CIST System for CL-SciSumm 2016 Shared Task

Lei Li, Liyuan Mao, Yazhao Zhang, Junqi Chi, Taiwen Huang, Xiaoyue Cong, Heng Peng

Center for Intelligence Science and Technology (CIST), School of Computer Science and Technology,
Beijing University of Posts and Telecommunications (BUPT), China
leili@bupt.edu.cn, circleyuan@bupt.edu.cn, yazhao@bupt.edu.cn, 709722796@qq.com,
zhidao2010@bupt.edu.cn, cxy0105@bupt.edu.cn, penghengp1@bupt.edu.cn

Abstract. This paper introduces the methods and experiments applied in CIST system participating in the CL-SciSumm 2016 Shared Task at BIRNDL 2016. We have participated in the TAC 2014 Biomedical Summarization Track, so we develop the system based on previous work. This time the domain is Computational Linguistics (CL). The training corpus contains 20 topics from Training-Set-2016 and Development-Set-Apr8 published by CL-SciSumm 2016. As to Task 1A and 1B, we mainly use rule-based methods with various features of lexicons and similarities; meanwhile we also have tried the machine learning method of SVM. As to Task 2, hLDA topic model is adopted for content modeling, which provides us knowledge about sentence clustering (subtopic) and word distributions (abstractiveness) for summarization. We then combine hLDA knowledge with several other classical features using different weights and proportions to evaluate the sentences in the Reference Paper from its cited text spans. Finally we extract the representative sentences to generate a summary within 250 words.

1 Introduction

With the rapid development of Computational Linguistics (CL), the scientific literature of this domain has grown into a rich, complex, and continually expanding resource. Literature surveys and review articles in CL do help readers to gain a gist of the state-of-the-art in research for a topic. However, literature survey writing is labor-intensive and a literature survey is not always available for every topic of interest. What are needed, are resources which can automate the synthesis and updating of text summarization of CL research papers. The CL-SciSumm 2016 (The 2nd Computational Linguistics Scientific Document Summarization Shared Task) has highlighted the challenges and relevance of the scientific summarization problem.

In this paper, we describe our strategies, methods and experiments applied for CL-SciSumm 2016. As to Task 1A, we firstly use different combination methods and strategies based on various feature rules of different lexicons and similarities to identify the spans of text (cited text spans) in the RP (Reference Paper). Then we also have tried the machine learning method of SVM (Support Vector Machine). As to Task 1B, we also use feature rules as a basis. Besides, SVM is used to judge which facet the cited text span belongs to. A voting method is also used to integrate different candidate results. And for Task 2, we firstly adopt hLDA (hierarchical Latent Dirichlet Allocation) topic model for document content modeling. The hLDA tree can provide us good knowledge about latent sentence clustering (subtopic in the document) and word distributions (abstractiveness of words and sentences) for summarization. Then we score the sentences in the RP according to several features including hLDA ones and extract candidate sentences to generate the final summary.

2 Related Work

There are many researches about document summarization [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20]. LDA has been widely applied [21,22]. Some improvements have been made [23,24,25]. One is to relax its assumption that topic number is known and fixed. [26] provided an elegant solution. [27] extended it to exploit the hierarchical tree structure of topics, hLDA, which is unsupervised method in which topic number could grow with the data set automatically. This could achieve a deeper semantic model similar with human mind and is especially helpful for summarization. [28] provided a multi-document summarization based on supervised hLDA with competitive results.

However, it has the disadvantage of relying on ideal summaries. [29] provided a contrastive theme summarization based on hLDA and SDPPs, which is sensitive to negative correlation.

In recent years, interest about information extraction and retrieval from scientific literature has increased considerably. Some researches [30] have shown that citations may contain information out of the abstracts provided by the authors. However, little work has been done on automatic gist extraction from research papers and their corresponding citation communities.

In order to identify the linkage efficiently between a paper citation and its cited text spans in the RP, we need to catch the deep meaning of natural language sentences. In fact, digging the deep meaning of sentences also has an important sense for information extraction and retrieval. Besides the traditional methods for sentence similarity calculation, recently the open-source tool released by Google -- word2vec [31] has a good performance in word semantic mining. And doc2vec [32] has promoted the information mining to the sentences.

