ATLANTIS PRESS

Advances in Intelligent Systems Research, volume 133

2nd International Conference on Artificial Intelligence and Industrial Engineering (AIIE2016)

Text Similarity Based on Semantic Analysis

Junli Wang, Qing Zhou*and Guobao Sun

College of Electronics and Information Engineering, Tong ji University, Shanghai, China

*Corresponding author

***Abstract*—One of the most important challenges in measuring text similarity is language variability: texts with the same meaning can be realized in several ways. A way to address the language variability is the notion of semantic similarity. This paper extracts the relevance of texts and terms through Singular Value Decomposition (SVD). According to Bayesian Network, we construct term-topic sets and then use Mutual Information (MI) to calculate the semantic similarity between terms. Finally, we use graph structures instead of term vectors to calculate text similarity.**

Keywords-semantic similarity; bayesian network; graph model

I. INTRODUCTION

With the rapid development of computer network, text similarity computation has been widely used in various fields. In Machine Translation[1], for example, the semantic similarity evaluates the quality of Machine Translation by measuring the degree of the equivalence between the reference target and the Machine Translation output. The computation of text similarity based on semantic analysis is to use the internal semantic relations of the text. One of the most important Natural Language Processing challenges is the variability of language, in other words, text that has the same meaning can be expressed in several different ways, thus it is necessary to identify the correlation of these different expressions in Natural Language Processing. Semantic similarity is a process of language variation. For instance, in text summarization[2], semantic similarity is used as a metric to select a sentence from a whole paragraph as the summary. The problem of semantic similarity is defined as the measurement and recognition of semantic relations between texts[3].

Latent Semantic Analysis (LSA) uses SVD to describe the relationship between texts, but the results of this method are slightly rough and it needs a great amount of memory. It is applicable for dealing with large-scale corpus coarse classification. In 2002, Google used Bayesian Network to establish links among texts, concepts and keywords, which clustered millions of keywords into some concepts, called Phil Cluster. In 2004, Google reconstructed Phil Cluster and named it Rephil. The use of data was many hundreds of times than the original, and the similarity of keywords was also expanded from co-occurrence in text to co-occurrence in context. Furthermore, they pointed out the concept of different particles. The rest of paper is organized as follows. Section 2 gives the problem statement. Section 3 presents our semantic similarity approach, including topic minding based on Bayesian Network, words similarity matching based on graph structure and word-couples similarity matching based on bipartite graph. Section 4 shows our experiment results. Section 5 summarizes this paper and discusses the future work.

II. PROBLEM STATEMENT

In Natural Language Processing, a text is usually represented by a word vector, and at the same time the linguistic features of word itself play a crucial role in measuring its weight and expressing the text semantic information. Take part-of speech for example, different part-of speeches of word have different contributions in text representation. Therefore, we retain nouns and verbs as the feature words of the text. Given a corpus of $\mathcal{D}$ documents as the input, where d-th document is a sequence of $\mathcal{N}_{\mathrm{d}}$ tokens: $\mathrm{w}_{\mathrm{d\_i}}, \mathrm{i} = 1, \cdots, \mathcal{N}_{\mathrm{d}}$. For convenience we index all the unique words in this corpus using a vocabulary of $\mathcal{M}$ words. And $\mathrm{w}_{\mathrm{d\_i}} = \mathrm{x}, \mathrm{x} \in \{1, \cdots, \mathcal{M}\}$ means that the i-th token in d-th document is the x-th word in the vocabulary.

Given a corpus, our task is to extract the text semantics and compare the semantic relevance between texts. In this paper, we use graph structures to describe the characteristic word sequences of texts. In Algorithm1, we present the graph construction process. Let $\mathcal{D}_{g} = \{\mathrm{g\_1}, \cdots, \mathrm{g\_d}, \cdots\}$ be a graph database where g_d signifies the d-th document. We use $\mathrm{V(g\_d)}$ and $\mathrm{E(g\_d)}$ to denote the vertex set and edge set of graph $\mathrm{g\_d}$, respectively. $|\mathrm{V(g\_d)}|$ and $|\mathrm{E(g\_d)}|$ represent the number of vertices and edges in graph g_d, respectively.

