International Journal of Database Theory and Application
Vol.9, No.10 (2016), pp.33-46
http://dx.doi.org/10.14257/ijdta.2016.9.10.04

Short Text Similarity Measure Based on Double Vector Space Model

Ying Liu, Dongmei Li* and Cong Dai

School of Information Science and Technology, Beijing Forestry University, Beijing 100083, China
lidongmei@bjfu.edu.cn

Abstract

Short text similarity measure is the basis of classification and duplicate checking of the short texts. Allowing for the insufficient consideration of the sentence semantic and structure information in similarity calculation between two short texts, we propose a novel method of short text similarity calculation based on double vector space model on the basis of traditional vector space model. Creatively transforming traditional vector space model into double vector space model. We utilize the numeral data link relations of Wikipedia to calculate semantic similarity between words, and calculate text structure similarity by dependency trees. Finally, we get the synthetic similarity by combining the semantic similar vector and structure similar vector. Our experiment results demonstrate that the proposed method has higher accuracy than other methods.

Keywords*: double vector space model; Wikipedia; semantic similarity; dependency tree*

1. Introduction

With the rapid development of Internet, the amount of information underlying in short texts (*e.g.* short messages, tweets and so on) is dramatically increasing. So it is quite essential to investigate how to handle these data. In order to analyze and sort the large amount of short text information, more often than not, scholars tend to calculate short text similarity. Traditional methods based on vector space model (VSM) convert each word in the sentence to a TF-IDF value with word frequency, part of speech and other relevant information, and then calculate the similarity between two sentences by cosine distance. However, these methods only take the superficial information of the sentences into account and they ignore the semantic relations between words in each sentence.

In order to capture semantic information in the sentences better, the previous methods of text similarity measure mainly depend on the HowNet [1], WordNet [2] and Synonyms [3] and other relevant lexical semantic dictionaries. But for these methods, high integrity and accuracy of semantic dictionaries are essential prerequisites for high accuracy of calculation results. Based on traditional TF-IDF methods, Chenghui Huang [4] introduced a new algorithm called TSemSim, which is based on external dictionary and combined word semantic information to calculate text similarity. TSemSim improves the accuracy of short text similarity to some extent, yet the algorithm has some limitations because of its dependence on dictionary.

Nowadays saw the burgeoning knowledge base from Wikipedia. Wikipedia has huge amounts of data and it has specific relations between data links network, which means that it greatly makes up for the drawbacks of traditional semantic dictionaries. Consequently, some new semantic relatedness measure methods based on Wikipedia are flourishingly proposed.

Strube and Ponzetto presented a new semantic relatedness calculation method called WikiRelate [5], which is based on Wikipedia classification structure and hierarchy

ISSN: 2005-4270 IJDTA

information to calculate the semantic relations between words. According to link relations between concepts in Wikipedia, WLM [6] combines the VSM and The Google Similarity Distance [7] so that it obtains more accurate results. However, WLM ignores the meticulous link relations between concepts. ESA [8] has the highest accuracy among all of the semantic relatedness measure methods utilizing Wikipedia recently. As for this approach, semantic information of each word appears in Wikipedia is represented as a high-dimensional vector. The weighted vectors of the Wikipedia articles represented as each term. This approach is able to obtain semantic representation of documents in Wikipedia and makes contributions to the classification of short texts. But this method requires a larger memory.

In addition to study semantic relations between words in the short texts, there are also some new ways focusing on the structure information of texts to measure short text similarity. Leusch G [9] proposed a new algorithm called edit distance based on exchanging blocks, which improves the accuracy of the results, but costs high space complexity. Bin Li [10] created a Chinese sentence semantic similarity method based on semantic dependency. It utilized dependency relations between words in dependency trees. The point of the method is that it calculates similarity between sentences by measuring the similarity of the effective matching pairs in dependency tree, considering the dependency relationship between words in dependency tree. However, this method only considers the core semantic dependency tree structure. So there are some limitations in this method. Li [11] presented an approach of texts similarity based on frame semantic parsing. Compared to traditional methods based on semantic similarity, the accuracy of results has some improvement. Yet it only takes the core framework into account, and this factor makes effects on the accuracy of the calculation results.