3 Task Description

There are two tasks in the CL-SciSumm 2016. Testing dataset, development dataset and training dataset, each contains 10 topics. Every topic consists of one reference paper (RP), some citing papers (CP) and one annotation file. There are five facets pre-defined, including Aim_Citation, Method_Citation, Results_Citation, Implication_Citation and Hypothesis_Citation. In Task 1, we need to identify the spans of text (cited text spans) in the RP that most accurately reflect the citance (Task 1A), and what facet of the paper that each cited text span belongs to, from the predefined set of five facets (Task 1B). Task 2 demands us to generate a structured summary of the RP from the cited text spans of the RP. The length of the summary should not exceed 250 words. Please refer to [8] for more details.

4 Methods

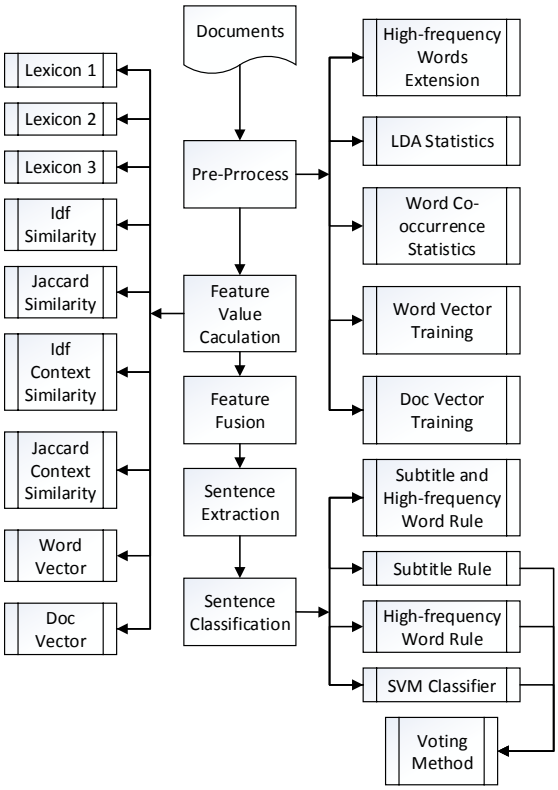

Fig. 1. Framework for Task 1

4.1 Task 1

Our main method is the rule-based method. Fig. 1 shows the framework for Task 1.

Task 1A.

We need to identify the linkage between a paper citation in the CP and its cited text spans in the RP. We think that the linkage is mainly represented by similar meaning between sentences. Hence our work is focused on sentence similarity computing based on various features.

Feature Extraction.

We have two kinds of features, one is from lexicons, and the other is from sentence similarities.

- Three Lexicons:

1. Reference Text high-frequency words (Lexicon 1): We picked up the words with high frequency from reference text in the training corpus artificially, and then expanded them through Wordnet and word vectors.
2. LDA lexicon (Lexicon 2): We used LDA (Latent Dirichlet Allocation) model to train the reference paper and citing papers to get a lexicon of 30 latent topics for files in every topic independently. Table 1 displays the relation of F-measure and the number of latent topics for the training corpus. We can see that the best performance is obtained when the number of latent topics is set to 30.

The number of topics	F-measure
10	0.03924
15	0.0493
20	0.03924
25	0.0503
30	**0.05131**
35	0.04527
40	0.04225
45	0.04024
50	0.03924

Table 1. The performance of LDA

3. Citation Text and Reference Text co-occurrence lexicon (Lexicon 3): We obtained the co-occurrence degree between words by the word frequency statistics of citation text and its reference text from the training corpus.

- Similarity between two sentences:

1. Idf similarity: We add up the idf (inverse document frequency) values of the same words between two sentences.

$$idf_i = log \frac{|D|}{\{j:t_i \in d_j\}} \quad (1)$$

where $|D|$ is the number of sentences, $\{j: t_i \in d_j\}$ is the number of the sentences which contain the word t_i.

2. Jaccard similarity: We use the division between the intersection and the union of the words in two sentences.

$$J = \frac{A \cap B}{A \cup B} \quad (2)$$

where A and B represent two sentences respectively.

- Similarity between two contexts:

The context of a sentence plays an important role in semantic parsing, so we calculate the context similarity.

$$SC_i = \sqrt{d_{i-1} * d_{i+1}} \quad (3)$$

Where SC_i is the context similarity of $sentence_i$, d_{i-1} and d_{i+1} is the imilarity of the sentence before and after it.

We also obtained two kinds of context similarities: Idf context similarity and Jaccard context similarity.

