Chapter 5

Ontology Learning Using Word Net Lexical Expansion and Text Mining

Hiep Luong, Susan Gauch and Qiang Wang

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/51141

1. Introduction

In knowledge management systems, ontologies play an important role as a backbone for providing and accessing knowledge sources. They are largely used in the next generation of the Semantic Web that focuses on supporting a better cooperation between humans and machines [2]. Since manual ontology construction is costly, time-consuming, error-prone, and inflexible to change, it is hoped that an automated ontology learning process will result in more effective and more efficient ontology construction and also be able to create ontologies that better match a specific application [20]. Ontology learning has recently become a major focus for research whose goal is to facilitate the construction of ontologies by decreasing the amount of effort required to produce an ontology for a new domain. However, most current approaches deal with narrowly-defined specific tasks or a single part of the ontology learning process rather than providing complete support to users. There are few studies that attempt to automate the entire ontology learning process from the collection of domain-specific literature and filtering out documents irrelevant to the domain, to text mining to build new ontologies or enrich existing ones.

The World Wide Web is a rich source of documents that is useful for ontology learning. However, because there is so much information of varying quality covering a huge range of topics, it is important to develop document discovery mechanisms based on intelligent techniques such as focused crawling [7] to make the collection process easier for a new domain. However, due to the huge number of retrieved documents, we still require an automatic mechanism rather than domain experts in order to separate out the documents that are truly relevant to the domain of interest. Text classification techniques can be used to perform this task.

In order to enrich an ontology's vocabulary, several ontology learning approaches attempt to extract relevant information from WordNet, a semantic network database for the English language developed by Princeton University [23]. WordNet provides a rich knowledge base in which concepts, called synonymy sets or synsets, are linked by semantic relations. However, a main barrier to exploiting the word relationships in WordNet is that most words have multiple senses. Due to this ambiguity, not all senses for a given word can be used as a source of vocabulary. Expanding a concept's vocabulary based on an incorrect word sense would add many unrelated words to that concept and degrade the quality of the overall ontology. Thus, candidate word senses must be filtered very carefully. Most existing approaches have had mixed results with sense disambiguation, so the vocabulary for a specific domain mined from WordNet typically requires further manual filtering to be useful.

In our work, we employ a general ontology learning framework extracts new relevant vocabulary words from two main sources, i.e., Web documents and WordNet. This framework can be used for ontologies in any domain. We demonstrate our approach to a biological domain, specifically the domain of amphibian anatomy and morphology. In this work, we are exploring two techniques for expanding the vocabulary in an ontology: 1) lexical expansion using WordNet; and 2) lexical expansion using text mining. The lexical expansion from WordNet approach accurately extracts new vocabulary for an ontology for any domain covered by WordNet. We start with a manually-created ontology on amphibian morphology. The words associated with each concept in the ontology, the concept-words, are mapped onto WordNet and we employ a similarity computation method to identify the most relevant sense from multiple senses returned by WordNetfor a given concept-word. We then enrich the vocabulary for that original concept in the amphibian ontology by including the correct sense's associated synonyms and hypernyms.

Our text mining approach uses a focused crawler to retrieve documents related to the ontology's domain, i.e., amphibian, anatomy and morphology, from a combination of general search engines, scholarly search engines, and online digital libraries. We use text classification to identify, from the set of all collected documents, those most likely to be relevant to the ontology's domain. Because it has been shown to be highly accurate, we use a SVM (Support Vector Machine) classifier for this task [6] [40]. Finally, we implemented and evaluatedseveral text mining techniques to extract relevant information for the ontology enrichment from the surviving documents.

In this paper, we describe our work on the ontology learning process and present experimental results for each of the two approaches. In section 2, we present a brief survey of current research on ontology learning, focused crawlers, document classification, information extraction, and the use of WordNet for learning new vocabulary and disambiguating word senses. In section 3, we present our ontology learning framework and our two approaches, i.e., lexical expansion using WordNet and text mining. Sections 4 and 5 describe these approaches in more detail and report the results of our evaluation experiments. The final section presents conclusions and discusses our ongoing and future work in this area.

http://dx.doi.org/10.5772/51141

2. Related Work

An ontology is an explicit, formal specification of a shared conceptualization of a domain of interest [11], where formal implies that the ontology should be machine-readable and the domain can be any that is shared by a group or community. Much of current research into ontologies focuses on issues related to ontology construction and updating. In our view, there are two main approaches to ontology building: (i) manual construction of an ontology from scratch, and (ii) semi-automatic construction using tools or software with human intervention. It is hoped that semi-automatic generation of ontologies will substantially decrease the amount of human effort required in the process [12][18][24]. Because of the difficulty of the task, entirely automated approaches to ontology construction are currently not feasible.

Ontology learning has recently been studied to facilitate the semi-automatic construction of ontologies by ontology engineers or domain experts. Ontology learning uses methods from a diverse spectrum of fields such as machine learning, knowledge acquisition, natural language processing, information retrieval, artificial intelligence, reasoning, and database management [29]. Gómez-Pérez et al [10] present a thorough summary of several ontology learning projects that are concerned with knowledge acquisition from a variety of sources such as text documents, dictionaries, knowledge bases, relational schemas, semi-structured data, etc. Omelayenko [24] discusses the applicability of machine learning algorithms to learning of ontologies from Web documents and also surveys the current ontology learning and other closely related approaches. Similar to our approach, authors in [20] introduces an ontology learning framework for the Semantic Web thatincluded ontology importation, extraction, pruning, refinement, and evaluation giving the ontology engineers a wealth of coordinated tools for ontology modeling. In addition to a general framework and architecture, they have implemented Text-To-Onto system supporting ontology learning from free text, from dictionaries, or from legacy ontologies. However, they do not mention any automated support to collect the domain documents from the Web or how to automatically identify domain-relevant documents needed by the ontology learning process. Maedche et al. have presented in another paper [21] a comprehensive approach for bootstrapping an ontology-based information extraction system with the help of machine learning. They also presented an ontology learning framework which is one important step in their overall bootstrapping approach but it has still been described as a theoretic model and did not deal with the specific techniques used in their learning framework. Agirre et al., [1] have presented an automatic method to enrich very large ontologies, e.g., WordNet, that uses documents retrieved from the Web. However, in their approach, the query strategy is not entirely satisfactory in retrieving relevant documents which affects the quality and performance of the topic signatures and clusters. Moreover, they do not apply any filtering techniques to verify that the retrieved documents are truly on-topic. Inspiring the idea of using WordNet to enrich vocabulary for ontology domain, we have presented the lexical expansion from WordNet approach [18] providing a method of accurately extract new vocabulary for an ontology for any domain covered by WordNet.

Many ontology learning approaches require a large collection of input documents in order to enrich the existing ontology [20]. Although most employ text documents [4], only a few

deal with ontology enrichment from documents collected from the Web rather than a manually created, domain-relevant corpus. To create a corpus from the Web, one can use general purpose crawlers and search engines, but this approach faces problems with scalability due to the rapid growth of the Web. Focused crawlers, on the other hand, overcome this drawback, i.e., they yield good recall as well as good precision, by restricting themselves to a limited domain [7]. Ester et al [7] introduce a generic framework for focused crawling consisting of two major components: (i) specification of the user interest and measuring the resulting relevance of a given Web page; and (ii) a crawling strategy.

Rather than working with domain-relevant documents from which vocabulary can be extracted, some ontology construction techniques exploit specific online vocabulary resources. WordNet is an online semantic dictionary, partitioning the lexicon into nouns, verbs, adjectives, and adverbs [23]. Some current researchers extract words from WordNet's lexical database to enrich ontology vocabularies [8][16][31]. In [27], Reiter et al describe an approach that combines the Foundational Model of Anatomy with WordNet by using an algorithm for domain-specific word sense disambiguation. In another approach similar to ours, Speretta et al exploit semantics by applying existing WordNet-based algorithms [31]. They calculate the relatedness between the two words by applying a similarity algorithm, and evaluated the effect of adding a variable number of the highest-ranked candidate words to each concept. A Perl package called Word-Net::Similarity [25] is a widely-used tool for measuring semantic similarity that contains implementations of eight algorithms for measuring semantic similarity. In our work, we evaluate the WordNet-based similarity algorithms by using the JSWL package developed by [26] that implements some of the most commons similarity and relatedness measures between words by exploiting the hyponymy relations among synsets.

Semantic similarity word sense disambiguation approaches in WordNet can be divided into two broad categories based on (i) path length and (ii) information content. Warinet al [37] describe a method of disambiguating an ontology and WordNet using five different measures. These approaches disambiguate semantic similarity based on the information content, (e.g., Lin [16], Jiang-Conrath [13], and Resnik [28]) and path length method (e.g., Leacock-Chodorow [15], and Wu-Palmer [38]). They present a new method that disambiguatesthe words in their ontology, the Common Procurement Vocabulary. Semantic similarity can also be calculated using edge-counting techniques. Yang and Powers [39] present a new path-weighting model to measure semantic similarity in WordNet. They compare their model to a benchmark set by human similarity judgments and found that their geometric model simulates human judgments well. Varelas et al [35] propose the Semantic Similarity Retrieval Model (SSRM), a general document similarity and information retrieval method suitable for retrieval in conventional document collections and the Web. This approach is based on the term-based Vector Space Model by computing TF-IDF weights to term representations of documents. These representations are then augmented by semantically similar terms (which are discovered from WordNet by applying a semantic query in the neighborhood of each term) and by re-computing weights to all new and pre-existing terms.

