A Metric-based Framework for Automatic Taxonomy Induction

Hui Yang
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
huiyang@cs.cmu.edu

Jamie Callan
Language Technologies Institute
School of Computer Science
Carnegie Mellon University
callan@cs.cmu.edu

Abstract

This paper presents a novel metric-based framework for the task of automatic taxonomy induction. The framework incrementally clusters terms based on *ontology metric*, a score indicating semantic distance; and transforms the task into a multi-criteria optimization based on minimization of taxonomy structures and modeling of term abstractness. It combines the strengths of both lexico-syntactic patterns and clustering through incorporating heterogeneous features. The flexible design of the framework allows a further study on which features are the best for the task under various conditions. The experiments not only show that our system achieves higher F1-measure than other state-of-the-art systems, but also reveal the interaction between features and various types of relations, as well as the interaction between features and term abstractness.

1 Introduction

Automatic taxonomy induction is an important task in the fields of Natural Language Processing, Knowledge Management, and Semantic Web. It has been receiving increasing attention because semantic taxonomies, such as WordNet (Fellbaum, 1998), play an important role in solving knowledge-rich problems, including question answering (Harabagiu et al., 2003) and textual entailment (Geffet and Dagan, 2005). Nevertheless, most existing taxonomies are manually created at great cost. These taxonomies are rarely complete; it is difficult to include new terms in them from emerging or rapidly changing domains. Moreover, manual taxonomy construction is time-consuming, which may make it unfeasible for specialized domains and personalized tasks. Automatic taxonomy induction is a solution to augment existing resources and to produce new taxonomies for such domains and tasks.

Automatic taxonomy induction can be decomposed into two subtasks: term extraction and relation formation. Since term extraction is relatively easy, relation formation becomes the focus of most research on automatic taxonomy induction. In this paper, we also assume that terms in a taxonomy are given and concentrate on the subtask of relation formation.

Existing work on automatic taxonomy induction has been conducted under a variety of names, such as ontology learning, semantic class learning, semantic relation classification, and relation extraction. The approaches fall into two main categories: *pattern-based* and *clustering-based*. *Pattern-based* approaches define lexical-syntactic patterns for relations, and use these patterns to discover instances of relations. *Clustering-based* approaches hierarchically cluster terms based on similarities of their meanings usually represented by a vector of quantifiable features.

Pattern-based approaches are known for their high accuracy in recognizing instances of relations if the patterns are carefully chosen, either manually (Berland and Charniak, 1999; Kozareva et al., 2008) or via automatic bootstrapping (Hearst, 1992; Widdows and Dorow, 2002; Girju et al., 2003). The approaches, however, suffer from sparse coverage of patterns in a given corpus. Recent studies (Etzioni et al., 2005; Kozareva et al., 2008) show that if the size of a corpus, such as the Web, is nearly unlimited, a pattern has a higher chance to explicitly appear in the corpus. However, corpus size is often not that large; hence the problem still exists. Moreover, since patterns usually extract instances in pairs, the approaches suffer from the problem of inconsistent concept chains after connecting pairs of instances to form taxonomy hierarchies.

Clustering-based approaches have a main advantage that they are able to discover relations

Proceedings of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, pages 271–279,
Suntec, Singapore, 2-7 August 2009.

which do not explicitly appear in text. They also avoid the problem of inconsistent chains by addressing the structure of a taxonomy globally from the outset. Nevertheless, it is generally believed that clustering-based approaches cannot generate relations as accurate as pattern-based approaches. Moreover, their performance is largely influenced by the types of features used. The common types of features include *contextual (*Lin, 1998), *co-occurrence* (Yang and Callan, 2008)*,* and *syntactic dependency* (Pantel and Lin, 2002; Pantel and Ravichandran, 2004). So far there is no systematic study on which features are the best for automatic taxonomy induction under various conditions.

This paper presents a metric-based taxonomy induction framework. It combines the strengths of both pattern-based and clustering-based approaches by incorporating lexico-syntactic patterns as one type of features in a clustering framework. The framework integrates contextual, co-occurrence, syntactic dependency, lexical-syntactic patterns, and other features to learn an *ontology metric*, a score indicating semantic distance, for each pair of terms in a taxonomy; it then incrementally clusters terms based on their ontology metric scores. The incremental clustering is transformed into an optimization problem based on two assumptions: *minimum evolution* and *abstractness*. The flexible design of the framework allows a further study of the interaction between features and relations, as well as that between features and term abstractness.

2 Related Work

There has been a substantial amount of research on automatic taxonomy induction. As we mentioned earlier, two main approaches are *pattern-based* and *clustering-based*.

Pattern-based approaches are the main trend for automatic taxonomy induction. Though suffering from the problems of sparse coverage and inconsistent chains, they are still popular due to their simplicity and high accuracy. They have been applied to extract various types of lexical and semantic relations, including *is-a*, *part-of*, *sibling*, *synonym*, *causal*, and many others.