In this paper, we make an alternative to the traditional vector space model, and make it transformed into two vector space model, combining semantic information of Wikipedia and structure information of semantic dependency tree. So we propose a new short text similarity algorithm called DVSM-WDT (Double Vector Space Models -Wikipedia and Dependency Tree), which performs better in promoting the accuracy of short text similarity calculation.

2. DVSM-WDT Model

As for two short texts, the former is marked as ST_1, the latter is marked as ST_2.

First of all, we have a process of filtering stop words and segmenting ST_1 and ST_2. Then we calculate the semantic similarity based on Wikipedia and the structure similarity based on semantic dependency trees. Combining link-in set similarity, link-out set similarity, double-link set similarity and concept structure similarity, we obtain the results of the semantic similarity. What's more, the structure similarity based on semantic dependency trees is the mixture of the ratio of relations, the ratio of parts of speech, the ratio of morphological patterns and the ratio of dependency depths. Finally, we obtain the similarity of ST_1 and ST_2, which is a combination of semantic similar vector and the structure similar vector. The DVSM-WDT model is shown in Figure 1:

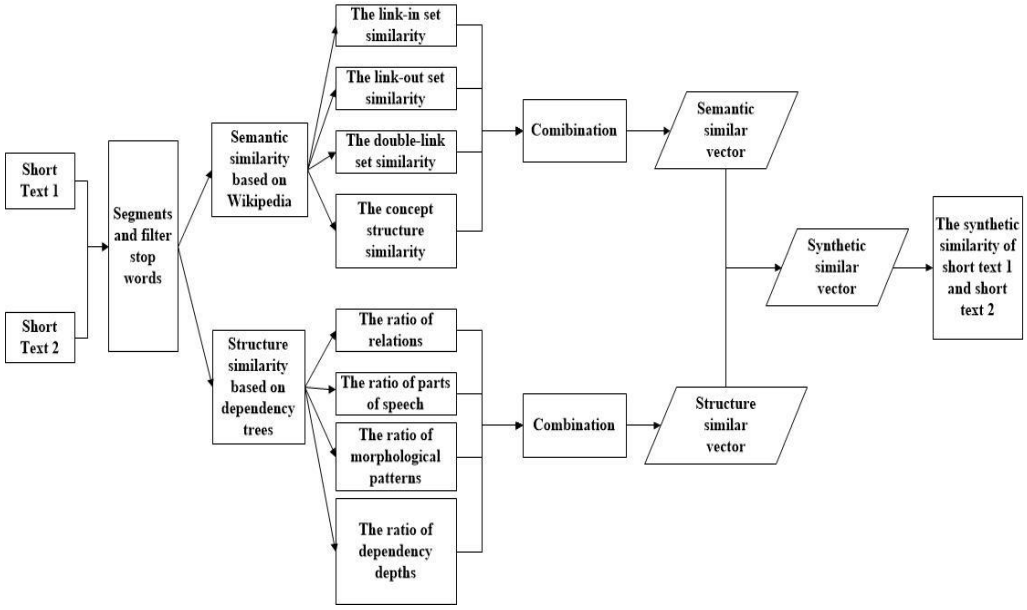

Figure 1. The DVSM-WDT Model

3. Short Text Similarity Measure Based on Double Vector Space Model

3.1. The Calculation Method of the Semantic Similarity

3.1.1. Relevant Definitions

Definition 1 (Concept) in Wikipedia, every explanation page is called a concept. The title of the explanation page is the name of the concept, denoted as WKC.

Definition 2 (Link-In Set) among the numerous link relations in Wikipedia, the link-in set of WKC is the set of the concepts which link to WKC, denoted as $LinkIn(WKC)$.

Definition 3 (Link-Out Set) the link-out set of WKC is the set of the concepts which WKC links to, denoted as $LinkOut(WKC)$.

Definition 4 (Double-Link Set) in the link-in set and link-out set of WKC, if there exists an element in the two sets that is double-link relation with WKC, these elements compose the double-link set, defined as $LinkDouble(WKC)$.

Definition 5 (Structure Vector) the vector containing relevant structure information of WKC is called the structure vector, denoted as SV_{WKC}.