Text mining, also known as text data mining or knowledge discovery from textual databases, refers generally to the process of extracting interesting and non-trivial patterns or knowl-

http://dx.doi.org/10.5772/51141

edge from unstructured text documents [3][12]. Tan [33] presents a good survey of text mining products/applications and aligns them based on the *text refining* and *knowledge distillation* functions as well as the *intermediate form* that they adopt. In terms of using text mining for the ontology learning task, Spasic et al. [30] summarizes different approaches in which ontologies have been used for text-mining applications in biomedicine. In another work, Velardi et al. [36] presents OntoLearn, a set of text-mining techniques to extract relevant concepts and concept instances from existing documents in a Tourism domain. Authors have devised several techniques to (i) identify concept and (ii) concept instances, (iii) organize such concepts in sub-hierarchies, and iv) detect relatedness links among such concepts.

In order to improve accuracy of the learned ontologies, the documents retrieved by focused crawlers may need to be automatically filtered by using some text classification technique such as Support Vector Machines (SVM), k-Nearest Neighbors (kNN), Linear Least-Squares Fit, TF-IDF, etc. A thorough survey and comparison of such methods and their complexity is presented in [40] and the authors in [6] conclude that SVM to be most accurate for text classification and it is also quick to train. SVM [34] is a machine learning model that finds an optimal hyper plane to separate two then classifies data into one of two classes based on the side on which they are located [5] [14].The k-nearest neighbors (kNN) algorithm is among the simplest of all machine learning algorithms: an object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors. In pattern recognition, the kNN is a method for classifying objects based on closest training examples in the feature space. It is a type of instance-based learning where the function is only approximated locally and all computation is deferred until classification. This algorithm has also been used successfully in many text categorization applications [41].

3. Ontology Learning Framework

3.1. Architecture

In this section, we present the architecture of our ontology learning process framework (c.f. Figure 1) that incorporates two approaches, i.e., lexical expansion and text mining, to identify new domain-relevant vocabulary for ontology enrichment. These approaches are presented in detail in sections 4 and 5.

3.1.1. Lexical expansion approach

This approach starts from a small manually constructed ontology, then tries to mine relevant words from WordNet in order to enrich the vocabulary associated with ontology concepts. It contains main following steps:

1. Extract all single concept-words from the seed ontology. Filter these words by removing stop-words; locate each remaining word's senses within WordNet.

2. Build a reference hypernym tree for each word sense as a reference source for semantic similarity disambiguation.
3. If a concept-word has multiple senses, identify the correct word sense usinga similarity computation algorithm on reference hypernym tree.
4. Select the most similar sense and add its synonyms and hypernyms to the corresponding concept in the ontology.

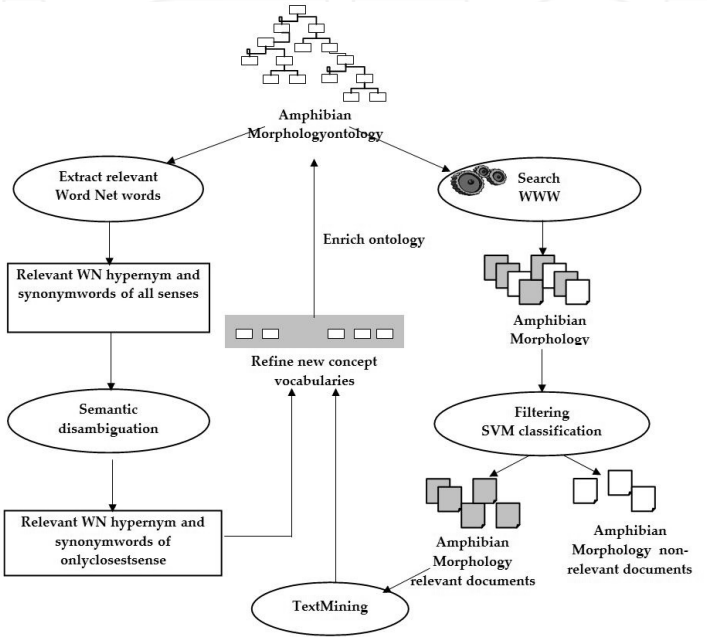

Figure 1. Architecture of ontology learning framework.

3.1.2. Text mining approach

The main processes are as following:

1. We begin with an existing small, manually-created amphibian morphology ontology [22]. From this, we automatically generate queries for each concept in the hierarchically-structured ontology.
2. We submit these queries to a variety of Web search engines and digital libraries. The program downloads the potentially relevant documents listed on the first page (top-ranked 10) results.
3. Next, we apply SVM classification to filter out documents in the search results that match the query well but are less relevant to the domain of our ontology. For example,

http://dx.doi.org/10.5772/51141

a document contains the word "cell" but it is in the context of cellphone, telecommunication… will be filtered out. Other documents containing that word "cell" with the context of amphibian, embryo structure or biological… will be kept.

After the above process, we have created a collection of documents relevant to amphibian morphology. These are input to an information extraction (IE) system to mine information from documents that can be used to enrich the ontology.

3.2. Domain and Ontology Application

The need for terminological standardization of anatomy is pressing in amphibian morphological research [22]. A long-term NSF-sponsored project, AmphibAnat, aims to integrate the amphibian anatomical ontology knowledge base with systematic, biodiversity, embryological and genomic resources. However, another important goal of this project is to semi-automatically construct and enrich the amphibian anatomical ontology. An amphibian ontology will facilitate the integration of anatomical data representing all orders of amphibians, thus enhancing knowledge representation of amphibian biology and diversity.

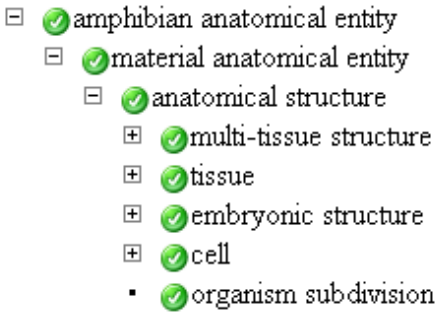

Figure 2. A part of the amphibian ontology

Based on information in a manually constructed seed ontology, we use a focused crawler and data-mining software in order to mine electronic resources for instances of concepts and properties to be added to the existing ontologies [17][19]. We also use concept-words of this ontology to match the corresponding relevant words and senses in WordNet. The current amphibian ontology created by this project consists of 1986 semantic concepts (with the highest depth level is 9) and 570 properties. Figure 2 presents a part of this ontology which is available in two main formats: (i) OWL and (ii) OBO - Open Biomedical Ontology.

4. Extracting New Vocabulary

In this section, we introduce our two main vocabulary-extraction approaches, one based on identifying information from an already-existing vocabulary resource (WordNet) and one

based on extracting vocabulary directly from the domain literature.The advantages of using WordNet is that it contains words and relationships that are, because it was manually constructed, highly accurate.However, the drawbacks are that the vocabulary it contains is broad and thus ambiguous.Also, for specialized domains, appropriate words and/or word senses are likely to be missing. Thus, we contrast this approach with one that works directlyon the domain literature.In this case, the appropriate words and word senses are present and ambiguity is less of a factor. However, the relationships between the word senses are implicit in how they are used in the text rather than explicitly represented.

4.1. Lexical Expansion Approach

This approach attempts to identify the correct WordNet sense for each concept-word and then add that sense's synsets and hypernyms as new vocabulary for the associated concept. We base our concept-word sense disambiguation on comparing the various candidate senses to those in a reference source of senses for the domain of the ontology. To provide this standard reference, we manually disambiguate the WordNet senses for the concept-words in the top two levels of the amphibian ontology. We then create a reference hypernym tree that contains the WordNethypernyms of these disambiguated WordNet senses. The problem of solving the lexical ambiguity that occurs whenever a given concept-word has several different meanings is now simplified to comparing the hypernym tree for each candidate word sense with the reference hypernym tree. We then compute a tree similarity metric between each candidate hypernym tree and the reference hypernym tree to identify the most similar hypernym tree and, by association, the word sense most closely related to the ontology domain.

4.1.1. WordNet Synonym and Hypernym

WordNet has become a broad coverage thesaurus that is now widely used in natural language processing and information retrieval [32]. Words in WordNet are organized into synonym sets, called *synsets*, representing a concept by a set of words with similar meanings. For example, frog, toad frog, and batrachian are all words in the same synset. Hypernyms, or the IS-A relation, is the main relation type in WordNet. In a simple way, we can define *"Y is a hypernym of X if every X is a (kind of) Y"*. All hypernym levels of a word can be structured in a hierarchy in which the meanings are arranged from the most specific at the lower levels up to the most general meaning at the top, for example "amphibian" is a hypernym of "frog", "vertebrate, craniate" is a hypernym of "amphibian", and so on.

For a given word, we can build a hypernym tree including all hypernyms in their hierarchy returned by WordNet. When a word has multiple senses, we get a set of hypernym trees, one per sense. In order to find the hypernyms of a given word in WordNet, we must provide the word and the syntactic class in which we are interested. In our approach, since on-

http://dx.doi.org/10.5772/51141

tologies generally represent information about objects, we are restricting the word senses considered to only the noun senses for a candidate word.

4.1.2. Reference Hypernym Tree

Because WordNet organizes nouns into IS-A hypernym hierarchies that provide taxonomic information, it is useful for identifying semantic similarity between words [25]. Our approach constructs a reference hypernym tree (or standard hypernym tree) from a few manually disambiguated words. When a word has multiple senses, we construct the hypernym tree for each sense. Then, we calculate how close each candidate sense's hypernym tree is to this reference source. We argue that the more similar a hypernym tree is to the reference tree, the closer the word sense is to the key concepts in the ontology.