DEFINITION 1. We formally define the graph of d-th document as follows:

- $g\_d = \{[w_{d\_i}, w_{d\_j}, e_{d\_ij}, weight_{d\_ij}] | w_{d\_i}, w_{d\_j} \in V(g\_d), e_{d\_ij} \in E(g\_d)\}$.

Algorithm 1: Graph Construction Process

Input: Corpus with $\mathcal{D}$ documents;
Output: Graph database $\mathcal{D}_{g}$;
for $d \in \mathcal{D}$ **do**
  $s_d \leftarrow$ {sentences of d-th document} $\forall s_{d\_i} \in s_d$;
  $w_d \leftarrow$ {sequence of d's tokens} $\forall w_{d\_i} \in w_d$;
  for $s_{d\_i} \in s_d$ **do**
    $w\_s_d \leftarrow$ {sequence of sentence s_{d_i}} $\forall w\_s_{d\_i} \in w\_s_d$;
    for $w\_s_{d\_i} \in w\_s_d$ **do**
      for $w\_s_{d\_j} \in w\_s_d$ **do**
        $e_{d\_ij} \leftarrow (w\_s_{d\_i}, w\_s_{d\_j})$;
        add(e_{d_ij}) to $E(g\_d)$;
        $\#(w\_s_{d\_i}, w\_s_{d\_j}) \leftarrow$ count(w_{d_i}, w_{d_j});
        for $e_{d\_ij} \in E(g\_d)$ **do**
          $weight_{d\_ij} \leftarrow \#(w_{d\_i}, w_{d\_j}) / (\#(w_{d\_i}) + \#(w_{d\_j}) - \#(w_{d\_i}, w_{d\_j}))$;
          add(g_d) to $\mathcal{D}_{g}$;
          return $\mathcal{D}_{g}$;

In addition, we use Bayesian Network to infer the document's underlying topics. Let $\mathcal{D}_{\mathcal{T}} = \{\mathrm{T\_1}, \cdots, \mathrm{T\_d}, \cdots\}$ be

a word-couple-topic database whereT_d indicates topics that d-th document contains.

DEFINITION 2. We define word-couple as follows:

- Given two words x and y, a word-couple is a combination of two words, denoted as$< x, y >$.
- Let $\mathcal{D}_c = \{< x_1, y_1 >, \cdots, < x_m, y_m >, \cdots\}$express all the word-couples in the corpus.

DEFINITION 3. We define the d-th document's word-couple-topic set T_das follows:

- $T\_d = \{t_{d\_k} | t_{d\_k} = BNT(w_{d\_i}, w_{d\_j})\}$.
- $BNT(x, y)$ is a mapping function that returns the corresponding topic of the word-couple $< x, y >$, where $x, y \in \mathcal{M}$.

In Algorithm2, we present the common word-coupe-topic set construction algorithm based on Bayesian Network.

Algorithm 2: Common Word-Couple-Topic Set Construction

Input: Word-couples $\mathcal{D}_c$, Bayesian Network *BN*;
Output: The common word-couple-topic set $\{(T, BNT(x, y))\}$;
1: **for**$< x, y > \in \mathcal{D}_c$**do**
2: **if**$\exists\, T \in BN$ where $T = BNT(x, y)$**then**
3: add(T, $BNT(x, y)$) to common word-couple-topic set;
4: **else**
5: add a new topic T' to *BN*;
6: $BNT(x, y) \leftarrow T'$
7: add(T', $BNT(x, y)$) to common word-couple-topic set;
8: **return** $\{(T, BNT(x, y))\}$;

The similarity calculation through comparing graph structures is called words similarity matching (WSM). Meanwhile, using word-couple-topics to measure semantic similarity among texts is called word-couples similarity matching, abbreviated to WCSM. Both characteristic words and topics are taken into account their influence on text similarity computation. Our text similarity calculation equation is defined as:

$$similarity(d1, d2) = \alpha WSM(d1, d2) + | \beta WCSM(d1, d2) \quad (1)$$

where α and β are two constants that weight these two similarity measure functions respectively.

