Deep News Event Ranker Based On User Relevant Query

Xiangfei Kong[1,2], Qingchao Kong[1*], Wenji Mao[1,2], Shaoqiang Tang[3]
[1]State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, University of Chinese Academy of Sciences
[2]School of Computer and Control Engineering, University of Chinese Academy of Sciences Beijing, China
[3]College of Engineering, Peking University, Beijing, China
{kongxiangfei2015,qingchao.kong,wenji.mao}@ia.ac.cn, maotang@pku.edu.cn

Abstract—News Event Ranking(NER), which takes event-related news documents as the ranking unit, has been addressed in many research work and implemented in security-oriented applications(e.g. public event monitoring, mining and retrieval). Previous work solely rank news event based on event relevant information, while user relevant information equally important for characterizing news event is totally neglected. In this paper, we depict news event with extra user comments sentiment polarity information, and address news event ranking problem by incorporating user relevant information into the input query. Given an input query, which contains event related objective aspects(e.g. actors, locations, date) and user related subjective aspects(e.g. public attention and opinion polarity), we develop a Deep News Event Ranker model to integrate objective event information and subjective user information. Firstly, a semantic similarity interaction module transforms query keywords, news document and news comments to their semantic vector representation and calculates query\document similarity and query\comment similarity. Then a Feature Extraction Based On CNNs and LSTM module extract query term importance features, query term frequency features and BM25-like relevance features for ranking. Finally, a Feature Aggregation module merges the extracted features with some auxiliary relevance features and produces a global relevance score. Experiments on a large news dataset demonstrate the effectiveness of our proposed model compared to several baseline models.

Keywords–News Event Ranking;User Relevant Query;user related subjective aspects;Deep News Event Ranker

I. INTRODUCTION

When a social news event happens, various media agencies will flock to report it. Meanwhile, people incline to express their views and opinions when reading news to learn about the latest events. In security-related applications, these user comments information such as public attention and opinion polarity, reflect public demand and individualized needs and can provide valuable information to facilitate government decision makers, security officials and information seeker. Moreover, fulfilling an event query such as "Event happened in Beijing which has a positive social influence or good public opinion", which includes both event related objective information and user related objective information, will enrich the functionality of existing event-related query systems greatly, and have promising application prospects in many security oriented applications.

To address the aforementioned issues, we develop an end-to-end learning framework, which aims to address the problem of ranking news events from a large news collection. This framework is based on the correlation between news events and user query containing event relevant information and user relevant information. Essentially, we aim to characterize news event not only the news report information but also the news comments which contain public attitude towards the news event.

News Event Ranking(NER) generates a ranked list by sorting in descending order of the relevance score between news event query and candidate news event, which can be viewed as a standardized information retrieval(IR) problem. A lot of attention rank news events based on event popularity and importance by carefully designed features[1][2][3]. Learning-To-Rank(LTR) has achieved great performance in [4][5] by combining relevance features and importance features(i.e. TF-IDF and PageRank) using machine learning methods in IR area[6]. However, these algorithms rely on the basis of effective handcrafted features, which is a complex empirical process and domain-related work. Recently, deep learning approaches lead to state-of-the-art results in many NLP tasks owning to preserving syntactic and semantic information of the input sentence. For example, some research work, such as DSSM[7], DRMM[8] transform the feature construction to raw sentence analysis between query and candidate documents by extracting semantic relevance. Without considering the more important relevance matching characteristics which proved to be effective in many ranking algorithms, such as query term frequency in titles, inverse-document frequency and document length, these deep learning methods have a relatively poor performance compared to LTR methods according to the conclusion of DeepRanker[9].

Inspired by news event ranking research, the deep learning ranking models and LTR models based on feature engineering, we propose an original news event ranking problem which characterizes event with extra user comments, and a new deep learning architecture, namely

* Corresponding author.

Deep News Event Ranker(DNER). The ranking features offered by LETOR mainly include length information, term frequency, inverse-document frequency and so on. Meanwhile, the success of deep learning in NLP demonstrate the effectiveness of catching semantic information in word and sentence.