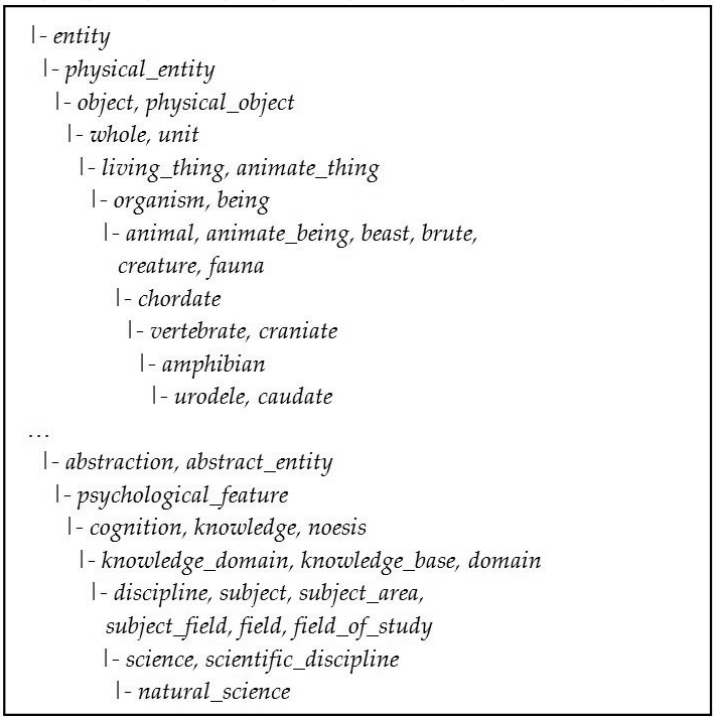

Figure 3. A part of the reference hypernym tree.

To build the reference hypernym tree, we consider only the top two levels of concepts of the amphibian ontology since they cover many vocabularies appearing at lower levels. In addition, since the concepts are also related using the "IS A" relation, the hypernyms of the concepts in top two levels of the ontology are also hypernyms of the concepts in the lower levels. The main steps of this process are as following: first, we convert each concept in top

two level into concept-words that can be submitted to WordNet, for instance the concept "anatomical_structure" is divided into two concept-words "anatomical" and "structure". The top two levels of our ontology contain 15 concepts and 19 concept-words. Then, we use WordNet to collect all the hypernyms for each concept-word. We manually choose the hypernym that best matches the meaning in our domain and then add to the reference hypernym tree. Figure 3 presents a part of our reference hypernym tree that contains 110 hypernyms covering 12 hierarchical levels. This reference tree will be used as truth to evaluate the correct sense extracted from WordNet for each experiment word.

4.1.3. Similarity Computation Algorithm

In this section, we present our tree-based similarity measurement technique that is used to compare hypernym trees corresponding to different senses of a concept-word with the reference hypernym tree. Our similarity measure is based on two factors:

- *Matched Tree Depth (D):* Since all hypernym trees are represented from the most general (top level) to the more specific meanings (lower levels), matches between hypernyms in the lower levels of the two trees should be ranked higher. For example, matching on "frog" is more important than matching on "vertebrate". We calculate the depth (D) as the distance from the most general hypernym (i.e., "entity") until the first hypernym that occurs in both the reference hypernym tree and a candidate hypernym tree. The higher valued matched tree depth indicates that the meaning of the matched word is more specific, thus it would be more relevant to the domain.

- *Number of common elements (CE):* Normally, if two trees contain many of the same entities, they are judged more similar. Once we determine the depth (D), we count the number of common elements between the reference hypernym tree and a candidate tree. If two candidate trees have the same match depth, then the tree with more elements in common with the reference hypernym tree would be considered the more similar one.

Once the hypernym tree is constructed for each sense, we are able to apply the similarity computation algorithm to calculate these two above factors (i.e. D and CE) for each case. Then, we weight these two factors to get the highest value.

Figure 4 shows an example of how the reference hypernym tree can be used to disambiguate word senses and select the most relevant one. Suppose that the hypernym trees in this example, i.e., HTree1, HTree2 and HTree3, each correspond to one sense of a given concept-word. We compare each hypernym tree to the reference hypernym tree by calculating the matched tree depth and the number of common elements between two trees. Results show that the first tree HTree1 is closest to the standard tree with the depth (D=6) and number of common elements (CE=13) are greater than others.

http://dx.doi.org/10.5772/51141

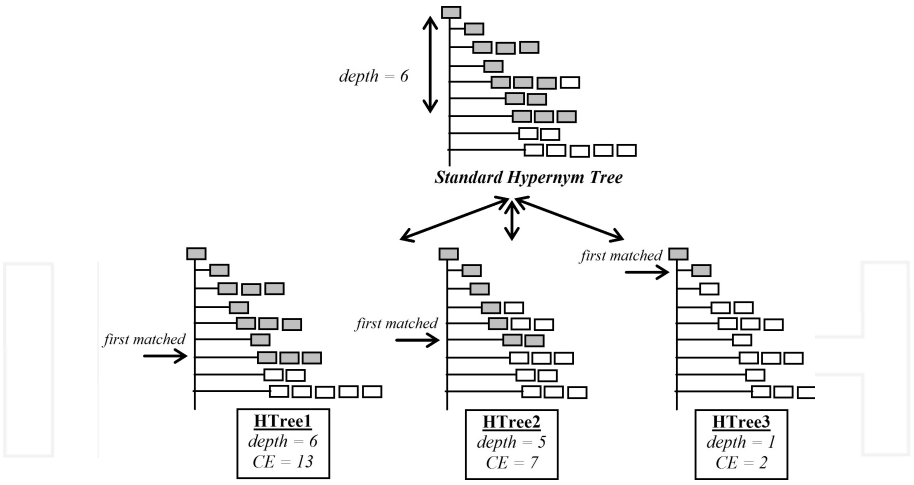

Figure 4. Similarity computation using the Reference (Standard) HypernymTree.

4.1.4. Experiments

a. Dataset and Experiments

As discussed in Section 3.2, we use the amphibian morphology ontology developed by the AmphibAnat project [22] for our experiments. We used WordNet (3.0) as the lexical source and the API library of MIT Java WordNet Interface (JWI 2.1.5) to locate the synsets and hypernyms for a given concept-word.

We begin by processing names for the concepts in the top two levels of the amphibian ontology and manually disambiguating them to create the reference hypernym tree. We then process the names for the 1971 concepts in levels 3 and below. Since many concepts are named with phrases of two or more wordsm and WordNet only contains single words, we need to convert these words into single words, i.e., the concept-words. After removing duplicate concept-words, e.g., concepts "anatomical_structure" and "multi-tissue_structure" have a duplicate concept-word, "structure", stopwords (e.g., of, to, for) and numbers, we end up with a set of 877 unique concept-words. However, because the ontology is very domain specific, many of these concept-words do not appear in WordNet at all. Very specific terms in the amphibian domain, e.g., premaxilla, dorsalis, adjectives, e.g., nervous, embryonic, hermaphroditic, and incomplete words added during the ontology creation process, e.g., sp, aa, do not appear in WordNet. Therefore, we report results using only the 308 concept-words that were contained in WordNet.

We matched these 308 words to WordNet to identify all their corresponding senses, synonyms and hypernyms. However, not all of the senses for a given word are relevant so we need to disambiguate the semantic similarity of these senses to choose the closest and most correct one for our amphibian domain. We applied each of three similarity computation al-

gorithms, to determine the closest sense for each word and evaluated them: 1) based on the number of matched tree depth (#Depth); 2) based on the number of common elements (#CE); and 3) based on the sum of both factors (#Depth+CE). For each method, we got a list of word senses selected by the algorithm and compared these senses to the truth-list provided by our human expert to evaluate the effectiveness of our algorithm.

b. Evaluation Measures

In order to evaluate the results of our both approaches, i.e., Lexical Expansion and Text Mining, we needed to know the correct word sense (if any) for each word. An expert in the amphibian domain helped us by independently judging the words and senses identified by each technique and identifying which were correct. These judgments formed the truth lists against which the words extracted by text mining and the senses chosen by the lexical expansion were compared.

Information extraction effectiveness is measured in terms of the classic Information Retrieval metrics of Precision, Recall and F-measure. We used these measures for both approaches, but with some different definitions of measure for each approach. In the lexical expansion approach, since each word in WordNet may have different senses and we have to find the correct one, we define:

- *Precision (P):* measures the percentage of the words having correct sense identified by our algorithm that matched the sense judged correct by the human expert.

$$P = \frac{\#\_words\_having\_correct\_sense\_identified}{\#\_words\_returned} \tag{1}$$

- *Recall (R):* measures the percentage of the correct words identified by our algorithm that matched those from the truth-list words.

$$R = \frac{\#\_words\_having\_correct\_sense\_identified}{\#\_truth-list\_words} \tag{2}$$

We use F-measure which is calculated as following for both approaches.

$$F_\beta = \frac{(1+\beta^2) * P * R}{\beta^2 * P + R} \tag{3}$$

Because we want to enhance the ontology with only truly relevant words, we want a metric that is biased towards high precision versus high recall. We chose to use the F-measure with a β value that weights precision four times higher than recall.

