

Learning the Distance Metric in a Personal Ontology

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

Personal ontology construction is the task of sorting through relevant materials, identifying the main topics and concepts, and organizing them to suit personal needs. Automatic construction of personal ontologies is difficult in part because measuring the semantic distance between two concepts is difficult. Knowledge-based approaches use either knowledge bases, such as WordNet, or lexico-syntactic patterns to induce the differences between concepts. However, these techniques are only applicable for a subset of concepts and leave the majority unmeasurable. On the other hand, statistical approaches are able to induce the differences between any concept pair but lack of human knowledge involvement and hence suffer from low precision.

In the context of personal ontology construction, semantic distances between concepts need to reflect personal preferences. Based on that, this paper presents a supervised hierarchical clustering framework to incorporate personal preferences for distance metric learning in personal ontology construction. In this framework, periodic manual guidance provides training data for learning a distance metric and the learned metric is used during automatic activities to further construct the ontology. A detailed user study demonstrates that the approach is effective and accelerates the construction of personal ontologies.

Categories and Subject Descriptors

H.4 [Information Systems Applications]: Miscellaneous

Keywords

Ontology Learning, Supervised Hierarchical Clustering

1. INTRODUCTION

With the tremendous growth of digitalized data in the daily life, when gathering information for a specific purpose, for example, buying a used car or searching for a good kindergarten for kids, an individual often needs to do

information triage: sorting through the relevant materials in the newly crawled document collection, identifying the main topics and concepts, and organizing them to suit the actual needs. This is exactly the task of building a personal domain-specific ontology. Personal ontology needs to match with personal needs, reflect knowledge in a specific domain, identify and organize the important concepts into a tree-structured hierarchy.

Due to the hierarchical nature of an ontology, hierarchical clustering is widely used in ontology learning. For example, [4] and [12] used bottom-up hierarchical clustering, [2] used bi-section k-means divisive clustering, and [5] used a hierarchical self-organized map. For any clustering algorithm, a distance metric is indispensable. It is essential to have a good distance metric to measure the semantic dissimilarity between two concepts in an ontology. Unfortunately, a good semantic distance metric, especially for personal ontologies which require tailoring the metric for individual needs, is hard to obtain.

A handy way to measure the semantic distance between two concepts, as in [11, 14], is to use Wordnet [6]. One way is to count the edge distance between two concepts in Wordnet. An even simpler way is to score 0 for two concepts when they are within the same Wordnet hypernym chain, 1 otherwise. The simplicity of Wordnet-based approaches is attractive, however, they are not applicable for all concept pairs and hence suffer from the low recall problem. WordNet includes over 120,000 words (and over 170,000 synsets) but few domain-specific terms. For example, transport and company are individually included, but not transport company. When building a personal, domain-specific ontology, often the concepts do not appear in Wordnet and hence there is no way to measure their Wordnet-based semantic distances. Moreover, even if they appear in Wordnet, the edge distance does not represent the actual semantic distance between two concepts due to unnecessary layers of intermediate concepts in Wordnet. For example, hunter, student, teacher are hyponyms of person. From hunter to person, there are skilled worker and worker as the intermediate concepts. From student to person, there is enrollee. From teacher to person, there are educator, professional and adult. The edge distance between (hunter, student) is 5, (teacher, student) is 6. However, it is clear that hunter is not semantically closer than teacher to student as suggested by the above scores. The inconsistent structure of Wordnet makes a direct use of such scores unreliable.

Lexico-syntactic patterns are also popular semantic distance measures [4, 10, 17]. This approach often makes bi-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CIKM'08, October 26–30, 2008, Napa Valley, California, USA.
Copyright 2008 ACM 978-1-59593-991-3/08/10 ...\$5.00.

nary decisions: two concepts are semantically close only if they appear in one of the valid lexico-syntactic patterns which indicate the presence of a semantic relationship. Such pattern matching techniques perform with high precision but low recall again due to the fact that valid lexico-syntactic patterns occur rarely in corpora. Many semantically close concept pairs may not match such patterns, hence become unmeasurable and remain unorganized in an ontology. The direct consequence is that the ontology constructed by this approach is not a complete tree, but a forest with many fragments (we call them ontology fragments), which clearly needs to be further built up. Unfortunately, it cannot be done by this approach.

Statistical measures are also proposed by researchers. They either make use of contextual distributions [3, 8] or concept co-occurrences as in the subsumption approach [13, 15], to measure the semantic distance between two concepts. Statistical measures resolve the low recall problem and are able to measure the distance between any pair of concepts. However, they purely rely on document statistics and have been isolated to useful knowledge resources such as Wordnet. Moreover, this approach suffers from the inconsistent long chain problem. For instance, car has hyponyms of coupe, sedan and sport, sport has basketball. It is clear that car should not be ascendent of basketball. The lack of syntactic and semantic knowledge about concepts limits the ability of this approach.

Note that ontology construction, especially personal ontology construction, is a highly subjective task. For instance, a user may consider “polar bear” and “seal” semantically close since they are both arctic marine mammals while another user may consider “polar bear” and “black bear” semantically close since they are both bears. Such personal preferences need to be reflected in a good semantic distance metric and hence manual guidance is a valuable resource to define a good metric. However, all the above approaches ignore such information.

