Assessing Sentence Similarity Using WordNet based Word Similarity

Hongzhe Liu
Beijing Key Laboratory of Information Service Engineering, Beijing Union University, Beijing, China
Email: liuhongzhe@buu.edu.cn

Pengfei Wang
Beijing Key Laboratory of Information Service Engineering, Beijing Union University, Beijing, China
Email: feipengwang767@163.com

***Abstract*— Sentence similarity assessment is key to most NLP applications. This paper presents a means of calculating the similarity between very short texts and sentences without using an external corpus of literature. This method uses WordNet, common-sense knowledge base and human intuition. Results were verified through experiments. These experiments were performed on two sets of selected sentence pairs. We show that this technique compares favorably to other word-based similarity measures and is flexible enough to allow the user to make comparisons without any additional dictionary or corpus information. We believe that this method can be applied in a variety of text knowledge representation and discovery applications.**

***Keywords*—Sentence similarity, WordNet, word similarity, natural language processing**

I. Introduction

Many natural language processing applications require that the similarity between very short text paragraphs or sentences be calculated quickly and reliably. The samples, usually pairs of sentences, are considered similar if they are judged to have the same meaning or discuss the same subject. A method that can automatically calculate semantic similarity scores is much more valuable than simple lexical matching for applications such as question answering(QA), information extraction(IE), multi-document summarization, and evaluations of machine translation(MT). Most existing word-similarity-based measures rely not only on ontology knowledge base like WordNet, but also on large text corpora which serve as additional knowledge resources. However, in many applications, especially in domain-based applications, large text corpus cannot be expected to be readily available. Many applications store ontology relations in a relational database, which do not fully represent the rich relations embedded in the original text collections. In these cases, the similarities between these short texts and sentences have to be extracted from the limited representations in the database only. In this paper, we focus on the challenge of assessing sentence similarities by using the structural information inherent in a given ontology structure.

Any means of assessing text semantic similarity must account for the fact that text has structure. In this project, we began with an approximate model, which we later adapted into a means of assessing sentence level similarity based on word-level similarity derived from the sentence. We used four similarity measures in our experiments to capture sentence similarities from different aspects.

II. Related Work

The existing sentence similarity evaluation methods can be grouped into five categories: word overlap measures, TF-IDF measures, word-similarity-based measures, word-order-based measures, and combined measures[1]–[6],[14]. Among these, word-based similarity measures provided the best human correlation[15],[16],[17]. There are many means of assessing word-to-word similarity. These include distance-oriented measures, knowledge-based measures, and measures of information theory. The main methods of determining word similarity, however, are those proposed by Leacock and Chodorow, Lesk, Wu and Palmer, Resnik, Lin, and Jiang and Conrath[7]–[12], please see[15] for the detailed evaluating of WordNet-based measures of semantic distance . Here, we used sentence semantic similarity measures, which are based on word similarity. We focused our attention on structure-based word similarity measures (RNCVM) [13], which is based on WordNet. This approach does not require a training corpus or other additional information, which is very important when building new applications. This approach produces high quality results with good human correlation.

III. Our Proposed Approach

A. WordNet Structure

On WordNet [20], information is presented in synsets, clusters of words that are considered synonyms or otherwise logically similar. WordNet also includes descriptions of the relationships in the synsets. A given

doi:10.4304/jsw.8.6.1451-1458

word may appear in more than one synset, which is logical considering that words can also appear in multiple parts of speech. The words in each synset are grouped so that they are interchangeable in certain contexts.

The WordNet pointers indicate both lexical and semantic relationships between words. Lexical relationships concern word form, and semantic relationships concern meaning. These include but are not limited to hypernymy/hyponymy (hyponymy/troponymy), antonymy, entailment, and meronymy/holonymy.

- Nouns and verbs are organized into hierarchies based on the ***hypernymy/hyponymy*** or ***hyponymy/troponymy*** relationships between synsets. Some verbs are not organized in hierarchies. These are connected by ***antonym*** relationships and ***similar*** relationships, like adjectives.
- Adjectives are organized into both head and satellite synsets, each organized around antonymous pairs of adjectives. These two adjectives are considered head synsets. Satellite synsets include words whose meaning is similar to that of a single head adjective.
- Nouns and adjectives that are derived from verbs and adverbs that are derived from adjectives have pointers indicating these relations.

