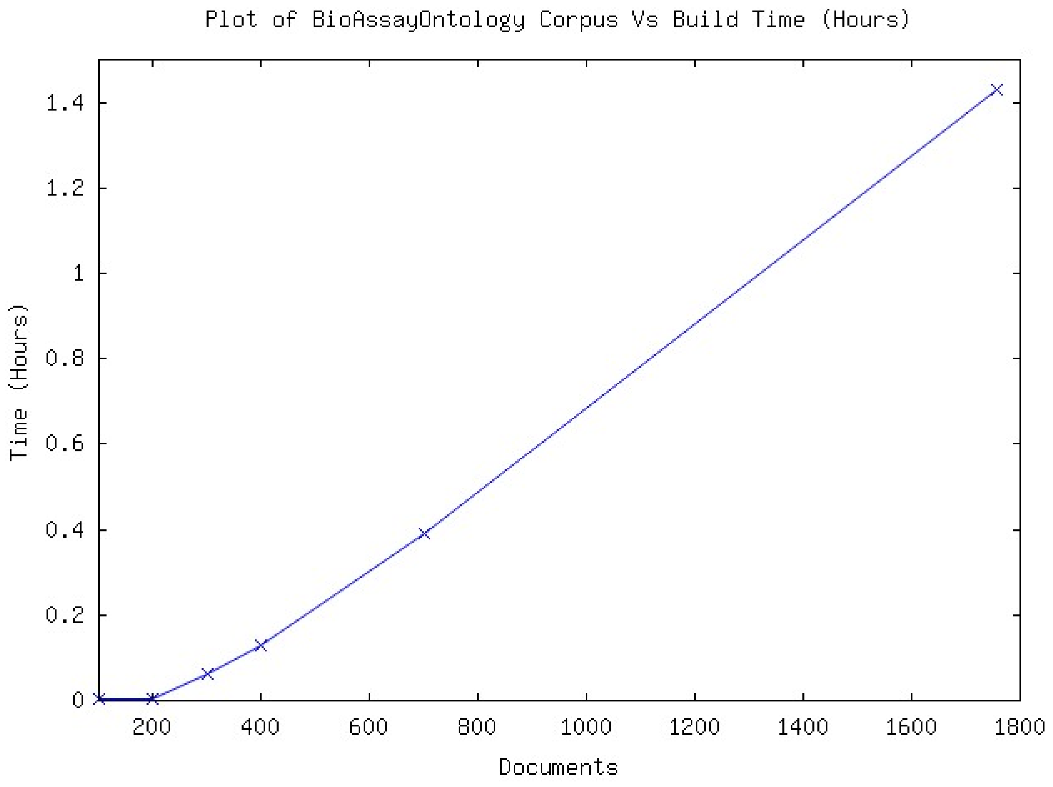
Figure 6.1: Lexicon distribution of Pdf corpus of 38 documents



of the corpus is approximately 6 Kb. We conducted these experiments in a Genuine Intel(R) CPU 585 @ 2.16GHz, 32 bits, 2 Gb Toshiba laptop. As Figure 6.2 shows, the time required to build the conceptualization grows linearly. It is evident from our implementation that the algorithms presented in different phases can be parallelized or use a MapReduce [70] framework to build the prior and posterior probabilities and conserve time. We have left this work future enhancements.

One of the other obstacles we have faced in terms of time complexity is in the representation layer. We have used Jena API [64] to serialize the probabilistic conceptualization into OWL DL [6]. When the system produced more than 1,000 concepts and relations, it is found that the Jena API takes a considerable amount of time to serialize the model. We have used different architectural schemes to improve its

Figure 6.2: BioAssayOntology corpus documents vs. build time



performance. With all optimization tasks we have put forward, still the presentation layer requires approximately 3.2 hours to serialize the model for the BioAssayOntology data corpus for full data set of 1,758 documents with capacity of 11.5 Mb. In order to provide a fast visualization of the conceptualization, we have written a simple yet flexible Java swing graphical user interface (GUI). This GUI has provided us visualizing and debugging the code as smoothly as possible. One of the other advantages of using a GUI is that it also provides the probabilities of the joint probability distribution $P(X, G)$, which is the representation of our probabilistic conceptualization.

The idea of our work is to generate an ontology without the supervision of a domain expert (unsupervised) for any given corpus. The user has to set system parameters such as the knowledge factor, relations factor and regular expression of the lexicon. Since we used data corpora from the bio medical domain, a collection of research papers and set of documents collected from computer science web site, the evaluation of the created ontology using standard techniques such as precision and recall is not easy. We evaluate the generated ontologies with human domain experts. We obtained the comments and recommendations from the domain expert on the importance of the generated ontology. The ontology that is generated is too large to show in here. Instead, we provide a few distinct snapshots of the ontology with the help of Protégé OWLViz plugin. Here are a few snapshots of the ontology created from the BioAssay ontology dataset for $KnowledgeFactor = 0.5$, N-gram $= 3$, and $RelationsFactor = 0.9$:

Figures 6.3, 6.4 and 6.5 show three snapshots of the created ontology. Figure 6.6 shows a snapshot of the ontology with $\exists$ relations. Figure 6.7 shows the Java swing visualization of BioAssayOntology for randomly chosen 200 documents.