4.1.5. Result Evaluation

Among 308 word returned from WordNet, human expert judged that 252 of the extracted words were correct. Of the 56 incorrect words returned, there were no senses in WordNet

http://dx.doi.org/10.5772/51141

relevant to the amphibian domain. Since each of above 252 returned words may have some correct senses judged by the human expert, we got finally 285 correct senses.In order to see how well each algorithm performed, we implemented different thresholds to see how results are varied as we increased our selectivity based on level of match and/or number of common elements matched. We varied the depth from 1 to 10, the common elements from 2 to 20, and the sum of the two from 3 to 30. For example with the #Depth+CE method, the range of threshold t is from 3 to 30; the value t=15 means that we take only senses having the number #Depth+CE is greater than 15. Figures 5, 6 and 7 show the F-measures achieved by three methods of #Depth, #CE and #Depth+CE respectively using various threshold values.

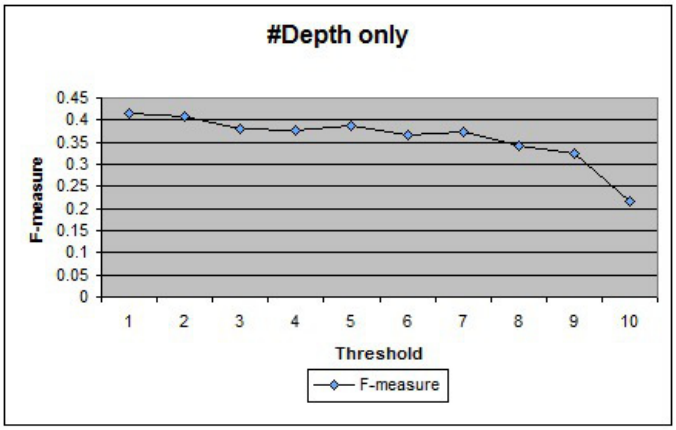

Figure 5. F-measure of the #Depth only method (β=0.25).

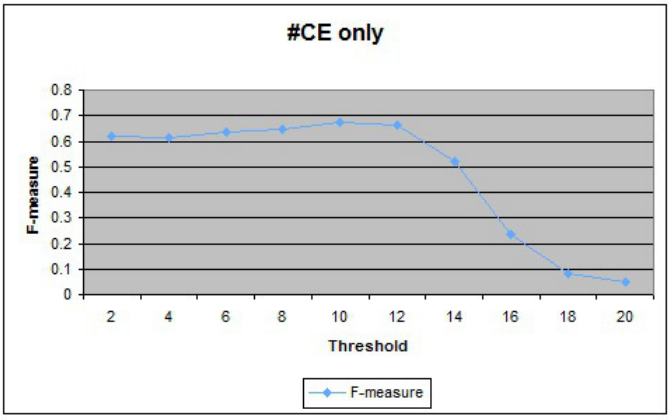

Figure 6. F-measure of the #CE only method (β=0.25).

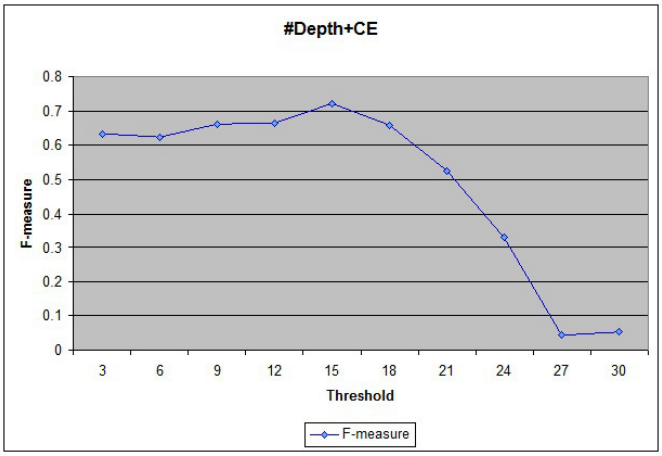

Figure 7. F-measure of the #Depth+CE method (β=0.25).

We found that we got poor results when using depth only, i.e., a maximum F-measure of 0.42 with a threshold of 1, but the common elements approach performed better with a maximum F-measure of 0.67 with a threshold of 10. However, the best result was achieved when we used the sum of depth and common element using a threshold t=15. This produced a precision of 74% and recall of 50% and an F-measure of 0.72. In this case, 155 words were found by the algorithm of which 115 were correct. Table 1 presents the number of unique synonym and hypernym words we could add to the ontology from WordNet based on our disambiguated senses.

Threshold	**3**	**6**	**9**	**12**	**15**	**18**	**21**	**24**	**27**	**30**
#words returned	308	291	271	225	155	98	62	17	8	4
# words correct	190	178	176	149	115	70	38	10	1	1
#SYN	417	352	346	308	231	140	78	26	1	1
#HYN	164	156	151	123	80	48	42	12	2	2

Table 1. Number of retrieved synonyms and hypernyms words.

To better understand this approach, we present an example based on the concept-word "cavity." WordNet contains 4 different senses for this word: 1) pit, cavity, 2) cavity, enclosed space; 3) cavity, caries, dental caries, tooth decay; and 4) cavity, bodily cavity, cavum. Based on the total value of #Depth+CE, our algorithm correctly selects the fourth sense as the most relevant for the amphibian ontology (c.f., Table 2). Based on this sense, we then consider the synonyms *[cavity, bodily_cavity, cavum]* and hypernyms *[structure, anatomical_structure, complex_body_part, bodily_structure, body_structure]* as words to enrich the vocabulary for the concept "cavity". This process is applied for all concept-words in the seed amphib-

http://dx.doi.org/10.5772/51141

ian ontology and new mined vocabularies will be used to enrich the corresponding concepts of the ontology.

	# Depth	#CE	#Depth+CE
1st sense	5	7	12
2nd sense	4	7	11
3rd sense	10	7	17
4th sense	**15**	**7**	**22**

Table 2. Example of semantic similarity computation for the concept-word "cavity".

4.2. Text Mining Approach

4.2.1. Searching and Collecting Documents

In order to collect a corpus of documents from which ontological enrichments can be mined, we use the seed ontology as input to our focused search. For each concept in a selected subset of ontology, we generate a query that is then submitted to two main sources, i.e., search engines and digital libraries. To aid in query generation strategies, we created an interactive system that enables us to create queries from existing concepts in the ontology and allows us to change parameters such as the Website address, the number of returned results, the format of returned documents, etc.

We next automatically submit the ontology-generated queries to multiple search engines and digital libraries related to the domain (e.g., Google, Yahoo, Google Scholar, http://www.amphibanat.org). For each query, we process the top 10 results from each search site using an HTML parser to extract the hyperlinks for collecting documents.

4.2.2. Classifying and Filtering Documents

Although documents are retrieved selectively through restricted queries and by focused search, the results can contain some documents that are less relevant or not relevant at all. Therefore, we still need a mechanism to evaluate and verify the relevance of these documents to the predefined domain of the ontology. To remove unexpected documents, first we automatically remove those that are blank or too short, are duplicated documents, or those that are in a format that is not suitable for text processing. We then use LIBSVMclassification tool [5] to separate the remaining documents into two main categories: (i) relevant and (ii) non-relevant to the domain of ontology. We have also varied different parameters of LIBSVM such as kernel type (-t), degree (-d), weight (-wi)… in order to select the best parameters for our classification. Only those documents that are deemed truly relevant are input to the pattern extraction process.

The SVM classification algorithm must first be trained, based on labeled examples, so that it can accurately predict unknown data (i.e., testing data). The training phase consists of find-

ing a hyper plane that separates the elements belonging to two different classes. Our topic focused search combining with the SVM classification as described in [17] is 77.5% accurate, in order to evaluate our text mining approach in the absence of noise, we report in this paper our results based on the labeled relevant training examples. In future, the topic focused crawler will be used to feed directly into the text mining process.

4.2.3. Information Extraction using Text Mining

After the topic-specific searching produces a set of documents related to the amphibian morphology domain, this phase information mines important vocabulary from the text of the documents. Specifically, our goal is to extract a set of words that are most related to the domain ontology concept-words. We have implemented and evaluated a vector space approach using two methods to calculate the *tf*idf* weights. Since most ontology concept-words are nouns, we also explored the effect of restricting our extracted words to nouns only. The weight calculation methods compared were the *document-based selection* and *corpus-based selection*. In both approaches, in order to give high weights to words important to the domain, we pre-calculated the *idf* (inverse document frequency) from a collection of 10,000 documents that were randomly downloaded from a broad selection of categories in the ODP[1] collection.

a. Document-based selection (L1)

This method, *L1*, first calculates weights of words in each document using *tf*idf* as follows:

$$W(i,\ j) = rtf_{(i,j)} * idf_i \tag{4}$$

$$rtf_{(i,j)} = \frac{tf_{(i,j)}}{N(j)} \tag{5}$$

$$idf_i = \log \frac{|D|}{|\{d : t_i \in d\}|} \tag{6}$$

where

$W(i,j)$ is the weight of term i in document j

$rtf_{(i,j)}$ is the relative term frequency of term i in document j

idf_i is the inverse document frequency of term i, which is pre-calculated across 10,000 ODP documents

$tf_{(i,j)}$ is the term frequency of term i in document j

$N(j)$ means the number of words in document j

$|D|$ is the total number of documents in the corpus

1 Open Directory Project http://www.dmoz.org/

http://dx.doi.org/10.5772/51141

$|\{d:t_i d\}|$ is number of documents in which t_i appears.