In this paper, we propose a novel framework to combine strengths of the above approaches and handle their weaknesses. By learning a distance metric from manual guidance, the framework assigns appropriate distance scores to concepts which remain unorganized. A supervised hierarchical clustering algorithm takes into account personal preferences and alters the clustering procedure to reflect such knowledge in the automatic activities. The learned distance metric is used in K-medoids clustering algorithm to create partitions for a level of concepts. The process repeats level by level until the final ontology is formed.

Notice and comment rulemaking [16] requires administrative agencies of the U.S. government to issue draft versions of proposed regulations, seek comments from stakeholders and the public, and respond in the final rule. Each year a few high profile rules attract hundreds of thousands of comments. Our research is tested on several corpora of public comments and aims on save agencies’ efforts by organizing concepts in the problem domain into ontologies.

The remaining of this paper is organized in the following way: section 2 introduces the framework which supports distance metric learning for semantic relationships. Section 3 describes the procedure to automatically produce ontology fragments. Section 4 shows a detailed user study and experimental results. Section 5 concludes the paper.



Figure 1: Ontology Fragments

2. LEARNING SEMANTIC DISTANCE IN A SUPERVISED HIERARCHICAL CLUSTERING FRAMEWORK

Supervised clustering provides an opportunity to learn the similarity measure from training data for clustering. [7, 9] make use of structural SVM algorithm as the general framework to learn the similarity measure. [1, 19] learn it using prior knowledge. In particular, [19] parameterizes Mahalanobis distance through a positive semi-definite matrix. In this section, we details our approach of distance metric learning in a framework of supervised hierarchical clustering for the task of personal ontology construction.

Note that this section presents our approach in an unconventional way. It goes directly to the most important part of our work, the distance metric learning, with an introduction of the problem settings for it and then details the learning process. After presenting the most important part, it then elaborates the main framework, which is the supervised hierarchical clustering algorithm, followed by a cluster labelling algorithm for the nameless clusters.

2.1 Problem Settings

The target of our research is to leverage the strengths of current technologies in personal ontology learning and solve the problems mentioned in section 1. The techniques developed in this work focus on learning the semantic distance metric for the concepts, which are lack of syntactic and semantic evidence in corpora and hence unmeasurable by the previous approaches.

There are two sources of training data available for this task. One is manual guidance, which reflects personal preferences. In this work, it is represented as manually created ontology fragments. Another is the high precision ontology fragments constructed by either lexico-syntactic pattern matching or Wordnet, as mentioned in section 1. We assume that they are available for now and will show details in section 3 on how to construct such high precision fragments. Note that both kinds of training data are with high precision and in the same representation form - ontology fragments.

Let us see an example. Figure 1 shows ontology fragments constructed either manually or by automatic pattern matching. They are training data for distance metric learning. Such training data suggests that (child, maker) is close since they are in the same group, (sport hunter, trophy hunter) is also close, (sea ice habitat, child) may be far away since they are in different groups. The goal is to find a mapping from such grouping information to their semantic distances and then use the mapping function to predict the semantic distances for unorganized concept pairs, such as (habitat, person) and (person, hunter). The mapping function is required to give reasonable scores to concept pairs such that (person, hunter) is closer than (habitat, person).

The problem is then turned into learning a semantic dis-

tance metric function from the available ontology fragments and predict distance scores between the unorganized concepts, which are usually the roots of ontology fragments (in this example, person, habitat, hunter).

2.2 Learning the Distance Metric

This section details the learning algorithm which finds a distance metric function from the given ontology fragments. A parameterized distance metric is derived and the parameter estimation is described. The prediction of semantic distances for unorganized concepts is also given.

We model the distance metric learning problem in a way that at each time, a set of concepts $\mathbf{x}^{(i)}$ at the i th level of an ontology hierarchy are taken as the training data. Another training input is a distance matrix $\mathbf{y}^{(i)}$, which indicates the distances between all pairs of concepts in $\mathbf{x}^{(i)}$. An entry of this matrix which corresponding to concept $x_j^{(i)}$ and $x_k^{(i)}$ is $y_{jk}^{(i)} \in \{1, 0\}$, where $y_{jk}^{(i)} = 0$, if $x_j^{(i)}$ and $x_k^{(i)}$ in the same cluster; 1, otherwise.

The training data consists n sets of training concepts $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$, ..., $\mathbf{x}^{(n)}$, each with $|\mathbf{x}^{(1)}|$, $|\mathbf{x}^{(2)}|$, ..., $|\mathbf{x}^{(n)}|$ concepts. Each set $\mathbf{x}^{(i)}$ represents a set of concepts at the level indexed by i . For each set of training data, the correct partition is given through the distance matrices $\mathbf{y}^{(1)}$, $\mathbf{y}^{(2)}$, ..., $\mathbf{y}^{(n)}$.

The estimated pairwise distance metric for two concepts at level i can be represented as a Mahalanobis distance:

$$d(x_j^{(i)}, x_k^{(i)}) = \sqrt{\Phi(x_j^{(i)}, x_k^{(i)})^T A \Phi(x_j^{(i)}, x_k^{(i)})} \quad (1)$$

where $\Phi(x_j^{(i)}, x_k^{(i)})$ represents a set of pairwise underlying feature functions, where each feature function is $\phi_d : (x_j^{(i)}, x_k^{(i)}) \mapsto r \in \mathbb{R}$ with $d=1, \dots, |\Phi|$. The underlying feature functions evaluate the relationship between $(x_j^{(i)}, x_k^{(i)})$ from various aspects. We will introduce them in more details in section 2.3. A is a parameter matrix, it gives different weights to each underlying distance feature function.

