Machine Learning Methods of Mapping Semantic Web Ontologies

Caden Howell
chowell4@students.depaul.edu
November 22, 2008

Abstract

This paper is an overview of the application of machine learning to ontology mapping at a high level. This paper introduces ontologies and ontology research for the Semantic Web. It compares several similarity measures and algorithms used to map or merge two ontologies with machine learning algorithms.

1 Introduction

In this paper, I will examine recent research in one aspect of the Semantic Web. Specifically, we will review examples of how machine learning has been used to combine Semantic Web ontologies. This paper is primarily a review of algorithms and methods, and will only focus qualitatively on research results.

2 Background

2.2 The Semantic Web

The Semantic Web is the next generation of the World Wide Web as conceived by Tim Berners-Lee in an article in Scientific American in May of 2001. [1] The World Wide Web is a large, interconnected body of loosely-formatted documents that are comprehensible to people, but not readable by machines. The goal of the Semantic Web is to allow machines to understand this data. This goal is interpreted differently by various researchers. However, a generally agreed-upon principle of the Semantic Web is the addition of machine-readable metadata to existing and new web data.

Metadata is literally data about data. In other words, metadata describes things, and metadata is itself data. For example, the metadata for a book would include the title, author, ISBN, and Library of Congress Subject Heading. It could also include the summary written on the dust jacket, an image of the cover, and a book review. Metadata is used to index and find items and to gain context and understanding.

With respect to the Semantic Web, metadata is supposed to help machines make the transition from processing data syntactically to processing the data according to its meaning. Berners-Lee visualizes that Semantic Web applications will exist that can find data on the web with related semantic meaning and synthesize that data or draw conclusions from it. For example, a person will be able to give a computer an instruction such as "find someone to walk my dog while I'm on vacation." Non-semantically, a computer might be able to do a search for dog walkers on a search engine, and return the results for the user to sort through. This search would be done on word-matching without the machine understanding what a dog is or the context of dog walking in the life of the computer user. With the Semantic Web, the computer would be able to access ontologies so that it could understand (or appear to understand) the relationship between dogs, walkers, times, and locations, and synthesize that information from the appropriate services to find and negotiate a dog walking service within 10 miles of the user's home with availability during her vacation.

Semantic data, which is metadata that computers can use to interpret the *meaning* of an item, has several emerging uses in the Semantic Web:

- Information can be found and organized based on meaning rather than text
- Semantic data can improve the way data is presented with meaningful clustering
- Semantic data can improve the discovery of semantically identified, relevant services
- Semantic data can integrate data from heterogeneous sources. [2]

2.3 Research areas in the Semantic Web

Some major areas of research and development in the Semantic Web include standards for semantic data, "triple store" databases of semantic data, ontology development, and the creation of discoverable, semantic web services. Prominent Semantic Web Standards include RDF, which is used to describe relationships between entities, and OWL, the Ontology Web Language.[3] Triple Store databases store semantic relationship data, and are named for the triple subject, predicate, object structure defined by RDF. [4] Semantic Web services provide semantically tagged data. They are valuable in exposing existing stores of data in a way that can be used by semantic applications. Ideally, they are also discoverable, so that an application that is looking for a specific concept, like "plane tickets" can find it without previous knowledge of the existence of such a service. An application would be able to find that service based on what the service is, rather than a syntactic match.

Whether or not humans will ever be removed from the cycle of Semantic Data creation and updates is an open issue. The consensus among the literature I reviewed was that human intelligence will always be necessary for the creation and processing of semantic data. [5] I hope that this is not true, because it presents a severe bottleneck in the development of the Semantic Web. [6]

2.4 Ontologies

In philosophy, an ontology is a theory about the nature of existence. [4] The generally agreed upon definition with respect to AI and the Semantic Web is "an explicit and formal specification of a conceptualization of a domain of interest." [2] An ontology is made up of concepts, which are also sometimes called "classes" or "entities", relations, which are relationships between the concepts, attributes of the concepts, and axioms. Concepts are classes of things. A concept might be a book or a document type. A specific document that falls into a concept is usually referred to as an "instance." Object oriented programmers will see a parallel between ontology concepts and object classes and ontology instances and object instances. The difference is that an ontological concept can itself be an instance, for example, in a meta-ontology which organizes concepts.