A word list sorted by weights is generated for each document from which the top k words are selected. These word lists are then merged in sorted order these words to only one list ranked by their weight. We performed some preliminary experiments, not reported here, which varied k from 1 to 110. The results reported here use k = 30, a value that was found to perform best.

b. Corpus-based selection (L2)

This method, *L2*, calculates weights of words by using *sum(tf)*idf*. Thus, the collection based frequency is used to identify a single word list rather than selecting a word list for each document based on the within-document frequency and then merging, as is done in method L1. The formula is thus:

$$W(i) = \sum_{j=1}^{n} rtf_{(i,j)} * idf_i \tag{7}$$

where

W(i) is the weight of term i in the corpus

The other factors are calculated as in L1 method.

c. Part-of-speech restriction (L1N, L2N)

For each of the previous approaches, we implemented a version that removed all non-nouns from the final word list. We call the word lists, *L1N* and *L2N*, corresponding to the subset of words in lists *L1* and *L2* respectively that are tagged as nouns using the WordNet library [25] using JWI(the MIT Java WordNet Interface).

4.2.4. Experiments

a. Dataset and Experiments

The current amphibian ontology is large and our goal is to develop techniques that can minimize manual effort by growing the ontology from a small, seed ontology. Thus, rather than using the whole ontology as input to the system, for our experiments we used a subset of only the five top-level concepts from the ontology whose meaning broadly covered the amphibian domain. Ultimately, we hope to compare the larger ontology we build to the full ontology built by domain expert.

We found that when we expand the query containing the concept name with keywords describing the ontology domain overall, we get a larger number of relevant results. Based on these explorations, we created an automated module that, given a concept in the ontology, currently generates 3 queries with the expansion added, e.g., “amphibian” “morphology” “pdf”. From each of the five concepts, we generate three queries, for a total of 15 automatically generated queries. Each query is then submitted to each of the four search sites from

which the top ten results are requested. This results in a maximum of 600 documents to process. However, because some search sites return fewer than ten results for some queries, we perform syntactic filtering, and duplicate documents are returned by search engines, in practice this number was somewhat smaller.

It is crucial to have a filtering stage to remove irrelevant and slightly relevant documents to the amphibian ontology. We have adopted an SVM-based classification technique trained on 60 relevant and 60 irrelevant documents collected from the Web. In earlier experiments, our focused search combining with the SVM classification was able to collect new documents and correctly identify those related to the domain with an average accuracy 77.5% [17][19].

Ultimately, the papers collected and filtered by the topic-specific spider will be automatically fed into the text mining software (with an optional human review in between). However, to evaluate the effectiveness of the text mining independently, without noise introduced by some potentially irrelevant documents, we ran our experiments using 60 documents manually judged as relevant, separated into two groups of 30, i.e., *Group_A* and *Group_B*. All these documents were preprocessed to remove HTML code, stop words and punctuation. We ran experiments to tune our algorithms using *Group_A* and then validated our results using *Group_B*.

For the document-based approach, we took the top 30 words from each document and merged them. This created a list of 623 unique words (*L1*). To be consistent, we also selected the top 623 words produced by the corpus based approach (*L2*). We merged these two lists and removed duplicates, which resulted in a list of 866 unique words that were submitted for human judgment. Based on our expert judgment, 507 of the total words were domain-relevant and 359 were considered irrelevant. These judgments were then used to evaluate the 6 techniques, i.e., *L1*, *L2*, *L1+L2*, *L1N*, *L2N* and *L1N+L2N*.

When we selected only the nouns from L1 and L2, this resulted in lists L1N and L2N that contained 277 and 300 words, respectively. When these were merged and duplicates were removed, 375 words were submitted for judgment of which 253 were judged relevant and 122 were judged irrelevant to the amphibian morphology domain

b. Evaluation Measures

In the text mining approach, we define:

- *Precision (P):* measures the percentage of the correct words identified by our algorithm that matched those from the candidate words.

$$P = \frac{\#\_correct\_words\_identified}{\#\_candidate\_words} \tag{8}$$

- *Recall (R)*: measures the percentage of the correct words identified by our algorithm that matched those from the truth-list words.

http://dx.doi.org/10.5772/51141

$$R = \frac{\#\_correct\_words\_identified}{\#\_truth-list\_words} \tag{9}$$

Based on these two measures, we also calculate the F-measure as the equation (3) to evaluate methods' performances.

4.2.5. Results

We evaluated our results by comparing the candidate word lists that were extracted from the relevant documents using our algorithms with the judgments submitted by our human domain expert. Since not all words on the word lists are likely to be relevant, we varied how many of the top weighted words were used. We chose threshold values t from 0.1 to 1.0 corresponding to the percentage of top candidate words that are extracted (e.g., t=0.1 means that top 10% words are selected). We carried out 6 different tests corresponding to the four candidate lists, i.e., *L1*, *L2*, *L1N*, *L2N* and two more cases *L1+L2* (average of *L1* and *L2*) and *L1N+L2N* (average of *L1N* and *L2N*) as input to our algorithm. These tests are named by their list names *L1*, *L2*, *L1+L2*, *L1N*, *L2N* and *L1N+L2N*. Figure 8 presents the F-measures achieved by these tests using various threshold values.

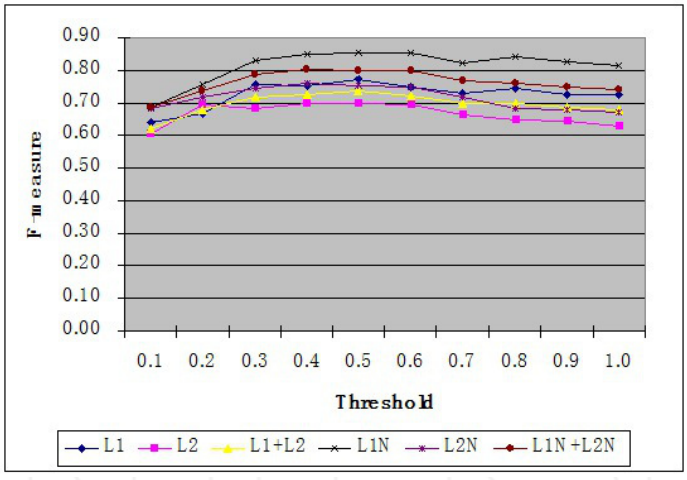

Figure 8. F-measure of the tests in Group_A (β=0.25).

The best result was achieved in the test L1N, using the highest weighted nouns extracted from individual documents. By analyzing results, we find that the best performance is achieved with a threshold t=0.6, i.e., the top 60% of the words (277 words total) in the candidate list are used. This threshold produced precision of 88% and recall of 58% meaning that 167 words were added to the ontology of which 147 were correct.

To confirm our results, we validated the best performing algorithm, *L1N* with a threshold of 0.6, using the 30 previously unused relevant documents in *Group_B*. We applied the document-based selection algorithm using nouns only with a threshold value 0.6. In this case, the achieved results are P = 77%, R = 58% and F-measure = 0.7. This shows that, although precision is a bit lower, overall the results are reproducible on a different document collection. In this case 183 words were added to the ontology of which 141 were correct.

Threshold	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
#candidatewords	28	55	83	110	139	167	194	222	249	277
# words added	22	50	77	101	124	147	162	188	206	225

Table 3. Number of words can be added.

Table 3 reports in more detail on the number of candidate words and how many correct words can be added to the ontology through the text mining process with the document-based selection and restricting our words to nouns only, i.e. the L1N test with threshold 0.6 on the validation documents, *Group_B*. We also observe that the top words extracted using this technique are very relevant to the domain of amphibian ontology, for example, the top 10 words are: frog, amphibian, yolk, medline, muscle, embryo, abstract, pallium, nerve, membrane.

4.3. Discussions

The experimental results show that the ontology enrichment process can be used with new relevant vocabularies extracted from both of our approaches of text mining and lexical expansion. Overall, the text mining approach achieved a better result than the WordNet approach. Table 4 shows the performance comparison of these two approaches in their best case.

Measures	Lexical Expansion Approach	Text Mining Approach
Precision	0.74	0.88
Recall	0.50	0.58
F-Measure	0.72	0.85
#Candidate Words	155	167
#Words mined correctly	115	147
#Words mined incorrectly	40	20

Table 4. Comparison of two approaches in the best case.

In the text mining approach, we got the best results using a vector space approach with the document-based selection and restricting our words to nouns only. Overall, our algorithm produced good accuracy, over than 81% for all cases. If we restrict our candidates to only

http://dx.doi.org/10.5772/51141

the top-weighted candidates extracted from the documents, the precision is higher but the recall decreases. In the best case, where the F-measure is maximized, the precision is 88% on the test collection. Our algorithm was also validated with another dataset (i.e. documents in *Group_B*), the precision in this case decreases to 77% which is still acceptable and does not affect significantly to the number and quality of relevant words extracted.

The results in the lexical analysis approach also show that our similarity computation method can provide an effective way to identify the correct sense for words in a given ontology. These senses then provide a source of new vocabulary that can be used to enrich the ontology. Our algorithm was best with the precision of 74% in the method of #Depth+CE. If we consider words with fewer senses, the accuracy of detection of correct words is higher. This level of accuracy was achieved using a reference hypernym tree with 110 hypernyms, built from only 14 manually disambiguated concept-words from the top two levels of our amphibian morphology ontology. From these words, we are able to identify a large collection of synonyms and hypernyms that are a good potential source for the ontology enrichment process.

5. Ontology Enrichment

In previous sections, we have presented our main approaches to mine new relevant vocabularies to enrich a domain ontology from two main sources: (i) the WordNet database; and (ii) domain-relevant documents collected from the Internet. In this section we describe how we enrich the domain ontology by adding these newly-mined words to the appropriate concepts. Our main task in this phase is finding a method to add correctly a new candidate word to the best appropriate ontology concept vocabulary.