Theoretically, the parameter estimation problem in our problem is to get A such that the expected loss is minimized. The expected loss, or the risk, can be represented as :

$$R(\hat{\mathbf{y}}) = \int \Delta(\mathbf{y}, \hat{\mathbf{y}}) p(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y} \quad (2)$$

where $p(\mathbf{x}, \mathbf{y})$ is the joint distribution of concepts and their clustering assignments. The loss function Δ in equation 2 is defined as the squared error loss of two distance matrices.

One important characteristic of such distance matrices is that they must represent valid clustering partitions, which means the clustering partitions represented by these distance matrices should be consistent. Therefore, certain constraints need to be satisfied. An obvious one is that concepts in the same cluster should have smaller distance scores than those in different clusters. Moreover, a valid distance metric should be non-negative and satisfying the triangle inequality. To ensure such regularities, we need to constrain the parameter matrix A to be positive semi-definite (PSD) [19].

In the parameter estimating process, the loss function Δ is minimized through minimizing the squared errors and the optimization function is defined as :

$$\min_A \sum_{j=1}^{|\mathbf{x}^{(i)}|} \sum_{k=1}^{|\mathbf{x}^{(i)}|} (y_{jk}^{(i)} - \sqrt{\Phi(x_j^{(i)}, x_k^{(i)})^T A \Phi(x_j^{(i)}, x_k^{(i)})})^2 \quad (3)$$

Algorithm 2.2 : Training Instance Selection

1. For each instance $x_j^{(i)}$, calculate the sum of distances from it to the rest in the training set:

$$s_j^{(i)} = \sum_k d(x_j^{(i)}, x_k^{(i)}), \forall k \neq j$$

2. Select instances with the above score in the middle range:

For each instance $x_j^{(i)}$,
 If $\alpha s_{max}^{(i)} \leq s_j^{(i)} \leq \beta s_{max}^{(i)}$, keep $x_j^{(i)}$,
 otherwise, remove it.
 where $s_{max}^{(i)} = \max(s_j^{(i)}), \forall j$.

subject to $A \succeq 0$

The optimization can be done by any standard semi-definite programming (SDP) solvers. Our system uses matlab software packages sedumi¹ and yalmip² to do the optimization.

Given the learned parameter matrix A , it is easy to generate distance metric for any pair of unmeasured concepts. Let $(x_l^{(i+1)}, x_m^{(i+1)})$ be an unmeasured concept pair from one level higher, i.e., the $(i+1)^{th}$ level in the ontology hierarchy. By calculating the distance for each concept pairs, we obtain the entries in a new distance matrix $\hat{\mathbf{y}}^{(i+1)}$. Note that this distance matrix should also result in a consistent clustering, which is guaranteed by the positive semi-definite parameter matrix A . The entry values for $\hat{\mathbf{y}}^{(i+1)}$ is defined as:

$$\hat{y}_{lm}^{(i+1)} = \sqrt{\Phi(x_l^{(i+1)}, x_m^{(i+1)})^T A \Phi(x_l^{(i+1)}, x_m^{(i+1)})} \quad (4)$$

The learned distance matrix $\hat{\mathbf{y}}^{(i+1)}$ contains the distance scores for concepts at the $(i+1)^{th}$ level. Note that previously they were unmeasured and unorganized. The scores are then passed into a clustering algorithm to produce partitions.

When there are many training instances from multiple levels of an ontology hierarchy, the learning speed of SDP solver is slow. To avoid congestion in this real-time interactive system, we employ a training instance selection algorithm. In a clustering assignment, singleton instances are usually less important since they have no relationship to the rest. Moreover, they have big distances, i.e., far away from, almost all the other instances in the set. Based on such observations, only instances having more relationships to others are kept as the training instances. Algorithm 2.2 details the process. The parameters α and β are set to 0.3 and 0.7 respectively.

2.3 Underlying Feature Functions

The learning of the distance metric function is actually a weight estimation for different underlying feature functions. Each feature function is a measurement of how distant two concepts are. Since it is easier to get similarity scores than to get dissimilarity scores for semantic relationships, we transform each distance feature function as one subtracts the pairwise similarity score for a concept pair.

Note that in theory, the design of the learning framework allows us to embed as many feature functions as we want into the framework as long as they are indicators of semantic difference/similarity between two concepts. However, due to the speed limitation of SDP solver, we only focus on the feature functions listed in Table 1.

