Ontology Enhancement and Concept Granularity Learning: Keeping Yourself Current and Adaptive

Shan Jiang[†], Lidong Bing[‡], Bai Sun[†], Yan Zhang[†*], Wai Lam[‡]

[†]Department of Machine Intelligence
Peking University
Beijing 100871, China
{jsh, sunbai}@pku.edu.cn
zhy@cis.pku.edu.cn

[‡]Department of Systems Engineering and Engineering Management
The Chinese University of Hong Kong
Shatin, Hong Kong
{ldbing, wlam}@se.cuhk.edu.hk

ABSTRACT

As a well-known semantic repository, WordNet is widely used in many applications. However, due to costly edit and maintenance, WordNet's capability of keeping up with the emergence of new concepts is poor compared with on-line encyclopedias such as Wikipedia. To keep WordNet current with folk wisdom, we propose a method to enhance WordNet automatically by merging Wikipedia entities into WordNet, and construct an enriched ontology, named as WorkiNet. WorkiNet keeps the desirable structure of WordNet. At the same time, it captures abundant information from Wikipedia. We also propose a learning approach which is able to generate a tailor-made semantic concept collection for a given document collection. The learning process takes the characteristics of the given document collection into consideration and the semantic concepts in the tailor-made collection can be used as new features for document representation. The experimental results show that the adaptively generated feature space can outperform a static one significantly in text mining tasks, and WorkiNet dominates WordNet most of the time due to its high coverage.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Thesauruses, Dictionaries*; I.5.2 [**Pattern Recognition**]: Design Methodology—*Feature evaluation and selection*

General Terms

Algorithms, Design, Experimentation, Performance

Keywords

WorkiNet, Ontology, Tailor-made Concept Representation Learning

*Corresponding author

KDD'11, August 21–24, 2011, San Diego, California, USA.

1. INTRODUCTION

WordNet [7] organizes terms by a variety of relations such as synonymy, antonymy, hyponymy and meronymy. The carefully designed structure and other characteristics make WordNet widely used in many applications [18]. However, maintaining WordNet with updated entries requires costly manual effort, which limits its capability to keep up with the emergence of new concepts. To enhance WordNet and keep it current, we propose a method to merge Wikipedia entities into it and construct an enriched ontology, named as WorkiNet.

As the largest on-line collaborative encyclopedia, Wikipedia[1] provides us a huge amount of knowledge in various domains. The compound annual growth rate of English Wikipedia is 88.4%[2] from 2002 to 2010. 17 million articles (over 3.5 million in English) have been written collaboratively by volunteers around the world. Due to folk wisdom, Wikipedia's ability of covering the newly-minted fact or knowledge is excellent compared with traditional expert-edited ontologies, such as WordNet. Take the term "Bing" as an example. In Wikipedia it may refer to a Web Search Engine from Microsoft, or a soft drink from UK, or others. But in WordNet this term does not even exist. Therefore, it is reasonable and beneficial to enrich WordNet with Wikipedia. Furthermore, it is also expected that the enhanced ontology will perform better in text mining tasks.

Traditionally, text document representation is usually based on the Bag of Words (BOW) approach, which represents the documents with features as weighted occurrence frequencies of individual words. This technique has several drawbacks: first, it breaks a phrase, say "text mining", into independent features. Second, it maps synonymous words into different features. Third, it merges different meanings of a polysemous word into a single feature. These drawbacks make BOW unable to compute document similarity accurately. Methods aiming at overcoming these drawbacks can be categorized into two classes: latent semantic analysis (including LSA [6], PLSA [10] and LDA [17]) and ontology-based methods [12, 21]. In this paper, we focus on the later methodology. It is observed that mapping synonyms into the same dimension can improve the quality of similarity calculation between documents which are described with different terms in the same synset [9]. Another kind of semantically expressive relation is hyponymy, and sibling hyponyms are often tightly related [24]. Therefore, merging these hyponyms can also offer useful hints for determining the similarity between related documents. For example, if we merge "overlip" and "underlip" into their common hypernym "lip", the similar-

[1]http://www.wikipedia.org
[2]http://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia

ity between documents containing these three different words will be increased.

We believe that the decision of the granularity of semantic term representation should be adaptive when dealing with different document collections. For instance, suppose we have two document collections. The first one is associated with coarse granularity categories, such as sports, military, etc, while the second one's granularity is finer, such as football, basketball, etc. In the first collection, "football" and "basketball" should be regarded as related, while in the second one they should be regarded as unrelated. However, most of the existing works are static [12, 29], consequently, these methods cannot handle the above scenarios. To tackle this problem, we propose a Tailor-made Concept Representation Learning (TCRL) model, which can learn a semantic concept collection from an ontology according to the characteristics of a given document collection and represent the documents adaptively.

The remainder of this paper is organized as follows. The details of WorkiNet construction are discussed in Section 2. The TCRL model is proposed in Section 3. Then the experimental results are given in Section 4. The related works are reviewed in Section 5. Finally, we conclude the paper in Section 7.

2. ENHANCED ONTOLOGY: WORKINET

Obviously, expert-edited ontologies are rather static. For example, the maintainers update WordNet once every few years. Ontologies of this kind have poor ability to cover new terms compared with on-line encyclopedias, such as Wikipedia. We propose a method to integrate Wikipedia's entities into WordNet, and construct an enriched ontology, named as WorkiNet.

2.1 Ontology of WordNet

Concepts are the basic components of an ontology, and represented as synonym sets. Among concepts, several semantic relations exist, such as *antonymy*, *hyponymy(is-a)*, *holonymy(part-of)*, etc. From the document representation point of view, synonymy and hyponymy relations are more useful. Thus they are the relations considered in this paper. We give a formal definition of a concept:

***Concept**: A semantic concept π is a triple $\langle S, g, H\rangle$, where S is the synonym set, g denotes the gloss, and H is the set of its hyponym concepts. We refer to the items with $\pi.S$, $\pi.g$ and $\pi.H$ respectively. A set of concepts is denoted as Π. If H is empty, π is a leaf concept.*

Organized by hyponymy, the ontology has a Hierarchical Directed Acyclic Graph (HDAG) structure. A fragment of WordNet is shown in Figure 1.

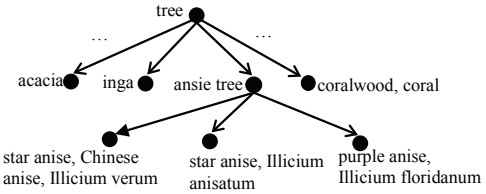

Figure 1: A fragment of WordNet structure. Each node is a concept with its synset attached as the label.

2.2 Wikipedia

Generally speaking, Wikipedia also has a hierarchical structure. However, because of the collaborative manner of editing and the massive number of editors, Wikipedia's structure is much more complicated than that of an expert-edited ontology. Another major difference between Wikipedia and an expert-edited ontology is that the former's categories are quite arbitrarily named, which makes it difficult to match a Wikipedia category with a concept in expert-edited ontologies accurately. To tackle this problem, we propose a concept expansion method to integrate Wikipedia entities into an expert-edited ontology.

For the convenience of describing our method, we first introduce two definitions and one data structure.