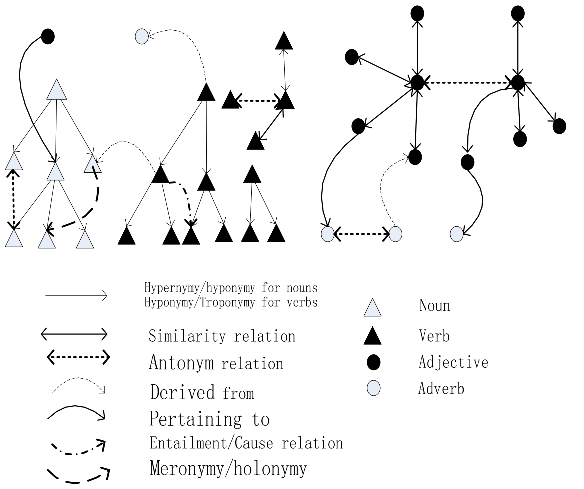


Figure 1. WordNet structure

B. Computing Word Similarity

Before introducing our method for calculating sentence similarity, we first describe how word similarity is calculated, which will be used in our study later.

1) Words Similarity in One Hierarchy

We now introduce how to compute similarity of two words in one hierarchy.

Density and similarity

With regard to the tree density, it can be observed that the densities in different part of the hierarchy are different. **The greater the density, the closer the distance between the nodes** . For example, the 'plant' section of the knowledge base is very dense, individual node having up to three and four hundreds children, collections of generally unpronounceable plant species; it can argue that the distance between nodes in such a section of structure should be very small relative to other less dense regions. That is in Fig. 2, the similarity value of the left part should be less than the similarity value of the right part of the hierarchy.

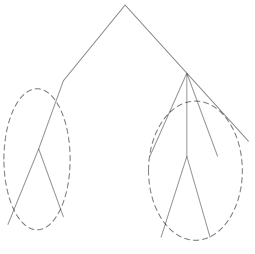


Figure 2. Local density effect

Depth and similarity

The deeper the depth of the nodes located, the higher the similarity of them. The foundation is that the distance shrinks as one descends the hierarchy, since differentiation is based on finer and finer details [5]. That is in Fig. 3, the value of sim (C_1, C_2) should be less than the value of sim (C_3, C_4).

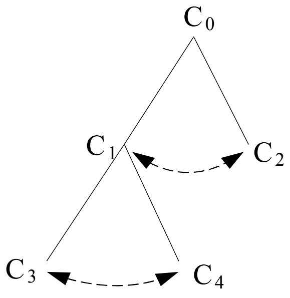


Figure 3. Depth effect

Path length and similarity

Semantic network includes concepts (usually nouns or noun phrases) that are linked to one another by named relations, for example, hyper/hyponym relation ('is-a' relation) and hol/meronym relation ('part-of' relation). If the semantic network is linked only by taxonomic 'is-a' relation, it is generally called 'is-a' semantic network or 'is-a' taxonomy. In this kind of semantic network, parent concept is more generalized than child concept, while child 'is a kind of' its parent concept.

Rada et al.[18]pointed out that the assessment of similarity in a semantic network can be in fact thought of as involving just taxonomic 'is-a' relation, and the simplest form of determining the distance between two elemental concept nodes, A and B, is the shortest path that links A and B, i.e. the minimum number of edges that separate A and B. But jiang and Conrath[11] then pointed out in a more realistic scenario, the distances between any two adjacent nodes are not necessarily equal. It is therefore necessary to consider that the edge connecting the two nodes should be weighted. To determine the edge weight automatically, certain aspects should be considered in the implementation. Most of these are typically related to the structural characteristics of a hierarchical network. Some conceivable features are: local network density (the number of child links that span out from a parent node), depth of a node in the hierarchy, type of link, and finally, perhaps the most important of all, the strength of an edge link. From Rada et al. and Jiang and Conrath, at least we can state that if **the shorter path is contained within the longer path in a 'is-a' taxonomy, the concept nodes pair with shorter path between them has greater concept similarity than that of with longer path between them.** That is in Fig. 4, the value of sim (C_0, C_3) should be less than the value of sim (C_0, C_1).

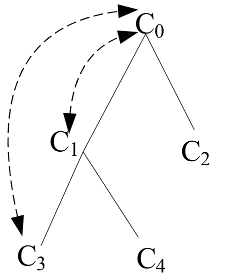


Figure 4. Path length effect

The premise of our method

The next two definitions are the premise of our method, definition 1 defines what is a hierarchy, definition 2 provides a mechanism through which similarity between concepts can be measured.