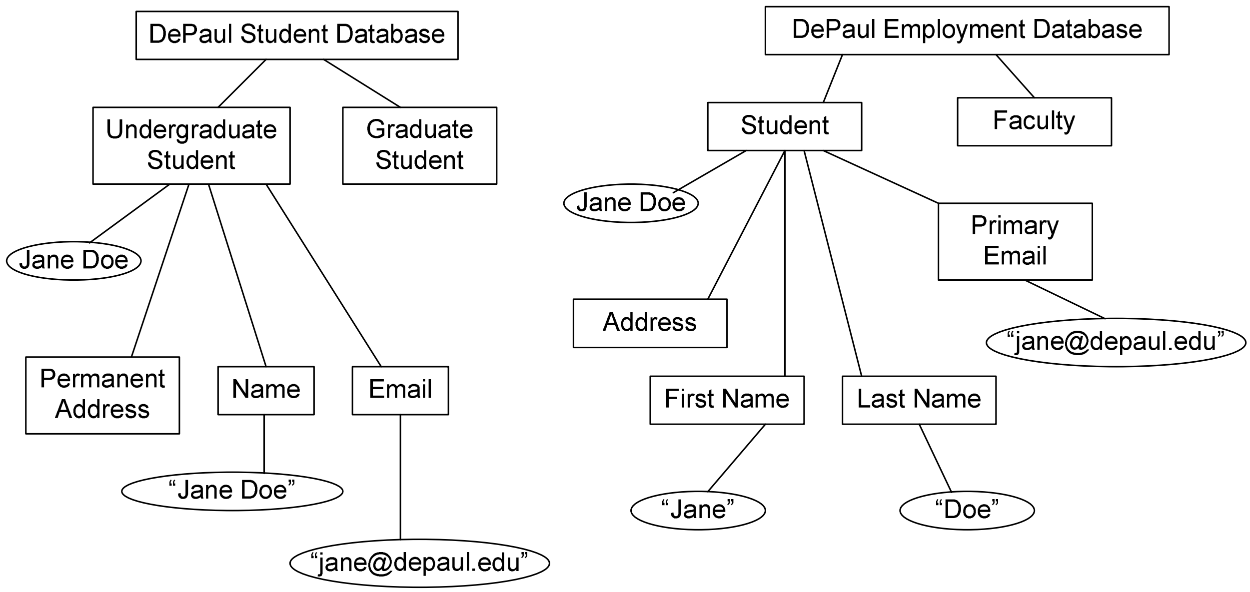

Figure 1. Two Simple Taxonomies [5]

Ontologies can be thought of as extended taxonomies. A taxonomy is a hierarchical structure, either a graph or a tree, used to organize and index concepts. It acts like a directory or catalog. Some common taxonomies include the Library of Congress Subject Headings, the Dewey Decimal System, the DMOZ open directory project [7], and the biological classification of species. A taxonomy can be used to represent the relationships in an ontology. Figure 1 illustrates two different taxonomies of people at DePaul. Each rectangle represents a concept, and each oval represents an instance.

Taxonomies and ontologies are not synonymous. Taxonomies lack the axioms, or rules for reasoning, and complex relationships that are present in ontologies. However, at this stage, most research into ontologies for the Semantic Web focuses on taxonomies.

It is important to note that ontologies are not monolithic, static structures. The goal of the semantic web is not to build a single ontology that represents all of existence. An ontology is a small, useful model which abstracts a part of reality in order to solve a problem. It is a microcosm. Two different ontologies can represent two different, conflicting models and be useful to solve different problems.

Additionally, ontologies are culturally dependent, and may contain cultural or personal biases. What one researcher considers important may become the top node of the hierarchy. A United States researcher may make the top nodes of a geographical hierarchy "United States" and "Other Countries", especially if the data being organized is heavy in instances from the United States. Such a choice would not occur to a researcher in one of the "other countries."