5.1. Lexical Expansion Approach

As presented in the section 4.1, our lexical expansion approach has identifies the correct WordNet sense for each concept-word and then considering that sense's synsets and hypernyms as new vocabulary for the associated concept.In the previous phase, we used the hypernyms trees to identify the correct sense for each concept-word. For each correct sense, we add the synonym words (synsets) of that sense to the ontology concept vocabulary. Because we know which concept the synset is associated with, an advantage of this approach is that it is trivial to attach the new vocabulary to the correct concept.

From the 285 correct senses mined in the first phase, our lexical expansion mineda total of 352 new synonym words over 155 concept words that were returned. Many synonym words were duplicates, so we refined the candidate vocabulary by removing redundant words, leaving 321 new words to add to the ontology.

To evaluate the accuracy of this approach, we presented each of these 155 word-concept pairings to a human domain expert and asked them to validate whether or not the pairing is accurate. In their judgment, 115 words were relevant to the amphibian morphology domain, and their 260 synonyms words were considered. We then refined these synonym words by

removing duplicates. Thus, we ultimately added 231 words to the appropriate concepts, almost entirely automatically. The precision of adding correct synonym words to the ontology was 71.9%. The only manual step in this process is the filtering out of the incorrect word-senses extracted from WordNet.

5.2. Text Mining Approach

5.2.1. Overview

To reiterate our task, given a list of potential synonyms to be added to an ontology, we want to develop an algorithm to automatically identify the best matching concept for each candidate. Unlike the WordNet approach, this is a much more difficult task for words mined from the literature. The candidate words are domain relevant, but exactly where in the ontology do they belong?We again turn to thedomain-relevant corpus of documents to determine the concept in the ontology to which these newly mined candidate words should be attached. The main differences between our approach and that of others are:

- The documents are used as the knowledge base of word relationships as well as the source of domain-specific vocabulary. Specifically, for each concept (and each candidate word), chunks of text are extracted around the word occurrences to create concept-word contexts and candidate word contexts. The words around each concept and/or candidate word occurrence thus contain words related to the semantics of the concept/candidate word.
- Word phrases can be handled. Instead of handling each word in a word phrase separately and then trying to combine the results, as we did with WordNet, we can process the word phrase as a whole. For each word phrase, we extract text chunks in which all of the words occur. Thus, the contexts are related to the concept-word as a whole, not just the individual component words.

The four main steps of our approach are:

1. Extract text chunks surrounding the concept-word(s) from each input document. Combine the text chunks extracted to create context files that represent the concept. Using the same method, create context files for each candidate word (i.e., potential synonym).
2. For each candidate, calculate the context-based similarity between the candidate context and each concept context.
3. Identify the most similar concepts using kNN (k nearest neighbors).
4. Assign the candidate words to the concept(s) with the highest similarity.

5.2.2. Extracting concept-word contexts

This process begins with a set of domain-related documents that are used as a knowledge base. From each preprocessed text file, we locate all candidate word occurrences and then

http://dx.doi.org/10.5772/51141

created windows of plus/minus N words around each occurrence. Overlapping windows are then combined to create non-overlapping text chunks. Because varying numbers of overlapping windows may be combined to create the text chunks, the text chunks vary in size and in the number of word occurrences they contain. In order to illustrate the text chunks selection process, consider a single document representing the concept name "neural_canal" with the query keywords are "neural" and "canal".

a. Step 1: Identify windows around each query keyword

In one part of the document under consideration with the window size of +-10, there are three occurrences of both terms "neural" and "canal" and thus three windows of size 21 with a word in the middle. We can see that Windows 1, 2, and 3 all contain keywords and they are overlapping.

Window 1: (middle keyword is "neural")

morphy pronounced reduction process boulengerella lateristriga maculata hypothesized synapomorphic species neural synapomorphy laterosensory canal system body majority characiforms completely developed posterior

Window 2: (middle keyword is "canal")

process boulengerella lateristriga maculata hypothesized synapomorphic species neural synapomorphy laterosensory canal system body majority characiforms completely developed posterior neural laterosensory canal

Window 3: (middle keyword is "neural")

synapomorphy laterosensory canal system body majority characiforms completely developed posterior neural laterosensory canalsystem body nearly lateral line scales minimally pore

b. Step 2: Combine overlapping chunks of text

In this step, we combine all overlapping windows to create the text chunks as following.

morphy pronounced reduction process boulengerella lateristriga maculata hypothesized synapomorphic species neural synapomorphy laterosensory canal system body majority characiforms completely developed posterior neural laterosensory canal system body nearly lateral line scales minimally pore

c. Step 3: Combining chunks of text across document

In this step, we simply combine the text chunks extracted for a given context from a document by appending them together.

5.2.3. Generating Vectors for the Extracted Contexts

After having extracted text chunks surrounding to keyword instances (i.e., concepts or synonyms), we have two sets of contexts: (1) **S** representing candidate synonyms; and (2) **C** representing concepts. In this step, we transform the contexts to vectors representing the keywords. We adopted *tf*idf* approach to index tokens from the context files and assign weight values to these tokens. Thus, all keywords can be represented by a series of features

(i.e., weighted tokens) extracted from the relevant chunks. We have generated two types of vectors for each keyword, i.e., *individual vectors* and a *centroid vector*. An *individual vector* summarizes the contexts around a keyword within a single document while a centroid vector combines the individual vectors to summarize all contexts in which a keyword appears.

a. Individual Vector

Let's take an example of a concept keyword C_k represented by 5 context files extracted from 5 relevant documents (i.e., N=5). This keyword will have 5 individual vectorsrepresenting for each individual context file. We used the KeyConcept package [9] to index features and calculate their weights; each individual vector has the following format:

$$IND_i\_C_k = (feature_{i1} : weight_{i1},\ feature_{i2} : weight_{i2},\ \ldots,\ feature_{im} : weight_{im}) \tag{10}$$

in which $weight_{ij} = rtf_{ij} * idf_j$, the relative term frequency rtf_{ij} and the inverse document frequency idf_{ij} are respectively calculated by the following formulas

$$rtf_{ij} = \frac{\#\ feature_j\_in\_context_i}{\#\ features\_in\_context_i} \tag{11}$$

$$idf_j = \log\left(\frac{\#\ documents\_in\_collection}{\#\ documents\_containing\_feature_j}\right) \tag{12}$$

In our experiments, the document collection we used is very specific to the amphibian morphology domain. In order to make the idf value more fair and accurate, we adopt the idf dataset based on the ODP that we had used in the previous paper [19].

b. Centroid Vectors

The centroid vector of the concept keyword C_k is calculated based on the individual vectors over N relevant documents.

$$CEN\_C_k = \sum_{i=1}^{N} IND_i\_C_k \tag{13}$$

5.2.4. Similarity Computation and Ranking

In the next sections, we present our methods to calculate the similarity between two sets, i.e., (1) S representing candidate synonyms; and (2) C representing concepts, to create and rank a list of synonym-concept assignments.

Since each context is essentially a collection of text, we use the classic cosine similarity metric from the vector space model to measure the semantic relatedness of two contexts. For each context S_m representing a candidate synonym and C_n representing a concept, the similarity of each pair (S_m, C_n) is the weight of cosine similarity value calculated by the following formula:

http://dx.doi.org/10.5772/51141

$$sim(S_m, C_n) = \frac{\sum_{i=1}^{t} w_{i,m} \times w_{i,n}}{\sqrt{\sum_{i=1}^{t} w^2_{i,m}} \times \sqrt{\sum_{i=1}^{t} w^2_{i,n}}} \quad (14)$$

where

$w_{i,m}$: weight of the feature i in the vector representing S_m

$w_{i,n}$: weight of the feature i in the vector representing C_n

t: total number of features

Note that these similarity values are normalized by the size of the context.

Since each keyword is presented either by N individual vectors (i.e., N context files) or a centroid vector over N context files, we propose to conduct four different methods to calculate the similarities between candidate synonym vectors and concept vectors:

- *Indi2Indi*: calculates the similarity value between each individual vector of a synonym keyword in S_m and each individual vector of the concept keyword in C_n.
- *Cent2Indi*: calculates the similarity value between the centroid vector of a synonym keyword in S_m and each individual vector of the concept keyword in C_n.
- *Indi2Cent*: calculates the similarity value between each individual vector of a synonym keyword in S_m and a centroid vector of the concept keyword in C_n.
- *Cent2Cent*: calculates the similarity value between the centroid vector of a synonym keyword in S_m and a centroid vector of the concept keyword in C_n.

Finally, for each candidate synonym it is assigned to the top concepts whose context C_n has the highest similarity values to the context for S_m.

Once the similarity between each word pair in (S_m, C_n) is calculated, we use the kNN method to rank the similarity computation results. The higher the similarity means the candidate synonym would be closer to the target concept. For each candidate word, we take top k items from the similarity sorted list. Then, we accumulate the weights of pairs having the same concept names. The final ranked matching list is determined by the accumulated weights. We then pick the pairs having highest weights value between S_m and C_n as the potential concepts to which the candidate synonym is added.

5.2.5. Experiments

a. Baseline: Existing WordNet-Based Algorithms

We evaluated four WordNet-based algorithms that had produced high accuracy as reported, i.e., Resnik [27], Jiang and Conrath [13], Lin [16], and Pirro and Seco [26], using all 191 test synonyms in our dataset.

