Word Embedding and Topic Modeling Enhanced Multiple Features for Content Linking and Argument/Sentiment Labeling in Online Forums

Lei Li and **Liyuan Mao** and **Moye Chen**
Center for Intelligence Science and Technology (CIST)
School of Computer Science
Beijing University of Posts and Telecommunications (BUPT) , China
leili@bupt.edu.cn circleyuan@bupt.edu.cn moyec@bupt.edu.cn

Abstract

Multiple grammatical and semantic features are adopted in content linking and argument/sentiment labeling for online forums in this paper. There are mainly two different methods for content linking. First, we utilize the deep feature obtained from Word Embedding Model in deep learning and compute sentence similarity. Second, we use multiple traditional features to locate candidate linking sentences, and then adopt a voting method to obtain the final result. LDA topic modeling is used to mine latent semantic feature and K-means clustering is implemented for argument labeling, while features from sentiment dictionaries and rule-based sentiment analysis are integrated for sentiment labeling. Experimental results have shown that our methods are valid.

1 Introduction

Comments to news and their providers in online forums have been increasing rapidly in recent years with a large number of user participants and huge amount of interactive contents. How can we understand the mass of comments effectively? A crucial initial step towards this goal should be content linking, which is to determine what comments link to, be that either specific news snippets or comments by other users. Furthermore, a set of labels for a given link may be articulated to capture phenomena such as agreement and sentiment with respect to the comment target.

Content linking is a relatively new research topic and it has attracted the focus of TAC 2014 (https://tac.nist.gov//2014/KBP/), BIRNDL 2016 (Jaidka et al., 2016) and MultiLing 2015 (Kabadjov et al., 2015) and MultiLing 2017. The main method is based on the calculation of sentence similarity (Aggarwal and Sharma, 2016; Cao et al., 2016; Jaidka et al., 2016; Saggion et al., 2016; Nomoto, 2016; Moraes et al., 2016; Malenfant and Lapalme, 2016; Lu et al., 2016; Li et al., 2016; Klampfl et al., 2016), with the key point of mining semantic information better.

Researchers have tried various features and methods for sentiment and argument labeling. The main features are different kinds of sentiment dictionaries, while the basic method is the rule-based one. The major method for sentiment and argument labeling is based on statistical machine learning algorithms (Aker et al., 2015; Hristo Tanev, 2015; Maynard and Funk, 2011).

2 Task Description

We work on three tasks for English and Italian in this paper. The first one is content linking, which is to find all the linking pairs for comment sentences. In every pair, one sentence belongs to the original article or a former comment by an author, the other belongs to a comment by a later commentator. The second and third tasks are to tag two kinds of labels to the linking pairs that were found in the first task. Labels involve argument label and sentiment label. For argument label, it focuses on whether or not a commentator agrees with the commentated author. For sentiment label, it cares about the sentiment of comment sentences. Experiments are implemented on the training data released by MultiLing 2017, including 20 English news (from The Guardian) and 5 Italian (from Le Monde) news with some comments.

3 Methods

For content linking, we adopt the Word Embedding Model to dig up word vectors as linking information of sentence pair with deeper semantic fea-

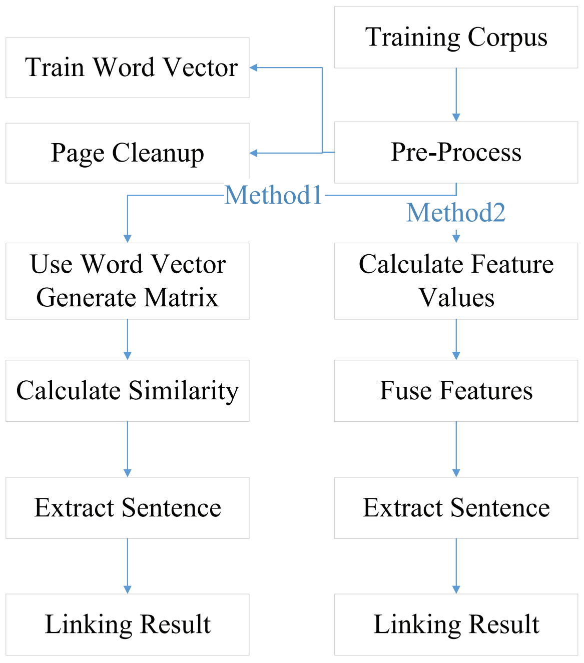

Figure 1: Content linking process

tures. Besides, we also use some traditional features of sentence similarity which performed well through experiments and explore how to fuse them together with Word Embedding features. For this purpose, first, we try to use every single feature to get one linking sentence, next, we choose the most repetitive sentence as final result via a voting method. Then we mainly use rule-based sentiment analysis to obtain the sentiment label. LDA (Latent Dirichlet Allocation) (Blei et al., 2003) topic model and K-means (Hartigan and Wong, 1979) are integrated to obtain the argument label.