The Java swing GUI shows the prior and posterior probabilities of the concepts and relations in the conceptualization. Before we discuss the pdf corpus, let us look at the University of Miami, Department of Computer Science corpus (*www.cs.miami.edu*). Statistics of this corpus are shown in tabel 6.3.

Gold standard based approaches such as precision (P), recall (R) and F-measure (F_1) are used to evaluate ontologies [71]. If the computed ontology is O_C and the reference ontology is O_R, then,

$$P(O_C, O_R) = \frac{C_C \bigcap C_R}{C_C} \tag{6.1}$$

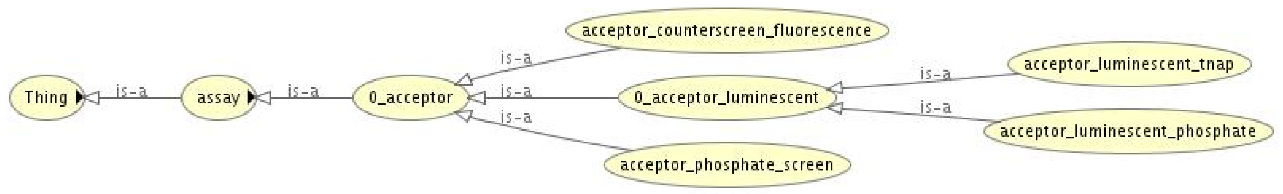

Thing
is-a
assay
is-a
0_acceptor
is-a
is-a
is-a
acceptor_counterscreen_fluorescence
0_acceptor_luminescent
acceptor_phosphate_screen
is-a
is-a
acceptor_luminescent_tnap
acceptor_luminescent_phosphate

Figure 6.3: An example snapshot of the BioAssayOntology corpus (No. 1)

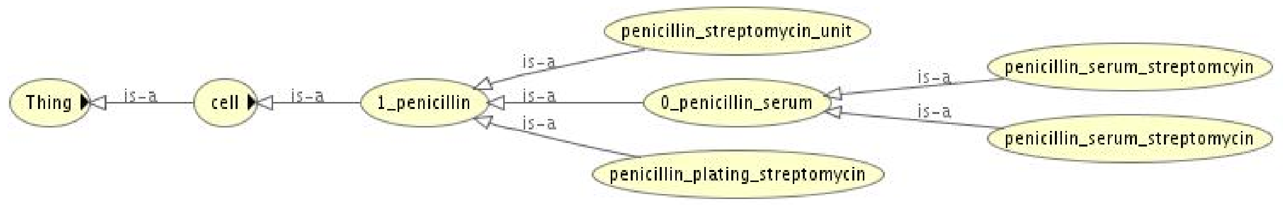

Thing
is-a
cell
is-a
1_penicillin
is-a
is-a
is-a
penicillin_streptomycin_unit
0_penicillin_serum
penicillin_plating_streptomycin
is-a
is-a
penicillin_serum_streptomcyin
penicillin_serum_streptomycin

Figure 6.4: An example snapshot of the BioAssayOntology corpus (No. 2)

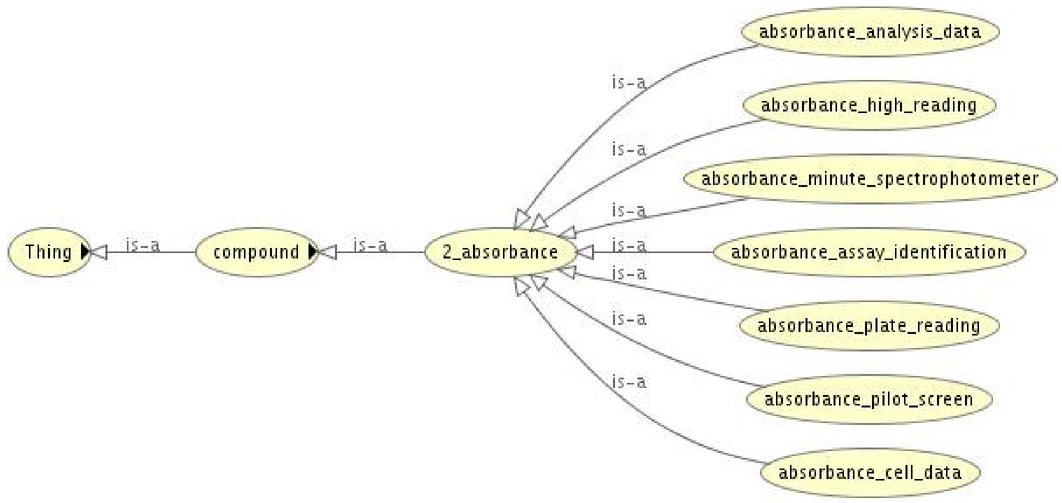


Figure 6.5: An example snapshot of the BioAssayOntology corpus (No. 3)