***Wikipedia Entity**: A Wikipedia entity ϕ is a triple $\langle t, d, C\rangle$ denoting an article in Wikipedia, where t is the title, d denotes the text description, and C is the set of categories that the article belongs to. We refer to the items with $\phi.t$, $\phi.d$ and $\phi.C$ respectively. Each element of C is denoted as c. The entire Wikipedia entity set is denoted as Φ.*

***Category Head**: In Wikipedia, each category name can be treated as a noun phrase, and its lemmatized head term is called a category head, denoted as h.*

***Index of Entity to Category Head** (IECH): For each ϕ, we construct a forward index: $\phi \rightarrow \{h_{i_1} : f_{i_1}, h_{i_2} : f_{i_2}, \cdots\}$, where f_{i_n} means the frequency of h_{i_n} appearing as a category head in all elements of $\phi.C$. The lemmatized head terms are sorted in descending order according to their frequencies.*

The notations introduced above are summarized in Table 1.

Table 1: Notation used in the paper.

Notation	Meaning
π	A semantic concept in WordNet
$\pi.S$	The synonym set of π
$\pi.g$	The gloss of π
$\pi.H$	The hyponym set of π
ϕ	A Wikipedia entity
$\phi.t$	The title of ϕ
$\phi.d$	The description of ϕ
$\phi.C$	The category set of ϕ
h	A category head of a Wikipedia entity
f_{i_n}	The frequency of h_{i_n} of a Wikipedia entity

2.3 WorkiNet generation

Given a Wikipedia entity ϕ which is not in WordNet, the following procedure is invoked to merge it into WordNet to construct an enhanced ontology, named as WorkiNet. The category information in Wikipedia can be used as clues to discover the higher level concepts. Unfortunately, the category names may not be exactly included in WordNet. We propose a two-phase strategy to generate higher-level concept candidates for a Wikipedia entity. After the candidates are selected, a context matching method is proposed to determine which candidate wins out.

2.3.1 Concept expansion

The design of our concept expansion algorithm is depicted in Algorithm 1. Given an entity ϕ, if $\phi.t$ is not contained by any of the concepts in WordNet, the concept expansion will be performed. First a candidate set Π^c which contains potential higher-level concepts of ϕ is constructed. Then based on the matching score between each candidate and ϕ, the most suitable one from Π^c is chosen as the hypernym of ϕ.

In the generation of Π^c, we employ a two-phase strategy. In the first phase, we retrieve the elements of $\phi.C$ in WordNet. If a certain concept π contains any element of $\phi.C$, π will be put into Π^c. After the first phase, if Π^c is empty, we proceed to the second phase. Utilizing the data structure *IECH*, the set of category heads of ϕ is obtained, denoted as $\{h_{i_1}, h_{i_2}, \cdots\}$. Then we retrieve these category heads in WordNet to compose Π^c. In this method, the original category names of ϕ are our first choice. If the category names are not in WordNet, we make use of the head terms of the

categories. For the sake of convenience, we use Π_1^c and Π_2^c to represent the candidate sets generated in Phase 1 and 2 respectively.

We use a running example to illustrate how a Wikipedia entity is processed. Consider the entity $\phi = \langle$"Windows Template Library", "a free software, object-oriented C++ template library ...", {"C++ libraries", "Free development toolkits and libraries", "Free software projects", "Microsoft application programming interfaces", "Object-oriented programming", "Windows-only free software"}$\rangle$[3]. Here $\phi.t$="*Windows Template Library*" is absent in WordNet and no element of $\phi.C$ is in WordNet either. Thus $\Pi_1^c = \emptyset$ in Phase 1. Then we turn to Phase 2. The *IECH* of ϕ is constructed as {library:1, toolkitslibrary:1, project: 1, interface:1, programming:1, software:1}. Among these category heads, "software" is contained by only one WordNet concept. And "project", "programming", "interface", "library" are contained by 2, 2, 4, 5 concepts respectively, while "toolkitslibrary" is absent in WordNet. Therefore, Π_2^c comprises the above 14 concepts.

After Π^c is generated, a matching score is used to determine which candidate is the most suitable one to be the hypernym of ϕ. The matching score between an entity and a concept is defined as

$$match(\phi, \pi_k) = \begin{cases} simT(\phi.d, \pi_k.g) & if\ \pi_k \in \Pi_1^c, \\ \log(f_{i_n} + 1) * (simT(\phi.d, \pi_k.g) + 1) & if\ \pi_k \in \Pi_2^c, \end{cases} \quad (1)$$

where f_{i_n} is the frequency of h_{i_n} ($h_{i_n} \in \pi_k.S$), and the $simT$ function is used to calculate the similarity between two text fragments, which will be discussed in detail in Section 2.3.2. Here we calculate the similarity between the description of ϕ and the gloss of π_k. The candidate whose matching score dominates others wins out and it is denoted as π^b.

We create a new concept π^ϕ with $\pi^\phi.S$ and $\pi^\phi.g$ setting to be $\{\phi.t\}$ and $\phi.d$ respectively. Finally, if Π_1^c is not empty, π^ϕ is attached under π^b directly. Otherwise, for each category $\phi.c$ whose lemmatized head term is contained in π^b, we create a new concept π^ψ with $\pi^\psi.S = \{c\}$. Then π^ϕ is attached under π^ψ, and π^ψ is added to the hyponym set of π^b.

Returning to the above running example, among the 14 candidates in Π^c, the concept π^b = $\langle$ {library, program library, subroutine library}, "(computing) a collection of standard programs...", $\emptyset\rangle$ in WordNet wins out. Here, "library" is the head term of "C++ libraries" which is one of the categories of ϕ. Therefore, we create a new concept $\pi^\psi = \langle$ {C++ libraries}, $NULL$, $\{\pi^\phi\}\rangle$, where π^ϕ is generated from ϕ and equals to $\langle$ {Windows Template Library}, "a free software,...", $\emptyset\rangle$. Here π^ϕ is in the hyponym set of π^ψ, which means π^ϕ is attached under π^ψ. And $\pi^b.H$ becomes $\{\pi^\psi\}$, which means π^ψ is attached under π^b.

In this way, we integrate Wikipedia entities into WordNet and generate the enhanced ontology WorkiNet. As mentioned above, we only integrate the Wikipedia entities whose titles are not in WordNet, thus the terms added into WorkiNet are guaranteed to be new. As for polysemants, different meanings of the same word are demonstrated in different articles with distinct titles in Wikipedia. For example, the computer company "Apple" is demonstrated in the article whose title is "Apple Inc"[4], while the title of the corresponding article of fruit "apple" is "Apple"[5]. Hence, we add new meanings of existing words as well.

2.3.2 Context based concept similarity

Leacock and Chodorow [15] propose the following formula to compute semantic similarity between two concepts:

$$sim_{LC}(\pi_1, \pi_2) = -log \frac{len(\pi_1, \pi_2)}{2 \times \max_{\pi \in WordNet} depth(\pi)} \quad (2)$$

where $len(\pi_1, \pi_2)$ is the length of the shortest path between concept π_1 and π_2 in WordNet and $depth(\pi)$ is the length of the path from π to the root. As noted by Sussna [26], sibling-concepts with larger depth appear to be more semantically related than the shallower ones. Based on this observation, we propose the following similarity calculation formula:

$$sim(\pi_1, \pi_2) = \frac{log \frac{len(\pi_1, \pi_2)}{depth(\pi_1) + depth(\pi_1)}}{log \frac{1}{2(\max_{\pi \in WordNet} depth(\pi) + 1)}}, \quad (3)$$

where the denominator is for normalization.