III. TEXT SEMANTIC SIMILARITY ALGORITHM

A. Topic Mining based on Bayesian Network

In Natural Language Processing, the two most common classification problems are classified texts according to topics and classified feature words according to topics. SVD can solve these problems once for all. Using SVD to deal with original data matrix, it can simultaneously complete synonyms classification and texts classification. After that, it forms a correlation relation network among texts, topics and characteristic words, which can be described by the Bayesian Network.

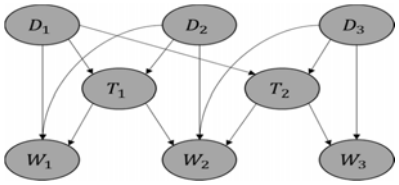

FIGURE I. BAYESIAN NETWORK FOR DESCRIBING DOCMENT(D), TOPIC(T) AND WORD(W).

A topic may contain more than one word, and a word may belong to more than one topic as well. Likewise, a document may correspond to multiple topics, and a topic also corresponds to a number of documents. Additionally, documents always have direct connections with words, and both of them are directly related to the topics, moreover, they have indirect relevance through the topics. However, this kind of word clustering only considers the relationship between words and documents, but less refer to the context of the words, which leads to the result of topic clustering too rough. We use Fp Grough [4] association rules algorithm to extract frequent item sets as word-couple set, and construct word-couple-topic set according to the relationships among documents, topics and words described by the Bayesian Network.

B. Words Similarity Matching

A text with centers and organization structures should be bound to have a certain semantic association in its internal vocabulary. Since using bag-of words model to calculate the similarity of word set directly will ignore the semantic information in a specific context, thus we use graph structures instead of word vectors. We mark the feature words after text preprocessing as nodes and the co-occurrences of words as edges in order to transform the text into a graph structure. This graph reveals the importance of words in the text through the associated relationships that exist between words. For any graphg_1, g_2 $\in \mathcal{D}_g$, WSM can be calculated as:

$$WSM = \gamma Sim\_node(g\_1, g\_2) + (1 - \gamma) Sim\_edge(g\_1, g\_2) \quad (2)$$

where $\gamma \in [0,1]$ is the amount of influence of the node similarity. The node similarity is defined as:

$$Sim\_node(g\_1, g\_2) = \frac{1}{k} \sum_{i=1}^{k} \frac{mcs_i}{max_i} simN_{maxi} \quad (3)$$

Where k is the number of matching nodes between g_1 and g_2 . $\mathrm{simN_{maxi}}$ representsi-th matching nodes' maximum similarity value. $\mathrm{mcs_i}$ and $\mathrm{max_i}$ are i-th matching nodes' maximum common appearance times and maximum individual appearance times, respectively. We quantify the relevance of two words based onMI. The specific solution is: 1) form massive texts, and find out a word vector that each word in the vector is co-occurrence and has a larger MI score with the target word; 2) find out another word vector for the other target word in the same way; 3) calculate the relevance of these two target words by Cosine Theorem. We use a matrix to describe the relevance between two graphs' node sets.

$$M_{g\_12} = \begin{pmatrix} w_{1\_1}w_{2\_1} & \cdots & w_{1\_1}w_{2\_n} \\ \vdots & \ddots & \vdots \\ w_{1\_m}w_{2\_1} & \cdots & w_{1\_m}w_{2\_n} \end{pmatrix} \quad (4)$$

Here $\mathrm{w_{1\_i}w_{2\_j}}$ is the semantic relevance value of node $\mathrm{w_{1\_i}}$ and node $\mathrm{w_{2\_j}}$, and$\mathrm{m} = \mathcal{N}_1, \mathrm{n} = \mathcal{N}_2$. We adopt a greedy

selection process to obtain optimal node matching. The steps are listed as follows.