Definition 1 (Concept hierarchical model). Denote as H (N, E) is a rooted tree. Where N is the set of concept nodes (corresponding to the concepts) in the tree and E is the set of edges between the parent/child pairs in H. The semantic coverage of the child concept nodes is the partition of the semantic coverage of their parent concept node.

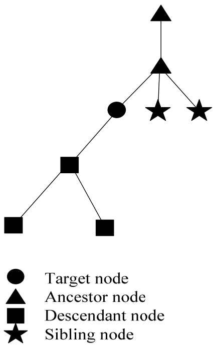


Figure 5. The concept node types illustration

A concept node is a parent of another concept node if it is one step higher in the hierarchy and closer to the root concept node. Each concept node in a tree has zero or more child concept nodes, which are one step below their parent concept node in the hierarchy. Sibling concept nodes share the same parent concept node. A concept node has at most one parent concept node. Concept nodes that do not have any children are called leaf concept nodes. The topmost concept node in the hierarchy is called the root concept node. Being the topmost concept node, the root concept node will not have parents, and it is the symbol of the universe. All concept nodes (except root concept node) can be reached from the root concept node by following edges and concept nodes on the path, and all these concept nodes on the path composed of the ancestor concept nodes of that concept node. All concept nodes below a particular concept node are called descendents of that concept node. Fig. 5 above illustrated the concept node types.

The concept hierarchical model is the premise of our method, and our similarity computation is from cosine similarity which is based on the orthogonality of its components, so the semantic coverage of the concept nodes should be independent. So we limit the semantic coverage of the child concept nodes are the partition (instead of covering) of the semantic coverage of their parent concept node. That is, the concepts subsumed by sibling concept nodes are usually non-overlapping; the relationship between two siblings is captured only through their ancestor concept nodes.

Definition 2 (Concept vector). Given a concept hierarchy model, H (N, E), with n concept nodes, the concept vector of a concept node C_i in this hierarchy has n dimensions. The concept node C_i's concept vector denote as $\vec{C_i}$ = ($v_{i,1}$, $v_{i,2}$,…,$v_{i,n}$), $v_{i,1}$, $v_{i,2}$,…,$v_{i,n}$ (i=1, 2, …, n) are the dimension values corresponding concepts C_1,C_2,…,C_n relative to concept C_i. Given two concept nodes, and their concept vectors, $\vec{C_i}$, $\vec{C_j}$ then their similarity is computed with (1):

$$sim(C_i, C_j) = \frac{\vec{C_i} \bullet \vec{C_j}}{\left\|\vec{C_i}\right\|\left\|\vec{C_j}\right\|} \quad (1)$$

Identifying the Concept Vectors for the concept nodes in the hierarchy

In the traditional corpus based method, the weight of concepts (the frequency of the concept) is derived from a large text corpus. We discuss a given hierarchy without a large corpus for frequency information extraction. Therefore, we need mechanisms to leverage the weights of concept nodes in the hierarchy. Essentially, our concept vectors would capture the semantic information inherent but hidden within the structure of the hierarchy which is the most challenge part of our work.

Consider that the document-document similarity computation, documents are represented as vectors; in the vector each dimension corresponds to a separate term. If a term occurs in a document, its value in the vector is non-zero. Usually a document is represented as a vector and the frequencies of a cluster of terms appeared in the document are used as dimension values. Vector operations can be used to compare document-document similarity. Here in a concept hierarchy model, the dimension values of each concept can be obtained only from the hierarchical structure. From observation, the density information of each concept node is inherent and hidden in the hierarchy.

Definition 3 (Local density). The density of a root concept node in a given concept hierarchy model is equal to 1, the density of other concept nodes equal the number of sibling concept nodes of that concept node plus 1(itself).

Definition 3 defines the situation of the uniform concept node local density. If sibling concept nodes have different density, it can be obtained from a large text corpus using traditional method as in reference[8], frequencies of concepts in the WordNet taxonomy were estimated using noun frequencies from the Brown Corpus of American English which is a large(more than 1,000,000 words) collection of text across generating from news articles to science fictions. Each noun that occurred in the corpus was counted as an occurrence of each taxonomic class containing it. But as mentioned above, such text corpus are usually hard to obtain in many domain specific applications (for example, biology and medicine,) and applications that rely on relational databases. Even the large text corpus is available, this kind of methods are slow due to the huge text statistics work, so we choose to use uniform density value in definition 3 to substitute their real distribution values. .