Pattern-based approaches started from and still pay a great deal of attention to the most common *is-a* relations. Hearst (1992) pioneered using a hand crafted list of hyponym patterns as seeds and employing bootstrapping to discover *is-a* relations. Since then, many approaches (Mann, 2002; Etzioni et al., 2005; Snow et al., 2005) have used Hearst-style patterns in their work on *is-a* relations. For instance, Mann (2002) extracted *is-a* relations for proper nouns by Hearst-style patterns. Pantel et al. (2004) extended *is-a* relation acquisition towards terascale, and automatically identified hypernym patterns by minimal edit distance.

Another common relation is *sibling*, which describes the relation of sharing similar meanings and being members of the same class. Terms in *sibling* relations are also known as *class members* or *similar terms*. Inspired by the conjunction and appositive structures, Riloff and Shepherd (1997), Roark and Charniak (1998) used co-occurrence statistics in local context to discover *sibling* relations. The KnowItAll system (Etzioni et al., 2005) extended the work in (Hearst, 1992) and bootstrapped patterns on the Web to discover siblings; it also ranked and selected the patterns by statistical measures. Widdows and Dorow (2002) combined symmetric patterns and graph link analysis to discover *sibling* relations. Davidov and Rappoport (2006) also used symmetric patterns for this task. Recently, Kozareva et al. (2008) combined a double-anchored hyponym pattern with graph structure to extract siblings.

The third common relation is *part-of*. Berland and Charniak (1999) used two meronym patterns to discover *part-of* relations, and also used statistical measures to rank and select the matching instances. Girju et al. (2003) took a similar approach to Hearst (1992) for *part-of* relations.

Other types of relations that have been studied by pattern-based approaches include question-answer relations (such as *birthdates* and *inventor*) (Ravichandran and Hovy, 2002), synonyms and antonyms (Lin et al., 2003), general purpose analogy (Turney et al., 2003), verb relations (including *similarity, strength, antonym, enablement* and *temporal*) (Chklovski and Pantel, 2004), entailment (Szpektor et al., 2004), and more specific relations, such as *purpose, creation* (Cimiano and Wenderoth, 2007), *LivesIn*, and *EmployedBy* (Bunescu and Mooney , 2007).

The most commonly used technique in pattern-based approaches is *bootstrapping* (Hearst, 1992; Etzioni et al., 2005; Girju et al., 2003; Ravichandran and Hovy, 2002; Pantel and Pennacchiotti, 2006). It utilizes a few man-crafted seed patterns to extract instances from corpora, then extracts new patterns using these instances, and continues the cycle to find new instances and new patterns. It is effective and scalable to large datasets; however, uncontrolled bootstrapping

soon generates undesired instances once a noisy pattern brought into the cycle.

To aid bootstrapping, methods of pattern quality control are widely applied. Statistical measures, such as point-wise mutual information (Etzioni et al., 2005; Pantel and Pennacchiotti, 2006) and conditional probability (Cimiano and Wenderoth, 2007), have been shown to be effective to rank and select patterns and instances. Pattern quality control is also investigated by using WordNet (Girju et al., 2006), graph structures built among terms (Widdows and Dorow, 2002; Kozareva et al., 2008), and pattern clusters (Davidov and Rappoport, 2008).

Clustering-based approaches usually represent word contexts as vectors and cluster words based on similarities of the vectors (Brown et al., 1992; Lin, 1998). Besides contextual features, the vectors can also be represented by verb-noun relations (Pereira et al., 1993), syntactic dependency (Pantel and Ravichandran, 2004; Snow et al., 2005), co-occurrence (Yang and Callan, 2008), conjunction and appositive features (Caraballo, 1999). More work is described in (Buitelaar et al., 2005; Cimiano and Volker, 2005). Clustering-based approaches allow discovery of relations which do not explicitly appear in text. Pantel and Pennacchiotti (2006), however, pointed out that clustering-based approaches generally fail to produce coherent cluster for small corpora. In addition, clustering-based approaches had only applied to solve *is-a* and *sibling* relations.

Many clustering-based approaches face the challenge of appropriately labeling non-leaf clusters. The labeling amplifies the difficulty in creation and evaluation of taxonomies. Agglomerative clustering (Brown et al., 1992; Caraballo, 1999; Rosenfeld and Feldman, 2007; Yang and Callan, 2008) iteratively merges the most similar clusters into bigger clusters, which need to be labeled. Divisive clustering, such as CBC (Clustering By Committee) which constructs cluster centroids by averaging the feature vectors of a subset of carefully chosen cluster members (Pantel and Lin, 2002; Pantel and Ravichandran, 2004), also need to label the parents of split clusters. In this paper, we take an incremental clustering approach, in which terms and relations are added into a taxonomy one at a time, and their parents are from the existing taxonomy. The advantage of the incremental approach is that it eliminates the trouble of inventing cluster labels and concentrates on placing terms in the correct positions in a taxonomy hierarchy.

The work by Snow et al. (2006) is the most similar to ours because they also took an incremental approach to construct taxonomies. In their work, a taxonomy grows based on maximization of conditional probability of relations given evidence; while in our work based on optimization of taxonomy structures and modeling of term abstractness. Moreover, our approach employs heterogeneous features from a wide range; while their approach only used syntactic dependency. We compare system performance between (Snow et al., 2006) and our framework in Section 5.