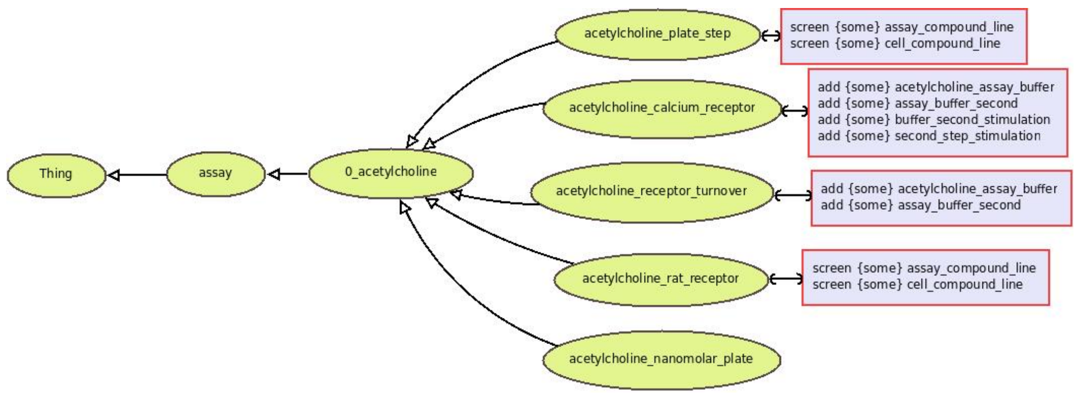
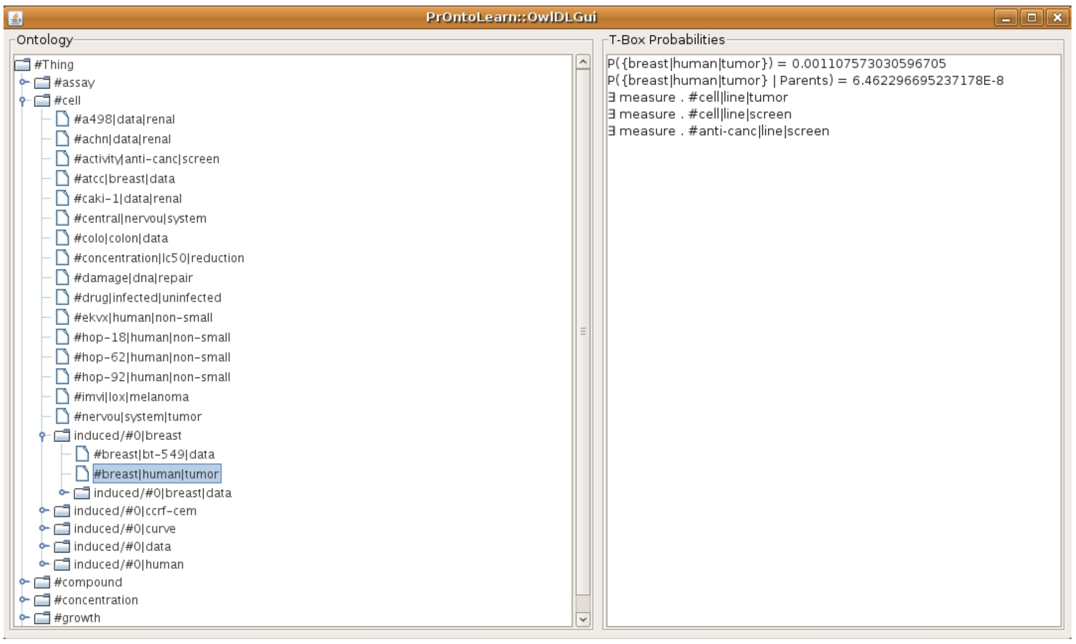

Thing
assay
0_acetylcholine
acetylcholine_plate_step
acetylcholine_calcium_receptor
acetylcholine_receptor_turnover
acetylcholine_rat_receptor
acetylcholine_nanomolar_plate
screen {some} assay_compound_line
screen {some} cell_compound_line
add {some} acetylcholine_assay_buffer
add {some} assay_buffer_second
add {some} buffer_second_stimulation
add {some} second_step_stimulation
add {some} acetylcholine_assay_buffer
add {some} assay_buffer_second
screen {some} assay_compound_line
screen {some} cell_compound_line

Figure 6.6: An example snapshot of the BioAssayOntology corpus with relations (No. 4)



```
#Thing
  #assay
  #cell
    #a498|data|renal
    #achn|data|renal
    #activity|anti-canc|screen
    #atcc|breast|data
    #caki-1|data|renal
    #central|nervou|system
    #colo|colon|data
    #concentration|lc50|reduction
    #damage|dna|repair
    #drug|infected|uninfected
    #ekvx|human|non-small
    #hop-18|human|non-small
    #hop-62|human|non-small
    #hop-92|human|non-small
    #imvi|lox|melanoma
    #nervou|system|tumor
    induced/#0|breast
      #breast|bt-549|data
      #breast|human|tumor
      induced/#0|breast|data
    induced/#0|ccrf-cem
    induced/#0|curve
    induced/#0|data
    induced/#0|human
  #compound
  #concentration
  #growth
```

```
P({breast|human|tumor}) = 0.001107573030596705
P({breast|human|tumor} | Parents) = 6.462296695237178E-8
∃ measure . #cell|line|tumor
∃ measure . #cell|line|screen
∃ measure . #anti-canc|line|screen
```

Figure 6.7: Java Swing GUI for BioAssayOntology dataset (for 200 documents).