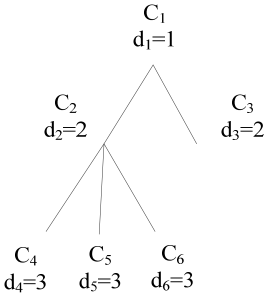


Figure 6. Tree example to show concept density

Fig. 6 provides a sample concept hierarchy. It shows how the concepts in the hierarchy share their local density. The density of root concept node C_1 is 1, the densities of C_2 and C_3 are 2, and densities of C_4, C_5, and C_6 are 3.

Consider the vector space model's approach origins in document-document similarity. The presumption is that, given a certain number of terms, the frequency of these terms in a document can be used as vectors to compute query-document similarity. In the situations we have described, the density information of a given node's relevancy nodes were used as vectors to compute internode similarity.

Relevancy nodes based concept vector

In the document-document similarity computation, a cluster of terms appeared in the document are used as dimension values. Given a concept node in the hierarchy, its ancestor concept nodes subsume its attributes, and its descendant concept nodes inherit it. So except the concept node itself, its ancestor and descendant concept nodes are relevancy to that concept node, which we used as "terms" in our structure.

Definition 4 (Relevancy Nodes). Given a concept node in the hierarchy, the concept node itself, its ancestor and descendant concept nodes compose its relevancy nodes.

Consider the vector space model's approach origins in document-document similarity. The presumption is that, given a certain number of terms, the frequency of these terms in a document was used as vectors to compute the query-document similarity. In our situations, the density information of a node's all relevancy nodes was used as vectors to compute internodes similarity.

Definition 5 (relevancy nodes based concept vectors for HCT). Given an HCT with n concept nodes, the concept vector of C_i is denoted as $\overrightarrow{C_i}$ = (vi,1, v i,2,…,v i,n) and v i,j (i=1,2,…n; j=1,2,…n) is the dimension value corresponding to all concept nodes relative to the particular concept node C_i , defined as follows using (2) :

$$v_{i,j} = \begin{cases} d_j (if\ C_j\ is\ the\ relevancy\ node\ of\ concept\ C_i) \\ d_j (if\ i = j) \\ 0\ \ (Otherwise\ ) \end{cases} \tag{2}$$

d_j is the local density of concept node C_j .

For example, for concept node C_2 in Fig. 6, the concept node C_2 itself, its ancestor concept node C_1, and its descendent concept nodes C_4, C_5, and C_6 compose C_2's relevancy nodes. Their local densities d_2, d_1, d_4, d_5, and d_6 are used as C_2's dimension values. C_3 is not a relevancy node of C_2, so its dimension value for concept vector $\overrightarrow{C_2}$ is 0. If we were to list all concept nodes in sequential order of concept vectors according to the tree's breadth-first traversal sequence, we would have C_2's concept vector $\overrightarrow{C_2}$ = (1, 2, 0, 3, 3, 3). Similarly, C_1's concept vector is $\overrightarrow{C_1}$ = (1, 2, 2, 3, 3, 3). C_3's concept vector is $\overrightarrow{C_3}$ = (1, 0, 3, 0, 0, 0), and C_4's concept vector is $\overrightarrow{C_4}$ = (1, 2, 0, 3, 0, 0),and C_5's concept vector is $\overrightarrow{C_5}$ = (1, 2, 0, 0, 3, 0). C_6's concept vector is $\overrightarrow{C_6}$ = (1, 2, 0, 0, 0, 3).

The similarity between any pair of words can be computed, for example, if we compute similarity values between $\overrightarrow{C_3}, \overrightarrow{C_4}$, their similarity values can be computed by the following (3):

$$sim(C_3,C_4) = \frac{\overrightarrow{C_3} \bullet \overrightarrow{C_4}}{\left\|\overrightarrow{C_3}\right\|\left\|\overrightarrow{C_4}\right\|} \approx 0.085 \tag{3}$$

2) Words Similarity In WordNet Structure

The semantic similarity of two words w_1 and w_2 is expressed as $sim(w_1, w_2)$. This value can be found through analysis of a lexical knowledge base, such as WordNet, in which words are organized into synonym sets (synsets).

- If the two target words are identical or in the same synset, then their similarity is 1.
- If one of the two target words is not in WordNet,

their similarity is 0.