3 The Features

The features used in this work are indicators of semantic relations between terms. Given two input terms c_x, c_y, a feature is defined as a function generating a single numeric score $h(c_x, c_y) \in \mathbb{R}$ or a vector of numeric scores $h(c_x, c_y) \in \mathbb{R}^n$. The features include *contextual*, *co-occurrence*, *syntactic dependency*, *lexical-syntactic patterns*, and *miscellaneous*.

The first set of features captures *contextual* information of terms. According to Distributional Hypothesis (Harris, 1954), words appearing in similar contexts tend to be similar. Therefore, word meanings can be inferred from and represented by contexts. Based on the hypothesis, we develop the following features: (1) *Global Context KL-Divergence*: The global context of each input term is the search results collected through querying search engines against several corpora (Details in Section 5.1). It is built into a unigram language model without smoothing for each term. This feature function measures the Kullback-Leibler divergence (KL divergence) between the language models associated with the two inputs. (2) *Local Context KL-Divergence:* The local context is the collection of all the left two and the right two words surrounding an input term. Similarly, the local context is built into a unigram language model without smoothing for each term; the feature function outputs KL divergence between the models.

The second set of features is *co-occurrence*. In our work, co-occurrence is measured by point-wise mutual information between two terms:

$$pmi(c_x, c_y) = \log \frac{Count(c_x, c_y)}{Count(c_x)Count(c_y)}$$

where *Count(.)* is defined as the number of documents or sentences containing the term(s); or *n* as in "Results 1-10 of about *n* for *term*" appearing on the first page of Google search results for a term or the concatenation of a term pair. Based

Hypernym Patterns	*Sibling Patterns*
NP_x (,)?and/or other NP_y	NP_x and/or NP_y
such NP_y as NP_x	*Part-of Patterns*
NP_y (,)? such as NP_x	NP_x of NP_y
NP_y (,)? including NP_x	NP_y's NP_x
NP_y (,)? especially NP_x	NP_y has/had/have NP_x
NP_y like NP_x	NP_y is made (up)? of NP_x
NP_y called NP_x	NP_y comprises NP_x
NP_x is a/an NP_y	NP_y consists of NP_x
NP_x , a/an NP_y	

Table 1. Lexico-Syntactic Patterns.

on different definitions of *Count(.),* we have (3) *Document PMI*, (4) *Sentence PMI*, and (5) *Google PMI* as the co-occurrence features.

The third set of features employs *syntactic dependency* analysis. We have (6) *Minipar Syntactic Distance* to measure the average length of the shortest syntactic paths (in the first syntactic parse tree returned by Minipar[1]) between two terms in sentences containing them, (7) *Modifier Overlap, (8) Object Overlap, (9) Subject Overlap, and (10) Verb Overlap* to measure the number of overlaps between modifiers, objects, subjects, and verbs, respectively, for the two terms in sentences containing them. We use Assert[2] to label the semantic roles.

The fourth set of features is *lexical-syntactic patterns*. We have (11) *Hypernym Patterns* based on patterns proposed by (Hearst, 1992) and (Snow et al., 2005), (12) *Sibling Patterns* which are basically conjunctions, and (13) *Part-of Patterns* based on patterns proposed by (Girju et al., 2003) and (Cimiano and Wenderoth, 2007). Table 1 lists all patterns. Each feature function returns a vector of scores for two input terms, one score per pattern. A score is 1 if two terms match a pattern in text, 0 otherwise.

The last set of features is *miscellaneous*. We have (14) *Word Length Difference* to measure the length difference between two terms, and (15) *Definition Overlap* to measure the number of word overlaps between the term definitions obtained by querying Google with *"*define*:term"*.

These heterogeneous features vary from simple statistics to complicated syntactic dependency features, basic word length to comprehensive Web-based contextual features. The flexible design of our learning framework allows us to use all of them, and even allows us to use different sets of them under different conditions, for instance, different types of relations and different abstraction levels. We study the interaction between features and relations and that between features and abstractness in Section 5.

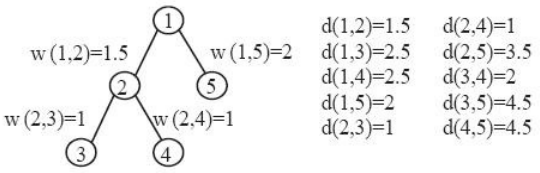

Figure 1. Illustration of Ontology Metric.

4 The Metric-based Framework

This section presents the metric-based framework which incrementally clusters terms to form taxonomies. By minimizing the changes of taxonomy structures and modeling term abstractness at each step, it finds the optimal position for each term in a taxonomy. We first introduce definitions, terminologies and assumptions about taxonomies; then, we formulate automatic taxonomy induction as a multi-criterion optimization and solve it by a greedy algorithm; lastly, we show how to estimate ontology metrics.

4.1 Taxonomies, Ontology Metric, Assumptions, and Information Functions

We define a *taxonomy* T as a data model that represents a set of terms C and a set of relations R between these terms. T can be written as $T(C,R)$. Note that for the subtask of relation formation, we assume that the term set C is given. A *full taxonomy* is a tree containing all the terms in C. A *partial taxonomy* is a tree containing only a subset of terms in C.