- Two vector similarities:

1. Word vector: We trained every word as a vector with 200 dimensions using Word2Vec. Then a sentence is:

$$W_i = (w_t, w_{t+1}, \cdots, w_{t+k}) \tag{4}$$

where w_t is the word vector and W_i is the vector set to represent a sentence. The vector sets of two sentences W_i and W_j can form a matrix $M_{i,j}$:

$$M_{i,j} = W_i W_j^T = \begin{bmatrix} w_t w_v & \cdots & w_t w_{v+l} \\ \vdots & \ddots & \vdots \\ w_{t+k} w_v & \cdots & w_{t+k} w_{v+l} \end{bmatrix} \tag{5}$$

where (w_t, w_v) is the cosine similarity of w_t and w_v, then the similarity of $sentence_i$ and $sentence_j$ is:

$$Sim_{i,j} = \frac{\sum_{m=i,n=j} max(M_{m,n})}{\sqrt{length_i length_j}} \tag{6}$$

where $length_i$ and $length_j$ represents the length. $max(M_{m,n})$ is the maximum of matrix $M_{m,n}$.

2. Doc vector: We represented every sentence as a vector with 200 dimensions by doc2vec, and used the cosine similarity between vectors to represent the sentence similarity.

We tested the performance of every feature independently for the training corpus as in Table 2. As we can see that Jaccard similarity performs the best.

Feature	F-measure
Lexicon 1	0.01793
Lexicon 2	0.05131
Lexicon 3	0.28437
Idf similarity	0.10589
Jaccard similarity	**0.11167**
Idf context similarity	0.07344
Jaccard context similarity	0.07042
Word vector	0.07771
Doc vector	0.05231

Table 2. The performance of single feature

Methods for linkage.

- SVM Method:

After we have obtained the above features, the first method we thought was to train a classifier based on SVM. But there is a problem of unbalanced data for training. According to our statistics, the number of negative samples is 125 times of the number of positive samples. So we divided the negative samples into 125 groups and train each group with all positive samples. Then using the trained 125 SVM models, we could predict the testing data and get 125 results. Finally, we choose the linkage sentences through a voting system based on these 125 results. When we tried this method on the testing data, unfortunately the performance was not so good as feature rule-based methods. As for the reason, we thought that the number of positive samples was too small, and every training set didn't have enough data to train a good model. To find a better performance, we tried the following methods.

- Voting Method:

Focusing on the F-measure of every feature, we tried different weights and proportions to combine them through experiments, and then got two results through a voting system.

Feature	Weight	Proportion
Idf similarity	1	8
Jaccard similarity	1	12

Jaccard context similarity	1	12
Word vector	1	10

Table 3. The parameters of V1.0

Feature	Weight	Proportion
Lexicon 2	0.3	2
Lexicon 3	0.4	2
Idf similarity	1	7
Jaccard similarity	1	3
Idf context similarity	0.5	4
Jaccard context similarity	0.5	8
Word vector	0.5	8
Doc vector	0.3	4

Table 4. The parameters of V2.0

Finally, the text span with the highest-number of votes is chosen as the citation text corresponding sentences in the reference paper. Table 3 and 4 show the two best combinations for the training corpus.

- Jaccard Focused Method:

From Table 2, we found out that Jaccard similarity behaves better than other features obviously. So we chose it as the major feature, and added other features as supplementary in this method. Table 5 shows the parameter setting.

Feature	weight	proportion
Jaccard similarity	10-fold of Jaccard value	7
Idf similarity	0.7	15
Jaccard context similarity	0.7	15
Idf context similarity	0.5	15
Word vector	0.5	25
Lexicon 3	0.2	25

Table 5. The parameters of Jaccard Focused

- Jaccard Cascade Method:

We chose the sentences with top two Jaccard values as the basic answer, then combined other features to find other two sentences with highest values as the improved answer in this method. Table 6 shows the parameter setting.

Feature	weight	proportion
Idf similarity	1.5	16
Jaccard context similarity	1.5	15
Idf context similarity	1	18
Lexicon 3	0.5	15

Table 6. The parameters of Jaccard Cascade

Task 1B.

- Rule-based method

1. Subtitle Rule: First of all, we examine whether the subtitles of reference sentences and cite sentences contains the following facet words: Hypothesis, Implication, Aim, Results and Method. If the subtitle contains any one of these words, it will be directly classified as the corresponding facet. If it contains more than one of these words, it will be classified into all the facets. Else if it contains none of them, we just classify it as the facet of Method.
2. High Frequency Word Rule: According to the High Frequency Word Rule, we firstly count the High Frequency Word of five facets from the Training Set and the Development Set. In order to improve the coverage of sentences, we expanded the High Frequency Word to get some similar words of each facet. We set an appropriate threshold for each facet. If the number of the High Frequency Word of any facet in the sentence is more than the corresponding facet threshold, then we just use the facet whose coverage is the highest as the final class. If some

facets' coverage are same, then we just classify according to the sequence of Hypothesis, Implication, Aim，Results and Method. If all facets have not reached the threshold of each facet, we classify it as the Method.