Table 6.3: *www.cs.miami.edu* corpus statistics

Title	**Statistics**	**Description**
Documents	218	All documents are xhtml formated with a give template
Unique *ConceptWords*	5,384	Normalized candidate concept words from NN, NNP, NNS, JJ, JJR & JJS using [a-zA-Z]+[-_]?\w*
Unique *Verbs*	835	Normalized verbs from VB, VBD, VBG, VBN, VBP & VBZ using [a-zA-Z]+[-_]?\w*
Total *ConceptWords*	39,455	
Total *Verbs*	4,797	
Total Lexicon	44,252	$Lexicon = ConceptWords \bigcap Verbs$
Total *Groups*	39,455	

$$R(O_C, O_R) = \frac{C_C \bigcap C_R}{C_R} \tag{6.2}$$

$$F_1(O_C, O_R) = \frac{2 \times P(O_C, O_R) \times R(O_C, O_R)}{P(O_C, O_R) + R(O_C, O_R)} \tag{6.3}$$

Where $C_C \in O_C$ and $C_R \in O_R$. We have used [3] as our reference ontology and it is given in Appendix A. Since a concept of the computed ontology is created using the form of $a\_b\_c\ldots$, a substring of the concept that matches with reference concept is accepted as a valid mapping.

Table 6.4: Precision, recall and F1 measurement for $N - gram = 4$ and $RF = 1$

KF	Precision	Recall	F1
0.1	0.209	1	0.309
0.2	0.194	1	0.325
0.3	0.257	1	0.410
0.4	0.257	1	0.410
0.5	0.257	1	0.410
0.6	0.248	1	0.397
0.7	0.244	1	0.393
0.8	0.236	1	0.383
0.9	0.237	1	0.383
1.0	0.13	1	0.232

According to the table 6.4, the precision of the generated ontology is approximately 21%. Then we have used another reference ontology to calculate recall, precision and F_1. Table 6.5 shows the results. According to this result, the precision of the constructed ontology is approximately 42%. This concludes one important fact. There is no gold standard ground truth available here. When we use another reference ontology the precision increases. This does not mean that this is ground truth. A quantitative evaluation of an ontology is still a open problem. Using a human domain expert to evaluate the ontology is an alternative approach. This is a qualitative evaluation method. In order to get a good qualitative evaluation, the ontology should be inspected with at least 10-15 experts. It is practically difficult job to find domain experts for a corpus such as BioAssay ontology dataset, which has bio medical molecular screen constructs. We evaluated our ontology for the first corpus with a bio medicine molecular expert. According to his comments, the ontology contains rich set of vocabulary, which is very useful for top-down ontology construction. But he also mentioned that the tree does have a flat structure. We use 3-gram generator to create the ontology. Therefore, the maximum level this model will give is at most 3.

Table 6.5: Precision, recall and F1 measurement for $N-gram = 4$ and $RF = 1$ using extended reference ontology

KF	Precision	Recall	F1
0.1	0.424	1	0.596
0.2	0.388	1	0.559
0.3	0.445	1	0.616
0.4	0.438	1	0.609
0.5	0.438	1	0.609
0.6	0.424	1	0.595
0.7	0.415	1	0.587
0.8	0.412	1	0.583
0.9	0.405	1	0.576
1.0	0.309	1	0.472

Finally, we have generated an ontology for the pdf corpus. The noise of the lexicon that is generated from this corpus depends on the pdf parser we used. We found that the version 1.0.0 PdfBox parser has inherent parsing noise. One of our assumptions is that the corpus contains correct information.