One of the challenges of the Semantic Web is to reason with conflicting data and conflicting ontologies. This issue may be addressed with specialized descriptive logics, or with a statistical "winner" in a probabilistic reasoning ontology. It is still largely unexplored.

2.5 Ontology Research

There are several facets of research with respect to ontologies for the Semantic web. The first is building ontologies. Some ontological structures, like those mentioned, already exist and must be adapted to the Semantic Web. Other research attempts to extract or learn an ontology from a body of data. Another facet is reasoning with ontologies. How can we draw conclusions from the relationships in ontologies? Additionally, there is ontology evolution, or how to update ontologies to reflect the changing world that they model. [8]

In this paper, we will focus on another facet, mapping ontologies, which is necessary part of reasoning with multiple ontologies, and which is addressed in the next section. [9]

3 Ontology Mapping

In order to reason with multiple ontologies, those ontologies must somehow be related to each other. There are several strategies to achieve this. First, two or more ontologies can be merged into one master ontology. This approach seems to be less favored, probably because it requires enough space to store n ontologies, and because it does not reflect the distributed nature of the web.

Another strategy is to define a transformation function from one ontology to another. This seems like an ideal solution, especially if the function is lightweight and easily adaptable. However, in practice mapping ontology classes can be complicated, context-dependent, and domain specific. An ontology mapping cannot just map classes with synonymous names. Sometimes, a class in one taxonomy maps to multiple classes in another taxonomy. [10] In Figure 1, for example, First Name and Last Name in the DePaul Employee Directory taxonomy both map to Name in the DePaul Student Directory taxonomy.

The strategy that we focus on in this paper is defining a mapping between concepts in two ontologies by finding pairs of related concepts. This strategy lends itself to machine learning. Ontology concept mappings can be inferred from overlapping instances in each ontology. Most of the research reviewed for this paper used variations of this strategy.

3.1 Domain Expert Ontologies vs. Machine Learned Ontologies

We have a great body of pre-Semantic Web ontologies that have been constructed by hand by domain experts, including the Library of Congress Subject Headings, biological classification, and many specialized ontologies that are used by domain experts. These ontologies tend to be hierarchical, domain-specific, and readable by humans. In addition, they are extensible, and are often coupled with complex problem-solving methods. [6] They can represent complex relationships. The relatively straightforward biological classification of Kingdom, Phylum, Class, Order, Family, Genus, Species is related to thousands of classification rules (Wings or gills? Fur or feathers?) and riddled with exceptions and reclassifications. For example, the classification tree is no longer divided at the root into Plants and Animals, but was redivided as recently as 1990 into Bacteria, Archaea, and Eukarya (which contains both animals and plants!) [11] This classification system remains relevant because it is continually updated.

In contrast, ontologies that are machine-learned tend to have a flatter structure, a fixed, non-extensible solution space, and classify data into clusters. They tend to use primitive problem solving methods, such as hill-climbing [6] and they are not always readable by humans. Researchers continue to experiment with the tradeoffs between human-generated ontologies and machine-learned ontologies. In some respects, such as in emulating complicated axioms, the human-generated ontologies seem more desirable, and researchers seek to duplicate these features in machine-generated ontologies. However, in other cases it seems more desirable to sacrifice fidelity to human reasoning for a simpler, more understandable model.

The ontology mapping research reviewed for this paper works at the taxonomy level, and disregards axioms. The simplest and most common taxonomy mappings are done on a one-to-one node level. These mappings disregard direct relationships between nodes and combinations of nodes, such as the mapping between "Name" and "First Name" and "Last Name" in Figure 1.

3.2 Ontology Mapping Process

Ehrig and Staab, authors of a process called Quick Ontology Mapping, break down the general machine learning-based ontology mapping process into six steps.