3. Combine Subtitle and High Frequency Word Rule: We firstly use the Subtitle Rule to classify the testing set. If the results are not in the five facets of Hypothesis, Implication, Aim, Results and Method, then we use the High Frequency Word Rule to get the final facet.

- SVM Classifier

We extract four features of each class. 1) Location of Paragraph: the order number of the paragraph in which the sentence is located. 2) Document Position Ratio: the ratio of sentence Sid to the total sentence number of the corresponding document. 3) Paragraph Position Ratio: the ratio of sentence Ssid to the total sentence number of the corresponding paragraph. 4) Number of Citations or References: the number of Citation Offset or Reference Offset. These features form an 8-dimension vector of a pair of reference sentence and citation sentence. We train SVM to get five classifiers. For the problem of unbalanced training data, we set different weights for different classes. If we cannot get any class of the five facets, then we classify it as Method class.

- Voting Method

We combine the results from Subtitle Rule, High Frequency Word and SVM classifier to generate the final results with most votes.

- Fusion Method

We run the above methods for each run result we obtained in Task 1A and choose a best one as the final result. Then we also tried a fusion method to combine all the run results of the above methods obtained in Task 1A. We counted the number of Method, Results, Aim, Hypothesis and Implication, and set an appropriate threshold for each facet class to get a final result of facet class.

4.2 Task 2

We provide a general overview of our method in Fig. 2.

Pre-processing.
The source documents provided by CL-SciSumm 2016 have some xml-coding errors. Besides, we need specific data format to train our hLDA feature. To obtain relatively a high-quality input dataset, we do pre-processing.

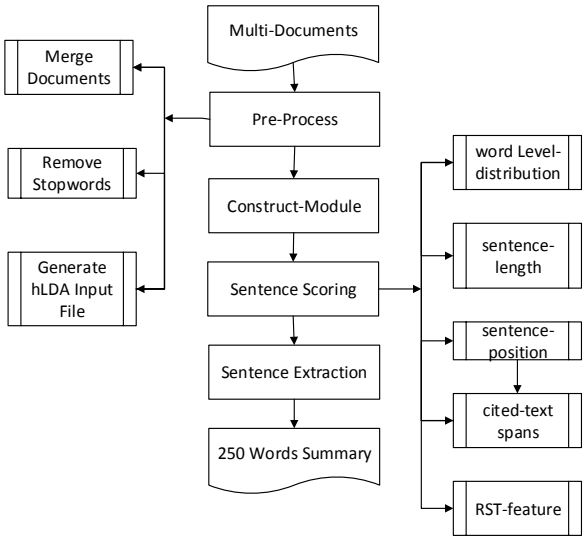

Fig. 2. Overview of our approach to RP summarization

1. Document merging: For each topic, we merge the reference paper and the cited text spans into a big text.
2. Stop words removing: We use the stop-word list to remove stop words. The stop-word list contains punctuation marks and some functional words. At the same time, all capitalized characters are changed to lower case.
3. Input file generation for hLDA: For the selected words, we build a dictionary for each document, which contains words and their corresponding frequency, and the index starts from 1 to word list size. Finally, we generate an input file for hLDA modeling, in which each line represents a sentence presented by word index - frequency pair, such as: [number of words in sentence i] [word-index A : frequency A] [word-index B : frequency B] …

Feature modeling.

1. hLDA-level distribution feature: To be known as exact description of latent topics over nested Chinese restaurant process, hLDA is one of the non-parametric topic models. Given the input file, unsupervised-hLDA generates tree-like topic structures over documents. Each sentence is assigned to a path starting from the root node to leaf nodes. Sentences sharing the same path are similar with each other and thus constitute a subtopic of the documents. Each node in the tree contains different words, whose distribution is referred to latent-topic. Besides, different level contains different nodes. Each word in a sentence is randomly assigned to the node at different level in the sentence path with some probability, and the probability to different level is the level message we would like to exploit which can represent the abstractiveness of the topic in the document to some extent. Since we have obtained the predefined facet information of cited test spans in Task 1B, we do not use the subtopic knowledge in hLDA here which may not directly match the five facets as defined by CL-SciSumm 2016. [33] investigated the influence of different hyper-parameters in hLDA modeling. Here we just use the result of these research, and set the depth of hierarchal tree to be three. Recently our work has been focused on the research in exploration of hLDA level information. According to recent research, we proposed a new level distribution scores as followed:

$$S_{levels} = \sum_{i=0}^{N}(W_i T_{i-Distribution} + T_{i-NodeFrequency}) \quad (7)$$

where W_i indicates the weight of the node level that word T_i is assigned in, $T_{i-Distribution}$ indicates the score of level distribution of word T_i, $T_{i-NodeFrequency}$ is the frequency of T_i in current node as followed.

$$T_{i-NodeFrequency} = \frac{counts(T_i)}{\sum_{i=0}^{V} counts(T_i)} \quad (8)$$

2. Sentence-length feature: It is a balance between length and sentence meaning. Gaussian Distribution is chosen in order to get a better combination along with the level feature.

$$S_{length} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{\sigma^2}} \quad (9)$$

where μ indicates the expectation of word frequency, σ indicates the standard deviation of word frequency.

3. Sentence-position feature: Sentence position is widely used in the summary extraction of news and science paper.

$$S_{position} = 1 - \frac{p_i}{n} \quad (10)$$

where n is the total number of sentences, p_i is the position of $i-th$ sentence. Number i is started from 0.

4. Cited text span: A citation is a summary of other paper authors, and we got all cited text spans of each reference paper from Task 1A. Thus we use it as a weak feature, for the reason that there must be some errors.

$$S_{spans} = sgn(x) = \begin{cases} 1, & sentence\ x\ is\ in\ result\ of\ task\ 1 \\ 0, & others \end{cases} \quad (11)$$

We use the Discourse Facet answer in Task 1B to provide a reference information for sentence extraction.

5. RST-feature: Rhetorical Structure Theory (RST) is the notion of rhetorical relation, a relation existing between two non-overlapping text spans called nucleus (N) and satellite (S). Text coherence in RST is assumed to arise due to a set of constraints and an overall effect that are associated with each relation. The constraints operate on the nucleus, on the satellite, and on the combination of nucleus and satellite. We call each sub-string cut by punctuations as a simple sentence. Using the RST-software we developed ourselves, we can get the RST-score of each simple sentence, then we have RST-score of a sentence as followed.

$$S_{rst} = \sum_{j=1,N} Sims_{rst-j} \quad (12)$$

where $Sims_{rst-j}$ is the RST-score of simple sentences, N is the total number of simple sentences in a sentence.

Structured summary extraction.

With the features above, we combine them to get a better summary result. A simple and efficient way is to combine them by a linear formula as shown below:

$$score_i = \omega_1 S_{level-i} + \omega_2 S_{length-i} + \omega_3 S_{position-i} + \omega_4 S_{spans-i} + \omega_5 S_{rst-i} \quad (13)$$

where ω_j indicates feature weight.

In Task 2, we need to generate a structured summary of the RP from the cited text spans of the RP. The length of the summary should not exceed 250 words. Besides, Task 2 pays more attention to the summary extracted from the third part, so we change the first person of the extracted sentences to third person. In order to extract a high-quality structured summary, we make full use of the prior knowledge that structured summary contains four parts: Introduction, Methods, Results and Conclusion. So we use the Discourse Facet to help extract the summary sentences. We will extract two or three sentences for each part of the summary if exists. Furthermore, we remove redundant candidate sentences. We use the following algorithm 1 to extract summary sentences.

```
for paper in Test-Set:
convert xml formed paper to txt, including title document merging of paper and cited text spans
stop word removing
input file generation for hLDA modeling
calculate scores of five features
while length ≤ 250
   extract sentence i according to feature scoring
   if sentence i is not redundant and length <= 250
    add i to candidate set
  person transformation
```

Algorithm 1 Feature-based summary extraction

To find the best use of hLDA level features, we also proposed two methods to extract level feature sentences. The first one is that, we ignore the clusters of the tree, only use the level feature score to choose best-N sentences. The other one is that each step extracts one sentence from one cluster in the first half of all clusters and adds it to the summary until the summary length is 250. We call the first one as ScoreDesc, the second one as In First Half Path.

5 Experiments

5.1 Task 1

Task 1A.

There are 5 runs that we submitted. From Run1 to Run4, we used the above mentioned methods respectively. Then, we got the performance of training dataset as followed in Table 7. In Run5, we used the SVM Method. And the accuracy we got in training set is 80.59% as a closed testing.