In our framework, automatic taxonomy induction is the process to construct a full taxonomy $\hat{T}$ given a set of terms C and an initial partial taxonomy $T^0(S^0, R^0)$, where $S^0 \subseteq C$. Note that T^0 is possibly empty. The process starts from the initial partial taxonomy T^0 and randomly adds terms from C to T^0 one by one, until a full taxonomy is formed, i.e., all terms in C are added.

Ontology Metric

We define an *ontology metric* as a distance measure between two terms (c_x,c_y) in a taxonomy $T(C,R)$. Formally, it is a function $d: C \times C \rightarrow \mathbb{R}_+$, where C is the set of terms in T. An ontology metric d on a taxonomy T with edge weights w for any term pair $(c_x,c_y) \in C$ is the sum of all edge weights along the shortest path between the pair:

$$d_{(T,w)}(c_x, c_y) = \sum_{e_{x,y} \in P(x,y)} w(e_{x,y})$$

[1] http://www.cs.ualberta.ca/lindek/minipar.htm.
[2] http://cemantix.org/assert.

where $P(x, y)$ is the set of edges defining the shortest path from term c_x to c_y . Figure 1 illustrates ontology metrics for a 5-node taxonomy. Section 4.3 presents the details of learning *ontology metrics*.

Information Functions

The amount of information in a taxonomy T is measured and represented by an information function *Info(T)*. An information function is defined as the sum of the ontology metrics among a set of term pairs. The function can be defined over a taxonomy, or on a single level of a taxonomy. For a taxonomy *T(C,R)*, we define its information function as:

$$Info(T) = \sum_{x<y, c_x, c_y \in C} d(c_x, c_y) \quad (1)$$

Similarly, we define the information function for an abstraction level L_i as:

$$Info_i(L_i) = \sum_{x<y, c_x, c_y \in L_i} d(c_x, c_y) \quad (2)$$

where L_i is the subset of terms lying at the i^{th} level of a taxonomy T. For example, in Figure 1, node 1 is at level L_1, node 2 and node 5 level L_2.

Assumptions

Given the above definitions about taxonomies, we make the following assumptions:

Minimum Evolution Assumption. Inspired by the minimum evolution tree selection criterion widely used in phylogeny (Hendy and Penny, 1985), we assume that a good taxonomy not only minimizes the overall semantic distance among the terms but also avoid dramatic changes. Construction of a full taxonomy is proceeded by adding terms one at a time, which yields a series of partial taxonomies. After adding each term, the current taxonomy T^{n+1} from the previous taxonomy T^n is one that introduces the least changes between the information in the two taxonomies:

$$T^{n+1} = \arg\min_{T'} \Delta Info(T^n, T')$$

where the information change function is $\Delta Info(T^a, T^b) = | Info(T^a) - Info(T^b) |$.

Abstractness Assumption. In a taxonomy, concrete concepts usually lay at the bottom of the hierarchy while abstract concepts often occupy the intermediate and top levels. Concrete concepts often represent physical entities, such as "basketball" and "mercury pollution". While abstract concepts, such as "science" and "economy", do not have a physical form thus we must imagine their existence. This obvious difference suggests that there is a need to treat them differently in taxonomy induction. Hence we assume that terms at the same abstraction level have common characteristics and share the same *Info(.)* function. We also assume that terms at different abstraction levels have different characteristics; hence they do not necessarily share the same *Info(.)* function. That is to say, $\forall$ concept $c \in T$, abstraction level $L_i \subset T$, $c \in L_i \Rightarrow c$ uses $Info_i(.)$.

4.2 Problem Formulation

The Minimum Evolution Objective

Based on the minimum evolution assumption, we define the goal of taxonomy induction is to find the optimal full taxonomy $\hat{T}$ such that the information changes are the least since the initial partial taxonomy T^0, i.e., to find:

$$\hat{T} = \arg\min_{T'} \Delta Info(T^0, T') \quad (3)$$

where T' is a full taxonomy, i.e., the set of terms in T' equals C.

To find the optimal solution for Equation (3), $\hat{T}$, we need to find the optimal term set $\hat{C}$ and the optimal relation set $\hat{R}$. Since the optimal term set for a full taxonomy is always C, the only unknown part left is $\hat{R}$. Thus, Equation (3) can be transformed equivalently into:

$$\hat{R} = \arg\min_{R'} \Delta Info(T'(C, R'), T^0(S^0, R^0))$$

Note that in the framework, terms are added incrementally into a taxonomy. Each term insertion yields a new partial taxonomy T. By the minimum evolution assumption, the optimal next partial taxonomy is one gives the least information change. Therefore, the updating function for the set of relations R^{n+1} after a new term z is inserted can be calculated as:

$$\hat{R} = \arg\min_{R'} \Delta Info(T(S^n \cup \{z\}, R'), T(S^n, R^n))$$

By plugging in the definition of the information change function $\Delta Info(.,.)$ in Section 4.1 and Equation (1), the updating function becomes:

$$\hat{R} = \arg\min_{R'} \left| \sum_{c_x, c_y \in S^n \cup \{z\}} d(c_x, c_y) - \sum_{c_x, c_y \in S^n} d(c_x, c_y) \right|$$