- If the two target words are not in the same synset but both in the WordNet hierarchy, then their word similarity is computed based on their relevancy nodes' local density in the hierarchy as introduced in section III.B, Reference [13] has more articulation about this method.
- If one word is in the synset hierarchy (in most of the nouns and verbs structure) and the other is in another part of the WordNet semantic net, then the user must determine if there is any relationship between them based on the WordNet semantic nets, If there is, their similarity value is *c*. For example, in Fig. 1, if there is a "derived from" relation between "verb" and "adjective", the similarity value of the target verb and adjective words is set to *c*.
- If neither of the two words is in the synset hierarchy (not in most of the nouns and verbs structure), then the user must determine there is any relationship between them based on the WordNet semantic nets, if there is, their similarity value is *c*. For example, in Fig. 1, if there is a "pertaining to" relation between "adverb" and "adjective", the similarity value of the target adverb and adjective words is set to *c*.

C. Computing Sentence Similarity

We use the two example sentences to illustrate our method of calculating sentence level similarities.

S_1: Consumers would still have to get a descrambling security card from their cable operator to plug into the set.

S_2: To watch pay television, consumers would insert into the set a security card provided by their cable service.

Sentence preprocessing

For each data set, WordNet is used as a resource for calculating word similarity. In order to use wordnet, we first transform sentences S_1 and S_2 to their bag-of-words representation, T_1 and T2, respectively.

T_1= {words in S_1}

T_2= {words in S_2}

We also form a superset T, which is the union of T_1 and T_2: T= $T_1 \cup T_2$.

In our example, T_1, T_2 and their union T can be represented as:

T_1=
{"consumer,""would,""still,""have,""to,""get,""a,""descramble,""security,""card,""from,""their,""cable,""operator,""to,""plug,""into,""the,""set"}

T_2=
{"to,""watch,""pay,""television,""consumer,""would,""insert,""into,""the,""set,""a,""security,""card,""provide,""by,""their,""cable,""service"}

T=
{"consumer,""would,""still,""have,""get,""a,""descramble,""security,""card,""from,""their,""cable,""operator,""to,""plug,""into,""the,""set,""watch,""pay,""television,""insert,""provide,""by,""service"}

Vector forming

The purpose of forming T, which contains all the words from T_1 and T_2, is to create semantic vectors for T_1 and T_2.The vector derived from the joint word set is called the lexical semantic vector and is denoted by V_1 and V_2.The dimension of the lexical semantic vector is the number of words in this joint word set. The value of each entry in V_1 or V_2 represents the semantic similarity of any word in V_1 or V_2 to any word in either of our sentences. For word w_i inset T, we would compute a similarity score for w_i and every word in T_1 as described above. The word to which w_i was most similar would be used to give the value of the corresponding vector entry.

Entry value of V_1:

{("a,""a"): 1.0,
("by,""a"): 0.0,
("cable,""cable"): 1.0,
("card,""card"): 1.0,
("consumer,""consumer"): 1.0,
("descrambling,""descrambling"): 1.0,
("from,""from"): 1.0,
("get,""get"): 1.0,
("have,""get"): 1.0,
("insert,""set"): 0.935
("into,""into"): 1.0,
("operator,""operator"): 1.0,
("pay,""have"): 0.836
("plug,""plug"): 1.0,
("provide,""set"): 0.999
("security,""security"): 1.0,
("service,""set"): 0.996
("set,""set"): 1.0,
("still,""still"): 1.0,
("television,""cable"): 0.999
("the,""the"): 1.0,
("their,""their"): 1.0,
("to,""to"): 1.0,
("watch,""set"): 0.857
("would,""would"): 1.0}

Entry value of V2:

{("a,""a"): 1.0,
("by,""by"): 1.0,
("cable,""cable"): 1.0,
("card,""card"): 1.0,
("consumer,""consumer"): 1.0,
("descrambling,""a"): 0.0,
("from,""a"): 0.0,
("get,""pay"): 0.807
("have,""pay"): 0.836
("insert,""insert"): 1.0,
("into,""into"): 1.0,
("operator,""card"): 0.0,
("pay,""pay"): 1.0,
("plug,""set"): 0.995
("provide,""provide"): 1.0,
("security,""security"): 1.0,
("service,""service"): 1.0,

("set,""set"): 1.0,
("still,""television"): 0.792
("television,""television"): 1.0,
("the,""the"): 1.0,
("their,""their"): 1.0,
("to,""to"): 1.0,
("watch,""watch"): 1.0,
("would,""would"): 1.0}