- **Step1:** Pick out the largest elements of each row in the matrix. Preserve their indexes as $rIndex = \{(1, rMax_1), \cdots, (m, rMax_n)\}$. If there is more than one largest element in a row, we retain all of them. For example, $w_{1\_i}w_{2\_j} = w_{1\_i}w_{2\_j'}, j \neq j'$.
- **Step2:** Pick out the largest elements of each column in the matrix in the same way and preserve their indexes as$cIndex = \{(cMax_1, 1), \cdots, (cMax_m, n)\}$.
- **Step3:** Select elements $(ind\_r, ind\_c) \in rIndex \cap cIndex$, except which the row indexind_ror the column index ind_cappears more than one time in the selection result. Assign the relevance value under the index to two nodes' similarity $simN_{maxi} = M_{g\_12}[ind\_r, ind\_c]$. Then, remove this row and this column fromM_{g_12}.
- **Step4:** Repeat step1 to step3 until there is no combination can be chosen.
- **Step5:** If $\mathrm{M_{g\_12}}$ is not empty, assign all the elements remained to the rest nodes, at this time, a node is always combined with more than one node.

We define the similarity between edge sets as:

$$Sim_{edge(g_1,g_2)} = \frac{\sum \forall E(g_1) \cap E(g_2) \left(\frac{min\left(weight_{1_{ij}}, weight_{2_{i'j'}}\right)}{max\left(weight_{1_{ij}}, weight_{2_{i'j'}}\right)} \right)}{max(|E(g_1)|, |E(g_2)|)} \quad (5)$$

Where $\mathrm{weight_{1\_ij}}$ and $\mathrm{weight_{2\_i'j'}}$ are the weight of edge $\mathrm{e_{1\_ij}}$ and $\mathrm{e_{2\_i'j'}}$ of graph $g_1$and $g_2$respectively, and $\mathrm{e_{1\_ij}}$ and $\mathrm{e_{2_i'j'}}$have the same node labels.

C. Word-coupe Similarity Matching

Based on the Bayesian Network, we have constructed a word-couple-topic database $\mathcal{D}_{\mathcal{T}}$, then any two different documents are represented by two independent sets of topics, WCSM(d1,d2)=WCSM(T_d1,T_d2). [5] introduced a method based on the sequence of vertices' adjacency list of two graphs to calculate the distance between them. The main idea is as follows.

Given two topic sets T_d1 and T_d2, we mark the topics as nodes and construct a bipartite graph $\mathrm{B(T_d1, T_d2)}$with $\mathrm{|V(T\_d1)|}$ nodes on one side and $\mathrm{|V(T\_d2)|}$ nodes on the other, where $\mathrm{V(T\_d1)}$ is the node set of T_d1. We use $\mathrm{b(u)}$ to denote the corresponding node u in $\mathrm{B(T\_d1, T\_d2)}$. For each pair of nodes $\mathrm{u \in V(T\_d1)}$ and $\mathrm{v \in V(T\_d2)}$, there is an edge between b(u) and b(v) in $\mathrm{B(T\_d1, T\_d2)}$), iff sim(u,v)>0 (i.e. u and v have some similarities). For each edge $\mathrm{(b(u), b(v)) \in E\big(B(T\_d1, T\_d2)\big)}$, the weight of edge $\mathrm{(b(u), b(v))}$ is defined as $\mathrm{w(b(u), b(v)) = sim(u, v)}$. After constructing the bipartite graph$\mathrm{B(T_d1, T_d2)}$, we find the maximum weighted bipartite matching $\mathrm{M(T\_d1, T\_d2)}$ of $\mathrm{B(T\_d1, T\_d2)}$ using Hungarian algorithm. Its main discovery loop by iterating is the following three main steps.

- *Construct the initial connections*: For each $u \in V(T\_d1)$ on the left side in $B(T\_d1, T\_d2)$, pick out the greatest similarity node $v \in V(T\_d2)$ on the right side and form a connection $(b(u), b(v))$ between them. If more than one right node meet the condition, then form multiple edges. For each node on the right side in $B(T\_d1, T\_d2)$, do the same as left nodes.
- *Maximum weighted matching*: Hungarian algorithm starts from an unmatched node via searching augmenting path ceaselessly to increase the amount of matching edges and matching nodes until there is no augmenting path any more. In this paper, we make changes to some properties of traditional augmenting path. We define that the terminal point of an augmenting path can be a matched point as well. Additionally, we suppose that the weight of all the odd edges is larger than that of their next edges on the path. Append all the odd edges on the augmenting path to the original matching result, and remove all the even edges append on the path from the original matching, so the number of matching will increase in one. Repeatedly, look for augment path and then form a new match constantly until it is unable to find out any augmenting path. Finally, receive the maximum weighted matching.
- *Iteration*: Recombine the nodes remained and repeat the above two steps until achieve the optimal bipartite matching.