¹<http://sedumi.mcmaster.ca/>

²<http://control.ee.ethz.ch/~joloef/yalmip.php>

Table 1: Underlying Feature Functions

The left half shows features for a single concept $x_j^{(i)}$, for the diagonal entries of the distance matrix. The right half shows features for two concepts $(x_j^{(i)}, x_k^{(i)})$, for the off-diagonal entries of the distance matrix. The feature functions are represented in the form of “1-f”, where f is a pairwise similarity function. “Normalized” means the data is normalized into the range of [0,1] by being divided by the maximum possible value. A verb predicate is in the form of verb(subject, object).

| | |
|-----------------------------------|--|
| 1 - normalized raw term frequency | 1 - normalized raw frequency of term co-occurrences of $(x_j^{(i)}, x_k^{(i)})$ |
| 1 - normalized log term frequency | 1 - normalized log frequency of term co-occurrences of $(x_j^{(i)}, x_k^{(i)})$ |
| Constant (always 1) | 1 - word overlap of Web definitions of $(x_j^{(i)}, x_k^{(i)})$ |
| | 1 - word overlap of modifiers of $(x_j^{(i)}, x_k^{(i)})$ |
| | 1 - object overlap of the same verb predicate when $x_j^{(i)}$ or $x_k^{(i)}$ as the subject |
| | 1 - subject overlap of the same verb predicate when $x_j^{(i)}$ or $x_k^{(i)}$ as the object |

Algorithm 2.4: Supervised Hierarchical Clustering

```

while not satisfied or not all concepts connected in a tree
  construct groups for level  $i$  by flat clustering;
  if in interactive mode
    wait for manual guidance;
    learn distance metric function from level  $i$ ;
    predict distance scores for level  $i + 1$ ;
     $i \leftarrow i + 1$ ;
Output the tree
  
```

Features in Table 1 are a balanced mixture of statistical, contextual and knowledge-based distance functions. Features in the left column of table 1 (for a single concept) and the first two features in the right column (for pairwise concepts), are basically various forms of term (co-)occurrences in corpora, which are statistical evidence of how distant two concepts are. Last two features in the right column are contextual features which measure the word overlap between the subjects/objects of verb predicates where each of the two concepts is the object/subject. For example, for concepts “polar bear” and “seal”, habitat(polar bear, arctic ice) and habitat(seal, sea ice) are two corresponding verb predicates, where the two concepts are the subjects. The word overlap between the objects is 1 (“ice” in this case). The fourth feature in the right column is also a contextual feature which measures the word overlap between noun or adjective modifiers in front of two concepts. For example, the overlap between modifiers in “high blood pressure” and “press pressure” is 0. The third feature function in the right column is based on the word overlap between the Web definitions of two concepts $(x_j^{(i)}, x_k^{(i)})$ by issuing query “define: $x_j^{(i)}$ ” and “define: $x_k^{(i)}$ ” to Google. Note that the Web definitions for concepts are mainly from Wordnet.

Notice that there is no feature function based on lexico-syntactic patterns. It is designed with purpose due to the fact that the unorganized concepts are unmeasurable by such patterns as we mentioned earlier.

2.4 Supervised Hierarchical Clustering

So far we have seen that the algorithm learns a distance metric function, and applies it to predict distance scores for unorganized concepts. With the predicted distance scores, any flat clustering algorithm can be used to cluster the unorganized concepts. However, the roots of the newly formed groups remain unorganized and need to be further clustered. It then needs to learn a new distance function for them and

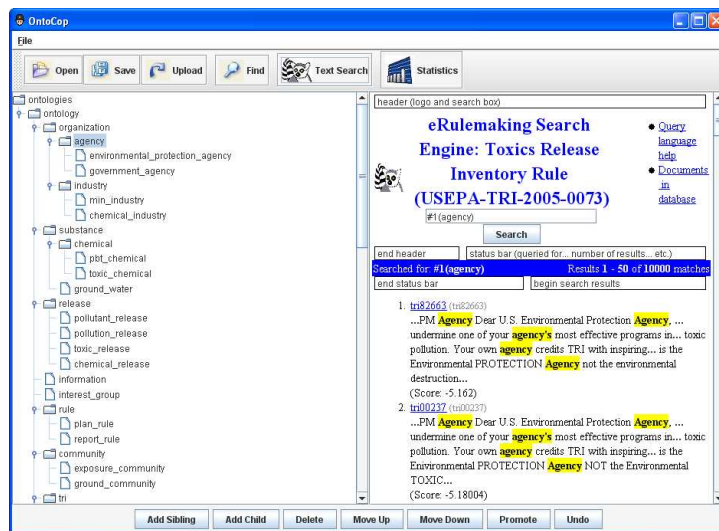


Figure 2: OntoCop Interface

predict their distance scores in order to further group them. This process can repeat many times until we are satisfied with the ontology or a complete tree is formed.

The above process is in a bottom-up fashion. Algorithm 2.4 gives the pseudo-codes for the supervised hierarchical clustering algorithm. Starting from ontology fragments at the bottom, the process builds up the ontology level by level by learning a new distance metric from the current level and applying it to the higher level. Note that the ontology fragments are built either manually or from high precision methods such as lexico-syntactic patterns, thus we can always trust the training data from the lower levels.

At each iteration, any flat clustering algorithm can be used to construct concept groups. The flat clustering algorithm used in this work is K-medoids. In K-medoids algorithm, the random initialization of K centers leads to somewhat unstable assignments. To get a stable cluster assignment and to remove the randomness introduced by the random initialization, K-medoids runs for 1000 times with different random initialization and then assigns each instance to its most frequent cluster center. We adopt Gap statistics [18] to estimate the optimal number of clusters.