The above updating function can be transformed into a minimization problem:

$$\min u$$
$$\text{subject to } u \le \sum_{c_x, c_y \in S^n \cup \{z\}} d(c_x, c_y) - \sum_{c_x, c_y \in S^n} d(c_x, c_y)$$
$$u \le \sum_{c_x, c_y \in S^n} d(c_x, c_y) - \sum_{c_x, c_y \in S^n \cup \{z\}} d(c_x, c_y)$$
$$x < y$$

The minimization follows the minimum evolution assumption; hence we call it the *minimum evolution objective*.

The Abstractness Objective

The *abstractness* assumption suggests that term abstractness should be modeled explicitly by learning separate information functions for terms at different abstraction levels. We approximate an information function by a linear interpolation of some underlying feature functions. Each abstraction level L_i is characterized by its own information function $Info_i(.)$. The least square fit of $Info_i(.)$ is: $\min | Info_i(L_i) - W_i^T H_i |^2$.

By plugging Equation (2) and minimizing over every abstraction level, we have:

$$\min \sum_i (\sum_{c_x, c_y \in L_i} d(c_x, c_y) - \sum_j w_{i,j} h_{i,j}(c_x, c_y))^2$$

where $h_{i,j}(.,.)$ is the j^{th} underlying feature function for term pairs at level L_i, $w_{i,j}$ is the weight for $h_{i,j}(.,.)$. This minimization follows the abstractness assumption; hence we call it the *abstractness objective*.

The Multi-Criterion Optimization Algorithm

We propose that both *minimum evolution* and *abstractness* objectives need to be satisfied. To optimize multiple criteria, the Pareto optimality needs to be satisfied (Boyd and Vandenberghe, 2004). We handle this by introducing $\lambda \in [0,1]$ to control the contribution of each objective. The multi-criterion optimization function is:

$$\begin{aligned} &\min \lambda u + (1-\lambda) v \\ \text{subject to} \quad &u \le \sum_{c_x, c_y \in S^n \cup \{z\}} d(c_x, c_y) - \sum_{c_x, c_y \in S^n} d(c_x, c_y) \\ &u \le \sum_{c_x, c_y \in S^n} d(c_x, c_y) - \sum_{c_x, c_y \in S^n \cup \{z\}} d(c_x, c_y) \\ &v = \sum_i (\sum_{\substack{c_x, c_y \in L_i \\ x < y}} d(c_x, c_y) - \sum_j w_{i,j} h_{i,j}(c_x, c_y))^2 \end{aligned}$$

The above optimization can be solved by a greedy optimization algorithm. At each term insertion step, it produces a new partial taxonomy by adding to the existing partial taxonomy a new term z, and a new set of relations $R(z,.)$. z is attached to every nodes in the existing partial taxonomy; and the algorithm selects the optimal position indicated by $R(z,.)$, which minimizes the multi-criterion objective function. The algorithm is:

foreach $z \in C \setminus S$

$S \rightarrow S \cup \{z\}$;

$R \rightarrow R \cup \{\arg\min_{R_{(z,.)}} (\lambda u + (1-\lambda) v)\}$;

Output $T(S, R)$;

The above algorithm presents a general incremental clustering procedure to construct taxonomies. By minimizing the taxonomy structure changes and modeling term abstractness at each step, it finds the optimal position of each term in the taxonomy hierarchy.

Statistics	*WN/is-a*	*ODP/is-a*	*WN/part-of*
#taxonomies	50	50	50
#terms	1,964	2,210	1,812
Avg #terms	39	44	37
Avg depth	6	6	5

Table 2. Data Statistics.

4.3 Estimating Ontology Metric

Learning a good ontology metric is important for the multi-criterion optimization algorithm. In this work, the estimation and prediction of ontology metric are achieved by ridge regression (Hastie et al., 2001). In the training data, an ontology metric $d(c_x,c_y)$ for a term pair (c_x,c_y) is generated by assuming every edge weight as 1 and summing up all the edge weights along the shortest path from c_x to c_y. We assume that there are some underlying feature functions which measure the semantic distance from term c_x to c_y. A weighted combination of these functions approximates the ontology metric for (c_x,c_y):

$$d(x, y) = \sum_j w_j h_j(c_x, c_y)$$

where w_j is the j^{th} weight for $h_j(c_x, c_y)$, the j^{th} feature function. The feature functions are generated as mentioned in Section 3.