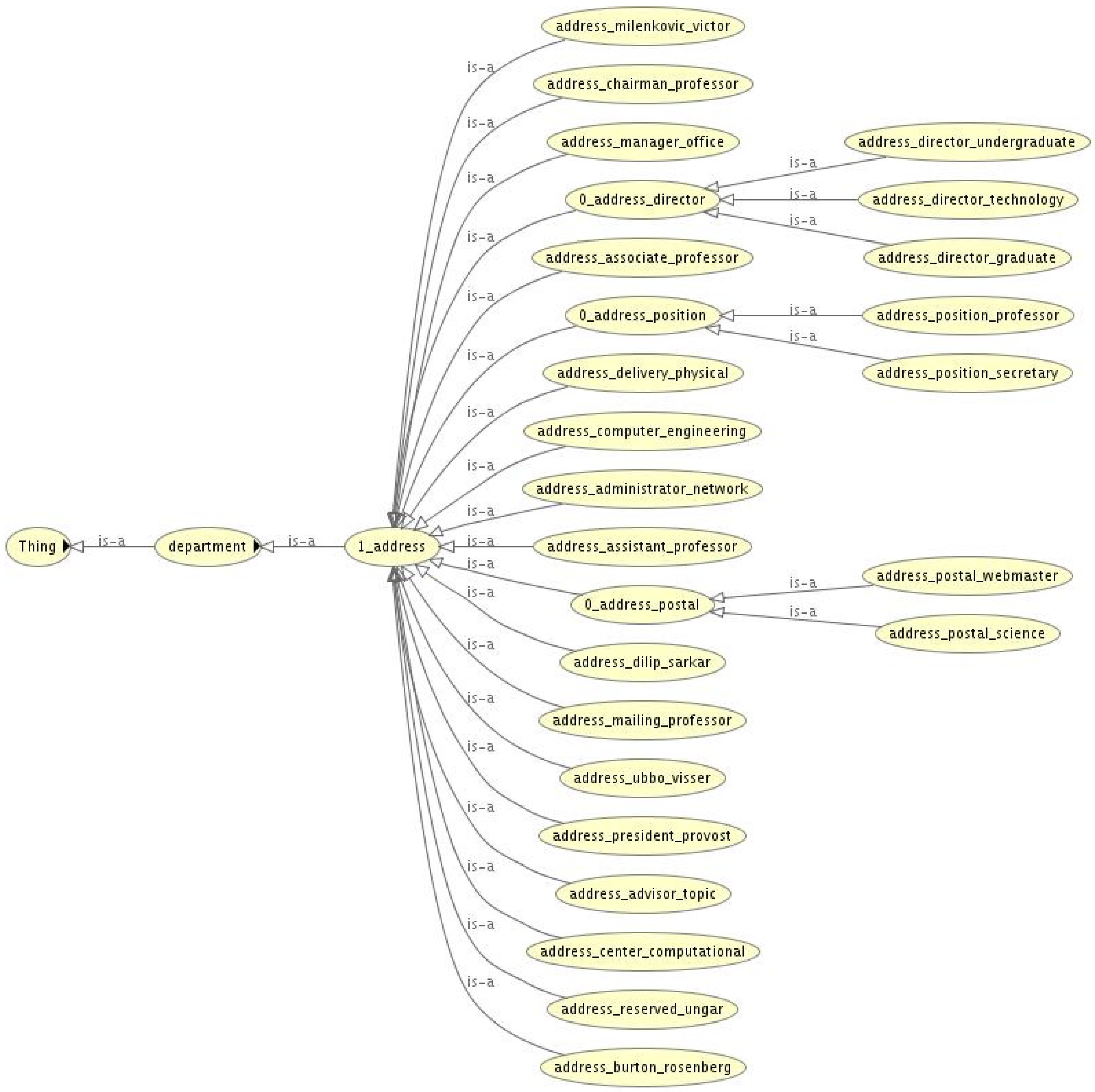

Thing
is-a
department
is-a
1_address
address_milenkovic_victor
address_chairman_professor
address_manager_office
0_address_director
address_director_undergraduate
address_director_technology
address_director_graduate
address_associate_professor
0_address_position
address_position_professor
address_position_secretary
address_delivery_physical
address_computer_engineering
address_administrator_network
address_assistant_professor
0_address_postal
address_postal_webmaster
address_postal_science
address_dilip_sarkar
address_mailing_professor
address_ubbo_visser
address_president_provost
address_advisor_topic
address_center_computational
address_reserved_ungar
address_burton_rosenberg

Figure 6.8: A shanpshot of *cs.miami.edu* ontology

This is a reasonable assumption because of the corpora we used was a technical or a publicly available in the web. If our method is used for a corpus containing a lot of noise, we need to provide another additional layer to filter out or smoothen the error. There are in many ways this could be achieved. The naive way is to use a standard dictionary to parse the words and use the first sense. The most common way is to send the word to a search engine and obtain the result and parse the result for the requested word. If the word appears more that a threshold $T > 0$, then that word can be considered a potential word for the lexion.

It is to be noted that the ontologies that are generated from the system are consistent with Pellet and Fact++ reasoners. We conducted reasoning experiments using Pellet and Fact++ plugins available in Protégé 4.X distribution.

6.2 Discussion

The results show that our method creates an ontology for any given domain with acceptable results. This is shown in the precision value, if the ground truth is available. On the other hand, if the domain does not have ground truth the results are subject to domain expert evaluation of the ontology. One of the potential problems we have seen in our approach is search space. Since our method is unsupervised, it tends to search the entire space for results, which is computationally costly. We thus need a better method to prune the search space so that out method provide better results. According to human domain experts, our method extracts good vocabulary but provides a flat structure. They have proposed a sort of a semi-supervised approach to correct this problem, by combining the knowledge from domain experts and results produced by our system. We left the detailed investigation for future work.

Since our method is based on Bayesian reasoning (which uses N-gram probabilities), it is paramount that the corpus contains enough evidence of the redundant information. This condition requires that the corpus is substantially large enough so that we can hypothesize that the corpus provides enough evidence to build the ontology. We did not conduct an experiment to show how much of evidence our system needs in order to create a considerable ontology.

We hypothesize that a sentence of the corpus would generally be subjected to the grammar rule given in Equation 2.14. This constituent is the main factor that uses to build the relationships among concepts. In NLP, there are many other finer grained grammar rules that specifically fit for given sentences. If these grammar rules are used, we believe we can build a better relationship model. We have left this for future work.

At the moment our system does not distinguish between concepts and the individuals of the concepts. The learned A-Box is primarily consist of the probabilities of each concepts. This is one area we are eager to work on. Using state-of-the art NLP techniques, we plan to fill this gap in a future work. Since our method has the potential to be used in any corpus, it could be seen that the lemmatizing and stemming algorithms that are available in WordNet would not recognize some of the words. Specially in the BioAssay dataset, we observed that some of the domain specific words are not recognized by WordNet. We used Porter stemming algorithm [67] to get the word form and it showed that this algorithm constructed peculiar word forms. Therefore, we deliberately removed this from the processing pipeline.

The complexity of our algorithms is as follows. The bootstrapping algorithm available in the syntactic layer has a worst case running time of $O(M \times max(sj) \times max(wk))$. The probabilistic reasoning algorithm has the worst case running time of $O(|\mathcal{L}| \times |SuperConcepts|)$. The generated has proven to be satisfiable under Pellet and Fact++ reasoners.