After concepts are clustered by K-medoids, if the system is in its interactive mode, it displays the learned ontology

and waits for manual guidance. Users can interact with the system via a tool called OntoCop (**O**ntology **C**onstruction **P**anel). Figure 2 shows the user interface of OntoCop. Users are able to add, delete, modify concepts, drag & drop concepts around and group them accordingly. Users can also search and view the documents relevant to a concept for a better understand of the domain knowledge when they are making decisions. When they are done with modifications to the concepts, they can upload the hierarchy to the server, which learns from the user modifications, predicts new distance scores for unorganized concepts and runs K-medoids to cluster them and returns the new hierarchy to users.

The supervised hierarchical clustering framework requests for manual guidance at each learning cycle, and adjusts the learning of the distance metric accordingly. In particular, by taking into account a user’s modification to the ontology, the system learns from his/her personalized grouping of concepts. The distance metric learning techniques presented in our approach make it possible that an ontology is formed based on the personal preferences of an individual user.

2.5 Higher Level Concept Names

In the bottom-up supervised hierarchical clustering framework, it needs to assign meaningful names to newly-created groups. The name for a new parent concept should be general enough to cover the meaning of all the child concepts, and be specific enough to just cover that of them. The general knowledge available on the Web is a good source to generate such names.

This work takes a Web-based approach for concept naming. A query formed by concatenating the child concepts (with comma as the delimiter) is sent to Google search engine. The top 10 returned snippets are parsed. After stop-words removal, the most frequent phrase is selected as the parent concept for a group. For example, “president” becomes the parent of “Bush” and “Reagan”.

3. CONSTRUCT ONTOLOGY FRAGMENTS

In the previous sections, we assume that the high precision ontology fragments constructed by the state-of-the-art technologies are available and based on that we learn the distance metric function of semantic relationships. In this section, we show our use of the current technologies to automatically construct such ontology fragments.

3.1 Concept Candidate Selection

The public comment corpora mainly contain email comments sent to agencies. The text collection is first processed through a duplicate detection tool [20]. The remaining texts after duplicate removal are split into sentences. Each sentence are then parsed by a POS tagger³. An noun sequence generator then outputs noun sequences, a sequence of words whose POS tags are NN (singular noun), NNP(singular proper name), NNS(plural noun), or NNPS(plural proper name), in the parsed sentences. For instance, “mercury/NN emissions/NNS” and “power/NN plants/NNS” are two such noun sequences extracted from “I/PRP strongly/RB urge/VBP you/PRP to/TO cut/VB mercury/NN emissions/NNS from/IN power/NN plants/NNS by/IN 90/CD percent/NN by/IN 2008/CD ./.”.

³<http://nlp.stanford.edu/software/tagger.shtml>

Each of these noun sequences is treated as a single text fragment, from which two kinds of candidates are generated. The first kind are bi-grams and tri-grams with frequency above a threshold (set to 2 in this work), for instance, “mercury emissions” and “power plant”. The second kind are noun sequences with length more than 3 and with first letters capitalized in every word. They are actually simple named entities. For instance, “Marine Mammal Protection Act”. These bi-grams, tri-grams and simple named entities are the initial concept candidates in the first pass of concept selection process.

3.2 Concept Filtering

During the first pass of concept candidate selection, the POS tagger outputs false positive nouns. For instance, “protect polar bear” is tagged as three nouns and then the noun sequence generator outputs it as a noun phrase. The POS tagging error is one of the major error sources in our approach. Given that POS taggers unavoidably generate such errors, a reliable way to detect the errors is needed.

Note that those POS tagging errors are mainly a verb or an adjective attached in front of a noun phrase or a weird verb phrase, for instance, “cause cause”. They are not common in the daily language. Hence, Google search results are used to evaluate whether a concept candidate is valid. Each concept is formulated as a query and sent to the Google search engine. Among the first 10 returned snippets, if a concept candidate appears above a threshold (set to 4), it is considered as a commonly-used phrase, otherwise, an error.

In this way, POS tagging errors are effectively identified and false positives are removed. Moreover, spelling errors in the concepts, for example “polor bear”, “pulution”, are also removed. In our experiments, concept filtering increases the precision by 14% by removing POS and spelling errors.

3.3 Initial Ontology Fragments

With the concept candidates, initial ontology fragments are created. They are first built from bi-grams. If the head nouns for two bi-grams are from the same Wordnet synset in their first senses (including the case that two bi-grams sharing the same head noun), then the two bi-grams are assigned to the same group. For these groups, one of their common head nouns is selected to as the parent concept. For example, “pollution” is selected to be the parent of “water pollution” and “air pollution”. A forest of ontology fragments with depth 2 is established.

Tri-gram concept candidates and simple named entity candidates are compared with the parent concepts and added as children of a parent concept if their suffixes match with it. For example, “heavy metal pollution” is added as “pollution”’s child. In Figure 1, fragments rooting from “hunter” and “habitat” are built in this way.