Given a text fragment Γ, we use the positive maximum matching method to segment Γ into different terms and each term is included in a certain concept in the ontology. For the words absent in the ontology, we treat each of them as a "pseudo-concept" with itself as synset. Suppose that two text fragments Γ_1 and Γ_2 cover n concepts in all, then they are represented as two n-dimension vectors v_1 and v_2. We design a semantic matrix as follows:

$$SM = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1n} \\ s_{21} & s_{22} & \dots & s_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ s_{n1} & s_{n2} & \dots & s_{nn} \end{pmatrix}, \quad (4)$$

where $s_{ij} =$

$$\begin{cases} 1 & if(\pi_i = \pi_j) \\ sim(\pi_i, \pi_j) & if((\xi(\pi_i) = 1 \wedge \xi(\pi_j) = 1) \wedge \pi_i \neq \pi_j) \\ 0 & if((\xi(\pi_i) = 0 \vee \xi(\pi_j) = 0) \wedge \pi_i \neq \pi_j) \end{cases}$$

in which $\xi(\pi_i)$ is an indicator, and takes value 1 if $\pi_i \in$ WordNet, otherwise it takes value 0. We calculate similarity between two text fragments Γ_1 and Γ_2 with the following formula:

$$simT(\Gamma_1, \Gamma_2) = \frac{v_1 \cdot SM \cdot v_2^T}{\sqrt{v_1 \cdot SM \cdot v_1^T} \sqrt{v_2 \cdot SM \cdot v_2^T}}. \quad (5)$$

One interesting property is that when SM is set to an identity matrix, $simT(\Gamma_1, \Gamma_2)$ becomes the cosine similarity between v_1 and v_2.

Algorithm 1: Concept expansion for WorkiNet generation

input : Wikipedia Φ, WordNet ontology O
output: The enhanced ontology WorkiNet
foreach $\phi \in \{\phi' | \phi' \in \Phi \wedge \nexists \pi(\phi'.t \in \pi.S \wedge \pi \in O)\}$ **do**
  concept set $\Pi_1^c \leftarrow \emptyset$, $\Pi_2^c \leftarrow \emptyset$, $\Pi^b \leftarrow \emptyset$
  /* *Phase 1* */
  foreach $c \in \phi.C$ **do**
    $\Pi_1^c \leftarrow \Pi_1^c \bigcup \{\pi | c \in \pi.S \wedge \pi \in O\}$
  /* *Phase 2* */
  if $\Pi_1^c = \emptyset$ **then**
    construct $IECH\ idx$ for ϕ
    foreach $h_{i_n} \in idx$ **do**
      $\Pi_2^c \leftarrow \Pi_2^c \bigcup \{\pi | h_{i_n} \in \pi.S \wedge \pi \in O\}$
  foreach $\pi_k \in \Pi_1^c \bigcup \Pi_2^c$ **do**
    calculate $match(\phi, \pi_k)$
  $\pi^b = \arg\max_{\pi_k \in \Pi_1^c \bigcup \Pi_2^c} macth(\phi, \pi_k)$
  create π^ϕ, $\pi^\phi.S \leftarrow \{\phi.t\}$, $\pi^\phi.g \leftarrow \phi.d$
  if $\pi^b \in \Pi_1^c$ **then**
    $\pi^b.H \leftarrow \pi^b.H \bigcup \{\pi^\phi\}$
  else foreach *c whose lemmatized head term is in* π^b **do**
    create π^ψ, $\pi^\psi.S \leftarrow \{c\}$
    $\pi^b.H \leftarrow \pi^b.H \bigcup \{\pi^\psi\}$, $\pi^\psi.H \leftarrow \pi^\psi.H \bigcup \{\pi^\phi\}$

[3] http://en.wikipedia.org/wiki/Windows_Template_Library
[4] http://en.wikipedia.org/wiki/Apple_Inc.
[5] http://en.wikipedia.org/wiki/Apple

3. TAILOR-MADE CONCEPT REPRESENTATION LEARNING MODEL

We design a Tailor-made Concept Representation Learning (TCRL) model that can adaptively generate tailor-made semantic concept set for a given document collection. As shown in Figure 2, given a document collection and an ontology such as WordNet or WorkiNet, the algorithm learns which concepts have better capability of discriminating the documents and generates a tailor-made collection $\mathcal{C}$, which is composed of all the concepts denoted by hollow circles. Each concept in $\mathcal{C}$ is generated by aggregating several concepts and part of their descendants. Some of the concepts in $\mathcal{C}$ encapsulate all of the descendants shown in the original ontology graph (located on the dashed line). For example, the terms in π_1 are added into $\pi_a.S$. Then we get a derived concept, π^{d1}, of π_a in the collection. π_2 and π_3 with all the descendants of π_3 in its subtaxonomy are merged into a new concept π^{d2}. The terms in π_4 and π_5 are added into $\pi_b.S$ and form a new concept π^{d3}.

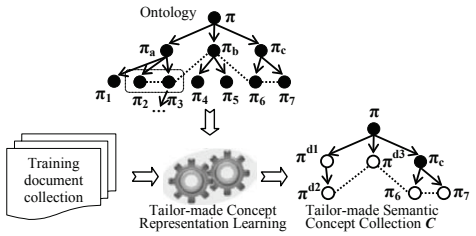

Figure 2: Semantic Concept Representation Learning.

$\mathcal{C}$ represents a semantic summary of the given domains, and it is generated specifically for a given document collection. As a result, each document collection has its own tailor-made semantic concept set. The tailor-made semantic concept set will be used as a new feature space for representing the documents. This new feature space is more effective when a text mining task, such as classification or clustering, is conducted on the corresponding document collection.

3.1 Concept merging

Suppose in the given document collection, the documents have been organized in classes. To select features with good capability of discriminating the documents in different classes, we employ the mutual information (MI) between a term t and a class c_i, denoted as $MI(t, c_i)$, to measure how much information the presence/absence of t contributes to make the correct class prediction on c_i. Formally:

$$MI(t, c_i) = MI(\varepsilon_t, \varepsilon_{c_i}) = \sum_{e_t \in \{0,1\}} \sum_{e_{c_i} \in \{0,1\}} p(\varepsilon_t = e_t, \varepsilon_{c_i} = e_{c_i}) log \frac{p(\varepsilon_t = e_t, \varepsilon_{c_i} = e_{c_i})}{p(\varepsilon_t = e_t) p(\varepsilon_{c_i} = e_{c_i})} \quad (6)$$

where ε_t and ε_{c_i} are two random variables that take the value of 1 or 0. $\varepsilon_t = 1$ denotes the document contains term t and $\varepsilon_t = 0$ denotes the document does not contain term t. Analogously, $\varepsilon_{c_i} = 1$ denotes the document is in class c_i and $\varepsilon_{c_i} = 0$ denotes the document is not in class c_i.