	Algos	Top1	Top2	Top3	Top4	Top5	Top6	Top7	Top8	Top9	Top10
Precision # Correct pairs identified	LIN	1	4	4	9	11	12	13	13	15	15
	JIANG	1	2	3	7	7	9	10	11	11	11
	RESNIK	2	9	9	10	15	16	16	16	17	17
	PIRRO	1	3	4	5	9	10	10	12	13	13
	LIN	0.52 %	2.09 %	2.09%	4.71%	5.76%	6.28%	6.81%	6.81%	7.85%	7.85%
	JIANG	0.52 %	1.05 %	1.57%	3.66%	3.66%	4.71%	5.24%	5.76%	5.76%	5.76%
	RESNIK	**1.05 %**	**4.71 %**	**4.71%**	**5.24%**	**7.85%**	**8.38%**	**8.38%**	**8.38%**	**8.90%**	**8.90%**
	PIRRO	0.52 %	1.57 %	2.09%	2.62%	4.71%	5.24%	5.24%	6.28%	6.81%	6.81%

Table 5. Evaluation of WordNet-based algorithms.

Our baseline approach using WordNet was complicated by the fact that it does not contain word phrases, which are very common in the concepts and synonyms in our ontology. We separate each word phrase (e.g., neural_canal) into single words, e.g., neural and canal, then submit each to WordNet to calculate the similarity. Then, we add together the individual word similarity score to produce the total similarity value for each phrase. For each of the 191 synonyms in the test set, the similarity between that synonym and each of the 530 concept names pairs was calculated, and then the concepts were ranked in decreasing order by similarity. Table 5 presents the accuracy for each method at different cutoff points in the list.

From the Table 5, we can see that Resnik's algorithm works best with this test set, although is only 1.05% accurate at the rank 1 location and 8.9% the correct concept appeared within the top 10 ranked concepts. The other approaches are not as accurate.

b. Context-Based Algorithms

We evaluated results by comparing the list of top ranked concepts for each synonym produced using our context-based method with the truth list we extracted from the ontology. The performance is reported over varying text chunk sizes (or text window sizes - *wsize*), from 4 to 40 tokens. Due to space limits, we only report here the best case achieved for each method (c.f. Figure 9). We also evaluated different values of k from 5 to 30 in the kNN methods to determine the k value that returned the best result. The numbers of k reported in each chart were different since the number or results generated depend on the number of vectors compared by each method. For example, when we measure the similarity between a synonym and a concept name, the *Indi2Indi* method used 5 individual vectors for each synonym and concept; but we have only one vector for each keyword in the *Cent2Cent* method.

http://dx.doi.org/10.5772/51141

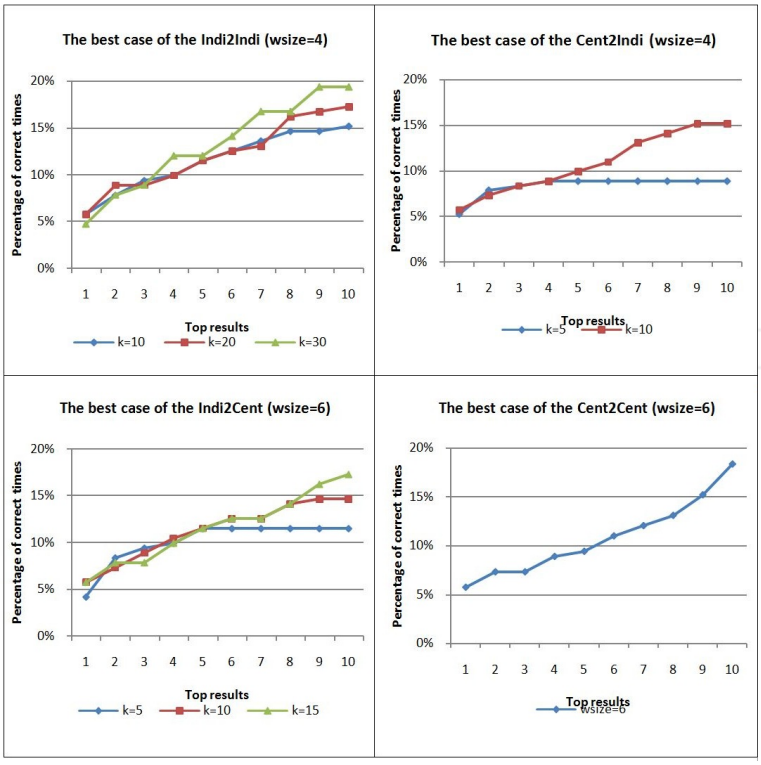

Figure 9. Result of the context-based similarity computation methods

By analyzing results, we find that the best performance is achieved in the Indi2Indi method with the window size (wsize) 4 and the k value 30 (c.f. Figure 9). Other methods slightly performed less well than the Indi2Ind. When the window size is increased, the performance is not improved.

c. Comparison with Baseline

From the above charts (c.f. Figure 9), we found that the context-based method works better with small text window size (i.e. size=4, 6). We evaluated the same reported pairs of synonym and concept-words with our text mining methods and the above WordNet-based algorithms. Figure 10 shows that our context-based similarity computation methods consistently outperform the best WordNet-based algorithm. Even the less promising *Cent2Cent* method produces performance better than the best case of the WordNet-based similarity algorithms (i.e., RESNIK's algorithm).

The experimental results support our belief that we might not need to rely on the WordNet database to determine similar words. The WordNet-based algorithms have not provided high precision results since many words are compound and/or do not exist in the WordNet database.

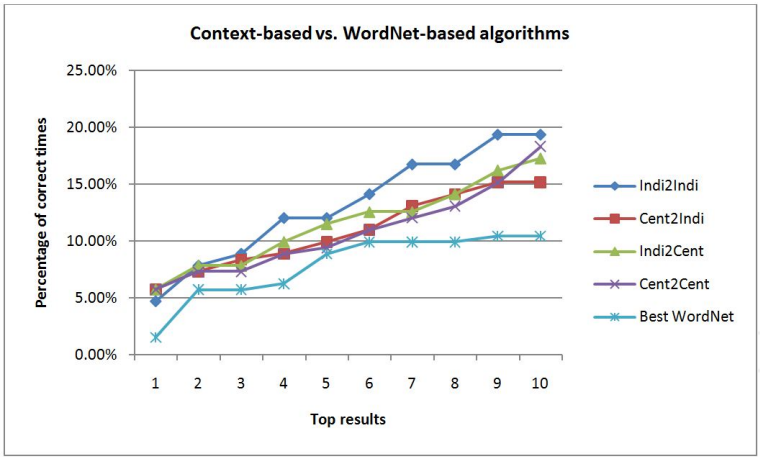

Figure 10. Comparison of our context-based methods vs. WordNet-based algorithms

5.3. Discussion

Our context-based approach was evaluated using synonyms from the amphibian ontology itself as our truth dataset. The experimental results show that our text mining approaches outperform the common similarity calculation algorithms based on WordNet in terms of correctly adding new words to the ontology concepts. Our best algorithm, Indi2Indi, with a window size of 4 and k=30 for kNN, selected the correct concept within the top 10 concepts 19.37% of the time. In comparison, the best WordNet-based algorithm only achieved 8.9% on the same task.

Overall, using WordNet similarity algorithms to assign a new vocabulary to an existing ontology might not be an effective solution since it depends on the WordNet database. One of the main drawbacks of the WordNet database is the unavailability of many words, particularly for narrowly-defined domains. Another problem with WordNet-based algorithms is that, because WordNet is so broad in scope, an extra step is neededfor semantic disambiguation thatcan further decrease accuracy.In contrast, the text mining approach can be applied to any domain and can use information from the Internet to mine new relevant vocabularies.

Compared with the lexical expansion approach, in terms of correctly assigning new words to specific ontology concepts, both context-based and WordNet-based text mining algorithms have lower precision (19.37% and 8.9% vs. 71.9%). This difference is to be expected because, with the lexical expansion approach, we already know the concepts to whichthe new words are to be attached.With that process, we start with a concept and try to find relevant domain-specific words.In contrast, the text mining approach starts with domain-relevant words and then tries to find matching concepts.

	Lexical Expansion	WordNet-based Algorithms	Context-based Algorithms
# Total words	321	191	191
# Correct words added	231	17	37
Precision	71.9%	8.9%	19.3%

Table 6. Comparison of different approaches on number of words added and accuracy.

Table 6 summarizes the results for the various approaches.It is clear that, in spite of the difficulty in finding domain-specific words in WordNet, the lexical expansion approach achieves both the best ontology enrichment, both in terms of the number of words added to the ontology and the quality of those additions.

6. Conclusions

The goal of this research study is to implement and validate an ontology learning framework in order to enrich the vocabulary of the domain ontology from Web documents related to the domain and from the WordNet semantic lexicon. We have presented two approaches, i.e., lexical expansion using WordNet and text mining, to perform these tasks. In the first approach, we built a reference hypernym tree by manually disambiguating top two levels of concepts from the ontology and compared three similarity computation methods to allow us to disambiguate the other concept-words in the ontology using WordNet. For the correct sense, we then enriched the corresponding concept with its synonyms and hypernyms. In our second approach, we implemented a focused crawler that retrieved documents in the domain of amphibian morphology that incorporates an SVM-based filter. The most relevant documents are submitted for information extraction using text mining.