Method	P	R	Fs
Voting 1.0	0.07627	0.18881	0.10865
Voting 2.0	0.08898	0.22028	0.12676
Jaccard Focused	0.09675	0.23951	0.13783
Jaccard Cascade	0.0911	0.22552	0.12978

Table 7. Task 1A results of training dataset

We also have tested our systems on the testing dataset with the golden standard and got the results as in Table 8.

Method	P	R	Fs
Voting 1.0	0.05735	0.16887	0.08562
Voting 2.0	0.05735	0.16887	0.08562
Jaccard Focused	0.05824	0.1715	0.08696

Jaccard Cascade	0.05376	0.15831	0.08027
SVM	0.02222	0.08179	0.03495

Table 8. Task 1A results of testing dataset

From Table 7 and Table 8, we can see that there isn't much difference among the first 4 methods. Although the performance of SVM on training dataset is good, its performance on the testing dataset is very poor. We think that the major reason comes from the shortage of training data. In spite of this, we still set it as one of our system runs. Because we thought that this is an important direction. Later maybe we can improve this method using other strategies like semi-supervised or unsupervised machine learning with more unlabeled data.

After more detailed analysis of the results, we can find two major problems. One is choosing too much sentences, in which the citation text in standard answer contains only one fitful sentence, but we chose more sentences. We got the right answer with the price of more error sentences included. The other is partial answer. For instance, there are 3 sentences in standard answer covering the two classes of features, yet we found only 1 and lost 2 of them with 3 error sentences covering only one class of features. Although there may exist some relevance between the Citation Text and Reference Text in our answers, the standard answers are much better with good precision and complete coverage. Our system cannot achieve this status. We need better methods to locate the best answer accurately and cover all aspects of the citation text.

Task 1B.

We tried all methods in the results from Task 1A, then got the best performances by Voting Method. At last, we submit 6 runs. The first 4 runs are using Voting Method respectively in the results of Task 1A from rule-based methods. We used Fusion Method in Run5. Table 9 shows our experiments of the training data. As for Run6, we used the Voting Method for all SVM results in Task 1A. Table 10 shows their performances on the testing dataset.

Task 1A Method	Best Run	Precision
Voting 1.0	Voting Method	0.6977401
Voting 2.0	Voting Method	0.69491525
Jaccard Focused	Voting Method	0.6920904
Jaccard Cascade	Voting Method	0.7005649
Fusion Method		0.7062146

Table 9. Task 1B results of training dataset

Task 1A Method	Best Run	Precision
Voting 1.0	Voting Method	0.7526881
Voting 2.0	Voting Method	0.7562724
Jaccard Focused	Voting Method	0.7491039
Jaccard Cascade	Voting Method	0.7562724
SVM	Voting Method	0.7598566
Fusion Method		0.7598566

Table 10. Task 1B results of testing dataset

As we can see from Table 9 and Table 10, generally speaking, the best run is Voting Method. Next best is High Frequency Word Rule. The other methods get almost the same precision. The Voting Method is the results combining Subtitle Rule, High Frequency Word Rule and SVM classifier. So it gets the best precision as our anticipation. Although the performance of SVM in Task 1A is poor, using its result for Task 1B is not too bad.

As to the errors, our system may give a wrong decision of the facet, or cover the right answer but with added errors, or cover only part of the answer. The major reason is that the input from Task 1A of the Reference Text in our system has a great difference from that of the standard answer. Our classification methods are based on information contained in Citation Text and Reference Text, thus the quality of the Reference Text can make a great effect on the performance of facet classification.

5.2 Task 2

Our experiment is based on a hypothesis that the training dataset and the testing dataset have similar structures. Thus we can use the parameters learned from the training dataset to the testing dataset evaluation. After adequate experi-

ments of the training dataset, we choose five different parameter values to calculate the testing dataset and get five runs for submission. The Manual ROUGE values of some experiments on the training set are shown in Table 11.

The results show that, definitely the sentence position feature achieves the best run. However, the level feature is the worst. The golden standard summary provided by CL-SciSumm 2016 includes abstract of the reference paper and the summary from the reference span. Sentence position and Cited text spans feature choose sentence from this point, so they get the best precision as our anticipation. Level feature aims to choose sentence from latent topics of the paper. And it is very difficult for unsupervised hLDA to catch correctly the facet information of the sentences just as human pre-defined. So the level feature does not achieve the best result. But, we also did the following experiments to find out the value of cluster message for summary extraction.