Suppose in the given document collection, there is no pre-defined class information. In this case, we calculate the mutual information between term t and an individual document d_i: $MI(t, d_i) = MI(\varepsilon_t, \varepsilon_{d_i})$. Random variable $\varepsilon_t = 1$ denotes the random selected term is t and random variable $\varepsilon_{d_i} = 1$ denotes the random term is in document d_i.

To generate the tailor-made semantic concept collection, the leaf concepts are merged into their hypernyms recursively, and an indicating function is used to measure whether the merging is profitable. We assume that the more unbalanced the distribution of MI value is, the more plausible it is to add terms into the concept collection. We define the indicating function $I(\pi)$ based on the inequality of the distribution.

3.2 Indicating function

An indicating function is used to indicate whether the merging should be performed, and it is based on the inequality of the distribution of MI values. Any effective method of measurement can be adopted. In this paper, we design two different patterns of the indicating function, namely, Gini coefficient based function and entropy-based function.

3.2.1 Gini indicating function

The Gini [8] coefficient is a measure of the imparity of a distribution which is firstly proposed by Corrado Gini in 1912 and is widely used in various fields such as economics, geography, etc. The Gini coefficient is defined based on the Lorenz curve. Its geometric significance is the proportion of the area between the Lorenz curve of the distribution and the uniform distribution line to the area under the uniform distribution line. The Gini coefficient ranges from 0 to 1, and a lower value indicates a more equal distribution while a higher one implicates more inequality. A number of formulas have been provided for the calculation of the Gini coefficient. In this paper, we take the method proposed by Chotikapanich and Griffiths [5]. Their work suggests an estimator obtained by approximating the Lorenz curve with a series of linear segments.

Supposing that MI values of a particular term set τ have been classified into M groups (for classifying task, each MI value forms an individual group). In the next step, we get the average MI value of each group, then sort them in ascending order. The following information is available for the i-th group:

- The proportion of terms: p_i
- Cumulative proportion of terms: $\varrho_i = p_1 + p_2 + \ldots + p_i$
- The proportion of MI_i: $\vartheta i = \frac{p_i MI_i}{\sum_{j=1}^{M} p_j MI_j}$

Then the coefficient value can be calculated as:

$$G(\tau, v) = 1 + \sum_{i=1}^{M} \frac{\vartheta i}{p_i} [(1 - \varrho_i)^v - (1 - \varrho_{i-1})^v] \quad (7)$$

where v is an inequality aversion parameter. $G(\tau, v)$ is defined for $v > 1$ and is equal to the original Gini coefficient when $v = 2$.

The indicating function based on Gini coefficient is defined as:

$$I_{\_Gini}(\pi) = \frac{Gini(\pi)}{\sum_{\pi' \in \pi.H} Gini(\pi') freq(\pi')} \quad (8)$$

where $Gini(\pi)$ is defined as:

$$Gini(\pi) = G(S_\pi^g, v) \quad (9)$$

$S_\pi^g = \pi.S \cup_{\pi' \in \Pi_\pi^d} \pi'.S$, and Π_π^d is the descendant set of π. And $freq(\pi')$ is defined as:

$$freq(\pi') = \frac{count(S_{\pi'}^g)}{\sum_{\pi_i' \in \pi.H} count(S_{\pi_i'}^g)}, \quad (10)$$

where $count(S_{\pi'}^g)$ is the cumulative frequency of π' and all the concepts in its subtaxonomy.

3.2.2 Entropy indicating function

Entropy is another common method to measure the degree of uniformity of a distribution. We define our entropy-based indicating function as follows:

$$I_{\_Entro}(\pi) = \frac{\sum_{\pi' \in \pi.H} entr(\pi') freq(\pi')}{entr(\pi)}, \quad (11)$$

where $entr(\pi)$ is defined as:

$$entr(\pi) = -\sum_{i=1}^{M} \widetilde{MI}(S_\pi^g, c_i) \cdot log(\widetilde{MI}(S_\pi^g, c_i)), \quad (12)$$

Here, $\widetilde{MI}(S^g_\pi, c_i)$ is a normalized MI value:

$$\widetilde{MI}(\pi, c_i) = \frac{MI(\pi, c_i)}{\sum_{j=1}^{M} MI(\pi, c_j)}, \quad (13)$$

where M is the number of classes.

For the convenience of explanation, $I_{\_Gini}$ and $I_{\_Entro}$ are uniformly referred to as I. The larger $I(\pi)$ is, the less noise is involved when merging the hyponyms of π into π. If $I(\pi) \geqslant \theta$ (a profitable threshold), the merging will be performed.

3.3 Semantic concept representation learning algorithm

The semantic concept representation learning algorithm is summarized in Algorithm 2. The key idea of the algorithm is to recursively merge leaves to their parent node if the merging is profitable. The merged node will be treated as a new leaf.

We introduce a border flag flg for each concept in the ontology. In the beginning, flg is initialized to be $false$ for each concept. Then in each loop, we get the unhandled deepest leaf π^l, that is, the deepest leaf with flg of $false$, and locate its hypernym π.

If the merging process stops in any branch of π, the root of the branch will be recognized as a non-leaf node. If π contains both non-leaf and leaf hyponyms, the merging will not be performed and flg of all the leaf hyponyms of π will be set to be $true$. When all the hyponyms of π are leaves, we will check if the merging is profitable. As long as $I(\pi) \geqslant \theta$, the merging will be performed.

Primarily, we gather leaves sharing high mutual information between each other into the same subset which is known as a "camp" in this paper. The camps and the remaining concepts compose a partition of the set $\pi.H$. Ideally, we should detect the optimal partition with the highest summation of mutual information values in every camp. However, to obtain the optimal solution is very costly due to the exponential number of possible subset partitions. Therefore, a heuristic method is adopted. First for each π'_i in $\pi.H$, $MI(\pi'_i, \pi)$ can be derived from Equation 6 and equals to $MI(\varepsilon_{\pi'_i}, \varepsilon_\pi)$. The random variable $\varepsilon_\pi = 1$ denotes that the document contains at least one term included in $\pi.S$, and $\varepsilon_{\pi'_i}$ is defined analogously. Let $\pi'_m = \arg\max_{\pi'_j \in \pi.H \wedge \pi'_j \neq \pi'_i} MI(\pi'_i, \pi'_j)$. If $MI(\pi'_i, \pi'_m) > MI(\pi'_i, \pi)$, π'_i and π'_m will be put into the same camp. After the camps are built, we pack each camp as a new node with $flg = true$ encapsulating all the concepts in it and their descendants, and attach the node under π. Then we merge the remaining concepts belonging to none of the camps and all of their descendants into π, and set the flg value of π to be $true$.

As a matter of fact, the camps are designed to redivide the concept set so that closely related concepts can be treated as one dimension. It will help the tailor-made semantic concept collection discriminate the documents better. However, if the new nodes generated from the camps are considered as new leaves, in the next loop, π will be recognized as non-leaf nodes and the merging process will stop for all the ancestors of π. To make sure that the merging process go on recursively, in the final step, π with all its child nodes are considered as a virtual leaf with flg=$false$.