Therefore, our proposed DNER model aim to construct important ranking features offered by LETOR with deep learning methods automaticlly, such as semantic similarity features, query term importance features, query term frequency features. This is achieved by a series of matrix operation on semantic representation vector of query and news collection, which contains news document and news comments. Our model also merge auxiliary features to better capture the relevance essence.

In a nutshell, we develop a new deep model for news event ranking problem which query and event contain user information. Our model extracts relevance features automatically by using CNNs and LSTM architecture. The experiments show that our model can better capture the relevance essence of news event ranking.

II. Problem Definition

In this study, we aim to rank a list of news events based on a given user query which contains event related objective information and user related subjective information. We cast it as a news event ranking problem where the news event collection $\mathcal{C}$ consists of news documents, denoted as d_i, and corresponding comments, denoted as r_i, that is $\mathcal{C} = \{(d_1, r_1), ..., (d_n, r_n)\}$, and the query Q contains news event keywords. Given a query $q_i \in Q$ and its candidate news event collections $\mathcal{C}_i = \{(d_{i1}, r_{i1}), ..., (d_{in}, r_{in})\}$, the target of the ranking model is to generate an ranking R, s.t. relevant news event collections appear at the top of the list. We tackle the news event ranking problem as a pointwise LTR problem. The output of our model is the relevance degree for each news event collection. That is to say, the task is to learn a ranking function: $h(w, \varphi(q_i, \mathcal{C}_i)) \rightarrow R \quad (R \in \{1, 2, ..., K\})$, where function $\varphi(\cdot)$ maps query\collection pairs to a feature vector representation where each element reflects a certain type of similarity. The weight vector w is the model parameter and is learned during the training process.

III. Deep News Event Ranker

Figure 1 gives an illustration of DNER model architecture. Our model architecture mainly consist of three modules: Semantic Similarity Interaction module, Feature Extraction Based On CNNs and LSTM module and Feature Aggregation module. In Semantic Similarity Interaction module, we first transform event related query keywords, user related query keywords, news document and news comments to their semantic vector representation by Word Embedding, then calculate semantic similarity matrix with these semantic vectors. The similarity matric include query\document similarity matrix and query\comment similarity matrix. The Feature Extraction module aims to extract query term importance vector, query term frequency vector and BM25-like relevance metric vector through stacked layers of operations, such as convolution, nonlinear transformation and pooling. The Feature Aggregation module merges the extracted features with some auxiliary relevances features for ranking. Next we describe these three modules separately in detail.

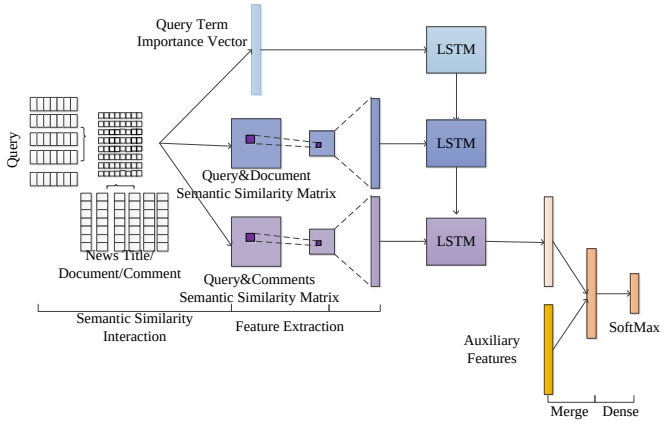

Fig. 1: DNER Model

A. Semantic Similarity Interaction

In this module, we adopt the word embedding technology to represent the semantic meanings of query keywords and documents. Word Embedding technology, such as Global Vectors for Word Representation(GloVe)[10], is the foundation of deep learning methods in NLP and IR. It facilitates researchers to explore the semantic similarity between query and document by mapping them to their intermediate vectors. Specifically, each query Q, news document D and news comment R is represented as a sequence of word $(s_1, s_2, ..., s_M)$, $(u_1, u_2, ..., u_N)$ and $(v_1, v_2, ..., v_O)$ respectively. Denote the word embedding of s_i as x_i, the query Q will be represented as $\left[\vec{x}_1^{(Q)}, \vec{x}_2^{(Q)}, ..., \vec{x}_M^{(Q)}\right]$, where each column stands for a word vector.