IV. Experiments And Results

A. Datasets and Evaluating Indicators

This paper uses the text categorization corpus provided by Natural Language Processing group of International Database Center of Computer Information and Technology Department of Fudan University, and we select 1114 documents from nine categories of the corpus for the experiments. After word segmentation and removing stop words, there are 11831 words remaining as features.

TABLE I. CATEGORIES AND DOCUMENTS

Category	Documents	Category	Documents	Category	Documents
Art	108	Computer	148	Economy	140
History	107	Environment	125	Politics	129
Space	102	Agriculture	151	Sports	104

In order to investigate the effect of text similarity algorithm proposed in this paper, we regard it as the basis of text clustering. Recall and precision are commonly used in evaluating the effect of classification algorithms, but there is no corresponding relation between machine clustering result and artificial classification result in the clustering, therefore we cannot directly use these two evaluation standards. In the literature [6], authors use the averaged accuracy (AA) as evaluation criterion, which evaluates the effect of clustering by examining whether the class relation between any two texts is consistent with the expected results. The class relation between two texts can be divided into the following four circumstances.

TABLE II. THE CLASS RELATION SIGN BETWEEN TWO TEXTS

Belong to the same class in Machine Clustering	**Belong to the same class in artificial classification**	**Sign**
Yes	Yes	a
Yes	No	b
No	Yes	c
No	No	d

The AA is defined as the arithmetic mean of the positive accuracy (PA) and the negative accuracy (NA), i.e. $AA = (PA + NA)/2$, where $PA = a/(a + c)$and$NA = d/(b + d)$.

B. Results

In order to verify the validity of the text similarity algorithm proposed in this paper, our algorithm is compared with the LSA model and the Maximum Common Subgraph (MCS)[7] model. The formula for calculating the similarity of text in LSA is:

$$A = U\Sigma V^T \tag{6}$$

$$L = U\Sigma^{-1} \tag{7}$$

$$\mathrm{Sim}(doc_1, doc_2) = cos(L^T doc_1, L^T doc_2) \tag{8}$$

where A is decomposed into the product of three matrices U, Σ and V^T. The column vectors of Uand V are orthogonal and normalized. Each term corresponds to a row vector of U, and each text corresponds to a column vector of V^T.

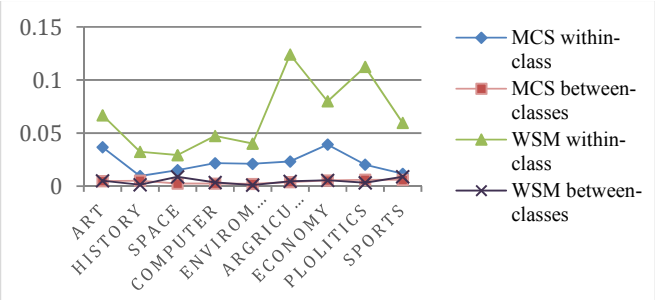

FIGURE II. SIMILARITY CALCULATION OF MCS AND WSM.

The text representation of WSM is similar to that of the MCS algorithm proposed in literature [7]. From Figure 2, we find the text similarity values based on graph structure are quite small. The main reasons lie in: 1) the test data are all long documents, each of whom always has hundreds or even thousands of terms, so the graph of each document is rather large; 2) due to the similarity of terms is ignored in WSM and MCS, the matching of nodes and edges is string matching essentially, thus the matching is quite less without semantics. For the text similarity algorithm, the similarity score of within-class should be as large as possible, and the similarity score of between-classes should be as small as possible. Obviously, in Figure 2, the average within-class similarity of WSM is larger than that of MCS, and the average between-classes similarity of WSM is approximately equal to that of MCS, moreover the differences between these two average scores of WSM are greater than those of MCS.