$\vec{V_1}$=(1.0 ,0.0 ,1.0 ,1.0 ,1.0 ,1.0 ,1.0 ,1.0 ,1.0 ,0.935 ,1.0 ,1.0 ,0.836 ,1.0 ,0.999 ,1.0 ,0.996 ,1.0 ,1.0 ,0.999 ,1.0 ,1.0 ,1.0 ,0.857 ,1.0)

$\vec{V_2}$=(1.0 ,1.0 ,1.0 ,1.0 ,1.0 ,0.0 ,0.0 ,0.807 ,0.836 ,1.0 ,1.0 ,0.0 ,1.0 ,0.995 ,1.0 ,1.0 ,1.0 ,1.0 ,0.792 ,1.0 ,1.0 ,1.0 ,1.0 ,1.0 ,1.0)

We give a smaller weight $\delta(0<\delta<1)$ to articles, prepositions, and conjunctions in the sentences due to their prevalence of usage, we set δ=0.2.

$\vec{V}'_1$ =(0.2 ,0.0 ,1.0 ,1.0 ,1.0 ,1.0 ,0.2 ,1.0 ,1.0 ,0.935 , 0.2 ,1.0 ,0.836 ,1.0 ,0.999 ,1.0 ,0.996 ,1.0 ,0.2 ,0.999 ,0.2 , 0.2 ,0.2 ,0.857 ,0.2)

$\vec{V}'_2$ =(0.2 ,0.2 ,1.0 ,1.0 ,1.0 , 0.0 ,0.0 ,0.807 ,0.836 ,1.0 , 0.2 ,0.0 ,1.0 ,0.995 ,1.0 ,1.0 ,1.0 ,1.0 ,0.792 ,1.0 ,0.2 ,0.2 , 0.2 ,1.0 ,0.2)

So, the cosine similarity between S_1 and S_2 is computed from (4):

$$\mathrm{sim}(S_1,S_2)=\frac{\vec{V_1}\bullet\vec{V_2}}{\left\|\vec{V_1}\right\|\left\|\vec{V_2}\right\|}=0.907 \qquad (4)$$

The cosine similarity between S1 and S2 is computed, and the result is 0.907 .

The intuition behind this calculation is to map semantic meanings contained in the union set sentence to each of the original sentence. In this way, we quantify how the semantic meaning of the sentence is conveyed in the two lexical realization formats. In the next section, we will empirically verify our approach on a large corpus.

IV. Experimental Evaluation

A. Evaluation Criteria

We look into four different evaluation measures, which capture different aspects of semantic similarity.

TP: Number of sentences predicted to be similar sentences that actually are similar.

TN: Number of sentences predicted to be dissimilar sentences that actually are dissimilar

FP: Number of sentences predicted to be similar that are actually dissimilar

FN: Number of sentences predicted to be dissimilar that are actually similar

Accuracy=(TP + TN)/(TP + TN+ FP + FN)
Precision= TP/(TP + FP)
Recall= TP/(TP + FN)
F-measure= $(1+\beta)$PR/(βP + R)
= 2PR / (P + R)
(when β=1, precision and recall have the same weight)

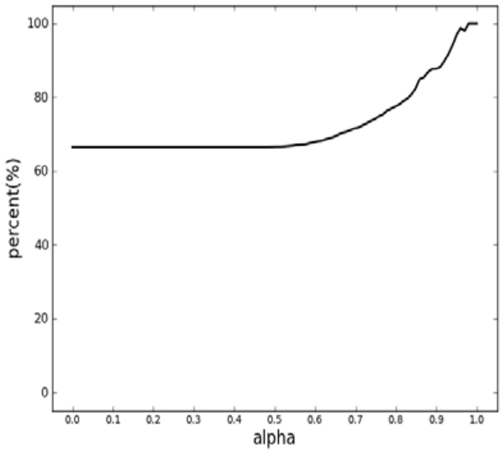
We evaluated the results in terms of accuracy, precision, recall, and F-measure.

B. Data Sets

The Microsoft Research paraphrase corpus (MSRP) data set is a known dataset for ground truth of sentence similarity calculation, and it includes 1,725 test pairs taken from Internet news articles [18]. Each sentence pair is judged by two human assessors whether they are semantically equivalent or not. Overall, 67% of the total sentence pairs are judged to be the positive examples. Semantically equivalent sentences may contain either identical information or the same information with minor differences in detail according to the principal agents and the associated actions in the sentences. Sentence that describes the same event but is a superset of the other is considered to be a dissimilar pair. Note that this rule is similar to other text entailment task.