If the merging is not profitable, all flg of π's leaf hyponyms will be set to $true$. Then the merging process will stop for all the ancestors of π. To restart the merging process, a non-leaf node whose child nodes are all leaves with $flg = false$ should be found. Finally, $\mathcal{C}$ is composed of all leaves (including the members of virtual leaves) with $flg = true$.

Let us return to the example illustrated in Figure 2. On the condition of $MI(\pi_2, \pi_3) > MI(\pi_2, \pi_a) > MI(\pi_1, \pi_2)$, π_2 and π_3 are put into the same camp. If we get $MI(\pi_1, \pi_3) < MI(\pi_1, \pi_2) < MI(\pi_2, \pi_3) \leqslant MI(\pi_1, \pi_a)$, π_1 will not be included by any camp. Then a new node π^{d2} is generated to encapsulate π_2, π_3 and all the descendants of π_3. The remaining concept π_1 is merged into π_a to form a new concept π^{d1}. π^{d1} and π^{d2} will be treated as a virtual leaf. $MI(\pi_4, \pi_5) \leqslant MI(\pi_4, \pi_b)$ and $MI(\pi_4, \pi_5) \leqslant MI(\pi_5, \pi_b)$, because there is no camp built for π_b, all the three nodes are merged into π^{d3}. The merging is not performed on π_c because $I(\pi_c)$ is less than θ. As a result, in the next loop, π contains the non-leaf hyponym π_c, so the merging process stops. If π_6 and π_7 are merged into π_c, all the child nodes of π will be treated as virtual leaves with flg=$false$. As long as $I(\pi) \geqslant \theta$, the merging will be performed on π recursively.

With an elaborately designed indicating function, a tailor-made concept collection can be selected to construct a better document representation. This selection is adaptive according to different document collections. For example, in our experiments, the concept π^e with synset {blue sky, blue, blue air, wild blue yonder} is merged into its grandparent concept with synset {atmosphere} for one document collection, while it is merged into the parent concept with synset {sky} for another. In the first case, terms in {atmosphere, sky, mackerel sky} $\bigcup \pi^e.S$ are treated as the same dimension. While in the later one, "atmosphere" is separated from others, because the grandparent concept $\pi^g = \langle$atmosphere,"the envelope of gases surrounding any celestial body",{"sky"}$\rangle$ has $I(\pi^g) < \theta$, and the merging is not performed.

3.4 Utilizing semantic concept for document representation

After the tailor-made semantic concept collection $\mathcal{C}$ is generated, we can utilize it to represent a document with a concept-based vector. Suppose t is a term from a particular document. If it is an unambiguous term, we map it to the unique concept in $\mathcal{C}$ that contains t. Otherwise, t is contained by more than one concept, and hence we need to disambiguate it first. For each π_i ($t \in \pi_i.S$) we calculate the similarity between t's context and the gloss of π_i according to Equation 5. t's context is composed of the surrounding sentences of t and denoted as Γ_t, while the gloss of π_i is denoted as $\pi_i.g$. Then the concept

$$\pi^m = \arg\max_{\pi_i} simT(\pi_i.g, \Gamma_t) \quad (14)$$

is selected as the correct concept of t. After the disambiguating, the reconstructed vectors can be applied in traditional vector space model for text mining tasks.

To evaluate the disambiguation approach, we select 60 common-used polysemants. On average, each of them is contained by 6.23 concepts in WordNet. We search these words in Wikipedia and collect the corresponding articles. Each article refers to a particular meaning of the related word. Then one article is randomly selected for each word, and the first sentence that contains the word in the article is extracted as the context of the word.

For instance, the term "orange" appears in 5 concepts in WordNet. It refers to "Orange (fruit)", "Orange (colour)" and "Orange (word)" in Wikipeida[6]. Then if we choose the meaning of orange color, and the context will be "The colour orange occurs between red and yellow in the visible spectrum at a wavelength of about $58 - 620$ nm, and has a hue of 30° in HSV colour space."[7]. After disambiguating as described above, it is mapped to concept $\langle$\{orange, orangeness\}, "orange color or pigment; any of a range of colors between red and yellow", \{reddish orange\}$\rangle$, which is the correct meaning of the word "orange" in the context. Overall, we achieve a disambiguation precision of 66.67% in our experiment.

[6]http://en.wikipedia.org/wiki/Orange

[7]http://en.wikipedia.org/wiki/Orange_(colour)

Algorithm 2: Semantic Concept Representation Learning Algorithm

input : the HDAG G of an ontology, a document collection
output: tailor-made semantic concept collection $\mathcal{C}$
assign each concept a flag flg with $false$
while *G has leaves with $flg = false$* **do**
  get the deepest leaf π^l with flg=$false$ and its hypernym π
  if *π has non-leaf hyponym concepts* **then**
    set_flag(π)
  else if $I(\pi) \geqslant \theta$ **then**
    merge_descendants(π)
  else *set_flag*(π)
set $\mathcal{C} = \{\pi | flg\ of\ \pi\ is\ true\}$

proc *set_flag*(π)
foreach *π' in $\pi.H$* **do**
  if $|\pi'.H| = 0$ **then**
    set $flg \leftarrow true$ for π'

proc *merge_descendants*(π)
create $Camps$
foreach *π'_i in $\pi.H$* **do**
  $\pi'_m \leftarrow \arg\max_{\pi'_j \in \pi.H \wedge \pi'_j \neq \pi'_i} MI(\pi'_i, \pi'_j)$
  if $MI(\pi'_i, \pi'_m) > MI(\pi'_i, \pi)$ **then**
    if $\exists c(c \in Camps \wedge (\pi'_m \in c \vee \pi'_i \in c))$ **then**
      $c \leftarrow c \bigcup \{\pi'_m, \pi'_i\}$
    else create $c \leftarrow \{\pi'_m, \pi'_i\}$
    $Camps \leftarrow Camps \bigcup \{c\}$
if $Camps \neq \emptyset$ **then**
  foreach *c in $Camps$* **do**
    create π_c with $flg = true$
    foreach *π_k in c* **do**
      $\pi_c.S \leftarrow \pi_c.S \bigcup S^g_{\pi_k}$
      $\pi.H \leftarrow \pi.H - \bigcup \{\pi_k\}$
    $\pi.H \leftarrow \pi.H \bigcup \{\pi_c\}$
foreach $\pi' \in \{\pi'_i | \nexists c(c \in Camps \wedge \pi'_i \in c)\}$ **do**
  $\pi.S \leftarrow \pi.S \bigcup S^g_{\pi'}$
  $\pi.H \leftarrow \pi.H - \bigcup \{\pi'\}$
set $flg \leftarrow true$ for π
pack π and all nodes in $\pi.H$ as a virtual leaf with flg=$false$

4. EXPERIMENTS

4.1 Information about data and resource

We use version 2.1 of WordNet, which contains 89,646 concepts. The Wikipedia data used is a dump in October 2009. After eliminating the articles without category information or valid article content, 2,722,437 articles are collected. Finally, 1,782,276 new leaves are added into 12,888 WordNet concepts. 57,573 of the new leaves are added into WordNet concepts directly and 1,724,703 of them are added into new concepts built from the categories, which are then attached under WordNet concepts containing the head terms of the categories.