Definition 6 (Semantic Similar Vector) the vector containing semantic similarity information between ST_1, ST_2 is called the semantic similar vector, denoted as $smv(ST_1, ST_2)$.

3.1.2. Relevant Calculating Formulas

WLM algorithm gains better results in semantic similarity between words by semantic link relations from Wikipedia. However, it ignores meticulous link relations. In this paper, we utilize the common links of link-in sets and link-out sets between two concepts. Besides, we consider the double-link sets and the structure information of concepts.

Assume there exists two concepts, WKC_1 and WKC_2. The similarity calculation between them is comprised by four parts below:

The link-in set similarity Sim_{linkin}, the link-out set similarity $Sim_{linkout}$, the double-link set similarity $Sim_{linkdouble}$ and the structure information similarity $Sim_{structure}$.

(1) The link-in set similarity Sim_{linkin}

$$Sim_{linkin} = \frac{2\left|LinkIn(WKC_1) \cap LinkIn(WKC_2)\right|}{\left|LinkIn(WKC_1)\right| + \left|LinkIn(WKC_2)\right|} \tag{1}$$

where $LinkIn(WKC_1)$ is the link-in set of WKC_1, $LinkIn(WKC_2)$ is the link-in set of WKC_2.

(2) The link-out set similarity $Sim_{linkout}$

$$Sim_{linkout} = \frac{2\left|LinkOut(WKC_1) \cap LinkOut(WKC_2)\right|}{\left|LinkOut(WKC_1)\right| + \left|LinkOut(WKC_2)\right|} \tag{2}$$

where $LinkOut(WKC_1)$ is the link-out set of WKC_1, $LinkOut(WKC_2)$ is the link-out set of WKC_2.

(3) The link-double set similarity $Sim_{linkdouble}$

$$Sim_{linkdouble} = \frac{2\left|LinkDouble(WKC_1) \cap LinkDouble(WKC_2)\right|}{\left|LinkDouble(WKC_1)\right| + \left|LinkDouble(WKC_2)\right|} \tag{3}$$

where $LinkDouble(WKC_1)$ is the link-double set of WKC_1, $LinkDouble(WKC_2)$ is the link-double set of WKC_2.

(4) The concept structure similarity $Sim_{structure}$

In order to calculate structure information of each WKC, we further divided the structure information into link-in information and link-out information. Link-in information is the sum of link weight in link-in set, and link-out information is the sum of link weight in link-out set. Hence, we define weights for all the elements in link-in set and link-out set of each WKC. We draw on the experience of WLM algorithm to calculate the weight:

$$w = \log \frac{|W|}{|T|} \tag{4}$$

$|W|$ refers to the total number of documents in Wikipedia.

$|T|$ represents the number of documents which contains the link in Wikipedia.

For WKC_1 and WKC_2, their structure vectors are respectively shown as follows:

$$SV_{WKC_1} = \left[\sum w_{linkin1}, \sum w_{linkout1}\right] \tag{5}$$

$$SV_{WKC_2} = \left[\sum w_{linkin2}, \sum w_{linkout2}\right] \tag{6}$$

$$Sim_{structure} = \frac{\sum w_{linkin1} \times \sum w_{linkin2} + \sum w_{linkout1} \times \sum w_{linkout2}}{\sqrt{\left(\sum {w_{linkin1}}^2 + \sum {w_{linkout1}}^2\right) \times \left(\sum {w_{linkin2}}^2 + \sum {w_{linkout2}}^2\right)}} \quad (7)$$

$\sum w_{linkin1}, \sum w_{linkin2}$ represent the sum of weight in link-in set of WKC_1 and WKC_2 respectively.

$\sum w_{linkout1}, \sum w_{linkout2}$ represent the sum of weight in link-out set of WKC_1 and WKC_2 respectively.

$Sim(WKC_1, WKC_2)$ represents the semantic similarity between WKC_1 and WKC_2.