1. Feature engineering. This step involves the extraction of representative features from the ontology, similar to the numeric and nominal features we saw in data sets we analyzed in class.
2. Selection of next search steps. Ehrig and Staab see ontology mapping as a search through the space of possible candidate mappings. Alternatively, it could be seen as the classification of instances from one ontology into the concepts in another ontology.
3. Similarity computation. Using selected similarity measures, compute the similarity values of candidate mappings.
4. Similarity aggregation. Aggregate the similarity metrics, for example, by using a linear weighted function.
5. Interpretation. Choose the mapping using the similarity values. Some methods incorporate additional processing of the similarity measures at this step.
6. Iteration. Some methods continue to iterate through steps 1-5 until no new mappings are proposed. [12]

3.3 Similarity Measures

As the mapping problem is essentially a search for matches, most ontology mapping algorithms incorporate several similarity measures. Similarity measures can be applied at a word-to-word level, a phrase-to-phrase level, and at a higher structural level in the ontology tree. Word and phrase level similarity measures are applied both at the instance and concept level.

3.3.1 Word level similarity measures

The two most popular word similarity measures are TF/IDF and string edit distance. TF/IDF, or term frequency, inverse document frequency considers a document as a "bag of words " or "bag of tokens" and is calculated as

$$TF/IDF = TF_{i,j} \times \log(N / (DF_j)) \quad [13]$$

where $TF_{i,j}$ is the frequency of token t_j in the i-th document and DF_j is the frequency of the document that contains the token t_j. [13] TF/IDF is analogous to a signal/noise ratio for a term in a document.

The string edit distance, or Levenshtein distance, is another common word similarity measure. The string edit distance is the "minimum cost of transforming one string into another through insertions, deletions, or substitutions." [13]

Other less popular word comparison measures include prefix comparison and suffix comparison. Prefix comparison is useful for finding equivalent terms and their abbreviations, like Illinois and Ill. Suffix comparison is useful for phone numbers which are often prepended with differently formatted area and international codes. [14]

The n-gram word comparison measure defines a substring length, and compares all substrings of that length between two strings. For example, a 3-gram word comparison of "head" and "bird" compares "hea" and "ead" to "bir" and "ird". [14]

A "synset" comparison is derived from the existing Wordnet ontology. Wordnet is a large ontology project resembling an English dictionary database with the enriched relationships of an ontology. [15] The synset is calculated as the path length between two words in the Wordnet ontology. [14]

3.3.2 Class or word list level similarity measures

Some word-level similarity measures are adaptable to class-level comparisons. TF/IDF can be used to find a list of tokens within a document as opposed to a single token.

Instead of string edit distance, the similarity of two word lists can be calculated as word edit distance, where each operation becomes the insertion, deletion, or substitution of a word rather than part of a word. [14]

3.3.3 Structural similarity measures

Two simple similarity measures for a concept that are derived from the structure of the ontology are hierarchy similarity and structural similarity. Hierarchy similarity compares all of the nodes from the root to the concept for each concept being compared in their respective taxonomy trees. Structural similarity compares the labels of the parent concepts. [14]

Another similarity measure made at the structural level is concept difference. Concept distance CD(i,j) between instances i and j is defined as

$$CD(i,j) = \begin{cases} 0 & A \equiv B \\ PT(A,B) & A \sqsubseteq B \text{ or } B \sqsubseteq A \\ PT(A,B) + P & A \not\sqsubseteq B \text{ or } B \not\sqsubseteq A \\ \infty & A \sqcap B = \bot \end{cases}$$

where PT(A,B) is path length between concepts A and B according to the computed hierarchical tree. [13]

3.4 Algorithms

Similarity measures can be used to give a numerical value to a potential ontological match, but an algorithm is used to combine and choose the best similarity measure. Most papers that I reviewed used support vector machines as their algorithm and varied in their process and similarity. A few papers that I came across used a Naïve Bayes approach [5, 16], but most other papers found support vector machines to be superior to Naïve Bayes. This is probably because the similarity measures used, addressed in the previous section, are not independent of each other. Although Naïve Bayes has been successful in similar applications, such as spam filtering, it seems intuitive that words in a document probably occur in meaningful, interdependent combinations.