3.4 Refinement Using Wordnet Hypernyms

The above string-matching technique effectively creates the initial set of ontology fragments, however, it also produces problematic fragments. For instance, “species” is with two children “animal species” and “bear species”. The sibling, “animal species” is actually the hypernym of “bear species”. Therefore, within a concept group, further investigation is needed. The concepts at the leaf level of each ontology fragment are examined in Wordnet. If one of the concept is another’s hypernym, the first is promoted as the

Table 2: Statistics of Public Comment Datasets

| Statistic | tri | wolf | polar bear | mercury |
|-------------|-----------|-----------|------------|------------|
| #docs | 86,740 | 283,265 | 546,896 | 536,967 |
| #words | 2,148,309 | 1,683,353 | 1,729,025 | 12,990,349 |
| #vocabulary | 12,838 | 51,938 | 67,110 | 102,503 |
| #concepts | 248 | 795 | 351 | 1084 |

parent of the latter’s. This process creates an intermediate level for the fragments.

It is also observed that there are concepts in different ontology fragments, but with hypernym relationships. The root concepts in the fragments are examined in Wordnet, and are connected into one fragment if they are in the same Wordnet hypernym chain. The ontology fragment rooting from “person” in Figure 1 is created in this way.

The ontology fragments created by the above techniques are with high precision since they are generated based on Wordnet synsets and syntactic pattern matching. They are used in the supervised hierarchical clustering framework as initial training data for distance metric learning.

4. USER STUDY AND EXPERIMENTS

As a system designed for personal ontology construction, a real user evaluation is necessary. We collaborated with the Qualitative Data Analysis Program (QDAP) at the University of Pittsburgh’s University Center for Social and Urban Research (UCSUR) to evaluate the system. Twelve professional coders participated in the experiments. They were divided into two groups, four for the manual group and eight for the interactive group. Users in the manual group were asked to construct ontology with the ontology fragments produced by the system in a bottom-up fashion until they felt satisfied with their work or reaching a 90-minutes limit (which is carefully evaluated by the experiment designers). The interactive group were asked to work interactively with the system until they felt satisfied with the work or reaching a 90-minutes limit. Each user in the interactive group worked on organizing the ontology fragments for a few minutes, then uploaded the modified hierarchy to the system; then the system learned from the user’s feedback, produced a new hierarchy and returned it to the user. It is the user’s decision to continue modifying the ontology and teaching the system to learn or stop. Both groups used the same editing tool provided in OntoCop, such as deleting, adding a node, dragging and dropping a node, promoting a node to the higher level, undoing previous actions, etc. The ontology fragments given to both groups were the same.

There are four public comment data sets used in the experiments, namely “toxic release inventory (tri)” (Docket id: USEPA-TRI-2005-0073), “wolf” (USEPA-RIN-1018-AU53), “polar bear” (USDOI-FWS-2007-0008), “mercury” (USEPA-OAR-2002-0056). Table 2 shows the number of documents, total number of words and unique words after duplicate detection, and number of concept candidates in these datasets. Among these four datasets, “tri” is the one with the smallest vocabulary and used for tool training for both manual and interactive users. The experimental results generated on “wolf”, “polar bear” and “mercury” datasets are reported in the following sections.

Since the problem addressed in this paper is to produce good distance metric for unmeasurable concepts in other

Table 3: Distance Metric Quality for Parent-Child Pairs

| | Baseline | Our Approach |
|------------|----------|--------------|
| wolf | 0.74 | 1 |
| polar bear | 0.61 | 1 |
| mercury | 0.68 | 1 |

Table 4: Distance Metric Quality for Sibling Pairs

| | Baseline | Our Approach |
|------------|----------|--------------|
| wolf | 0.46 | 1 |
| polar bear | 0.26 | 1 |
| mercury | 0.28 | 1 |

approaches, recall is an important evaluation metric. F3-measure, which weighs recall three times as much as precision, is used in the experiments. Two criteria are used to evaluate the performance of ontology construction: F3-measure for parent-child pairs and F3-measure for sibling pairs. The first measures whether a concept is assigned to the correct parent while the second measures whether a concept belongs to the correct group. For a given hierarchy, two lists of node pairs are generated. The first is a list of all the parent-child pairs in the hierarchy. The second is a list of all the sibling pairs. Both gold standard runs and test runs are represented in such two lists. The F3-measures generated from such two lists are used in the following experiments.

Since the ontology learning task studied in this work is personalized, the gold standard used in the evaluation is the final version of the ontology constructed by each user.

4.1 Quality of Distance Metrics

This experiment investigates the quality of distance metrics for personal ontology construction. We compare the distance metrics learned in our approach with the distance metrics generated from a baseline system. The baseline system is a state-of-the-art system which uses lexico-syntactic patterns as presented in [10] and Wordnet edge distance (see section 1) to generate distances for concept pairs. We indirectly evaluate the quality of distance metrics by evaluating the clustering results based on them. The clustering performance is measured by F3-measure of parent-child pairs and F3-measure of sibling pairs.

Table 3 shows the F3-measures of parent-child pairs in the resulting clusters against the gold standard ontologies for three datasets. The reported F3-measures are averaged over 8 interactive users. The baseline system gives F3-measure ranging from 0.61 to 0.74 for different datasets, while our approach gives F3-measure of 1 for all datasets. It clearly shows that our approach learns from personal preferences of individual users and gives a perfect personalized distance metric for the task of ontology learning.