Consequently, the semantic similarity between WKC_1 and WKC_2 is:

$$Sim(WKC_1, WKC_2) = 0.5 \times \left(\frac{Sim_{linkin} + Sim_{linkout}}{2} + Sim_{linkdouble} \right) + 0.5 \times Sim_{structure} \quad (8)$$

3.2. The Calculation Method of the Structure Similarity Based on Semantic Dependency Trees

3.2.1. The Establishment of Semantic Dependency Tree

L. Tesniere is the first one who proposed the concept of dependency syntax, a French linguist. He analyzed a sentence by utilizing a dependency syntax tree which describes dependency relations between words in the sentence. [12] We can understand and analyze the structure of sentence better by using dependency relation between words. In this paper, we use Stanford Parser to construct semantic dependency trees.

An example sentence: My dog likes eating sausage.

The new semantic dependency tree is originated from the sentence is shown in Figure 2:

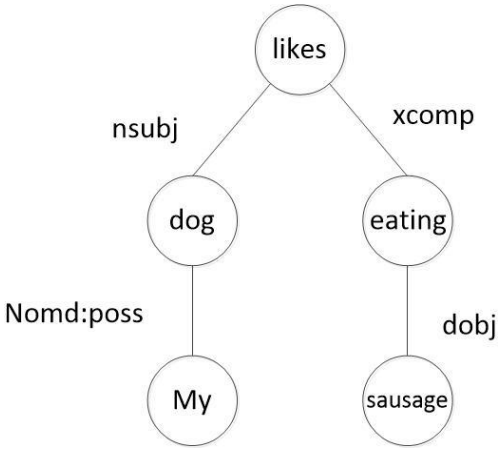

Figure 2. A Sample of Semantic Dependency Tree

As the semantic tree is constructed, we can calculate the structure similarity based on semantic dependency trees by the relevant information between nodes in the tree.

3.2.2. Relevant Definitions

Definition 1 (Dependency Node) in the dependency tree DT (in the following definitions, we use DT to represent the dependency tree), the node combined with the

information of the part of speech and morphological pattern is called the dependency node, denoted as N.

Definition 2 (Relation Branch) in DT, DT consists of several relation branches. Each of relation branches is denoted as R. R is made up of a pair of parent-child nodes and the relation between them defined as r. We define triple R as $R=(N_1,r,N_2)$. N_1 is the parent of N_2.

Definition 3 (Direct Subtree) in DT, a subtree which consists of N (not a leaf) and the children of N is called a direct subtree, denoted as DS.

Definition 4 (Subtree Relation Set) in DS, the set constructed by r is subtree relation set, denoted as $D(r)$. The number of element in $D(r)$ is denoted as $\lceil D(r) \rceil$.

Definition 5 (Relation Frequency) in DS, the number of r is the relation frequency of r in DS, denoted as $f(r)_{DS}$.

Definition 6 (Dependency Depth) in DT, every N should have its related dependency depth from the distribution of R in DS, denoted as $dd(N)$. The $dd(N)$ of root is 1.

If $R=(N,r,N')$, then:

$$dd(N')=dd(N)+\frac{1}{\lceil DS(r) \rceil \times f(r)_{DS}} \tag{9}$$

Definition 7 (Ratio of Dependency Depth) it is the ratio between the dependency depth of N_1 and N_2, denoted as $depth(N_1,N_2)$.

$$depth(N_1,N_2)=\begin{cases}\dfrac{dd(N_1)}{dd(N_2)} & dd(N_1)<dd(N_2)\\ \dfrac{dd(N_2)}{dd(N_1)} & dd(N_1)\geq dd(N_2)\end{cases} \tag{10}$$

Definition 8 (Ratio of Relation) it is the ratio of the count of relation which is matched between R_1,R_2 and the sum of relations between R_1,R_2, denoted as $rel(R_1,R_2)$.

Definition 9 (Ratio of Part of Speech) it is the ratio of the count of the part of speech which is matched between R_1,R_2 and the sum of part of speech between R_1,R_2, denoted as $ps(R_1,R_2)$.

Definition 10 (Ratio of Morphological Pattern) it is the ratio of the count of the morphological pattern which is matched between R_1,R_2 and the sum of morphological pattern between R_1,R_2, denoted as $mp(R_1,R_2)$.

Definition 11 (Structure Similar Vector) the vector which mixes the similarity information between DT_1 and DT_2 is called the structure similar vector of DT_1 and DT_2, denoted as $ssv(DT_1,DT_2)$.