4.2 Case study in WorkiNet

Table 2 presents some examples about under which concept a Wikipedia entity is attached. The entities are given in the second column, and their expanded WordNet concepts (denoted by the corresponding synsets) are shown in the fourth column. The third column, "Added Category", denotes the category of the Wikipedia entity. If the cell in "Added Category" is not empty, the Wikipedia entity will be attached directly to category first, and then this category is subsumed by the WordNet concept in the fourth column.

Table 2: Case study 1.

#	Wikipedia Entity	Added Category	WordNet Concept
1	dekopon		citrus, citrus fruit, citrous fruit
2	J2EE application	java platform	platform
3	Proxi		freeware
4	Thenar eminence		hand
5	Yahoo! Meme	real-time web	World Wide Web, WWW, web
6	Conditional entropy	selective information	information, entropy,
7	Standard Template Library	generic programming	program, programme, computer program, computer programme

Table 3: Case study 2.

#	WordNet Concept	Wikipedia Entities and category(if added)	
1	black tea	Golden needle tea; Nepal tea; Nilgiri tea; Tibeti;	
2	thermochemistry	Principle of maximum work; Heat of formation group additivity; Isodesmic reaction; Nernst heat theorem; Exergonic reaction; Endergonic reaction; Van't Hoff equation; Thermochemical equation; Thomsen-Berthelot principle	
3	Supercomputer	NEC supercomputers	SUPER-UX
		IBM supercomputers	PERCS; ACS-1; IBM Roadrunner; Blue Gene; Blue Waters; Cyclops64

We can see that reasonable higher level concepts for the entities are found. "J2EE application" is recognized as a software platform and is subsumed by the WordNet concept $\langle$\{platform\}, "the combination of a particular computer and a particular operating system", $\emptyset\rangle$. We create a new hypernym for it from its category, namely "java platform". A Japanese fruit "dekopon", which is a rare term and cannot even be found in many dictionaries, is retrieved as well.

Table 3 gives some examples from another perspective to show the consistency of the semantic meaning of the entities attached under the same concept. In the first example, different kinds of black tea are supplemented. In the second example, chemistry-related phrases are added to "thermochemistry". In the final one, we find 7 examples of supercomputers. All the 6 supercomputers of IBM supercomputers are reasonable, while SUPER-UX is actually the version of the Unix operating system used on NEC SX architecture supercomputers. In spite of some inaccurate information, WorkiNet still digs out quite abundant useful ontology information.

4.3 Quality evaluation of WorkiNet

For evaluation purpose, we define five grades to score the correctness of the semantic relation between a Wikipedia entity and its hypernyms (including the category if added) in WorkiNet. Examples are given in Table 4. In **Excellent** and **Good**, the matching quality is high, and these cases agree with people's general knowledge. In **Fair**, the relation between the entity and the concept is not strong, but still reasonable. If the quality is difficult to judge, we put it into **Neutral**. Finally, the wrong cases are put into **Bad**. Five volunteers have been invited to conduct the evaluation. For a cer-

tain case, each volunteer gives a score, then we average all scores and round the average score to its nearest grade as the final result.

Table 4: Evaluation grades.

Grade (Score)	Examples (Wiki entity (⇒ category) ⇒ WordNet concept)
Excellent(5)	Irisbus⇒bus manufacturers⇒manufacturer producer; Guantĺcnamo Bay ⇒ bay, embayment
Good(4)	The Moorcock ⇒ English contract case law⇒ law, practice of law; Helpless Romantic (album) ⇒ Jon B. albums ⇒ album
Fair(3)	Red Fraction ⇒ 2006 singles ⇒ single, bingle Services cricket team ⇒ Military
Neutral(2)	Log-a-Log ⇒ Redwall characters ⇒ character, eccentric, type, case; Stewart v. Abend ⇒ United States Supreme Court cases ⇒ sheath, case
Bad(1)	Dwight Dickinson⇒1916 births⇒birth; Deferipr-one ⇒ Chelating agents ⇒ agent, factor, broker

We randomly sample 300 cases and the evaluation result is given in Table 5. The overall average score is 4.26 and 80% of the cases fall into **Excellent** or **Good**. We examine the **Bad** and **Neutral** cases, and find that a main reason for the mis-attaching is that the support of the article's content to the most profitable head term is not dominant. Take "Dwight Dickinson⇒1916 births⇒birth" as an example, "Dwight Dickinson" was a United States diplomat and Navy veteran, but the entity is wrongly categorized into "1916 births" because the matching score of the intended category "officer" (the head term of the original category "United States Navy officers") is lower than that of "birth". Another factor is that $simT$ function does not work well for some cases. "Deferiprone" is a chelating agent, but it is wrongly connected with the concept "agent, factor, broker" because the $simT$ value of this concept is bigger than that of any other concept that contains "agent". Here "agent" is the intended category head, but the WordNet concept selected is not correct.

4.4 WordNet and WorkiNet's performance in text mining

We investigate whether the TCRL model can outperform a static one by conducting experiments on clustering and classification. A previous method denoted as "Hotho" [12] is also implemented for conducting the comparison. In this method, each synonym set, with a fixed number of levels of hypernyms added, is used as one dimension in the vector space. As shown in [12, 13], "Hotho" can dominate BOW approach in different text mining tasks. We also conduct a comparison between WorkiNet and WordNet to see whether WorkiNet can achieve competitive performance or even better under the TCRL model.

4.4.1 Document collections

Two data sets are used in the experiments: 20 Newsgroups (NG20)[8] and TREC 2004 (TREC) data extracted from the document collection Disc 5. All the 20 groups of NG20 are used, while the biggest 10 categories are used in TREC.

4.4.2 Document classification

In this experiment, LibSVM [4] with a linear kernel is employed. The threshold θ is set to 1.2.

Evaluation criteria. Two types of the F-measure scores are computed, namely, macro-average and micro-average [28].

In macro-average, the precision and recall for each category c_i are calculated first, denoted as P_i and R_i. Then F-measure for c_i is: $F_i = \frac{2P_iR_i}{P_i+R_i}$. The macro-averaged F-measure is the average of F-measure for each category: $F^{ma} = \frac{\sum_i F_i}{|C|}$. In micro-averaging, F-measure is computed globally over all category decisions. The global precision and recall are calculated as: $P = \frac{\sum_i TP_i}{\sum_i (TP_i+FP_i)}$, $R = \frac{\sum_i TP_i}{\sum_i (TP_i+FN_i)}$, where TP_i, FP_i and FN_i are true positive, false positive and false negative numbers for category c_i. Micro-averaged F-measure is then computed as: $F^{mi} = \frac{2PR}{P+R}$.

[8]http://people.csail.mit.edu/jrennie/20Newsgroups/

Table 5: Evaluation result.

Excellent	Good	Fair	Neutral	Bad
193	47	26	12	22

Table 6: Classification performance comparison.