Table 4 shows the F3-measures of sibling pairs in the resulting clusters against the gold standard ontologies. The reported F3-measures are also averaged over 8 interactive users. The baseline system gives F3-measure ranging from 0.26 to 0.46 for different datasets, while our approach again gives F3-measure of 1 for all datasets. Comparing the baseline performance for parent-child pairs and for sibling pairs, we notice that the baseline system suffers from a performance drop in the second task. It may suggest that the task

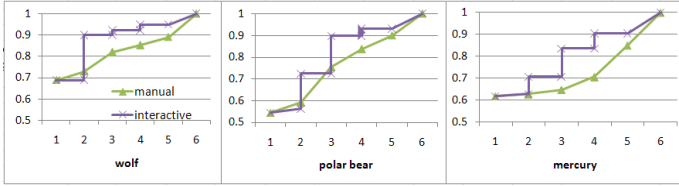


Figure 3: F3 for Parent-Child Pairs over Cycles

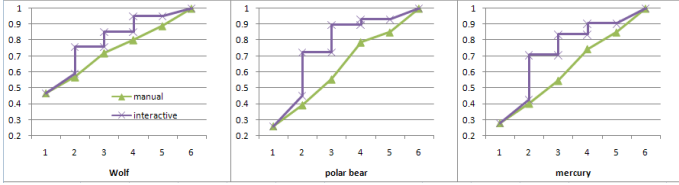


Figure 4: F3 for Sibling Pairs over Cycles

of grouping concepts together is more difficult than identifying parent-child relationship. Nevertheless, our approach again demonstrates its ability to learn a perfect distance metric. It shows that our approach is very effective to learn the distance metric in a personal ontology.

4.2 Learning Effect over Cycles

In the interactive runs, human users and the system work together to build a personal ontology in several cycles. This experiment investigates the system’s ability to learn from different users and eventually fulfil individual user’s needs.

Figure 3 shows the changes of average F3-measure for parent-child pairs over six learning cycles. Figure 4 shows the changes of average F3-measure for sibling pairs. The x-axis are the learning cycles for each dataset. The y-axis indicates the averaged F3-measures.

Results for both interactive and manual users before and after each learning cycle are shown. For manual users, we use their partially constructed ontologies with 20%, 40%, 60%, and 80% modifications in the editing log and plot the F3-measures. Each individual’s partial ontologies are compared with his/her own finalized ontology. The F3-measure is averaged over the 4 users in the manual group. For interactive users, we take the ontologies that uploaded by them each time to the server and plot the F3-measures of each uploaded version and the learned ontology afterwards against his/her own finalized ontology. The F3-measure for the interactive group is averaged over the 8 members.

In both Figure 3 and Figure 4, the F3-measures for both manual and interactive groups converge to 1 at the end of the learning process. It can be attributed to the fact that this is a personalized task. For the interactive users, we notice an obvious performance gain between an uploaded ontology and the ontology learned automatically from it. It clearly shows that the interactive system learns from individual users and improves the system performance at each learning cycle.

Moreover, comparing the performances of interactive and manual users, we notice that the learning curve of the interactive users are steeper than that of the manual users. It indicates that the learning process in the interactive runs helps the interactive users move faster towards their personal satisfaction levels.

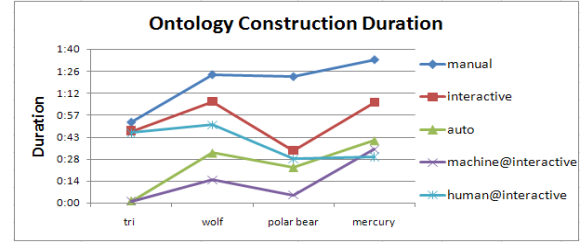


Figure 5: Ontology Construction Duration

4.3 Structure Changes over Cycles

This experiment studies the changes of ontology structures over the learning cycles. Table 5 lists the total number of concepts, hierarchy depth, number of concepts at level 1 (the top level) and level 2, averaged over the 8 interactive users for the three datasets in six learning cycles. The structure of the manually constructed ontologies is also presented and averaged over the 4 manual users.

We notice that with more learning cycles, the depth of the ontology increases, number of top level nodes decreases. Comparing to the manual runs, interactive runs produce deeper hierarchies and fewer top level nodes. It indicates that interactive users have developed more ontology layers and created more groupings thus produced fewer top level nodes than manual users. It confirms that our learning algorithm creates more opportunities for users to explore the structure of an ontology and saves efforts for users in building a personal ontology.

4.4 Duration

A big gain of our system is time. Figure 5 shows the time used by the manual, interactive and automatic runs. The automatic runs turn off the manual guidance component and directly learn from the flat clustering results from the lower levels at each learning cycle (See section 2.4).

Almost all runs are done within 90 minutes, except for the manual runs on mercury dataset where users deal with more than 1000 concepts. In general, interactive runs save 30 to 60 minutes for building one ontology. The automatic run unsurprisingly runs the fastest. The machine time and manual time in part of the interactive runs are also presented as “machine@interactive” and “human@interactive”. Figure 5 shows that the machine time is much less than the human time in the interactive runs except for the mercury dataset. It is because that our algorithm employs internet search engines to name the new clusters, the air time makes the machine time higher than what we have expected. Nevertheless, the proposed learning framework and interactive system accelerates the construction of personal ontologies.