3.2.3. Relevant Calculation Formulas

Assume that there exists two dependency trees, the former will be marked as DT_1, the latter will be marked as DT_2.

We make an assumption that N_1 is a dependency node(not a leaf) of DT_1, N_2 is a dependency node(not a leaf) of DT_2, $DS_1 = \{R_{1_1}, R_{1_2} \ldots R_{1_m}\}$ is an direct subtree of N_1, $DS_2 = \{R_{2_1}, R_{2_2} \ldots R_{2_n}\}$ is an direct subtree of N_2.

$match(N_1, N_2)$ is the combination of the ratio of relations, the ratio of parts of speech and the ratio of morphological pattern.

$$match(N_1, N_2) = \frac{\sum rel(R_{1_i}, R_{2_j}) + ps(R_{1_i}, R_{2_j}) + mp(R_{1_i}, R_{2_j})}{3}, i \in [1, m], j \in [1, n] \quad (11)$$

If N_1 or N_2 is a leaf:

$$match(N_1, N_2) = \frac{\sum ps(R_{1_i}, R_{2_j}) + mp(R_{1_i}, R_{2_j})}{2}, i \in [1, m], j \in [1, n] \quad (12)$$

We utilized $sim(N_1, N_2)$ to represent the similarity between N_1 and N_2:

$$sim(N_1, N_2) = A \times match(N_1, N_2) + B \times depth(N_1, N_2) \quad (13)$$

A, B are adjustment factors, and $A + B = 1$

$$DT_1 = \{N_{1_1}, N_{1_2} \ldots N_{1_{m'}}\} \; DT_2 = \{N_{1_1}, N_{1_2} \ldots N_{1_{n'}}\}$$
$$ssv_{i'}(DT_1, DT_2) = \max\{sim(N_{1_{i'}}, N_{2_1}) \ldots sim(N_{1_{i'}}, N_{2_{n'}})\} \quad (14)$$
$$ssv(DT_1, DT_2) = [ssv_1(DT_1, DT_2), ssv_2(DT_1, DT_2) \ldots ssv_{m'}(DT_1, DT_2)] \quad (15)$$

3.3. The Calculation Method of Short Text Similarity

We denote two short texts as X, and Y. Thus, X can be regarded as the word sequence $X_1, X_2, \ldots, X_n$, and Y as $Y_1, Y_2 \ldots, Y_m$.

X is reflected by: $X = (x_1, x_2, x_3, \ldots x_n)$

Y is reflected by: $Y = (y_1, y_2, y_3, \ldots y_m)$

(1) Construct the semantic similarity matrix M_{xy} of X and Y:

$Sim(x_i, y_j)$ is the semantic similarity of x_i, y_j

$$M_{xy} = \begin{bmatrix} Sim(x_1, y_1) & \cdots & Sim(x_1, y_m) \\ \vdots & \ddots & \vdots \\ Sim(x_n, y_1) & \cdots & Sim(x_n, y_m) \end{bmatrix} \quad (16)$$

If x_i or y_j is stop word, $Sim(x_i, y_j) = 0$

(2) For each row in the matrix, we use the $\max\left(Sim\left(x_i, y_j\right)\right)$ to calculate the maximum of concept similarity between a certain word in X and all the words in Y. Thus, the matrix will be reduced to one dimension. Therefore, we can construct the semantic similar vector of X and Y, which is the first layer of VSM:

$$smv(X,Y)=\left[\max\left(Sim\left(x_1, y_j\right)\right), \max\left(Sim\left(x_2, y_j\right)\right)\ldots\max\left(Sim\left(x_n, y_j\right)\right)\right], j\in[1,m] \quad (17)$$

(3) We can construct two semantic dependency trees DT_X, DT_Y of X, Y by using Stanford Parser.