We test the different sentence choosing method for level feature using both ScoreDesc Method and In First Half Path Method, Table 12 shows our experiments. From the result we can learn that compared with the ScoreDesc algorithm, the In First Half Path algorithm achieves a better result. Thus we believe that the clusters modeled by unsupervised hLDA could implicate some latent topic messages, although they are not directly matched to the pre-defined five facets, which leads to the poor performance in this pre-defined structured summarization task. But we think that it possibly will work better for other open domains without pre-defined structures.

	ω_1	ω_2	ω_3	ω_4	ω_5	ROUGE-1 F	ROUGE-L F
0	0	1	1	1	0	0.4237	0.3762
1	0	1	1	0	0	0.4628	0.4174
2	1	0	9	0	0	0.3602	0.3202
3	1	0	9	9	0	0.3926	0.3511
4	0.5	1	9	0	0	0.4026	0.3651
5	1	0	0	0	0	0.1770	0.1426
6	0	1	0	0	0	0.3114	0.2588
7	0	0	1	0	0	0.5099	0.4682
8	0	0	0	1	0	0.4449	0.3987
9	0	0	0	0	1	0.2913	0.2399

Table 11. The ROUGE values for training dataset

method	ROUGE-1 F	ROUGE-L F
Score Desc	0.1770	0.1426
In First Half Path	0.2754	0.2447

Table 12. ROUGE values of different sentence choosing method for level feature

Finally, we choose the top five parameters in Table 11 to model the testing data, and got the following answers submitted, detailed in Table 13.

Run	ROUGE-1 F	ROUGE-L F
1	0.5272	0.4881
2	0.4708	0.4238
3	0.4378	0.3931
4	0.4200	0.3685
5	0.3879	0.3451

Table 13. ROUGE values of selected answer for testing data.

6 Conclusion and Future Work

Our system has tried to add some semantic information like word vector, doc vector and word distributions in hLDA tree to improve the citance linkage and summarization performance. Yet the result isn't very satisfied. Our future work is to find some better ways to mine and use more semantic features for citance linkage. As to summarization, we will try to add more predefined semantic features to unsupervised hLDA model and sentences combination and compression for better summary sentences. Furthermore, we will try to find a better method to choose least sentences covering most information.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant 91546121, 61202247, 71231002 and 61472046; EU FP7 IRSES MobileCloud Project (Grant No. 612212); the 111 Project of China under Grant B08004; Engineering Research Center of Information Networks, Ministry of Education; Beijing Institute of Science and Technology Information; CapInfo Company Limited.