We then calculate query\document semantic similarity matrix and query\comment semantic similarity matrix using Equation (1) and (2). For word i in query Q and word j in document D or comment R, cosine value z_{ij} is first calculated between them, and then f function makes a nonlinear transformation on z_{ij} in order to approximate the term frequency. Here we utilize the rapid change property and derivability of tanh function by threshold θ in Equation (2) to measure the relevance. If the similarity value is less than θ, the value of $f(z)$ will be close to 0, while the value of $f(z)$ will be close to 1 when the similarity value is more than θ. The derivability of function tanh guarantees the query and document semantic vector

is learnable to adapt our problem.

$$z_{ij}^{(Q,D)} = \frac{< \vec{x}_i^{(Q)}, \vec{x}_j^{(D)} >}{\left\| \vec{x}_i^{(Q)} \right\| \cdot \left\| \vec{x}_j^{(D)} \right\|} \quad (1)$$

$$f(z) = \tanh(z-\theta) = \frac{e^{z-\theta} - e^{-z+\theta}}{e^{z-\theta} + e^{-z+\theta}} = \frac{e^{z-2\theta} - e^{-z}}{e^{z-2\theta} + e^{-z}} \quad (2)$$

B. Feature Extraction

1) Term Importance Features: According to the conclusion of DeepRanker[9], query term importance is an important ranking matric in LTR models. Usually important query keyword has a high term frequency in candidate documents. For example, in the query "trump's asian tour", the keyword "trump" and "asian" show more times than "tour" in many news reports. Since the semantic similarity matrix we design can depict the term frequency, we conduct a Conv1D operation on the query\document semantic similarity matrix to get a query term importance vector as shown in Equation (3).

$$\overrightarrow{QI}_i = Conv1D([z_{i1}, z_{i2}, ..., z_{in}]) \quad (3)$$

2) Term Frequency Features: Query term frequency reflects how important the query term is to a document in the collection. By using different size of convolution kernels as shown in Equation (4), we can extract local matching signals from the semantic similarity matrix. Then a local normalization operation is operated to amplify the extracted matching signals. Finally we filter significant matching patterns by max-pooling to get query term frequency vector as shown in Equation (5).

$$lms_{ij} = Conv2D\left(\begin{vmatrix} z_{i-k,j-k} & \cdots & z_{i-k,j+k} \\ \vdots & z_{ij} & \vdots \\ z_{i+k,j-k} & \cdots & z_{i+k,j+k} \end{vmatrix}\right) \quad (4)$$

$$h = \max\{h_{x,y}\}$$
$$x \in [i-k_2, i+k_2], y \in [j-k_2, j+k_2] \quad (5)$$

3) BM25-like Relevance Features: BM25 matric is commonly used in learning-to-rank approaches. In recent years, many researchers have achieved good results by using LSTM[11] to approximate BM25 metric in Question&Answer(QA) areas. Since our query and document are all raw sentences, we tackle our news event ranking problem using LSTM approaches just like QA problem. As shown in Equation (6),(7) and (8), the input of our LSTM module is query\document semantic similarity vector $\overrightarrow{SS}$, query term importance vector: $\overrightarrow{QI}$ and query term frequency vector: $\overrightarrow{TF}$. These vectors are sequentially read by LSTM. In this way, the relevance features are aggregated and stored into cell memory vector. For each step in the LSTM layer, the hidden vector of the output vector is generated by combing the cell memory vectors from LSTM. In other words, all the contextual information across the entire sequence(both query and document sentences) has been taken into consideration. The final output of each time step is the label indicating whether the candidate features should be selected as the proper features for news event ranking.