Then construct the structure similar vector of DT_X and DT_Y from formula (9)-(15):

$$ssv\left(DT_X, DT_Y\right)=\left[ssv_1\left(DT_X, DT_Y\right), ssv_2\left(DT_X, DT_Y\right)\ldots ssv_n\left(DT_X, DT_Y\right)\right] \quad (18)$$

(4)According to the semantic similar vector and structure similar vector, we construct a new matrix M'_{xy}:

$$M'_{xy}=\begin{bmatrix}\max\left(Sim(x_1, y_j)\right) & \cdots & \max\left(Sim(x_n, y_j)\right)\\ \vdots & \ddots & \vdots\\ ssv_1\left(DT_X, DT_Y\right) & \cdots & ssv_n\left(DT_X, DT_Y\right)\end{bmatrix}, j\in[1,m] \quad (19)$$

Reduce the dimension of M'_{xy} and construct the n-dimension synthetic similar vector. Each dimension in vector is denoted as Sim'_i

$$Sim'_i=\alpha\times\max\left(Sim\left(x_i, y_j\right)\right)+\beta\times ssv_i\left(DT_X, DT_Y\right), i\in[1,n], j\in[1,m] \quad (20)$$

α, β are used to adjust weight in the formula, and $\alpha+\beta=1$

(5) We obtain the synthetic similar vector $syv(X,Y)$, which is the second layer of VSM:

$$syv(X,Y)=\left[Sim'_1, Sim'_2\ldots Sim'_n\right] \quad (21)$$

In addition, we construct a n-dimension vector $syv(Y,Y)$:

$$syv(Y,Y)=[1,1\ldots1] \quad (22)$$

From that, we can calculate the synthetic similarity of X, Y $synsim(X,Y)$:

$$synsim(X,Y)=\frac{\sum_{k=1}^{n} Sim'_k}{\sqrt{\sum_{k=1}^{n} Sim'^{\,2}_k \sum_{k=1}^{n} 1}} \quad (23)$$

4. Experiment

4.1. The Source of Data and the Datasets

In this paper, the version of Wikipedia data is updated at December 1st, 2015. We download the pages-articles.xml.bz2, which is the page information data(including id of pages, titles and content), and the page links data which is called pagelinks.sql.gz(it

contains multi-links between pages). We use JWPL [13] to analyze and deal with these data.

In order to test the semantic similarity more accurately by the algorithm proposed in this paper, we use a common dataset of semantic similarity test, the WordSim-353 datasets (Finkelstein *et al*., 2002), which contains 353 word pairs.

When it comes to the evaluation of short text similarity, we process the Reuters Transcribed Subset dataset and select 107 short texts which the number of words ranges between 110 and 180. These 107 short texts are sorted into 7 categories: acq, coffee, crude, earn, grain, ship, trade. The new dataset is called Reuters Transcribed Subset-107.

4.2. The Evaluation Method of Algorithm

In order to assure the accuracy of the experiment of semantic similarity, we use Spearman's rank correlation coefficient to measure the relatedness between the results of the experiment and human judgements:

$$\rho = 1 - \frac{6 \times \sum_{i} d_i^2}{n(n^2 - 1)} \tag{24}$$

where d_i is the difference between the semantic similarity of the i th pair words and human judgements, n represents the size of dataset.

To assure the accuracy of the experiment of short text similarity, in Reuters Transcribed Subset-107, every short text is compared with other short texts in the dataset by order. For every short text, we choose 3 of the other short texts(order by similarity) to match it. So there are 321 matched pairs of short texts. If a short text and a matched short text belong to the same category, they are the successful match pairs.

$$accuracy = \frac{\sum MatchSen}{\sum TotalSen} \times 100\% \tag{25}$$

$\sum MatchSen$ is the count of the successful match pair. $\sum TotalSen$ is the count of the total match pair.

4.3. The Analysis of Experiment Results

4.3.1. Experiment Results of the Semantic Similarity

Figure 3 shows the comparison of the accuracy of the semantic similarity algorithm in this paper and paper [2,5-6].

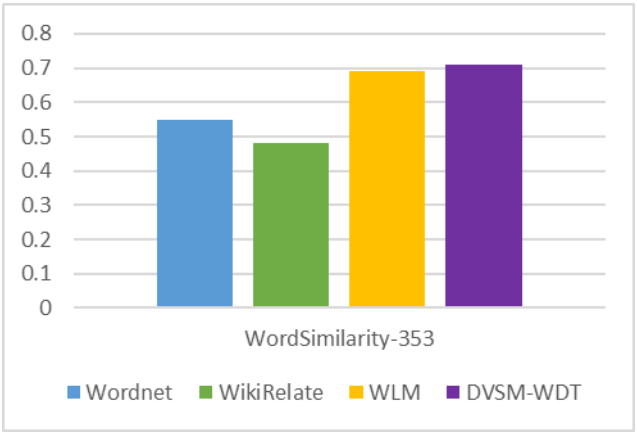

Figure 3. The Comparison of the Accuracy of the Semantic Similarity Algorithm in this Paper and Traditional Algorithm