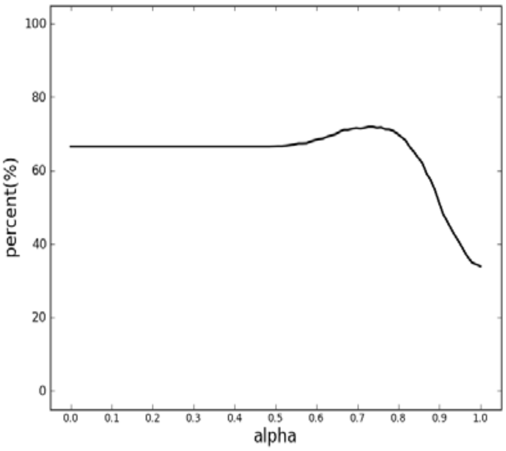
C Analysis of Results

Fig. 7,8 and 9 depict the accuracy, precision, and recall values of our method. Table I shows the sentence similarity performance on the MSRP data (C=0.2. δ=0.2).



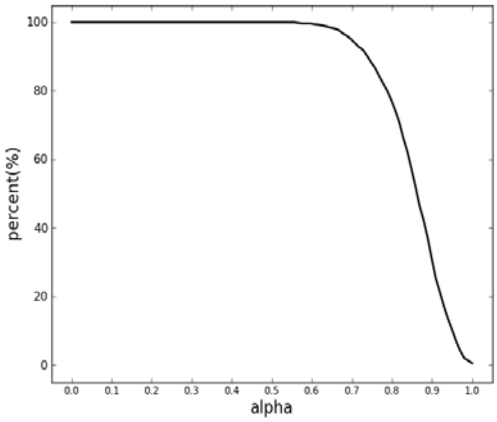
x-axis: similarity threshold (alpha)
y-axis: Precision values of our method(percent(%))

Figure 7. Precision values of our method with different similarity threshold (alpha)



x-axis: similarity threshold (alpha)
y-axis: Accuracy values of our method(percent(%))

Figure 8. Accuracy values of our method with different similarity threshold (alpha)



x-axis: similarity threshold (alpha)
y-axis: Recall values of our method(percent(%))

Figure 9. Recall values of our method with different similarity threshold (alpha)

TABLE I.
PERFORMANCE OF THE SENTENCE SIMILARITY METHOD ON THE MSRP DATA SET

Methods	Accuracy	Precision	Recall	F-measure
J & C	69.3	72.2	87.1	79.0
L & C	69.5	72.4	87.0	79.0
Lesk	69.3	72.4	86.6	78.9
Lin	69.3	71.6	88.7	79.2
W & p	69.0	70.2	92.1	80.0
Resnik	69.0	69.0	96.4	80.4
Ours (Alpha=0.737)	72.0	73.8	90.2	81.1

Note: The results of related methods are taken from Mihalcea et al. [5]

In addition to word order, the related method requires that the specificity of words should be taken into account, so a higher weight was given to specific words and a low weight to the similarity of generic concepts. The specificity of each word was generally determined using the inverse document frequency (*idf*). We took a first rough cut at this problem and then attempted to model the semantic similarity of texts as a function of purely semantic similarity of the component words.

We evaluated these results in terms of accuracy, precision, recall, and f-measure. Most methods yield good results for precision or recall, but very few do so for both. The average related method *accuracy* was 69.2%, though it climbed as high as 72% when we used 0.737 as the similarity threshold score. When related methods' *precision* values were found to be 69.0–72.4, recall tended to be 86.6–96.4.The *recall* of ours method was 0.928–0.988 at the same level of precision, which was also higher than that of related methods. When related methods *'recall* values were 86.6–96.4, *precision* was 69.0–72.4.The *precision* of our method was 0.706–0.748 for the same *recall*, which was also higher than that of related methods. We also obtained the highest F-measure value (81%)of any related method.

V. IMPLICATIONS AND CONCLUSION

In this paper, we propose an unsupervised approach to automatically calculate sentence levels similarities based on word level similarities, without using any external knowledge from other training corpora. The main contributions of our work to the field are as follows:

1. A means of automatically computing the similarity values of any two words in WordNet is defined.
2. A means of computing sentence similarity based on the word composition of each sentence is proposed.

Experiments prove that our method compares favorably to other word-similarity-based measures. This method allows the user make comparisons between sentences based solely on ontological structure without requiring on any additional dictionary or corpus of information. In this way, our approach can be applied directly to any domain application. We believe this is a very attractive feature in building new applications.