5 Experiments

5.1 Data

The gold standards used in the evaluation are hypernym taxonomies extracted from WordNet and ODP (Open Directory Project), and meronym taxonomies extracted from WordNet. In WordNet taxonomy extraction, we only use the word senses within a particular taxonomy to ensure no ambiguity. In ODP taxonomy extraction, we parse the topic lines, such as "Topic r:id=`Top/Arts/Movies'", in the XML databases to obtain relations, such as *is_a(movies, arts)*. In total, there are 100 hypernym taxonomies, 50 each extracted from WordNet[3] and ODP[4], and 50 meronym taxonomies from WordNet[5]. Table 2

[3] WordNet hypernym taxonomies are from 12 topics: *gathering, professional, people, building, place, milk, meal, water, beverage, alcohol, dish,* and *herb.*

[4] ODP hypernym taxonomies are from 16 topics: *computers, robotics, intranet, mobile computing, database, operating system, linux, tex, software, computer science, data communication, algorithms, data formats, security multimedia,* and *artificial intelligence.*

[5] WordNet meronym taxonomies are from 15 topics: *bed*, car, *building*, *lamp*, *earth*, *television*, *body, drama, theatre, water, airplane, piano, book, computer,* and *watch.*

WordNet/*is-a*			
System	*Precision*	*Recall*	*F1-measure*
HE	**0.85**	0.32	0.46
GI	n/a	n/a	n/a
PR	0.75	0.73	0.74
ME	0.82	**0.79**	**0.82**
ODP/*is-a*			
System	*Precision*	*Recall*	*F1-measure*
HE	0.31	0.29	0.30
GI	n/a	n/a	n/a
PR	0.60	**0.72**	0.65
ME	**0.64**	0.70	**0.67**
WordNet/*part-of*			
System	*Precision*	*Recall*	*F1-measure*
HE	n/a	n/a	n/a
GI	**0.75**	0.25	0.38
PR	0.68	0.52	0.59
ME	0.69	**0.55**	**0.61**

Table 3. System Performance.

summarizes the data statistics.

We also use two Web-based auxiliary datasets to generate features mentioned in Section 3:

- *Wikipedia corpus.* The entire Wikipedia corpus is downloaded and indexed by Indri[6]. The top 100 documents returned by Indri are the global context of a term when querying with the term.
- *Google corpus*. A collection of the top 1000 documents by querying Google using each term, and each term pair. Each top 1000 documents are the global context of a query term.

Both corpora are split into sentences and are used to generate contextual, co-occurrence, syntactic dependency and lexico-syntactic pattern features.

5.2 Methodology

We evaluate the quality of automatic generated taxonomies by comparing them with the gold standards in terms of precision, recall and F1-measure. F1-measure is calculated as *2*P*R/(P+R)*, where *P* is *precision*, the percentage of correctly returned relations out of the total returned relations, *R* is *recall*, the percentage of correctly returned relations out of the total relations in the gold standard.

Leave-one-out cross validation is used to average the system performance across different training and test datasets. For each 50 datasets from WordNet hypernyms, WordNet meronyms or ODP hypernyms, we randomly pick 49 of them to generate training data, and test on the remaining dataset. We repeat the process for 50 times, with different training and test sets at each time, and report the averaged precision, recall and F1-measure across all 50 runs.

We also group the fifteen features in Section 3 into six sets: contextual, co-concurrence, patterns, syntactic dependency, word length difference and definition. Each set is turned on one by one for experiments in Section 5.4 and 5.5.

Feature	*is-a*	*sibling*	*part-of*	***Benefited Relations***
Contextual	0.21	**0.42**	0.12	*sibling*
Co-occur.	**0.48**	**0.41**	**0.28**	*All*
Patterns	**0.46**	**0.41**	**0.30**	*All*
Syntactic	0.22	**0.36**	0.12	*sibling*
Word Leng.	0.16	0.16	0.15	*All but limited*
Definition	0.12	0.18	0.10	*Sibling but limited*
Best Features	Co-occur., patterns	Contextual, co-occur., patterns	Co-occur., patterns	

Table 4. F1-measure for Features vs. Relations: WordNet.

5.3 Performance of Taxonomy Induction

In this section, we compare the following automatic taxonomy induction systems: *HE*, the system by Hearst (1992) with 6 hypernym patterns; *GI*, the system by Girju et al. (2003) with 3 meronym patterns; *PR*, the probabilistic framework by Snow et al. (2006); and *ME*, the metric-based framework proposed in this paper. To have a fair comparison, for *PR*, we estimate the conditional probability of a relation given the evidence $P(R_{ij}|E_{ij})$, as in (Snow et al. 2006), by using the same set of features as in *ME*.

Table 3 shows precision, recall, and F1-measure of each system for WordNet hypernyms (*is-a*), WordNet meronyms (*part-of*) and ODP hypernyms (*is-a*). Bold font indicates the best performance in a column. Note that *HE* is not applicable to *part-of*, so is *GI* to *is-a*.

Table 3 shows that systems using heterogeneous features (*PR* and *ME*) achieve higher F1-measure than systems only using patterns (*HE* and *GI*) with a significant absolute gain of >30%. Generally speaking, pattern-based systems show higher precision and lower recall, while systems using heterogeneous features show lower precision and higher recall. However, when considering both precision and recall, using heterogeneous features is more effective than just using patterns. The proposed system *ME* consistently produces the best F1-measure for all three tasks.

The performance of the systems for ODP/*is-a* is worse than that for WordNet/*is-a*. This may be because there is more noise in ODP than in

[6] http://www.lemurproject.org/indri/.