Finally, our method provides a process to create a lexico-semantic ontology for any domain. For our knowledge, this is a very first research on this line of work. So we continue our research along this line and to provide better results for future use.

Chapter 7

Summary & Future Work

We have introduced a novel process to generate an ontology for any random text corpus. We have shown that our process constructs a flexible seed ontology. It is also shown that in order to achieve high precision, it is paramount that the corpus should be large enough to extract important evidence. Our research has also shown that probabilistic reasoning on lexico-semantic structures is a powerful solution to overcome or at least mitigate the knowledge acquisition bottleneck. Our method also provides evidence to domain experts to build ontologies using a top-down approach.

Though we have introduced a powerful technique to construct ontologies, we believe that there is a lot of work that can be done to improve the performance of our system. One of the areas our method lacks is the separation between concepts and individuals. We would like to use the generated ontology as a seed ontology to generate instances for the concepts and extract the individuals already classified as concepts. We would use NLP technique to obtain this classification. In addition to this, our system can improve the quality of the relations if we introduce more specific grammar rules to sentences. We are looking at computational lexical semantics to prune the search space, so that the algorithms are efficient. Finally, we would like to increase the lexicon of the system with more tags available from the Penn treebank tagset. We believe that if we introduce more tags into the system, our system can be trained to construct human readable (friendly) concepts and relations names.

Appendix: A

Figure 1 shows one of the reference ontologies that we used to evaluate the 2nd dataset.

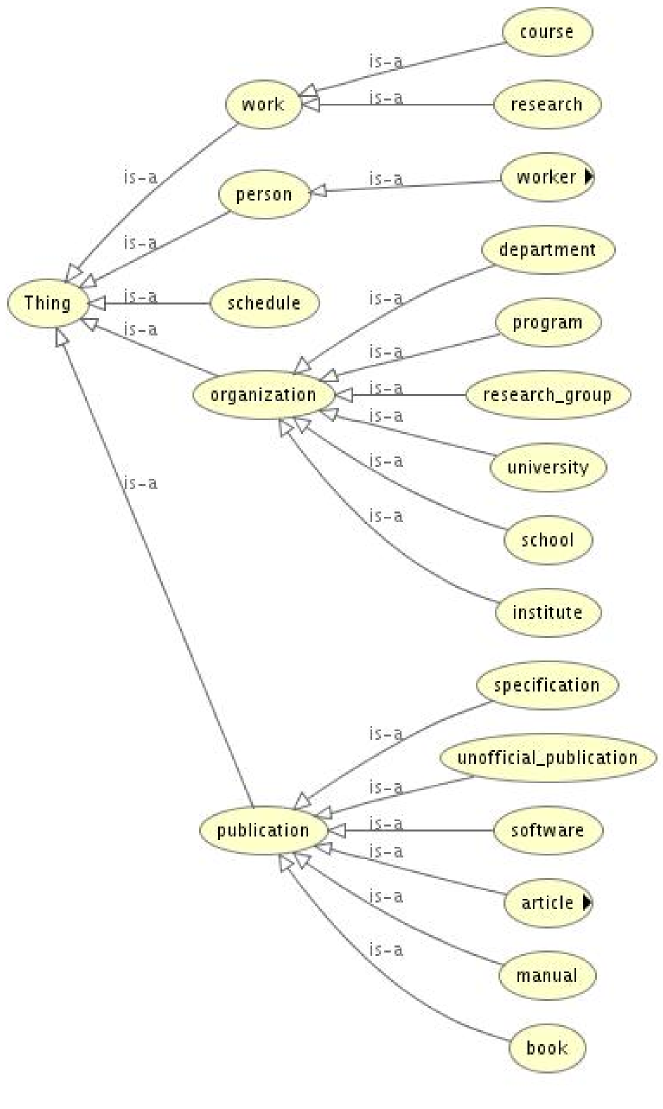


Figure 1: Reference Computer Science Department ontology [3]

References

[1] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, vol. 5, no. 2, pp. 199–220, 1993.

[2] R. Studer, V. R. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," *Data and Knowledge Engineering*, vol. 25, pp. 161–197, March 1998.

[3] SHOE, "Example computer science department ontology," *http://www.cs.umd.edu/projects/plus/SHOE/cs.html*, 2010.

[4] "Bioassay ontology," *http://bioassayontology.org/*, 2010.

[5] WordNet, "Wordnet 3.0," *http://wordnet.princeton.edu/*, 2010.

[6] D. L. McGuinness and F. van Harmelen, "Owl web ontology language overview," Tech. Rep. REC-owl-features-20040210, W3C, 2004.

[7] W. Borst, *Construction of Engineering Ontologies.* PhD thesis, Institute for Telematica and Information Technology, University of Twente, Enschede, The Netherlands, 1997.

[8] D. Guarino, N. Obrele and S. Staab, "What is an ontology ?," *International Handbooks on Information Systems*, pp. 1–17, 2009.

[9] I. Horrocks and P. Patel-Schneider, "Reducing owl entailment to description logic satisfiability," *Web Semant.*, vol. 1, no. 4, pp. 345–357, 2004.

[10] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach.* Prentice Hall, 3rd ed., 2009.

[11] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics).* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[12] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition.* Prentice Hall, Pearson Education International, 2. ed., [pearson international edition] ed., 2009.