3.1 Content Linking

Figure 1 shows the process for content linking.

3.1.1 Pre-Processing

We crawl 1.5G data from the English Guardian website to train word vectors for English, and about 1G data from Wikipedia for Italian. Then we use the tool named word2vec (Mikolov et al., 2013) for training.

3.1.2 Method 1-Word Vector Algorithm

After the training of word embedding models, a sentence in the corpus can be expressed as:

$$W_t = (w_t, w_{t+1}, \cdots, w_{t+k}) \tag{1}$$

Where w_t is the word vector of 300 dimensions of word t. Then two sentences W_i and W_j can form a calculating matrix $M_{i,j}$:

$$M_{i,j} = W_i W_j^T = \begin{bmatrix} w_t w_v & \cdots & w_t w_{v+l} \\ \vdots & \ddots & \vdots \\ w_{t+k} w_v & \cdots & w_{t+k} w_{v+l} \end{bmatrix} \tag{2}$$

Before the computation of (w_t, w_v), we need some processing steps: stemming as well as stop words and punctuation removing. Besides, it is essential to check relations between word t and word v based on WordNet. If they exist in the hyponyms/hypernyms part of each other, they can be seen as the same.

The cosine distance can represent (w_t, w_v), and the similarity of sentences i and j is:

$$Sim_{i,j} = \frac{\sum_{m=i,n=j} max(M_{m,n})}{\sqrt{length_i length_j}} \tag{3}$$

Where $maxM_{m,n}$ is obtained through the following concrete steps. First, find out the maximum of $M_{i,j}$, then delete the row and column of the maximum. Next, find the maximum of the remaining matrix and remove row and column like the former step. Do the same procedure until the matrix is empty. Finally add up all the maximum values. $length_i$ is the number of word vectors in the sentence, and $\sqrt{length_i length_j}$ is used to reduce the influence of sentence length.

We think that the maximum value in the matrix can represent the most matching word pairs in the two sentences. We just choose the maximum value in each step and delete the word pairs in the matrix for next iteration until the matrix is empty. As a result, we can find out all the best matching word pairs in the two sentences. Hence accumulation of word similarities of all these best matching word pairs can represent the similarities of the two sentences.

Based on the above sentence similarities, we can extract those sentences with the highest similarity to a comment sentence as its linking result.

3.1.3 Method 2-Feature Fusion Algorithm

This algorithm is only for English. We have two kinds of features, one is from lexicons, and the other is from sentence similarities.

We have three lexicons: Linked Text high-frequency word lexicon (Lexicon 1), LDA lexicon (Lexicon 2), Comment Text and Linked Text co-occurrence lexicon (Lexicon 3). For Lexicon 1, we pick up the words with high frequency from standard answers artificially, and then expand them

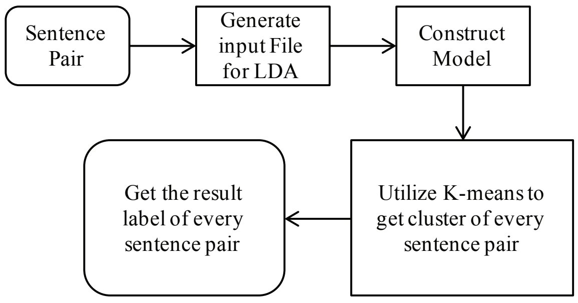
Sentence Pair
Generate input File for LDA
Construct Model
Utilize K-means to get cluster of every sentence pair
Get the result label of every sentence pair

Figure 2: Argument label process

through WordNet and word vectors, resulting in a lexicon. For Lexicon 2, we use LDA model to train the news and comments to get a lexicon of 25 latent topics in every file independently. For Lexicon 3, we obtain the co-occurrence degree between words by the word frequency statistics of comment and its linked sentence from the training corpus.

As for sentence similarities, we have word vector similarity, jaccard similarity, idf similarity, res similarity, jcn similarity and path similarity. Word vector similarity is calculated through Method 1. We add up the idf values of the same words between two sentences to represent their idf similarity. The last three similarities are from WordNet.

For every feature, we can use it to get a sentence with the highest score. Then among these nine sentences chosen by nine features, we use voting method to choose the most repetitive sentence as the final linking result. When some sentences get the same votes, we choose the first one according to sentence order in the input news and comments.

3.2 Argument Label

Figure 2 shows the process for argument label.

Given a collection of sentences in the input file, we wish to discover topic distribution of every sentence through LDA model. We generate the input file for LDA first. For every sentence, we change it into its bag-of-words model representation, which assumes that the order of words can be neglected. During LDA modeling, we set the topic number to 15 according to the experiments. That is to say, later in K-means clustering, our feature is the 15-dimension vector. We run K-means to cluster all sentences into two categories. For every sentence pair, if the two sentences belong to the same category, then we set the label to in_favour, else, against.