Our approaches were empirically tested based on the seed amphibian ontology with WordNet lexical synsets and with retrieved Web document. While both approaches performed well, the text mining approach had higher precision for extracting relevant vocabulary to enrich the ontology. However, that approach had greater difficulty attaching the new vocabulary to the correct concept. Considering the ultimate utility of these two approaches, the text mining approach depends on the number and quality of the documents collected by the topic specific crawler. In addition, although it extracts good words, these words are not matched with particular concepts within the ontology. A further pairing process, most likely involving WordNet, is needed to complete the ontology enrichment process. In contrast, the lexical expansion approach using WordNet is only dependent on the concept-words in the ontology itself. It also extracts words from WordNet on a concept-by-concept basis, so no extra process is required to match new words with concepts. However, it does suffer from inaccuracies when the incorrect senses of words are used for expansion. It also requires a small amount of manual effort in order to disambiguate a few concept-words in order to construct the reference hypernym tree.

In future, we hope to combine these two approaches to exploit the strengths of each. For example we can use WordNet pair the text mining with concepts and use the documents to identify and disambiguate the multiple senses for the concept-words found in WordNet. In order to mine more vocabularies for the ontology, we will deal with the concept-words that currently do not appear in WordNet due to the very narrowness of the domain. Our other main task is to validate our approach on ontologies from other domains, to confirm that it is domain-independent. Finally, we need to incorporate the results of this work into a complete system to automatically enrich our ontology.

Acknowledgements

This research is partially supported by the NSF grant DBI-0445752: Semi-Automated Construction of an Ontology for Amphibian Morphology.

Author details

Hiep Luong*, Susan Gauch and Qiang Wang

*Address all correspondence to: hluong@uark.edu

Department of Computer Science and Computer Engineering, University of Arkansas, U.S.A.

References

[1] Agirre, E., Ausa, O., Havy, E., & Martinez, D. (2000). Enriching Very Large Ontologies Using the WWW. ECAI 1st Ontology Learning Workshop Berlin, August

[2] Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The Semantic Web. *Scientific American*, 35-43.

[3] Berry, M. W. (2003). Survey of Text Mining. Springer-Verlag New York Inc., Secaucus, NJ,

[4] Buitelaar, P., Cimiano, P., & Magnini, B. (2005). Ontology Learning from Text: Methods, Evaluation and Applications. IOS Press (Frontiers in AI and applications, , 123

[5] Chang, C. C., & Lin, C. J. (2001). LIBSVM : a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[6] Dumais, S. T., Platt, J., Heckerman, D., & Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. Proceedings of CIKM-98, 148-155.

http://dx.doi.org/10.5772/51141

[7] Ester, M., Gross, M., & Kriegel, H. P. (2001). Focused Web Crawling: A Generic Framework for Specifying the User Interest and for Adaptive Crawling Strategies. Paper presented at 27th Int. Conf. on Very Large Databases, Roma, Italy.

[8] Gangemi, A., Navigli, R., & Velardi, P. (2003). The OntoWordNet Project: extension and axiomatization of conceptual relations in WordNet. In Proceedings of ODBASE03 Conference. Springer

[9] Gauch, S., Madrid, J. M., Induri, S., Ravindran, D., & Chadlavada, S. (2010). KeyConcept: A Conceptual Search Engine. *Center, Technical Report: ITTC-FY2004-TR8646--37*, University of Kansas.

[10] Gómez-Pérez, A., & Manzano-Macho, D. (2003). A survey of ontology learning methods and techniques. *Deliverable 1.5, IST Project IST-20005-29243- OntoWeb*.

[11] Gruber, T. (1994). Towards principles for the design of ontologies used for knowledge sharing. *Int. J. of Human and Computer Studies* [43], 907-928.

[12] Hotho, A., Nürnberger, A., & Paaß, G. (2005). A Brief Survey of Text Mining. *LDVForum*, 20(1), 19-62.

[13] Jiang, J., & Conrath, D. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings of International Conference Research on Computational Linguistics Taiwan,

[14] Joachims, T. (1998). Text categorization with support vector machines: learning with many relevant features. Paper presented at Proceedings of the 10th ECML-,1998. 137-142.

[15] Leacock, C., & Chodorow, M. (1998). Combining Local Context and WordNet Similarity for Word Sense Identification. MIT Press, Cambridge, 265-283.

[16] Lin, D. (1998). An information-theoretic definition of similarity. Paper presented at Proceedings of the Fifteenth International Conference on Machine Learning,. 296-304.

[17] Luong, H., Gauch, S., & Wang, Q. (2009, Feb. 1-7). Ontology-based Focused Crawling. Paper presented at International Conference on Information, Process, and Knowledge Management (eKNOW 2009),, Cancun, Mexico,. 123-128.

[18] Luong, H., Gauch, S., & Speretta, M. (2009, August 2-4,). Enriching Concept Descriptions in an Amphibian Ontology with Vocabulary Extracted from WordNet. Paper presented at The 22nd IEEE Symposium on Computer-Based Medical Systems (CBMS 2009), New Mexico, USA. 1-6.

[19] Luong, H., Gauch, S., Wang, Q., & Maglia, A. (2009). An Ontology Learning Framework Using Focused Crawler and Text Mining. *International Journal on Advances in Life Sciences*, 1(23), 99-109.

[20] Maedche, A., & Staab, S. (2001, March). Ontology Learning for the Semantic Web. *IEEE Intelligent Systems, Special Issue on the Semantic Web*, 16(2), 72-79.

[21] Maedche, A., Neumann, G., & Staab, S. (2003). Bootstrapping an Ontology-Based Information Extraction System. *Studies in Fuzziness and Soft Computing, Intelligent exploration of the web*, Springer, 345-359.

[22] Maglia, A. M., Leopold, J. L., Pugener, L. A., & Gauch, S. (2007). An Anatomical Ontology For Amphibians. *Pacific Symposium on Biocomputing* [12], 367-378.

[23] Miller, G. A. (1995). WordNet: a lexical database for English. *Comm. ACM*, 38(11), 39-41.

[24] Omelayenko, B. (2001). Learning of ontologies for the Web: the analysis of existent approaches. Paper presented at Proceedings of the international workshop on Web dynamics, London.

[25] Pedersen, T., Patwardhan, S., & Michelizzi, J. (2004). WordNet::Similarity- Measuring the Relatedness of Concepts. Paper presented at In Proc. of 19th National Conference on Artificial Intelligence, USA. The MIT Press, 1024-1025.

[26] Pirro, G., & Seco, N. (2008). Design, implementation and evaluation of a new semantic similarity metric combining features and intrinsic information content. *OTM Mexico, LCNS 5332*, 1271-1288.

[27] Reiter, N., & Buitelaar, P. (2008). Lexical Enrichment of a Human Anatomy Ontology using WordNet. In: Proc. of GWC08 (Global WordNet Conference), Hungary , 375-387.

[28] Resnik, P. (1995). Using Information Content to evaluate semantic similarity in a taxonomy. In IJCAI-95, Montreal, Canada , 448-453.

[29] Shamsfard, M., & Barforoush, A. (2003). The State of the Art in Ontology Learning. *The Knowledge Engineering Review*, Cambridge Univ. Press, 18(4), 293-316.

[30] Spasic, I., Ananiadou, S., Mc Naught, J., & Kumar, A. (2005). Text mining and ontologies in biomedicine: making sense of raw text. *Brief Bioinform*, 6, 239-251.

[31] Speretta, M., & Gauch, S. (2008). Using Text Mining to Enrich the Vocabulary of Domain Ontologies. Paper presented at ACM International Conference on Web Intelligence, Sydney. 549-552.

[32] Stevenson, M. (2002). Combining disambiguation techniques to enrich anontology. In Proceedings of the 15th ECAI workshop on Machine Learning and Natural Language Processing for Ontology Engineering

[33] Tan, A. H. (1999). Text mining: The state of the art and the challenges. In Proceedings of the Pacific Asia Conf on Knowledge Discovery and Data Mining PAKDD'99 workshop on Knowledge Discovery from Advanced Databases , 65-70.

[34] Vapnik, V. (1995). The Nature of Statistical Learning Theory. Springer-Verlag.

[35] Varelas, G., Voutsakis, E., Raftopoulou, P., Petrakis, E. G., & Milios, E. E. (2005). Semantic similarity methods in WordNet and their application to information retrieval

on the Web. In Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management (WIDM'05). ACM Press , 10-16.

[36] Velardi, P., Fabriani, P., & Missikoff, M. (2001). Using text processing techniques to automatically enrich a domain ontology. In Proceedings of the international conference on Formal Ontology in Information Systems- Volume 2001 (FOIS'01), ACM New York, NY, USA DOI=10.1145/505168.505194 http://doi.acm.org/10.1145/505168.505194 , 2001, 270-284.

[37] Warin, M., Oxhammer, H., & Volk, M. (2005). Enriching an ontology with wordnet based on similarity measures. In MEANING-2005 Workshop

[38] Wu, Z., & Palmer, M. (1994). Verb semantics and lexical selection. In 32nd Annual Meeting of the Association for Computational Linguistics , 133-138.

[39] Yang, D., & Powers, W. (2005). Measuring semantic similarity in the taxonomy of WordNet. Paper presented at Proc of the 28th Australasian Computer Science Conference. 315-332.

[40] Yang, Y., Zhang, J., & Kisiel, B. (2003, July-August). A scalability analysis of classifiers in text categorization. Paper presented at Proceedings of 26th ACM SIGIR Conference. 96-103.

[41] Yang, Y. (1994). Expert Network: Effective and Efficient Learning from Human Decisions in Text Categorization and Retrieval. *Proceedings of 17th ACM SIGIR*, 13-22.

[42] Yang, Y., & Pederson, J. O. (1997). A comparative study on feature selection in text categorization. *ICML*.

INTECH