[13] J. Völker, P. Haase, and P. Hitzler, "Learning expressive ontologies," in *Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, (Amsterdam, The Netherlands, The Netherlands), pp. 45–69, IOS Press, 2008.

[14] D. Koller, A. Levy, and A. Pfeffer, "P-classic: A tractable probabilistic description logic," in *In Proceedings of AAAI-97*, pp. 390–397, 1997.

[15] T. M. Mitchell, *Machine Learning.* New York: McGraw-Hill, 1997.

[16] J. Pearl, *Probabilistic reasoning in intelligent systems: networks of plausible inference.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

[17] Z. Ding and Y. Peng, "A probabilistic extension to ontology language owl," in *HICSS '04: Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 4*, (Washington, DC, USA), p. 40111.1, IEEE Computer Society, 2004.

[18] T. Lukasiewicz, "Probabilistic description logics for the semantic web," tech. rep., Nr. 1843-06-05, Institut fur Informationssysteme, Technische Universitat Wien, 2007.

[19] "Pellet pronto," *http://pellet.owldl.com/pronto/*, 2008.

[20] "Read the web project," *http://rtw.ml.cmu.edu /readtheweb.html*, 2010.

[21] A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka, Jr., and T. M. Mitchell, "Coupled semi-supervised learning for information extraction," in *WSDM '10: Proceedings of the third ACM international conference on Web search and data mining*, (New York, NY, USA), pp. 101–110, ACM, 2010.

[22] T. M. Mitchell, J. Betteridge, A. Carlson, E. Hruschka, and R. Wang, "Populating the semantic web by macro-reading internet text," in *ISWC '09: Proceedings of the 8th International Semantic Web Conference*, (Berlin, Heidelberg), pp. 998–1002, Springer-Verlag, 2009.

[23] A. Carlson, J. Betteridge, E. R. Hruschka, Jr., and T. M. Mitchell, "Coupling semi-supervised learning of categories and relations," in *SemiSupLearn '09: Proceedings of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*, (Morristown, NJ, USA), pp. 1–9, Association for Computational Linguistics, 2009.

[24] R. C. Wang and W. W. Cohen, "Language-independent set expansion of named entities using the web," in *ICDM '07: Proceedings of the 2007 Seventh IEEE International Conference on Data Mining*, (Washington, DC, USA), pp. 342–350, IEEE Computer Society, 2007.

[25] R. C. Wang and W. W. Cohen, “Iterative set expansion of named entities using the web,” in *ICDM ’08: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, (Washington, DC, USA), pp. 1091–1096, IEEE Computer Society, 2008.

[26] R. C. Wang and W. W. Cohen, “Automatic set instance extraction using the web,” in *ACL-IJCNLP ’09: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1*, (Morristown, NJ, USA), pp. 441–449, Association for Computational Linguistics, 2009.

[27] D. A. Waterman, *A guide to expert systems*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1985.

[28] P. Clark and P. Harrison, “Large-scale extraction and use of knowledge from text,” in *K-CAP ’09: Proceedings of the fifth international conference on Knowledge capture*, (New York, NY, USA), pp. 153–160, ACM, 2009.

[29] C. Havasi, “Conceptnet 3: a flexible, multilingual semantic network for common sense knowledge,” in *the 22nd Conference on Artificial Intelligence*, 2007.

[30] B. Van Durme and L. Schubert, “Open knowledge extraction through compositional language processing,” in *STEP ’08: Proceedings of the 2008 Conference on Semantics in Text Processing*, (Morristown, NJ, USA), pp. 239–254, Association for Computational Linguistics, 2008.

[31] D. Lin and P. Pantel, “Discovery of inference rules for question-answering,” *Nat. Lang. Eng.*, vol. 7, no. 4, pp. 343–360, 2001.

[32] J. Bos and K. Markert, “Recognising textual entailment with logical inference,” in *HLT ’05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, (Morristown, NJ, USA), pp. 628–635, Association for Computational Linguistics, 2005.

[33] L. Balby Marinho, K. Buza, and L. Schmidt-Thieme, “Folksonomy-based collabulary learning,” in *ISWC ’08: Proceedings of the 7th International Conference on The Semantic Web*, (Berlin, Heidelberg), pp. 261–276, Springer-Verlag, 2008.

[34] H. C. Cankaya and D. Moldovan, “Method for extracting commonsense knowledge,” in *K-CAP ’09: Proceedings of the fifth international conference on Knowledge capture*, (New York, NY, USA), pp. 57–64, ACM, 2009.

[35] W. Salloum, “A question answering system based on conceptual graph formalism,” in *KAM ’09: Proceedings of the 2009 Second International Symposium on Knowledge Acquisition and Modeling*, (Washington, DC, USA), pp. 383–386, IEEE Computer Society, 2009.

[36] D. S. Kim, K. Barker, and B. Porter, "Knowledge integration across multiple texts," in *K-CAP '09: Proceedings of the fifth international conference on Knowledge capture*, (New York, NY, USA), pp. 49–56, ACM, 2009.

[37] K. Tomanek and U. Hahn, "Reducing class imbalance during active learning for named entity annotation," in *K-CAP '09: Proceedings of the fifth international conference on Knowledge capture*, (New York, NY, USA), pp. 105–112, ACM, 2009.