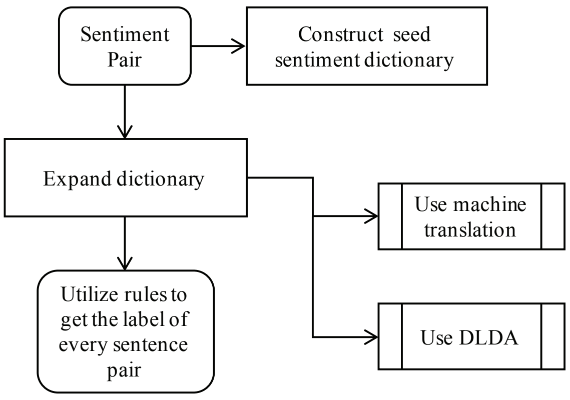
Sentiment Pair
Construct seed sentiment dictionary
Expand dictionary
Use machine translation
Use DLDA
Utilize rules to get the label of every sentence pair

Figure 3: Sentiment label process

3.3 Sentiment Label

Figure 3 shows the process for sentiment label.

There are three kinds of seed sentiment dictionaries discovered from OpinionFinder system (MPQA, http://mpqa.cs.pitt.edu/). One is subjectivity lexicon, the other two are called Intensifier and Valenceshifters lexicon. Intensifier lexicon involves words which can improve the sentiment level. Valenceshifters lexicon involves words which can alter the sentiment label.

The original dictionary is in English. We use machine translation to add Italian vocabularies. With DLDA (Chen et al., 2014), we can get all sentiment weights of words in corpus. At last, the word which is not included in seed has the same polarity with a seed word if their sentiment weight distance can be ignored.

Through DLDA, every word gets a sentiment state. We map the sentiment state to a number of word score as in Table 1. We accumulate word score in a sentence to obtain the sentence score, which is then mapped to the sentiment label as in Table 2.

Sentiment state	**Word score**
Weak neg(only)	-1
Strong neg(only)	-2
Strong pos(only)	2
Weak pos(only)	1
Neutral	0
Intensifier+weak neg	-2
Intensifier+weak pos	2

Table 1: Scoring strategy

Note that when current sentence score is bigger than 0 and current word is in Valenceshifters and the score of current word is less than 0, sentence score = sentence score * (-1), or current sentence score is less than 0 and current word is in Valenceshifters and the score of current word is more than 0, sentence score strategy is the same. For any other conditions, we simply accumulate the word score.

Sentence final score	**Label**
>0	Positive
=0	Neutral
<0	Negative

Table 2: Mapping sentence score to sentiment label

4 Experiments

4.1 Content Linking

Threshold is used for extracting sentence. We choose the sentence as linking result only when the score (for Method 1) or the vote (for Method 2) is bigger than the threshold.

Thres hold	**Linking Precision**	**Thres hold**	**Linking Precision**
0	77.9	0.5	81.9
0.1	78.2	0.6	80.6
0.2	80.8	0.7	84.2
0.3	80.6	0.8	87.0
0.4	80.8	0.9	87.8

Table 3: The performance of Method 1

Table 3 and Table 4 show the performance of Method 1 and Method 2 in our experiments respectively. The first and third rows in Table 4 are the threshold. F1 to F9 refer to 9 features respectively (word vector, jaccard, idf, res, jcn, path, lexicon 2, lexicon 1 and lexicon 3) and the number means the vote for corresponding feature.

From Table 3, we can find out that, for Method 1, the bigger threshold usually can bring the higher precision. But the sentences we obtain may be fewer, too. This will cause low recall rate. According to the precision evaluation method used by MultiLing 2015, precision of 86 is high. Thus we can have good precision here. For Method 2 in Table 4, although its precision is a little lower than that of Method 1, it can also achieve good result. Lexicon 3 shows its good performance, other features like jaccard and idf perform well, too. Hence, how to combine them is important for us in the future. Besides, the linking precision of Italian is 10.1 with the threshold of 0.6 as shown in Table 5.

Threshold	**2**	**2**	**2**
F1	1	0	1
F2	1	1	1
F3	1	0	0
F4	1	0	0
F5	1	1	0
F6	1	0	0
F7	1	0	0
F8	1	0	0
F9	1	2	2
Linking Precision	**78.4**	**85.2**	**85.2**
Threshold	**3**	**3**	**3**
F1	1	1	1
F2	1	0	1
F3	1	0	1
F4	1	1	0
F5	1	0	0
F6	1	0	0
F7	1	0	1
F8	1	0	0
F9	1	2	2
Linking Precision	**78.2**	**82.7**	**82.6**

Table 4: The performance of Method 2

Thres hold	**Linking Precision**	**Thres hold**	**Linking Precision**
0.3	8.14	0.5	8.8
0.4	8.25	0.6	10.1

Table 5: The performance for Italian

4.2 Argument and Sentiment Label

From Table 6, we can find out that when we set the threshold at 0.2 and 0.3, we can get the highest precision in both argument label and sentiment label. However, unlike the linking precision mentioned above, the bigger thresholds result in lower precision. The reason may be that when we set a bigger threshold, the linking sentences we obtain are much fewer. Sometimes we can only get one or two sentence pairs. If there are any wrong answers in the results, it will obviously decrease the precision.