As we discussed in class, support vector machines attempt to find a hyperplane that divides the space of test instances, with a maximum margin between the hyperplane and the instances of the two separate classes. [17]

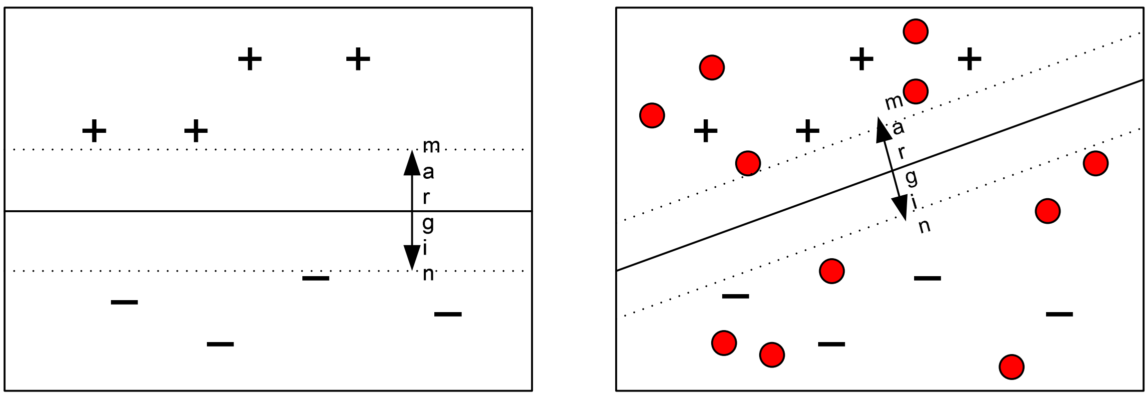

Figure 2. Left: Example of SVM dividing positive and negative classes with a hyperplane with a maximum margin. Right: Example of TSVM dividing positive and negative classes with a hyperplane with a maximum margin to classified and unclassified instances. [18]

Zhang's implementation had an interesting addition to SVM. Instead of dividing only the classified training instances with a maximum margin, the hyperplane also divides the unclassified test instances with a maximum margin, as illustrated in Figure 2. This implementation had improved performance over regular SVM implementations. [18]

3.5 Measuring result quality

Measures for ontology mapping quality are generally borrowed from information retrieval. The most frequently used quality measures were precision, recall, and F-measure, where

$$\textit{Precision} = \#\ \textit{relevant and retrieved} \,/\, \textit{total number retrieved} \quad [4]$$

$$\textit{Recall} = \#\ \textit{relevant and retrieved} \,/\, \textit{total number relevant} \quad [4]$$

$$\textit{F-Measure} = (2 \times \textit{precision} \ x \ \textit{recall}) \,/\, (\textit{precision} + \textit{recall}) \quad [14]$$

4 Individual implementations

4.1 GLUE

The most influential proposal for machine-aided ontology matching of those I reviewed was GLUE by Doan et al. [5] GLUE calculates the joint probability distribution of ontology concepts A and B from two different ontologies as P(A, B), P($\overline{A}$, B), P(A, $\overline{B}$), and P($\overline{A}$, $\overline{B}$). P(A,B) is estimated as

$$P(A,B) = [N(U_1^{A,B}) + N(U_2^{A,B})\ ] / [N(U_1) + N(U_2)\ ]$$

where $N(U_1^{A,B})$ is the number of instances from ontology 1 that match both concept A and B, $N(U_2^{A,B})$ is the number of instances from ontology 2 that match both concept A and B, $N(U_1)$ is the total number of input instances from ontology 1, and $N(U_2)$ is the total number of input instances from ontology 2. The other 3 join probability distributions are calculated similarly.

The joint probability distributions are then used to calculate the Jaccard similarity, which is a measure of overlap of two sets, where the result of the calculation is 0 for completely disjoint sets, and 1 for identical sets. The Jaccard similarity is calculated as

$$Jaccard\text{-}sim(A,B) = P(A \cap B) / (A \cup B)\ = P(A,B) / P(A, B) + P(\overline{A}, B) + P(A, \overline{B}))\ [5]$$

where A and B are sets of the instances that match some concepts A and B from two different ontologies.