		WorkiNet		WordNet	
		F^{mi}	F^{ma}	F^{mi}	F^{ma}
NG20	Hotho	.854	.748	.875	.756
	Gini	.881	.771	.893	.764
	Entro	.883	.773	.893	.764
TREC	Hotho	.591	.542	.498	.450
	Gini	.649	.589	.604	.556
	Entro	.648	.588	.603	.555

Results. The classification result is given in Table 6. "Gini" and "Entro" denote the performance of TCRL with Gini and entropy indicating function respectively. We can see that TCRL outperforms Hotho under both evaluation metrics, no matter which ontology is used. WorkiNet performs better than WordNet in TREC but a slight decline is found in NG20. One direct cause is that the dimension is over reduced by TCRL when WorkiNet is employed. Too many concepts are merged so that the capability of discriminating documents in the new feature space is weakened. The difference between the performance of WordNet and WorkiNet on NG20 is not significant for both F^{ma} and F^{mi}. It may be due to the insensitivity of NG20 to new words.

4.4.3 Document clustering

In this experiment, we run K-Means for three times on each data set to get an average result. The adopted value of θ is 0.6.

Evaluation criteria. The purity measure is employed to evaluate the clustering performance. Let $G = \{g_1, g_2, \cdots\}$ denote the cluster set generated by K-Means, and $C = \{c_1, c_2, \cdots\}$ denote the pre-defined clusters. To compute the purity, each g_k is assigned to the pre-defined cluster c_i which is the most frequent one in g_k, then the purity is measured as:

$$Pur(C, G) = \frac{1}{|D|} \sum_k \max_i |g_k \cap c_i|, \quad (15)$$

Results. The clustering result is given in Table 7. It can be seen that our adaptive TCRL model works better than the static one significantly. On average, TCRL improves the results by more than 49% for both Gini and entropy indicating function. Moreover, WorkiNet outperforms WordNet.

4.5 Parameter analysis in TCRL

The effects of the threshold θ under different settings are shown in Table 8, Table 9 and Table 10. "AVG" is the average value of the results of TCRL with different θ and "SD(/100)" is the standard deviation between these values divided by 100. We can see that in the classification task, the TCRL model is not sensitive to θ. It is because the number of classes is so small that the distribution is too sparse. In the clustering, the TCRL model is a bit more sensitive, but the difference is still inapparent. For both classification and clustering, performances of Gini and entropy indicating function are similar. From the values of the standard deviation, we can see

Table 7: Clustering performance comparison.

		WorkiNet	WordNet
NG20	Hotho	.601	.696
	Gini	.853	.815
	Entro	.852	.824
TREC	Hotho	.332	.291
	Gini	.543	.509
	Entro	.547	.507

that the performance of Gini indicating function is a bit more stable under different values of θ, especially for classification. Overall, the TCRL model is not sensitive to θ in our experiments.

In general, WorkiNet outperforms WordNet. However, WorkiNet's improvement of classification in NG20 is not significant. It implies that WorkiNet contributes little given an old data set.

Table 8: Parameter θ's effect in classification(Gini indicating function).

θ	NG20				TREC			
	F^{mi}		F^{ma}		F^{mi}		F^{ma}	
	WD	WK	WD	WK	WD	WK	WD	WK
Hotho	.875	.854	.756	.748	.498	.591	.450	.542
0.2	.890	.876	.763	.767	.603	.626	.554	.570
0.4	.890	.876	.763	.767	.602	.628	.553	.571
0.6	.890	.876	.763	.767	.604	.626	.555	.570
0.8	.890	.881	.763	.771	.604	.631	.555	.573
1.0	.890	.881	.763	.771	.602	.649	.554	.589
1.2	.893	.881	.764	.771	.604	.649	.555	.589
AVG	.891	.879	.763	.769	.603	.635	.554	.577
SD(/100)	.122	.274	.041	.219	.098	1.11	.082	.936

Table 9: Parameter θ's effect in classification(entropy indicating function).

θ	NG20				TREC			
	F^{mi}		F^{ma}		F^{mi}		F^{ma}	
	WD	WK	WD	WK	WD	WK	WD	WK
Hotho	.875	.854	.756	.748	.498	.591	.450	.542
0.2	.887	.868	.759	.760	.600	.599	.551	.545
0.4	.887	.868	.759	.760	.600	.599	.551	.545
0.6	.887	.868	.759	.760	.601	.600	.552	.547
0.8	.887	.872	.759	.763	.600	.640	.551	.582
1.0	.893	.876	.764	.767	.603	.644	.554	.585
1.2	.893	.883	.764	.773	.603	.648	.555	.588
AVG	.889	.873	.761	.764	.601	.622	.551	.565
SD(/100)	.310	.606	.258	.527	.147	2.46	.362	2.16

5. RELATED WORK

Researchers have conducted extensive work on automatic ontology generation and enrichment. The work in [2] extends WordNet with syntagmatic information by adding information about the co-occurrence of word meaning in texts. Similar work focusing on WordNet and Wikipedia is reported by Ruiz-Casado et al. [22]. Ponzetto and Navigli [20] present a method to link WikiTaxonomy to WordNet which maximizes the structural overlap between WordNet's taxonomy and Wikipedia's Category by disambiguating with WordNet hierarchy. Ferrández's method [1] presents an algorithm to align FrameNet units to WordNet synsets by exploiting the particular traits of each hierarchy.

In [27] a document-specific hierarchy is automatically extracted from the text and combined with WordNet relations to build an extended WordNet hierarchy, which is trimmed with a disambiguation method. A *head-modifier* relation is used for the hierarchy building. Magnini and Speranza [16] propose an approach to integrate generic and specialized wordnet-like lexical database with *plug-in* relation.

Table 10: Parameter θ's effect in clustering.

	Gini				Entropy			
	NG20		TREC		NG20		TREC	
θ	WD	WK	WD	WK	WD	WK	WD	WK
Hotho	.696	.601	.291	.332	.696	.601	.29	.332
0.2	.818	.852	.511	.548	.815	.852	.502	.539
0.4	.820	.851	.510	.544	.812	.848	.510	.547
0.6	.815	.853	.509	.543	.824	.852	.507	.547
0.8	.802	.855	.508	.542	.813	.850	.509	.541
1.0	.813	.849	.508	.544	.813	.855	.511	.544
1.2	.813	.851	.501	.541	.813	.842	.507	.546
AVG	.814	.852	.508	.544	.815	.850	.508	.544
SD(/100)	.629	.204	.354	.242	.452	.449	.320	.335

YAGO [25] mainly focuses on finding the "individuals" and "facts" from Wikipedia. The authors try to extract 14 kinds of relations, and WordNet is utilized to help them to generate *subClassOf* and *means* relations. In [19], the authors present a method to automatically create an independent lexical ontology based on terms from a semantic network and their ontology has a similar model with WordNet.

As an important semantic bank, WordNet has been used to improve the performance of clustering and classification. Scott and Matwin [23] successfully integrate the WordNet resource for classification. Hotho et al. [11, 12] incorporate synonym and hypernym as background knowledge into document representation. Shehata [24] analyzes each term at sentence-level and improves clustering using terms and the corresponding synonyms and hypernyms. In comparison, both approaches in [12] and [24] provide static feature analyses to all document collections, while ours agilely construct semantic concept collections for different document collections according to their own characteristics. Jing et al. [14] construct a term similarity matrix using WordNet to improve text clustering. Their approach only uses synonyms and hyponyms, but fails to handle polysemy. In Recupero's work [21], two strategies, namely, WordNet lexical categories (WLC) technique and WordNet ontology (WO) technique, are proposed to create a new vector space.