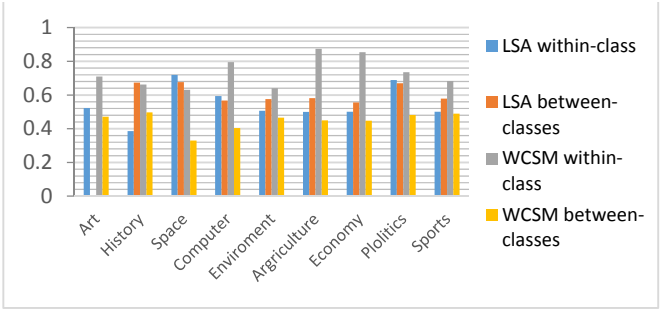

FIGURE III. SIMILARITY CALCULATION OF LSA AND WCSM.

In this paper, we construct word-couple-topic sets based on Bayesian Network and adopt bipartite graph to match topic sets. As shown in Figure 3, for LSA the average between-classes similarity is approximated to zero in Art category, while the average between-classes similarity is much larger than the average within-class similarity in History category. Therefore, we think that LSA is unstable in calculating text similarity. And the differences of the two average scores of WCSM are almost greater than those of LSA except the differences in Art and History. We use the PAM clustering method to carry on the further experiment analysis. The average accuracy of clustering results is shown in Table III.

TABLE III. AVERAGE ACCURACY OF PAM CLUSTERING

	LSA	**MCS**	**WSM**	**WCSM**	**αWSM+βWCSM**
AA	52.65%	63.13%	73.39%	78.96%	84.73%

From Table III, our similarity method is effective. As the weights of two similarity-matching methods, α and β embody the influence degree of word set and word-couple set on text similarity calculation. Due to the large difference of similarity between WSM and WCSM, after many experiments it is found that when $\alpha = 6.45$ and $\beta = 0.58$ we may get a better AA score of clustering.

V. CONCLUSION AND FUTURE WORK

In this paper, we focus on measuring the semantic similarity of text based on word set matching and word-couple set matching. Compared with ordinary LSA and simple MCS models, experiments have proved the effectiveness of our approach.There is no denying that our algorithm still has some defects. Firstly, we only extract the nouns and verbs after pretreatment as the feature words roughly, which makes the vocabulary huge and not simplification enough and lastly leads to the low similarity scores. Secondly, the bipartite graph-matching algorithm has high time complexity. Finally, the determination of the weight α and β, which reflect the influence degree of word set and word-couple set on the text similarity matching, is artificial so that it is not very reliable and accurate. In future work, we will try to improve our approach based on the above three points.

ACKNOWLEDGMENT

This work is supported by the Science and Technology Committee of Shanghai Support Project under grant No.14JC1405800, the Project of the Central Universities Fundamental Research of Tongji University.

REFERENCES

[1] S. Banerjee and A. Lavie,"METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization, 2005.

[2] D. Das and A. F. T. Martins,"A survey on automatic text summarization," Literature Survey for the Language and Statistics II course at Carnegie Mellon University, 2007.

[3] V. Rus, M. Lintean, R. Banjade, N. Niraula and S. Dan,"SEMILAR: The semantic similarity toolkit," in 51st Annual Meeting of the Association for Computational Linguistics, 2013.

[4] J. Han, J. Pei, Y. Yin, and R. Mao, "Mining frequent patterns without candidate generation: a frequent tree approach,"Data mining and knowledge discovery, vol. 8, pp. 53-87, January 2004.

[5] Y. Zhu, L. Qin, J. X. Yu, and H. Cheng,"Finding top-k similar graphs in graph databases," in Proceedings of the 15th International Conference on Extending Database Technology. ACM, pp. 456-467, March 2012.

[6] M. Iwayama and T. Tokunaga, "Hierarchical Bayesian clustering for automatic text classification,"Proceedings of the 14th international joint conference on Artificial intelligence-Volume 2, 1995, pp. 1322-1327.J.

[7] N. Wu and F. Q. Liu, "Research on text similarity computing based on max-common subgraphs," Journal of the China Society for Scientific and Technical Information, 2010, vol. 29, pp. 785-791[in Chinese].