To determine whether two an instance from ontology 1 matches a concept in ontology 2, the GLUE method combines two similarity measures. One is naïve Bayes, which treats the instance as a bag of tokens. The other measures the similarity of the fully qualified names. In other words, the second measure takes together the name of the concept with the names of all of its ancestor concepts in the taxonomical tree. These two measures are combined as a linearly weighted expression.

One unique aspect of the GLUE approach is that a relaxation labeling step takes place after it computes the concept similarities. GLUE performs an iterative local optimization based on the heuristic that nodes are more likely to match if the neighborhoods of nodes around them match.

GLUE was tested on a mapping of between the Cornell and Washington course catalogs, where accuracy ranged from 66% to 97%. On a matching between directories of Yahoo! and Standard company profiles, accuracy was approximately 70-80%.

In addition to GLUE, Doan et al. introduced CGLUE. CGLUE addresses the issue that ontologies do not always have a one-to-one concept matching. For example, as mentioned previously, Figure 1 shows how First Name and Last Name in one ontology might map to Name in another ontology.

CGLUE uses the beam searching technique, which keeps track of the k most promising candidate matches, where k is a predefined constant, as it makes comparisons. To illustrate, suppose that we use the beam searching technique to map concept A in ontology 1 to the best concept or set of concepts in ontology 2. Initially, the set of candidate matches is the entire set of concepts in ontology 2. These are scored for similarity, and all but the k most similar are thrown out. Then, each of the k remaining concepts will be unioned with each concept in ontology 2, creating *k × the number of ontology 2 concepts* potential matches. These will be scored for similarity and all but k will be thrown out. This process repeats until the search finds a similarity score that exceeds a preset stopping criterion.

4.2 Wang, Lu, and Zhang [19]

The process proposed by Wang et al combines a number of similarity measures: string edit distance, TF/IDF, concept distance, and context similarity. Context similarity is a measurement based on relationships between classes, which seems to have a similar intention to the relaxation labeling technique in GLUE.

Wang et al's solution builds feature vectors composed of these similarity measures and uses a support vector machine to classify matched instance pairs between the two ontologies. Some matched pairs have correct ontology labels, and some are paired with incorrect ontology labels, creating classes of correct and incorrect ontology matches. The experiments performed in this paper showed that the combined similarity measure performed as much as 50% better than string edit distance alone or as much as 7% better than TF/IDF similarity alone.

4.3 Zhang and Lee [18]

Unlike most other papers I read, which created mappings between ontologies, Zhang and Lee proposed a method for merging a source taxonomy into a master taxonomy. This paper was also unique in that it introduced the TSVM technique described previously in the Algorithms section.

Another unique feature of this implementation was that it introduced the idea of "cluster shrinkage." Clusters of instances which belonged to the same source taxonomy concept in the feature space were shrunk to a centroid. That centroid was then categorized using TSVM into one of the concepts in the master taxonomy.

Zhang and Lee compared their TSVM implementation to SVM, Naïve Bayes, and their own version of enhanced Naïve Bayes. Using Google and Yahoo link directories as taxonomies, they found that TSVM performed better than all three.

5 Conclusion

When I first heard of the Semantic Web eight years ago, I thought that it sounded like something that I could never see in my lifetime. Although we are still far from seeing computers be able to convincingly emulate semantic understanding, I am astounded at how much research has already been produced. I had a difficult time reducing my topic from "Machine Learning Applications to the Semantic Web" to "Machine Learning Applications to Semantic Web Ontologies" to "Machine Learning Applications to Semantic Web Ontology Mapping." Even within this relatively narrow scope, there was much research available.

There are many straightforward similarity measures being used to integrate ontologies, but I am not sure how well they approximate similarity of meaning. They seem to be all derivative of syntactic features. Additionally, most of the papers that I looked at differed minimally from the steps described by Ehrig and Staab, using different combinations of the same similarity measures.

My greatest satisfaction in researching this paper was finding that there is still much to be done regarding Semantic Web research. I also find it easy to imagine that some things that seem impossible now, like Natural Language Processing and the realization of the Semantic Web vision, may seem possible as we make incremental developments in research, hardware and software. I would be very interested in understanding how ontologies are used to draw logical conclusions. I am especially intrigued by the problem of how to reason in the face of ontologies with conflicting structures or conflicting logical axioms.