4.3.2. Experiment Results of the Structure Similarity of Semantic Dependency Trees

In formula 13, A, B are adjustment factors, and $A+B=1$. In order to choose appropriate value of A, B, we obtain the relations between the value of A, B and the accuracy of formula 13, as is shown in Figure 4. From Figure 4, we know that when A=0.7, B=0.3, the accuracy of the structure similarity of dependency trees is the best.

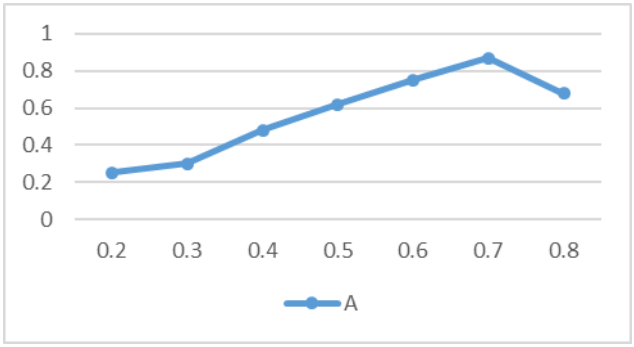

Figure 4. The Relationship Between the Rate of A and the Accuracy of the Structure Similarity of Dependency Trees

4.3.3. Experiment Results of the Short Text Similarity

In formula 20, α, β are adjustment factors, and $\alpha+\beta=1$. In order to choose appropriate value of α, β, we analyze the relations between the value of α, β and the error rate of formula 20, as is shown in Figure 5. (0.4,0.6) is the best choice in the accuracy of short text similarity.

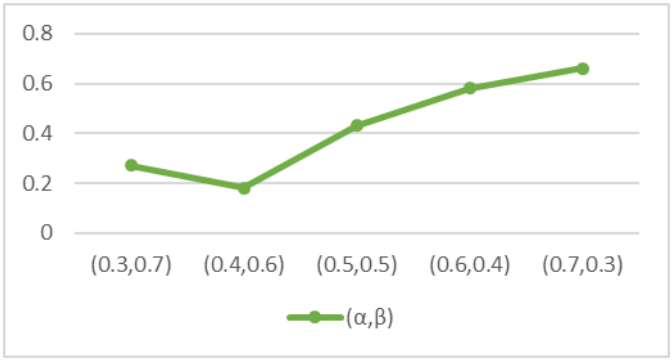

Figure 5. The Relationship Between the Rate of α, β and the Error Rate of the Structure Similarity of Dependency Trees

Figure 6 demonstrates the comparison of the accuracy of DVSM-WDT and other short text similarity algorithms.

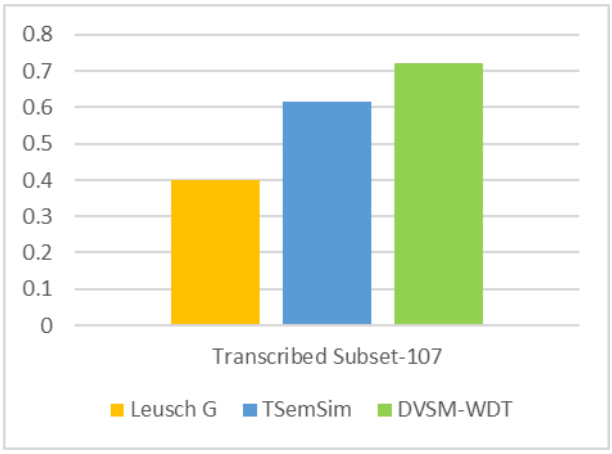

Figure 6. The Comparison of the Accuracy of DVSM-WDT and other Algorithms

From Figure 6, Leusch G algorithm improves the traditional edit distance, but this calculation method is based on block-exchanging and morphological pattern. It consumes a lot of memories and has some limitations. Compared to the traditional TF-IDF algorithm, TSemSim improves the accuracy of short text similarity. However, it focuses on the semantic between words in the text but do not concentrate on the structure information of text. DVSM-WDT combines the semantic and structure information of short texts so that it has a high accuracy in experiments.