$$h_{qi} = LSTM\left(\overrightarrow{QI}\right) \quad (6)$$

$$h_{ss} = LSTM\left(\overrightarrow{SS}\right) \quad (7)$$

$$h_{tf} = LSTM\left(\overrightarrow{TF}\right) \quad (8)$$

Finally, all the significant matching patterns obtained from different kernels are concatenated to conform a vector to represent the local relevance. This vector will be treated as the input of the Feature Aggregation module.

C. Feature Aggregation

The output of the above module includes the semantic similarity information, term frequency and query term importance information. However, this is not enough to model the relevance. Thus we merge some auxiliary features: query length, document length, comment num and PageRank value which can be easily obtained by google search engine. The merged vector is then operated by a dense layer and finally projected as a one-dimension vector. The output of this module is a softmax vector representing the rank level of the input document.

IV. Experiments

A. Datasets

As elaborated in Section I, the comparison between LTR methods and deep learning methods need a thorough dataset with enough query sentence and document sentence, and a well hand-crafted features construction is also needed. We manually construct event query with event objective aspects keywords(e.g. actors, locations, date) and expand these keywords using synonym dictionaries and abbreviations expansion dictionaries. In this way, we construct 1000 queries with 15 words on average. Then search these keywords in search engine. Usually the search results returned by search engine are from various websites, so we filter these results and retain news url posted by Yahoo News, which contains a large number of user comments and emotional polarity label. The correlated news title, news content and comments are also crawled. We utilize the tagged sentiment rate information to build ground truth, that is, news events rank by positive sentiment rate corresponds to the results for query with positive user information and vice versa. Meanwhile, we refer to the LETOR datasets to construct 40 features for implementing LTR approaches, which also contains our self-constructed user information features. These features contain 'reading user count', 'count of labeled positive sentiment', 'count of labeled negative sentiment', 'count of labeled neutral sentiment', 'comment

count', 'up vote count', 'down vote count' and 'reply count'.

B. Baseline Methods

To show the superiority of our model, we have performed a comparison between our model and several baseline LTR mthods, including BM25[12], RankSVM[13], RankNet[14], LambdaMART[4] and deep learning based methods, DSSM[7]. For RankSVM method, we adopt l2 normalization and set penalty coefficient as 1.1; For RankNet model, we transform the candidate documents to partial order documents pair, that is $\{c_i, c_j\}$, where c_i is more relevant than c_j; For LambdaMART model, the learning rate value is 0.01. All the experiment code are posted on Github[1]. The evaluation metric is NDCG@K.

C. Experimental Results

In our DNER model, the learning rate is 0.01, the size of batch is 20. We learn word semantic vector using GloVe and fine tune them in training process to adapt the domain of task. All queries sentences are padded into maximum length T=8, titles sentences are padded into maximum length T=15, news contents and news comments are padded into maximum length T=100.

TABLE I: Performance Comparison of News Event Ranking

Methods	$NDCG$@1	$NDCG$@5	$NDCG$@10
BM25	0.1990	0.3868	0.5887
RankSVM	0.2093	0.4049	0.5958
RankNet	0.2875	0.3973	0.5925
LambdaMART	0.3199	0.4551	0.6577
DSSM-QD	0.1958	0.3767	0.6040
DSSM-QC	0.1718	0.3686	0.5750
DNER-UC	0.3128	0.4915	0.6526
DNER-FE	0.2941	0.4641	0.6429
DNER	0.3790	0.5059	0.6686

Table I shows the comparison results of different methods on our datasets. It is clear that our DNER model produces the best results in NDCG@K(K=1,5,10). The LTR methods, RankSVM, RankNet and LambdaMART, only use hand-crafted features. The three models perform better than BM25 metric, demonstrating the powerful performance of the LTR methods in our news event ranking task. Compared to RankSVM, RankNet performs better in NDCG@K metric, which shows that neural network method for news event ranking is an effective method. LambdaMART outperforms other methods since it is an ensemble learning method and adapt state-of-the-art multiple additive regression tree method.