Feature	L_2	L_3	L_4	L_5	L_6
Contextual	0.29	0.31	**0.35**	**0.36**	**0.36**
Co-occurrence	**0.47**	**0.56**	**0.45**	**0.41**	**0.41**
Patterns	**0.47**	**0.44**	**0.42**	**0.39**	**0.40**
Syntactic	0.31	0.28	**0.36**	**0.38**	**0.39**
Word Length	0.16	0.16	0.16	0.16	0.16
Definition	0.12	0.12	0.12	0.12	0.12

Table 5. F1-measure for Features vs. Abstractness: WordNet/*is-a*.

Feature	L_2	L_3	L_4	L_5	L_6
Contextual	**0.30**	**0.30**	**0.33**	**0.29**	**0.29**
Co-occurrence	**0.34**	**0.36**	**0.34**	**0.31**	**0.31**
Patterns	0.23	0.25	**0.30**	**0.28**	**0.28**
Syntactic	0.18	0.18	0.23	**0.27**	**0.27**
Word Length	0.15	0.15	0.15	0.14	0.14
Definition	0.13	0.13	0.13	0.12	0.12

Table 6. F1-measure for Features vs. Abstractness: ODP/*is-a*.

WordNet. For example, under *artificial intelligence*, ODP has *neural networks*, *natural language* and *academic departments*. Clearly, *academic departments* is not a hyponym of *artificial intelligence*. The noise in ODP interferes with the learning process, thus hurts the performance.

5.4 Features vs. Relations

This section studies the impact of different sets of features on different types of relations. Table 4 shows F1-measure of using each set of features alone on taxonomy induction for WordNet *is-a*, *sibling*, and *part-of* relations. Bold font means a feature set gives a major contribution to the task of automatic taxonomy induction for a particular type of relation.

Table 4 shows that different relations favor different sets of features. Both co-occurrence and lexico-syntactic patterns work well for all three types of relations. It is interesting to see that simple co-occurrence statistics work as good as lexico-syntactic patterns. Contextual features work well for *sibling* relations, but not for *is-a* and *part-of*. Syntactic features also work well for *sibling,* but not for *is-a* and *part-of*. The similar behavior of contextual and syntactic features may be because that four out of five syntactic features (*Modifier, Subject, Object,* and *Verb overlaps*) are just surrounding context for a term.

Comparing the *is-a* and *part-of* columns in Table 4 and the *ME* rows in Table 3, we notice a significant difference in F1-measure. It indicates that combination of heterogeneous features gives more rise to the system performance than a single set of features does.

5.5 Features vs. Abstractness

This section studies the impact of different sets of features on terms at different abstraction levels. In the experiments, F1-measure is evaluated for terms at each level of a taxonomy, not the whole taxonomy. Table 5 and 6 demonstrate F1-measure of using each set of features alone on each abstraction levels. Columns 2-6 are indices of the levels in a taxonomy. The larger the indices are, the lower the levels. Higher levels contain abstract terms, while lower levels contain concrete terms. L_1 is ignored here since it only contains a single term, the root. Bold font indicates good performance in a column.

Both tables show that abstract terms and concrete terms favor different sets of features. In particular, contextual, co-occurrence, pattern, and syntactic features work well for terms at L_4-L_6, i.e., concrete terms; co-occurrence works well for terms at L_2-L_3, i.e., abstract terms. This difference indicates that terms at different abstraction levels have different characteristics; it confirms our *abstractness assumption* in Section 4.1.

We also observe that for abstract terms in WordNet, patterns work better than contextual features; while for abstract terms in ODP, the conclusion is the opposite. This may be because that WordNet has a richer vocabulary and a more rigid definition of hypernyms, and hence *is-a* relations in WordNet are recognized more effectively by using lexico-syntactic patterns; while ODP contains more noise, and hence it favors features requiring less rigidity, such as the contextual features generated from the Web.

6 Conclusions

This paper presents a novel metric-based taxonomy induction framework combining the strengths of lexico-syntactic patterns and clustering. The framework incrementally clusters terms and transforms automatic taxonomy induction into a multi-criteria optimization based on minimization of taxonomy structures and modeling of term abstractness. The experiments show that our framework is effective; it achieves higher F1-measure than three state-of-the-art systems. The paper also studies which features are the best for different types of relations and for terms at different abstraction levels.

Most prior work uses a single rule or feature function for automatic taxonomy induction at all levels of abstraction. Our work is a more general framework which allows a wider range of features and different metric functions at different abstraction levels. This more general framework has the potential to learn more complex taxonomies than previous approaches.

Acknowledgements

This research was supported by NSF grant IIS-0704210. Any opinions, findings, conclusions, or recommendations expressed in this paper are of the authors, and do not necessarily reflect those of the sponsor.

References

M. Berland and E. Charniak. 1999. Finding parts in very large corpora. ACL'99.

S. Boyd and L. Vandenberghe. 2004. Convex optimization. In Cambridge University Press, 2004.

P. Brown, V. D. Pietra, P. deSouza, J. Lai, and R. Mercer. 1992. Class-based ngram models for natural language. Computational Linguistics, 18(4):468–479.

