FIRE2019@AILA: Legal Retrieval Based on Information Retrieval Model

Jiaming Gao[1], Hui Ning[1], Huilin Sun[2], Ruifeng Liu[2], Zhongyuan Han[2,*], Leilei Kong[2] and Haoliang Qi[2]

[1] Harbin Engineering University Harbin, China
[2] Heilongjiang Institute of Technology Harbin, China
gaojiaming24@gmail.com,ninghui@hrbeu.edu.cn
{sunhuilin24, liuruifeng812, Hanzhongyuan, Kongleilei1979, haoliang.qi}@gmail.com

Abstract. This paper describes our evaluation methods in the task 1 of AILA (Artificial Intelligence for Legal Assistance) tasks in FIRE 2019. The task 1 is to identify relevant prior cases for a given situation. We deem the task as an information retrieval task. We first extract the topic words from the given situation and use the topic words as a query to identify the relevant prior cases by using the information retrieval model. The best result gets the second place on the MAP.

Keywords: Artificial Intelligence for Legal Assistance, Legal Retrieval, Information Retrieval.

1 Introduction

During the judge's trial of the case, prior cases help a lawyer understand how the Court has dealt with similar cases in the past and prepare the legal reasoning accordingly. Therefore, it is important to help judges find similar prior cases for a given situation. In this evaluation, the organizer provides 3000 cases of documents of cases that were judged in the Supreme Court of India. The task is to identify the most relevant cases related to the given situation [1]. In this paper, we introduced the methods in this task. Our method is divided into two steps. First, the topic words are extracted from the given situation. And then, the topic words are deemed as a query to search the relevant prior cases by using the retrieval model. We use experimental data provided by the Fire2017 legal retrieval task, and select the number of keywords according to the change of the MAP value.

[*]corresponding author

2 Model Framework

The Framework is shown as following figure 1.

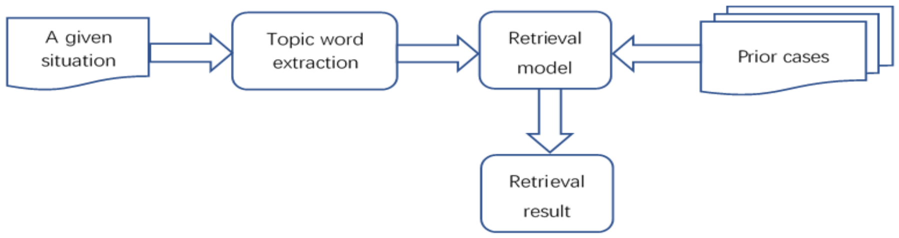


Fig. 1. Model Framework

First, the topic extraction module uses TF-IDF and TextRank methods to extract the topic words from a given situation. And then the top-k topic words, which are deemed to be a query, are input to the retrieval model such as the VSM(Vector Space Model), the BM25 model, and the LM(Language Model). The top-n cases in the retrieval result are deemed the relevant prior cases.

3 Topic Extraction Method

3.1 TF-IDF-based topic extraction method

TF-IDF (term frequency-inverse document frequency) [2] is an important indicator of how important a word is in a document. The weight assigned by TF-IDF to the word t in document d can be expressed as: $TF-IDF = tf_{t,d} \times idf_t$, $tf_{t,d} = \frac{n_{t,d}}{\sum_k n_{k,d}}$ represents the frequency of the term in the document d, $idf_t = \log_{10} \frac{N}{df_t+1}$ indicates the inverse document frequency. Where df_t indicates the number of all documents in which the term t appears, plus 1 is to prevent the denominator from being zero, and N is the number of all documents.

3.2 TextRank-based topic extraction method

TextRank [3] is a graph-based topic extraction algorithm that can extract topic words from a single document. Formally, let $G = (V, E)$ is a directed graph consisting of a series of vertices V and edges E. w_{ji} represents the weight between two vertices V_i and V_i. $\mathrm{In}(V_i)$ represents a collection of points pointing to that point. $\mathrm{Out}(V_j)$ indicates that vertex V_j points to a collection of points. Where d is the damping coefficient, which is between 0 and 1, and is generally set to 0.85. The score of a vertex V_i is defined as follows, according to the formula for calculating the score of each word.