DSSM-QD method ranks news events based on cosine similarity value of query semantic vector and document semantic vector got by DSSM model architecture, while DSSM-QC method corresponds to query semantic vector and comment semantic vector. As a whole, deep learning methods based on cosine similarity between query\document semantic vector is weakly compared to LTR methods, which is in agreement with the conclusion of DeepRanker[9].

DNER-FE only uses auxiliary hand-crafted features without considering user comments information. While the result is less superior than our DNER models, it performs better than LTR methods and DSSM-based methods. DNER-UC adopts queries term importance features, query term frequency features and BM25-like relevance features construct by our model. DNER-UC is superior than DNER-FE which shows that our automatic generated features are comparable with hand-crafted features. Finally, the terminal approach DNER which aggregate our automatic generated features and auxiliary features achieve best results on all the three NDCG@K metrics. It demonstrates the effectiveness of our proposed model since it extracts semantic information and interaction relationship between query and news event and merges the extracted features with some auxiliary relevance features.

V. Conclusions

In this paper we propose an original news event ranking problem which characterizes news event with news document and news comments. Given user's event query containing event related objective information and user related subjective information, we rank news events based on our automatic generated relevance features and auxiliary features. These features fully consider semantic information and interaction relationship between query and news event. Experiments on a news datasets demonstrate our proposed model can better capture the relevance essence. What's more, the analysis of the experiment results show that our automatic generated features are comparable with hand-crafted features. In summary, Deep News Event Ranker Based On User Relevant Query can enrich the functionality of existing event-related query systems greatly, and have promising application prospects in many security oriented applications. In the future work, we will explore the application scenarios by merging more abundant user-relevant information in query.

References

[1] Igor Trajkovski. Pagerank-like algorithm for ranking news stories and news portals. In ICT Innovations 2013, pages 87–96. Springer, 2014.

[2] Derek Davis, Gerardo Figueroa, and Yi-Shin Chen. Socirank: identifying and ranking prevalent news topics using social media factors. IEEE transactions on

[1] https://github.com/highflykxf/eventranker

systems, man, and cybernetics: systems, 47(6):979–994, 2017.

[3] Vinay Setty, Abhijit Anand, Arunav Mishra, and Avishek Anand. Modeling event importance for ranking daily news events. In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, pages 231–240. ACM, 2017.

[4] Christopher JC Burges. From ranknet to lambdarank to lambdamart: An overview. Learning, 11(23-581):81, 2010.

[5] Olivier Chapelle and S Sathiya Keerthi. Efficient algorithms for ranking with svms. Information retrieval, 13(3):201–215, 2010.

[6] Hang Li and Zhengdong Lu. Deep learning for information retrieval. In Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, pages 1203–1206. ACM, 2016.

[7] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In Proceedings of the 22nd ACM international conference on Conference on information & knowledge management, pages 2333–2338. ACM, 2013.

[8] Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. A deep relevance matching model for ad-hoc retrieval. In Proceedings of the 25th ACM International on Conference on Information and Knowledge

[13] Thorsten Joachims. Svm-rank: Support vector machine for ranking. Cornell University, 2009.

Management, pages 55–64. ACM, 2016.

[9] Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, Jingfang Xu, and Xueqi Cheng. Deeprank: A new deep architecture for relevance ranking in information retrieval. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, pages 257–266. ACM, 2017.

[10] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pages 1532–1543, 2014.

[11] Di Wang and Eric Nyberg. A long short-term memory model for answer sentence selection in question answering. In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), volume 2, pages 707–712, 2015.

[12] Stephen E Robertson and Steve Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, pages 232–241. Springer-Verlag New York, Inc., 1994.

[14] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In Proceedings of the 22nd international conference on Machine learning, pages 89–96. ACM, 2005.