References

[1] Berners-Lee, T., J. Hendler, O. Lassila, and Et al. "The Semantic Web." Scientific American 284 (2001): 28-37.
[2] Davies, John, ed. Semantic Web Technologies : Trends and Research in Ontology-Based Systems. New York: John Wiley & Sons, Incorporated, 2006.
[3] http://www.w3.org/2001/sw/
[4] Morville, Peter. Ambient Findability. Danbury: O'Reilly Media, Incorporated, 2005. 50.
[5] Doan, AnHai, Jayant Madhavan, Robin Dhamankar, Pedro Domingos, and Alon Halevy. "Learning to match ontologies on the SemanticWeb." The VLDB Journal 12 (2003): 303-19. The ACM Digital Library. Nov. 2003.
[6] Omelayenko, Borys. "Learning of Ontologies for the Web: The Analysis of Existing Approaches." Birkbeck College, University of London, 8th International Conference on Database Theory (ICDT 2001), Jan. 2001, London, England. Department of Computer Science. Vrije Universiteit Amsterdam. 9 Nov. 2008 <http://www.cs.vu.nl/~borys/papers/webdyn01.pdf>.
[7] http://www.dmoz.org/
[8] Bloehdorn, Stephen, Peter Haase, York Sure, and Johanna Voelkner. "Ontology Evolution." Semantic Web Technologies : Trends and Research in Ontology-Based Systems. Ed. John Davies. New York: John Wiley & Sons, Incorporated, 2006.
[9] Grobelnik, Marko, and Dunja Mladenic. "Knowledge Discovery for Ontology Construction." Semantic Web Technologies : Trends and Research in Ontology-Based Systems. Ed. John Davies. New York: John Wiley & Sons, Incorporated, 2006.
[10] Noy, Natalya F. "Tools for Mapping Ontologies." Handbook on Ontologies. Ed. Steffen Staab and Rudi Studer. New York: Springer, 2004. 365-84.
[11] http://en.wikipedia.org/wiki/Biological_classification
[12] Ehrig, Mark, and Steffen Staab. "QOM - Quick Ontology Mapping." International Semantic Web Conference, 2004. Lecture Notes in Computer Science: The Semantic Web ISWC 2004. Vol. 3298/2004. Berlin, Heidelberg: Springer, 2004. 683-97.
[13] Wang, Chao, Jie Lu, and Guangquan Zhang. "Integration of Ontology Data through Learning Instance Matching." Dec. 2006. 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI'06). Los Alamitos, CA: IEEE Computer Society, 2006. 536-39.
[14] Ichise, Ryutaro. "Machine Learning Approach for Ontology Mapping Using Multiple Concept Similarity Measures." Seventh IEEE/ACIS International Conference on Computer and Information Science (icis 2008), May 2008, Paris, France. Washington, DC: IEEE Computer Society, 2008. 340-46.
[15] http://wordnet.princeton.edu/
[16] Pan, Rong, Zhongli Ding, Yang Yu, and Yun Peng. "A Bayesian Network Approach to Ontology Mapping." Proceedings of the Fourth International Semantic Web Conference (ISWC 2005), 17 Oct. 2005, Galway, Ireland. Lecture Notes in Computer Science: The Semantic Web â€“ ISWC 2005. Berlin, Heidelberg: Springer, 2005. 563-77.
[17] Segaran, Toby. Programming Collective Intelligence : Building Smart Web 2.0 Applications. Danbury: O'Reilly Media, Incorporated, 2007.
[18] Zhang, Dell, and Wee Sun Lee. "Web taxonomy integration using support vector machines." ACM, WWW '04: Proceedings of the 13th international conference on World Wide Web, May 2004, New York, United States.
[19] Wang, Chao, Jie Lu, and Guangquan Zhang. "Integration of Ontology Data through Learning Instance Matching." Dec. 2006. 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI'06). Los Alamitos, CA: IEEE Computer Society, 2006. 536-39.