Thres hold	Argu ment	Senti ment	Thres hold	Argu ment	Senti ment
0	85.9	77.6	0.5	76.7	78.3
0.1	86.1	79.6	0.6	67.6	60.7
0.2	86.3	79.1	0.7	52.9	52.0
0.3	84.8	81.1	0.8	47.6	59.1
0.4	80.5	75.4	0.9	47.3	58.9

Table 6: The performance of Labeling

5 Conclusion

For content linking, our system has tried to mine both syntactic and semantic information, and the performances are good. For argument and sentiment labeling, we focus on machine learning algorithm and sentiment dictionaries. And there is still space for us to improve. Our future work is to find some better ways to mine and use more semantic features for both content linking and labeling.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 91546121, 71231002 and 61202247; National Social Science Foundation of China under Grant 16ZDA055; EU FP7 IRSES MobileCloud Project 612212; the 111 Project of China under Grant B08004; Engineering Research Center of Information Networks, Ministry of Education; the project of Beijing Institute of Science and Technology Information; the project of CapInfo Company Limited.

References

Peeyush Aggarwal and Richa Sharma. 2016. Lexical and syntactic cues to identify reference scope of citance. In *BIRNDL@ JCDL*, pages 103–112.

Ahmet Aker, Fabio Celli, Adam Funk, Emina Kurtic, Mark Hepple, and Rob Gaizauskas. 2015. Sheffield-trento system for sentiment and argument structure enhanced comment-to-article linking in the online news domain.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Ziqiang Cao, Wenjie Li, and Dapeng Wu. 2016. Polyu at cl-scisumm 2016. In *BIRNDL@ JCDL*, pages 132–138.

Xue Chen, Wenqing Tang, Hao Xu, and Xiaofeng Hu. 2014. Double lda: A sentiment analysis model based on topic model. In *International Conference on Semantics, Knowledge and Grids*, pages 49–56.

J. A. Hartigan and M. A. Wong. 1979. Algorithm as 136: A k-means clustering algorithm. *Applied Statistics*, 28(1):100–108.

Alexandra Balahur Hristo Tanev. 2015. Tackling the onforums challenge.

Kokil Jaidka, Muthu Kumar Chandrasekaran, Sajal Rustagi, and Min-Yen Kan. 2016. Overview of the cl-scisumm 2016 shared task. In *BIRNDL@ JCDL*, pages 93–102.

Mijail Kabadjov, Josef Steinberger, Emma Barker, Udo Kruschwitz, and Massimo Poesio. 2015. Onforums: The shared task on online forum summarisation at multiling'15. In *Forum for Information Retrieval Evaluation*, pages 21–26.

Stefan Klampfl, Andi Rexha, and Roman Kern. 2016. Identifying referenced text in scientific publications by summarisation and classification techniques. In *BIRNDL@ JCDL*, pages 122–131.

Lei Li, Liyuan Mao, Yazhao Zhang, Junqi Chi, Taiwen Huang, Xiaoyue Cong, and Heng Peng. 2016. Cist system for cl-scisumm 2016 shared task. In *BIRNDL@ JCDL*, pages 156–167.

Kun Lu, Jin Mao, Gang Li, and Jian Xu. 2016. Recognizing reference spans and classifying their discourse facets. In *BIRNDL@ JCDL*, pages 139–145.

Bruno Malenfant and Guy Lapalme. 2016. Rali system description for cl-scisumm 2016 shared task. In *BIRNDL@ JCDL*, pages 146–155.

Diana Maynard and Adam Funk. 2011. Automatic detection of political opinions in tweets. In *Extended Semantic Web Conference*, pages 88–99. Springer.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Computer Science*.

Luis Moraes, Shahryar Baki, Rakesh Verma, and Daniel Lee. 2016. University of houston at cl-scisumm 2016: Svms with tree kernels and sentence similarity. In *BIRNDL@ JCDL*, pages 113–121.

Tadashi Nomoto. 2016. Neal: A neurally enhanced approach to linking citation and reference. In *BIRNDL@ JCDL*, pages 168–174.

Horacio Saggion, Ahmed AbuRa'ed, and Francesco Ronzano. 2016. Trainable citation-enhanced summarization of scientific articles. In *BIRNDL@ JCDL*, pages 175–186.