$$WS(V_i) = (1-d) + d * \sum_{V_j \epsilon In(V_i)} \frac{w_{ji}}{\sum_{V_k \epsilon Out(V_j)} w_{jk}} WS(V_j) \tag{1}$$

4 Identifying the Relevant Cases via Information Retrieval Model

In our approach, we consider the extracted topic as a query and the prior cases as a document collection. The task of this evaluation can be formulated as a retrieval problem. We used three search models, as follows:

4.1 Vector Space Model

The Vector Space Model [4] uses cosine similarity to calculate the similarity between a query and a document. Suppose the vector corresponding to the query q is represented by $\bar{V}(q) = (w_{11}, w_{12}, \cdots, w_{1n})$, and the vector corresponding to the document d is represented by $\bar{V}(d) = (w_{21}, w_{22}, \cdots, w_{2n})$, each of the vectors The components correspond to one term.

$$score(q,d) = cos\theta = \frac{\bar{V}(q) \cdot \bar{V}(d)}{|\bar{V}(q)||\bar{V}(d)|} = \frac{\sum_{k=1}^{n} w_{1k} \times w_{2k}}{\sqrt{\sum_{k=1}^{n} w_{1k}^2 \sum_{k=1}^{n} w_{2k}^2}} \tag{2}$$

During the retrieval process, the similarity scores of the query and each document are calculated and sorted according to the similarity score. The higher the ranking, the more relevant the document is to the query.

4.2 Probability Model

BM25 [5] is a method for establishing a probability model based on factors such as word frequency and document length. For a query q is a collection containing n terms t. $tf_{t,d}$ represents the word frequency of the term t in the document d, N represents the number of all documents, df_t represents the number of all documents in which the term t appears, L_d represents the length of the document d, and L_{avg} represents the whole The average document length of the document collection. k_1 is a tuning parameter with a positive value, and b is another tuning parameter. The value ranges from 0 to 1. k_3 represents a tuning parameter with a positive value, which is used to scale and control the frequency of the term in the query. n represents the number of terms in the query. The relevant line scores of each query word and document are calculated in the calculation process, and finally, the sum is obtained to obtain the correlation score between the query and the document.

$$score(q,d) = \sum_{t \in q}^{n} \log\left[\frac{N}{df_t}\right] \cdot \frac{tf_{t,d} \cdot (k_1+1)}{tf_{t,d} + k_1 \cdot \left[(1-b) + b \cdot \frac{L_d}{L_{avg}}\right]} \cdot \frac{(k_3+1) \cdot tf_{t,d}}{k_3 + tf_{t,d}} \tag{3}$$

4.3 Language Model

We build language models θ_Q and θ_D for queries and documents. According to the risk minimization model, the KL (Kullback-Leibler) distance metric is applied to the correlation between the query language model θ_Q and the document language model θ_D[6]. The scores for query q and document d become:

$$score(q,d) = KL(\theta_{\mathrm{Q}} \mid \theta_{\mathrm{D}}) = \sum_{w \subset V} P(w \mid \theta_{\mathrm{Q}}) \log \frac{P(w \mid \theta_{\mathrm{Q}})}{P(w \mid \theta_{\mathrm{D}})} \tag{4}$$

Where V is the entire vocabulary, w is the term in V, and is the language model of the query and document, respectively. P($w|\theta_Q$) and P($w|\theta_D$) are the probabilities of w in the query language model θ_Q and the document language model θ_D, respectively. P($w|\theta_Q$) uses the maximum likelihood estimation, and P($w|\theta_D$) uses the Dirichlet smoothing[7] document language model.

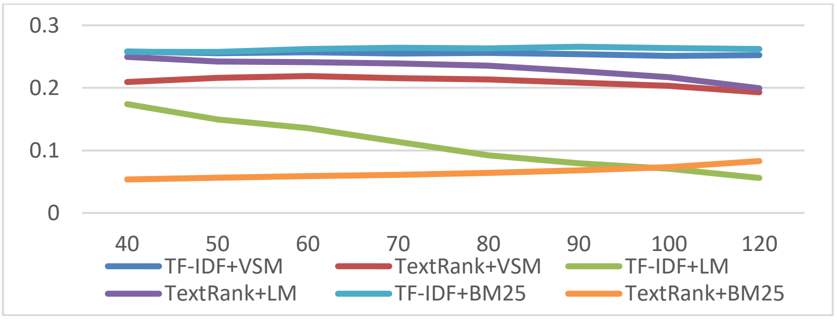
5 Experiment

5.1 Experiment Setting

In the process of extracting the topic words using TextRank, we use the TextRank method in the open-source toolkit HanLP[†] to extract the topic words. The VSM retrieval model was implemented using the Lucene [8] toolkit[‡], and the BM25 retrieval model and language model were implemented using the Lemur[9] toolkit3[§].