ACKNOWLEDGEMENTS

This project was supported by the Project of Construction of Innovative Teams and Teacher Career Development for Universities and Colleges under Beijing Municipality (CIT&TCD20130513), Funding Project for Academic Human Resources Development in Institutions of Higher Learning under the Jurisdiction of Beijing Municipality (Grant No. PHR201108419), and the National Natural Science Foundation of China (Grant No. 60972045 and grant No.61271369).

REFERENCES

[1] Metzler D., Bernstein Y., Croft W., Moffat A., Zobel J. (2005) Similarity measures for tracking information flow. Proceedings of CIKM, 517–524.

[2] Ponzetto S.P, Strube M. (2007) Knowledge Derived From Wikipedia for Computing Semantic Relatedness, Journal of Artificial Intelligence Research, 30, 18–-212.

[3] Allan J., Bolivar A., and Wade C. (2003) Retrieval and novelty detection at the sentence level. In Proceedings of SIGIR’03, 314–321.

[4] Hoad T., Zobel J.(2003) Methods for identifying versioned and plagiarized documents. Journal of the American Society of Information Science and Technology,54(3), 203–215.

[5] Mihalcea R., Corley C., Strapparava C. (2006) Corpus-based and knowledge-based measures of text semantic similarity, in Proceedings of AAAI 2006, Boston, July.

[6] Chukfong H.,Masrah A., Azmi M.,Rabiah A.K., Shyamala C.(2010).Doraisamy.Word sense disambiguation-based sentence similarity, Proceedings of the 23rd International Conference on Computational Linguistics, 418-426

[7] Leacock C, Chodorow. M (1998). Combining local context and WordNet sense similarity for word sense identification. In WordNet, an Electronic Lexical Database, 265–283.

[8] Resnik P. (1999), Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language, Journal of Articial Intelligence Research, 95–130.

[9] Banerjee S.,Pedersen T.(2003).Extended gloss overlaps as a measure of semantic relatedness. In Proceedings of IJCAI, 805–810.

[10] Wu Z., Palmer. M.(1994) Verb semantics and lexical selection. In Proceedings of ACL. 133–138

[11] Jiang J.,Conrath D.(1997).Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings of COLING.

[12] Lin D.(1998). An information-theoretic definition of similarity. In Proceedings of ICML, 296–304.

[13] Hongzhe L., Hong B., De X.(2011), Concept Vector for Similarity Measurement based on Hierarchical Domain Structure, computing and informatics, Vol. 30, 1001–1021,

[14] Li, Y., Bandar, Z., and McLean, D.(2003) An approach for measuring semantic similarity using multiple information sources. IEEE Trans. Knowledge Data Eng. 15(4).871–882.

[15] Budanitsky, A. and Hirst, G.(2003) Evaluating WordNet-based measures of semantic distance. Computational Linguistics, 32(1), 13–47.

[16] Palakorn A., Xiaohua H., Xiajiong S.(2008) The Evaluation of Sentence Similarity Measures, Proceedings of the 10th international conference on Data Warehousing and Knowledge Discovery, 305-316

[17] Aminul I., Diana I.(2008) Semantic text similarity using corpus-based word similarity and string similarity, ACM Transactions on Knowledge Discovery from Data (TKDD), 2 (2).

[18] Dolan, W., Quirk, C., and Brockett, C.(2004) Unsupervised construction of large paraphrase corpora: Exploiting massively parallel new sources. In Proceedings of the 20thInternational Conference on Computational Linguistics.

[19] Rada R., Mili H., Bicknell E., and Bletner M. (1989) Development and Application of a Metric on Semantic Nets”, IEEE Transactions on Systems, Man, and Cybernetics, 19(1).17-30.

[20] Wordnet website, available on: http://wordnet.princeton.edu/.2011.5.1

Hong-zhe Liu, Ph.D of school of computer and information technology from Beijing Jiao tong University, Beijing, China, she receive her M.S. degree in computer science from California State University, USA, in 1999. She is now an Assistant Processor of Computer Science Department, Beijing union University, vice director of Beijing Key Laboratory of Information Service Engineering, Beijing Union University, Beijing, China. Her research interests include semantic computing, artificial intelligence and distributed systems.

Peng-fei Wang, master of software engineering of Beijing Key Laboratory of Information Service Engineering from Beijing Union University, Beijing, China, he received his bachelor degree in electronic information engineering from Beijing Union University, Beijing, China, in 2012. His research interests include semantic computing, artificial intelligence and distributed systems.