References

1. Wan, X., Yang, J., Xiao, J.: Using Cross-Document Random Walks for Topic-Focused Multi-Document. In: IEEE / Wic / ACM International Conference on Web Intelligence, pp. 1012-1018. (2006).
2. Garc á, J., Laurent, F., Gillard, O. F.: Bag-of-senses versus bag-of-words: comparing semantic and lexical approaches on sentence extraction. In: TAC 2008 Workshop - Notebook papers and results. (2008).
3. Bellemare, S., Bergler, S., Witte, R.: ERSS at TAC 2008. In: TAC 2008 Proceedings. (2008).
4. Conroy, J., Schlesinger, J. D.: CLASSY at TAC 2008 Metrics. In: TAC 2008 Proceedings. (2008).
5. Zheng, Y., Takenobu, T.: The TITech Summarization System at TAC-2009. In: TAC 2009 Proceedings. (2009).
6. Annie, L., Ani, N.: Predicting Summary Quality using Limited Human Input. In: TAC 2009 Proceedings. (2009).
7. Darling, W.M.: Multi-document summarization from first principles. In: Proceedings of the third Text Analysis Conference, TAC-2010. NIST (Vol. 150). (2010).
8. Kokil, J., Muthu, K.C., Sajal, R., Min-Yen, K.: Overview of the 2nd Computational Linguistics Scientific Document Summarization Shared Task (CL-SciSumm 2016). In: The Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL 2016), Newark, New Jersey, USA. (2016).
9. Genest, P., Lapalme, G., Qu ébec, M.: Text Generation for Abstractive Summarization. In: TAC 2010 Proceedings. (2010).
10. Jin, F., Huang, M., Zhu, X.: The THU Summarization Systems at TAC 2010. Text Analysis Conference. (2010)
11. Abu-Jbara, A., Radev, D.: Coherent citation-based summarization of scientific papers. In: Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 500-509. Portland, Oregon (2010).
12. Zhang, R., Ouyang, Y., Li, W., Zhang, R., Ouyang, Y., Li, W.: Guided Summarization with Aspect Recognition. In: TAC 2011 Proceedings. (2011).
13. Marina, L., Natalia, V.: Multilingual Multi-Document Summarization with POLY. In: Proceedings of the MultiLing 2013 Workshop on Multilingual Multi-document Summarization. (2013).
14. Steinberger, J.: The UWB Summariser at Multiling-2013. In: Proceedings of the MultiLing 2013 Workshop on Multilingual Multi-document Summarization. (2013).
15. Ardjomand, N., Mcalister, J.C., Rogers, N.J., Tan, P.H., George, A.J., Larkin, D. F.: Multilingual Summarization: Dimensionality Reduction and a Step Towards Optimal Term Coverage. In: Multiling 2013 Workshop on Multilingual Multi-Document Summarization, pp. 3899-3905. (2013).
16. Anechitei, D.A., Ignat, E.: Multi-lingual summarization system based on analyzing the dis-course structure at MultiLing 2013. In: Proceedings of the MultiLing 2013 Workshop on Multilingual Multi-document Summarization. (2013).
17. El-Haj, M., Rayson, P.: Using a Keyness Metric for Single and Multi Document Summarisation. Multiling 2013 Workshop, ACL. (2013).
18. Fattah, M.A.: A hybrid machine learning model for multi-document summarization. Applied Intelligence, 40(40), 592-600. (2014).
19. Zhang, R., Li, W., Gao, D., Ouyang, Y.: Automatic twitter topic summarization with speech acts. IEEE Transactions on Audio Speech & Language Processing, 21(3), 649-658. (2013).
20. Xu, Y. D., Zhang, X. D., Quan, G. R., Wang, Y. D.: MRS for multi-document summarization by sentence extraction. Telecommunication Systems, 53(1), 91-98. (2013).
21. Arora, R., Ravindran, B.: Latent dirichlet allocation based multi-document summarization. In: The Workshop on Analytics for Noisy Unstructured Text Data, pp. 91-97. ACM. (2008)
22. Krestel, R., Fankhauser, P., Nejdl, W.: Latent dirichlet allocation for tag recommendation. In: ACM Conference on Recommender Systems, pp. 61-68. (2009).
23. Griffiths, T.L., Steyvers, M., Blei, D.M., Tenenbaum, J.B.: Integrating topics and syntax. In: Advances in Neural Information Processing Systems, 17, 537--544. (2010).
24. Blei, D.M., Lafferty, J. D.: Dynamic topic models. In: Proceedings of the 23rd international conference on Machine learning, pp. 113--120. (2006).
25. Wang, C., Blei, D.M.: Decoupling Sparsity and Smoothness in the Discrete Hierarchical Dirichlet Process. Advances in Neural Information Processing Systems 22. In: Conference on Neural Information Processing Systems 2009. Proceedings of A Meeting Held 7-10 December 2009, Vancouver, British Columbia, Canada, pp. 1982-1989. (2009).

26. Teh, Y.W., Jordan, M.I., Beal, M.J., Blei, D.M.: Hierarchical dirichlet processes. Journal of the American statistical association. (2012).
27. Blei, D.M., Griffiths, T.L., Jordan, M.I.: The nested Chinese restaurant process and bayesian nonparametric inference of topic hierarchies. Journal of the ACM, 57(2), 87-103. (2010).
28. Celikyilmaz, A., Hakkani-Tur, D.: A Hybrid Hierarchical Model for Multi-Document Summarization. ACL 2010, Proceedings of the, Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden, pp. 815-824. (2010).
29. Ren, Z., De Rijke, M.: Summarizing Contrastive Themes via Hierarchical Non-Parametric Processes. International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 93-102. ACM. (2015).
30. Elkiss, A., Shen, S., Fader, A., Gü&#x f; Erkan, States, D., Radev, D.: Blind men and elephants: what do citation summaries tell us about a research article?. Journal of the American Society for Information Science & Technology, 59(1), 51-62. (2008).
31. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. Computer Science. (2013).
32. Le, Q. V., Mikolov, T.: Distributed representations of sentences and documents. Computer Science, 4, 1188-1196. (2014).
33. Heng, W., Yu, J., Li, L., Liu, Y.: Research on Key Factors in Multi-document Topic Modelling Application with HLDA. Journal of Chinese Information Processing, 27(6): 117–127. (2013).