P. Buitelaar, P. Cimiano, and B. Magnini. 2005. Ontology Learning from Text: Methods, Evaluation and Applications. Volume 123 Frontiers in Artificial Intelligence and Applications.

R. Bunescu and R. Mooney. 2007. Learning to Extract Relations from the Web using Minimal Supervision. ACL'07.

S. Caraballo. 1999. Automatic construction of a hypernym-labeled noun hierarchy from text. ACL'99.

T. Chklovski and P. Pantel. 2004. VerbOcean: mining the web for fine-grained semantic verb relations. EMNLP '04.

P. Cimiano and J. Volker. 2005. Towards large-scale, open-domain and ontology-based named entity classification. RANLP'07.

P. Cimiano and J. Wenderoth. 2007. Automatic Acquisition of Ranked Qualia Structures from the Web. ACL'07.

D. Davidov and A. Rappoport. 2006. Efficient Unsupervised Discovery of Word Categories Using Symmetric Patterns and High Frequency Words. ACL'06.

D. Davidov and A. Rappoport. 2008. Classification of Semantic Relationships between Nominals Using Pattern Clusters. ACL'08.

D. Downey, O. Etzioni, and S. Soderland. 2005. A Probabilistic model of redundancy in information extraction. IJCAI'05.

O. Etzioni, M. Cafarella, D. Downey, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. 2005. Unsupervised named-entity extraction from the web: an experimental study. Artificial Intelligence, 165(1):91–134.

C. Fellbuam. 1998. WordNet: An Electronic Lexical Database. MIT Press. 1998.

M. Geffet and I. Dagan. 2005. The Distributional Inclusion Hypotheses and Lexical Entailment. ACL'05.

R. Girju, A. Badulescu, and D. Moldovan. 2003. Learning Semantic Constraints for the Automatic Discovery of Part-Whole Relations. HLT'03.

R. Girju, A. Badulescu, and D. Moldovan. 2006. Automatic Discovery of Part-Whole Relations. Computational Linguistics, 32(1): 83-135.

Z. Harris. 1985. Distributional structure. In Word, 10(23): 146-162s, 1954.

T. Hastie, R. Tibshirani and J. Friedman. 2001. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer-Verlag, 2001.

M. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. COLING'92.

M. D. Hendy and D. Penny. 1982. Branch and bound algorithms to determine minimal evolutionary trees. Mathematical Biosciences 59: 277-290.

Z. Kozareva, E. Riloff, and E. Hovy. 2008. Semantic Class Learning from the Web with Hyponym Pattern Linkage Graphs. ACL'08.

D. Lin, 1998. Automatic retrieval and clustering of similar words. COLING'98.

D. Lin, S. Zhao, L. Qin, and M. Zhou. 2003. Identifying Synonyms among Distributionally Similar Words. IJCAI'03.

G. S. Mann. 2002. Fine-Grained Proper Noun Ontologies for Question Answering. In Proceedings of SemaNet' 02: Building and Using Semantic Networks, Taipei.

P. Pantel and D Lin. 2002. Discovering word senses from text. SIGKDD'02.

P. Pantel and D. Ravichandran. 2004. Automatically labeling semantic classes. HLT/NAACL'04.

P. Pantel, D. Ravichandran, and E. Hovy. 2004. Towards terascale knowledge acquisition. COLING'04.

P. Pantel and M. Pennacchiotti. 2006. Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations. ACL'06.

F. Pereira, N. Tishby, and L. Lee. 1993. Distributional clustering of English words. ACL'93.

D. Ravichandran and E. Hovy. 2002. Learning surface text patterns for a question answering system. ACL'02.

E. Riloff and J. Shepherd. 1997. A corpus-based approach for building semantic lexicons. EMNLP'97.

B. Roark and E. Charniak. 1998. Noun-phrase co-occurrence statistics for semi-automatic semantic lexicon construction. ACL/COLING'98.

R. Snow, D. Jurafsky, and A. Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. NIPS'05.

R. Snow, D. Jurafsky, and A. Y. Ng. 2006. Semantic Taxonomy Induction from Heterogeneous Evidence. ACL'06.

B. Rosenfeld and R. Feldman. 2007. Clustering for unsupervised relation identification. CIKM'07.

P. Turney, M. Littman, J. Bigham, and V. Shnayder. 2003. Combining independent modules to solve multiple-choice synonym and analogy problems. RANLP'03.

S. M. Harabagiu, S. J. Maiorano and M. A. Pasca. 2003. Open-Domain Textual Question Answering Techniques. Natural Language Engineering 9 (3): 1-38, 2003.

I. Szpektor, H. Tanev, I. Dagan, and B. Coppola. 2004. Scaling web-based acquisition of entailment relations. EMNLP'04.

D. Widdows and B. Dorow. 2002. A graph model for unsupervised Lexical acquisition. COLING '02.

H. Yang and J. Callan. 2008. Learning the Distance Metric in a Personal Ontology. Workshop on Ontologies and Information Systems for the Semantic Web of CIKM'08.