5. Conclusion

In this paper, we integrate the semantic information of Wikipedia and the structure information from semantic dependency trees and improve the traditional VSM. We create a new short text similarity method based on double-VSM. In the experiments, the comparison of the proposed method and the traditional methods shows that the novel method has higher accuracy. The new short text similarity calculation model provides a new research goal in the classification and duplicate checking of short texts. In the future, the adjustment factors of algorithm and the short text similarity measure in some special areas can be conducted further.

Acknowledgements

This work is supported by Beijing Forestry University Education Reform Project (Data Structure Resources Sharing Course), Beijing Higher Education Reform (No. 2013-ms047) and Beijing Undergraduate Training Programs for Innovation and Entrepreneurship (No. 201510022051).

References

[1] D. Zhendong and D. Qiang, "HowNet", [EB/OL]. http: //www.keenage.com.
[2] G. A. Miller and G. A. Miller, "WordNet: An on-line lexical database", International Journal of Lexicography, vol. 3, no. 4, **(1990)**, pp. 235-244.
[3] M. Jiaju, Z. Yiming and G. Yunqi, "Synonyms", Shanghai: Shanghai Lexic graphical Publishing House, **(1983)**.
[4] H. Chenghui, Y. Jian and H. Fang, "A Text Similarity Measurement Combining Word Semantic Information with TF-IDF Method", Chinese Journal of Computers, vol. 34, no. 5, **(2011)**, pp. 856-864.
[5] M. Strube and S. P. Ponzetto, "Wikirelate! Computing semantic relatedness using Wikipedia", National Conference on Artificial Intelligence-volume. **(2006)**, pp. 1419-1424.
[6] I. H. Witten and D. Milne, "An Effective, Low-Cost Measure Of Semantic Relatedness Obtained From Wikipedia Links", Proceedings of Aaai, **(2008)**.
[7] R. L. Cilibrasi and P. M. B. Vitanyi, "The Google Similarity Distance", IEEE Transactions on Knowledge & Data Engineering, vol. 19, no. 3, **(2007)**, pp. 370-383.
[8] E. Gabrilovich and S. Markovitch, "Computing semantic relatedness using Wikipedia-based explicit semantic analysis", In Proceedings of the 20th International Joint Conference on Artificial Intelligence, **(2007)**, pp. 1606-1611.
[9] G. Leusch, N. Ueffing and H. Ney, "A novel string-to-string distance measure with applications to machine translation evaluation", Proceedings of Mt Summit IX, **(2003)**, pp. 240-247.
[10] L. Bin, L. Ting, Q. Bing and L. Sheng, "Chinese Sentence Similarity Computing Based on Semantic Dependency Relationship Analysis", Application Research of Computers, vol. 20, no. 12, **(2003)**, pp. 15-17.
[11] R. Li, S. Li and Z. Zhang, "The Semantic Computing Model of Sentence Similarity Based on Chinese FrameNet", (2009) IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology. IEEE Computer Society, **(2009)**, pp. 255-258.
[12] L. Haitao, "Dependency Syntax and Machine Translation", The application of languages, no. 3, **(1997)**, pp. 91-95.
[13] https://code.google.com/p/jwpl

Authors

Ying Liu, she is an undergraduate student in School of Information and Technology, Beijing Forestry University. She is a member of Institute of Artificial Intelligence in Beijing Forestry University. Her main research interests include semantic web and data mining.

Dongmei Li, she received her master degree in Institute of Software, Chinese Academy of Sciences and her Ph.D. degree from Beijing Jiaotong University. Currently she is an associate professor in School of Information and Technology, Beijing Forestry University. Her main research interests include artificial intelligent, knowledge engineering and semantic web.

Cong Dai, she is an undergraduate student in School of Information and Technology, Beijing Forestry University. She joined Institute of Artificial Intelligence in Beijing Forestry University in 2013 as a research assistant. Her main research interests include intelligent information retrieval and semantic web.