5. CONCLUSIONS

This paper has proposed a learning framework to combine the strengths of current technologies for defining semantic distances between concepts and to construct personal ontology given a text corpus. By incorporating personal preferences as guidance, the proposed distance metric learning and supervised hierarchical clustering framework is able to predict good semantic distance scores for concepts and further organize them into ontology hierarchies.

Other than simple case studies or reconstructing exist-

Table 5: Ontology Structure over Learning Cycles: Averaged over 12 Users

| | wolf | | | | polar bear | | | | mercury | | | |
|--------|--------|-------|--------|--------|------------|-------|--------|--------|---------|-------|--------|--------|
| cycle | #nodes | depth | level1 | level2 | #nodes | depth | level1 | level2 | #nodes | depth | level1 | level2 |
| manual | 766 | 5.8 | 13.8 | 74.5 | 335 | 5.5 | 7 | 45.8 | 1084 | 4.3 | 104.8 | 186.8 |
| 1 | 795 | 4 | 103 | 235 | 351 | 4 | 87 | 102 | 1046 | 3 | 240 | 296 |
| 2 | 766 | 4.3 | 85.7 | 203 | 324 | 4.3 | 70.3 | 87.3 | 1032 | 4.2 | 190.2 | 239.2 |
| 3 | 611 | 4.8 | 23.2 | 96.8 | 318 | 4.7 | 30.8 | 69 | 1006 | 4.5 | 158 | 234.7 |
| 4 | 694 | 5.7 | 10.7 | 75 | 331 | 4.8 | 20 | 61 | 938 | 4.8 | 35 | 159.4 |
| 5 | 648 | 7 | 10 | 49 | 314 | 5 | 8.5 | 52.5 | 918 | 4 | 53.5 | 170.5 |
| 6 | - | - | - | - | - | - | - | - | 868 | 5 | 84 | 217 |

ing ontologies, a detailed user study on concept ontology construction over large email datasets has been done. The results show that the interactive learning process not only saves time and human efforts, but also produces high quality ontology by learning from individual users and accelerates the process of developing personal ontologies.

The system has been tested on datasets which do not contain broad, diverse concepts like many other research have been worked on, which makes the ontology construction a harder problem. Such specific domains contain many similar concepts and the task requires stronger ability to disambiguate the subtle differences among similar concepts. Our system has successfully demonstrated its ability to deal with such domains.

6. ACKNOWLEDGMENTS

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.

7. REFERENCES

- [1] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *Proceedings of the International Conference on Machine Learning*, 2003.
- [2] P. Cimiano, A. Hotho, and S. Staab. Comparing conceptual, divisive and agglomerative clustering for learning taxonomies from text. In *Proceedings of the European Conference on Artificial Intelligence*, 2004.
- [3] P. Cimiano and J. Vlker. Text2onto: A framework for ontology learning and data-driven change discovery. In *Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems*, 2005.
- [4] M. Degeratu and V. Hatzivassiloglou. Building automatically a business registration ontology. In *Proceedings of the 2nd National Conference on Digital Government Research*, 2002.
- [5] D. Elliman and J. R. G. Pulido. Automatic derivation of on-line document ontologies. In *International Workshop on Mechanisms for Enterprise Integration: From Objects to Ontology*, 2001.
- [6] C. Fellbaum. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.
- [7] T. Finley and T. Joachims. Supervised clustering with support vector machines. In *Proceedings of the International Conference on Machine Learning*, 2005.
- [8] G. Grefenstette. Explorations in automatic thesaurus construction. In *Kluwer*, 1994.
- [9] P. Haider, U. Brefeld, and T. Scheffer. Supervised clustering of streaming data for email batch detection. In *Proceedings of the International Conference on Machine Learning*, 2007.
- [10] M. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics*, 1992.
- [11] H. Hu and X. Du. An ontology learning model in grid information services. In *Proceedings of the First International Conference on Innovative Computing, Information and Control*, 2006.
- [12] L. Khan and L. Wang. Automatic ontology derivation using clustering for image classification. In *In Proc. of 8th International Workshop on Multimedia Information Systems*, 2002.
- [13] D. J. Lawrie and W. B. Croft. Generating hierarchical summaries for web searches. In *Proceedings of the Annual International ACM SIGIR Conference*, 2003.
- [14] N. R. P. Velardi, and A. Gangemi. Ontology learning and its application to automated terminology translation. In *Intelligent Systems, IEEE, Volume 18, Issue 1*, 2003.
- [15] M. Sanderson and B. Croft. Deriving concept hierarchies from text. In *Proceedings of the 22nd Annual International ACM SIGIR Conference*, 1999.
- [16] S. Shulman, J. Callan, E. Hovy, and S. Zvestoski. Language processing technology for electronic rulemaking: A project highlight. In *Proceedings of the 6th National Conference on Digital Government Research*, 2005.
- [17] R. Snow, D. Jurafsky, and A. Y. Ng. Semantic taxonomy induction from heterogenous evidence. In *COLING/ACL*, 2006.
- [18] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a dataset via the gap statistic. In *Tech. Rep. 208, Dept. of Statistics, Stanford University*, 2000.
- [19] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Advances in Neural Information Processing Systems*, 2002.
- [20] H. Yang and J. Callan. Near-duplicate detection by instance-level constrained clustering. In *Proceedings of the 29th Annual International ACM SIGIR Conference*, 2006.