In our previous work [3], we propose two concrete methods, namely, top-down (GBG-g) and bottom-up (GBG-s) algorithms for document presentation. In this paper, we focus on proposing a general framework for concept granularity learning. Compared with merging all the descendants to their roots, the construction of camps helps to redivide the concept set so that the granularity can be learned more precisely and adaptively. With the general framework of merging and redividing, the algorithms can be designed to be more efficient. For instance, the camp building method and the indicating function can be flexibly modified as long as they can help to learn the concept granularity better.

6. ACKNOWLEDGMENT

We thank the anonymous reviewers for their valuable and constructive comments. This work is supported by NSFC with Grant No. 61073081, HGJ with Grant No. 2011ZX01042-001-001, the Research Grant Council of the Hong Kong Special Administrative Region, China (Project Codes: CUHK4128/07 and CUHK413510). This work is also affiliated with the Microsoft-CUHK Joint Laboratory for Human-centric Computing and Interface Technologies.

7. CONCLUSIONS

In this paper, we construct an enhanced ontology, WorkiNet, by integrating the information of Wikipedia into the expert-edited on-

tology WordNet. In general, the performance of WorkiNet in text mining applications is very encouraging. We have also proposed an adaptive model called TCRL to reconstruct document presentation. The efficacy of the TCRL model is investigated in classification and clustering, and the experimental results show that it outperforms a static one no matter which indicating function is used.

The framework of the TCRL model offers opportunities to further optimize the task of feature selection. Indicating functions can be flexibly modified as long as it is effective to conduct feature selection. In this paper we propose two different types of indicating functions, and both of them perform well. And the camp building algorithm can be further improved as well. Another possible future direction is to further improve the accuracy of WorkiNet generation, and solve the problem of mis-attaching.

8. REFERENCES

[1] *Ó*. Ferrández, M. Ellsworth, R. Muñoz, and C. F. Baker. Aligning framenet and wordnet based on semantic neighborhoods. In *LREC '10: Proceedings of the 7th Conference on International Language Resources and Evaluation*, pages 310–314, 2010.

[2] L. Bentivogli and E. Pianta. Extending wordnet with syntagmatic information. In *Proceedings of Second Global WordNet Conference*, pages 47–53, 2004.

[3] L. Bing, B. Sun, S. Jiang, Y. Zhang, and W. Lam. Learning ontology resolution for document representation and its applications in text mining. In *CIKM '10: Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, pages 1713–1716, 2010.

[4] C. C. Chang and C.J.Lin. *LIBSVM: a library for support vector machines*. 2001.

[5] D. Chotikapanich and W. Griffiths. On calculation of the extended gini coefficient. *Review of Income and Wealth*, pages 541–547, 2001.

[6] S. Deerwester, S. Dumais, G. Furnas, T.K.Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[7] C. Fellbaum. *WordNet: An Electronic Lexical Database*. The MIT Press, Cambridge, MA, 1998.

[8] C. Gini. Variabilità e mutabilità. *Bologna: Tipografia di Paolo Cuppini*, 1912.

[9] J. Gonzalo, F. Verdejo, I. Chugur, and J. Cigarrán. Indexing with wordnet synsets can improve text retrieval. In *Proceedings ACL/COLING Workshop on Usage of WordNet for Natural Language Processing*, pages 38–44, 1998.

[10] T. Hofmann. Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 50–57, 1999.

[11] A. Hotho, A. Maedche, and S. Staab. Ontology-based text document clustering. *Künstliche Intelligenz*, 4:48–54, 2002.

[12] A. Hotho, S. Staab, and G. Stumme. Wordnet improves text document clustering. In *Proceeding of the SIGIR 2003 Semantic Web Workshop*, 2003.

[13] X. Hu, N. Sun, C. Zhang, and T.-S. Chua. Exploiting internal and external semantics for the clustering of short texts using world knowledge. In *CIKM '09: Proceedings of the 18th International Conference on Information and Knowledge Management*, pages 919–928, 2009.

[14] L. Jing, L. Zhou, M. K. Ng, and J. Z. Huang. Ontology-based distance measure for text clustering. In *Proceedings of the Text Mining Workshop, SIAM International Conference on Data Mining*, 2006.

[15] C. Leacock and M. Chodorow. Combining local context and wordnet similarity for word sense identification. *Christiane Fellbaum, editor, WordNet: An Electronic Lexical Database*, pages 265–283, 1998.

[16] B. Magnini and M. Speranza. Integrating generic and specilized wordnets. In *RANLP: Recent Advances in Natural Language Processing*, pages 149–153, 2001.

[17] D. M.Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[18] J. Morato, M. *Á*. Marzal, J. Lloréns, and J. Moreiro. Wordnet applications. In *GWC '04: Proceedings of the Second Global Wordnet Conference*, 2004.

[19] H. G. Oliveira and P. Gomes. Towards the automatic creation of a wordnet from a term-based lexical network. In *TextGraphs-5 Proceedings of the 2010 Workshop on Graph-based Methods for Natural Language Processing*, pages 10–18, 2010.

[20] S. P. Ponzetto and R. Navigli. Large-scale taxonomy mapping for restructuring and integrating wikipedia. In *IJCAI '03: Proceedings of the 21st International Joint Conference on Artificial Intelligence*, pages 2083–2088, 2009.

[21] D. R. Recupero. A new unsupervised method for document clustering by using wordnet lexical and conceptual relations. *Information Retrieval*, 10(6):563–579, 2007.

[22] M. Ruiz-Casado, E. Alfonseca, and P. Castells. Automatic assignment of wikipedia encyclopedic entries to wordnet synsets. In *AWIC '05: Proceedings of Advances in Web Intelligence*, pages 380–386, 2005.

[23] S. Scott and S. Matwin. Text classification using wordnet hypernyms. In *COLING-ACL '98: Workshop on Usage of WordNet in NLP Systems*, pages 45–51, 1998.

[24] S. Shehata. A wordnet-based semantic model for enhancing text clustering. In *ICDMW '09: Proceedings of the IEEE International Conference on Data Mining Workshops*, pages 477–482, 2009.

[25] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW '07: Proceedings of the 16th International Conference on World Wide Web*, pages 697–706, 2007.

[26] M. Sussna. Word sense disambiguation for free-text indexing using a massive semantic network. In *CIKM '93: Proceedings of the Second International Conference on Information and Knowledge Management*, pages 67–74, 1993.

[27] P. Vossen. Extending, trimming and fusing wordnet for technical documents. In *NAACL Workshop on WordNet and Other Lexical Resources*, 2001.

[28] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 42–49, 1999.

[29] I. Yoo, X. Hu, and I.-Y. Song. Integration of semantic-based bipartite graph representation and mutual refinement strategy for biomedical literature clustering. In *KDD '06: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 791–796, 2006.