[38] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *Proceedings of the 14th conference on Computational linguistics*, (Morristown, NJ, USA), pp. 539–545, Association for Computational Linguistics, 1992.

[39] A. Maedche, E. Maedche, and S. Staab, "The text-to-onto ontology learning environment," in *Software Demonstration at ICCS-2000 - Eight International Conference on Conceptual Structures*, 2000.

[40] E. Maedche and S. Staab, "Ontology learning for the semantic web," *IEEE Intelligent Systems*, vol. 16, pp. 72–79, 2001.

[41] A. Maedche, E. Maedche, and R. Volz, "The ontology extraction maintenance framework text-to-onto," in *In Proceedings of the ICDM01 Workshop on Integrating Data Mining and Knowledge Management*, 2001.

[42] P. Velardi, R. Navigli, A. Cucchiarelli, and F. Neri, "Evaluation of OntoLearn, a methodology for automatic population of domain ontologies," in *Ontology Learning from Text: Methods, Applications and Evaluation* (P. Buitelaar, P. Cimiano, and B. Magnini, eds.), IOS Press, 2006.

[43] P. Buitelaar, D. Olejnik, and M. Sintek, "A protege plug-in for ontology extraction from text based on linguistic analysis," in *In Proceedings of the 1st European Semantic Web Symposium (ESWS*, 2004.

[44] N. Guarino and C. A. Welty, "An overview of ontoclean," *Handbook on Ontologies*, pp. 201–220, 2009.

[45] P. Cimiano and J. Völker, "Text2onto - a framework for ontology learning and data-driven change discovery," 2005.

[46] P. Haase and J. Völker, "Ontology learning and reasoning - dealing with uncertainty and inconsistency," in *Proceedings of the Workshop on Uncertainty Reasoning for the Semantic Web (URSW*, pp. 45–55, 2005.

[47] P. Cimiano, A. Hotho, and S. Staab, "Learning concept hierarchies from text corpora using formal concept analysis," *Journal of Artificial Intelligence research*, vol. 24, pp. 305–339, 2005.

[48] P. Cimiano, *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications.* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[49] C. Chemudugunta, A. Holloway, P. Smyth, and M. Steyvers, "Modeling documents by combining semantic concepts with unsupervised statistical learning," in *ISWC '08: Proceedings of the 7th International Conference on The Semantic Web*, (Berlin, Heidelberg), pp. 229–244, Springer-Verlag, 2008.

[50] D. Maynard, Y. Li, and W. Peters, "Nlp techniques for term extraction and ontology population," in *Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, (Amsterdam, The Netherlands, The Netherlands), pp. 107–127, IOS Press, 2008.

[51] H. Tanev and B. Magnini, "Weakly supervised approaches for ontology population," in *Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, (Amsterdam, The Netherlands, The Netherlands), pp. 129–143, IOS Press, 2008.

[52] P. Pantel and M. Pennacchiotti, "Automatically harvesting and ontologizing semantic relations," in *Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, (Amsterdam, The Netherlands, The Netherlands), pp. 171–195, IOS Press, 2008.

[53] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval.* New York, NY, USA: Cambridge University Press, 2008.

[54] "Lexicon," *http://en.wikipedia.org/wiki/Lexicon/*, 2010.

[55] "The penn treebank project," *http://www.cis. upenn.edu/ treebank/*, 2010.

[56] "Stanford log-linear part-of-speech tagger," *http://nlp.stanford.edu/software/ tagger.shtml*, 2010.

[57] K. Toutanova and C. D. Manning, "Enriching the knowledge sources used in a maximum entropy part-of-speech tagger," in *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*, (Morristown, NJ, USA), pp. 63–70, Association for Computational Linguistics, 2000.

[58] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, (Morristown, NJ, USA), pp. 173–180, Association for Computational Linguistics, 2003.

[59] OpenNLP, "Open nlp," *http://opennlp.sourceforge.net/*, 2010.

[60] JWI, "Mit java wordnet interface series 2," *http://projects.csail.mit.edu/jwi/*, 2008.

[61] "Alfresco bioassayontology," *http:// share.ccs.miami.edu/share/page/site-index*, 2009.

[62] "Html parser: Sourceforge," *http://htmlparser.sourceforge.net/samples.html*, 2010.

[63] PDFBox, "Pdfbox: Apache software foundation," *http://pdfbox.apache.org/*, 2010.

[64] Jena, "Jena: A semantic web framework for java," *http://jena.sourceforge.net/*, 2010.

[65] "Java language 1.6," *http://www.java.com/en/*, 2010.

[66] "Protege ontology editor and knowledge-base framework," *http://prot ege.stanford.edu/*, 2010.

[67] M. F. Porter, "An algorithm for suffix stripping," pp. 313–316, 1997.

[68] E. Prud'hommeaux and A. Seaborne, "Sparql query language for rdf," tech. rep., W3C, http://www.w3.org/TR/rdf-sparql-query/, 2008.

[69] "Intellij idea 9: Jet brains," *http://www.jetbrains.com/idea/*, 2010.

[70] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[71] K. Dellschaft and S. Staab, "Strategies for the evaluation of ontology learning," in *Proceeding of the 2008 conference on Ontology Learning and Population: Bridging the Gap between Text and Knowledge*, (Amsterdam, The Netherlands, The Netherlands), pp. 253–272, IOS Press, 2008.