5.2 Parameter Settings

We conducted six experiments according to topic extracting methods and retrieval models. The value of TopK is between 40 and 120, and the experimental results are as shown in **Fig.2**. And select the number of topic words according to MAP.



[†] https://github.com/hankcs/HanLP
[‡] http://lucene.apache.org/
[§] http://www.lemurproject.org/

Fig.2. The Map with different number of topic words

It can be seen from the figure 2 that TF-IDF+VSM, TF-IDF+BM25, TextRank+LM are relatively stable, so we submitted the search results of these three models, labeled HGC_1, HGC_2, HGC_3, and the model parameter settings are as following Table 1 :

Table 1. Experimental parameter setting.

Runs	Query Extraction	Model	Parameter	Topk
HGC_1	TF-IDF	Vector Space Model	None	80
HGC_2	TF-IDF	BM25	K1:1.2,b:0.75,K3:7	70
HGC_3	TextRank	Language Model	μ=100	60

5.3 Experimental Result

The results of the top seven submissions are as following Table 2.

Table 2. Results of the AILA Task 1.

Runs	MAP	P@10	BPREF	1/rank of first relevant document
HLJIT2019-AILA_task1_2	0.1492	0.07	0.1286	0.288
HGC_1	0.1382	0.0575	0.1207	0.28
HLJIT2019-AILA_task1_1	0.1335	0.06	0.1134	0.282
HGC_2	0.1286	0.05	0.1092	0.256
IITP_BM25_case	0.0984	0.0275	0.0869	0.175
TFIDF	0.0956	0.05	0.067	0.203
HGC_3	0.0946	0.0316	0.0804	0.18

It can be seen from the experimental results that TF-IDF+VSM has achieved good performance and ranked second. The language model combined with the TextRank topic extraction method did not show excellent performance.

6 Conclusions

This paper describes the evaluation method we used in the FIRE2019 AILA. The TF-IDF and TextRank based topic extraction methods are used in the topic extraction stage. And the VSM, BM25, and LM are used in the retrieval stage. Through the final evaluation results, it can be seen that the TF-IDF-based topic extraction method combined with the VSM model is superior to our other submission results.

Acknowledgment

This work is supported by the National Social Science Fund of China (No. 18BYY125).

References

1. P. Bhattacharya, K. Ghosh, S. Ghosh, A. Pal, P. Mehta, A. Bhattacharya., P. Majumder, Overview of the Fire 2019 AILA track: Artificial Intelligence for Legal Assistance. In Proc. of FIRE 2019 - Forum for Information Retrieval Evaluation, Kolkata, India, December 12-15, 2019.
2. Singhal, Amit, et al. Document length normalization. Information Processing & Management 32.5 (1996): 619-633.
3. Mihalcea, Rada, and Paul Tarau. Textrank: Bringing order into text. Proceedings of the 2004 conference on empirical methods in natural language processing. 2004: 404-411.
4. Salton, Gerard, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. Communications of the ACM 18.11 (1975): 613-620.
5. Robertson, Stephen, and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. Foundations and Trends® in Information Retrieval 3.4 (2009): 333-389.
6. Song, Fei, and W. Bruce Croft. A general language model for information retrieval. Proceedings of the eighth international conference on Information and knowledge management. ACM, 1999: 316-321.
7. Zhai Chengxiang, and John Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. ACM SIGIR Forum. Vol. 51. No. 2. ACM, 2017: 268-276.
8. McCandless, Michael, Erik Hatcher, and Otis Gospodnetic. Lucene in action: covers Apache Lucene 3.0. Manning Publications Co., 2010.
9. Ogilvie, Paul, and Jamie Callan. Experiments using the Lemur toolkit. TREC. Vol. 10. 2001: 103-108.