HITSZ-ICRC at NTCIR-11 Temporalia Task

Yongshuai Hou, Cong Tan, Jun Xu*,
Youcheng Pan, Qingcai Chen and Xiaolong Wang
Key Laboratory of Network Oriented Intelligent Computation
Department of Computer Science and Technology
Harbin Institute of Technology Shenzhen Graduate School, 518055 Shenzhen, China
houyongshuai@hitsz.edu.cn, hit.xujun@gmail.com

ABSTRACT

Temporal Information Access (Temporalia) task is a pilot task at NTCIR-11 for the first year. HITSZ-ICRC group participated in Temporalia task, worked in both Temporal Query Intent Classification (TQIC) subtask and Temporal Information Retrieval (TIR) subtask. In TQIC subtask, firstly, we extracted different linguistic level features from user query, extracted expanding features for the query by downloading search results from search engine Bing; then we designed rule based method and multi-classifier voting method to classify user query intent separately; in formal run step, we combined the classification results produced by rule based method and multi-classifier voting method as final classification result to submit. In TIR subtask, firstly, we built an index for documents in the aim corpus using Lucene tool kit; secondly we calculated the content relevant score using BM25 model and the temporal relevant score based on the date distance between the query date and time expression tagged in document content; thirdly, we developed two rank methods, relevant score weight sum method and learning to rank method, to calculate the final relevant score for each document and rank relevant documents based on the final score; the subtopic classification method we used in TIR subtask is same as in TOIC subtask.

Keywords

query intent classification, temporal information retrieval, learning to rank, text classification, multi-classifiers voting

Team Name

HITSZ-ICRC

Subtasks

Temporal Query Intent Classification (TQIC) and Temporal Information Retrieval (TIR)

1. INTRODUCTION

Web search engines developed fast in recent years, it became mainly way for users to search information from web. The content update quickly in current internet web, and more and more user queries' intent have timeliness requirement. For example, when user input query "gold price", the query intent needs not only information retrieval returning pages with content about price of gold, but also needs recent web page about gold price. Temporal information retrieval has become a research focus in information retrieval and related research communities in recent years [1, 2, 3].

Temporal Information Access (Temporalia) task has been hosted at the 11th NTCIR workshop on Evaluation of Information Access Technologies (NTCIR-11) [1] as a pilot task. The task focused on two major sub-problems: query intent understanding and document ranking considering their temporal aspects. The first sub-problem was called *Temporal Query Intent Classification* (TQIC) as one subtask in Temporalia. The second sub-problem was called *Temporal Information Retrieval* (TIR) as another subtask in Temporalia [2].

TQIC subtask required to classify query intent into one of the following classes based on temporal element: *past*, *recency*, *future* and *atemporal*. Each query was given to participants with its query submitting date in this task. TIR subtask required to retrieve a set of documents in response to a search topic that incorporates time factor. Each search topic in this subtask was given to participants with topic title, topic description, search date, and four subtopics in the four temporal query intent classes.

Intelligent Computing Research Center of Harbin Institute of Technology Shenzhen Graduate School (HITSZ-ICRC) participates in Temporalia task of NTCIR-11 this year and worked on both subtask TQIC and subtask TIR, and submitted 3 formal run results for each subtask.

In TQIC subtask, we thought query intent classification as short text classification problem. Firstly, query intent was classified using methods based on user query literal features; secondly, query intent was classified using methods based on features expanding from user query; finally, formal run query intent classification results were produced by combining and voting the query classification results got by all the classification methods in first and second steps.

In TIR subtask, for the subtopic intent class information is not allowed to use in TIR formal run, it is necessary to classify subtopic intent of each subtopic in every search topic. The classification methods developed in TQIC subtask was used to classify subtopic intent here. To retrieval the related page quickly and effectively, an index was built based on the corpus for TIR subtask. In relevant documents ranking step, two methods were developed: the first method called relevant score weight sum, it used temporal element as a rank weight for candidate relevant document, and add the weight to the content relevant weight; the second rank method used the temporal element as rank feature, and use learning to rank algorithms to rank candidate relevant document.

* Corresponding Author

2. TEMPORAL QUERY INTENT CLASSIFICATION

TQIC subtask required to classify user query into four classes: *past*, *recency*, *future* and *atemporal* [2]. In this step we tried methods for text classification to do temporal query intent classify.

2.1 Rule Based Method

Some special obvious features can be easily extracted after analysis and statistic user queries. For example, queries such as "French Open 2013 Live Scores", "Movies 2012" and "Upcoming Movies in 2013" contain digital year strings, its intent class is easy to judge based on the digital year in query and the query submitting date; queries such as "current price of gold", "Did the Pirates Win Today" and "weather for tomorrow" contain explicit time words which also can be used to judge query intent class directly; queries such as "fb stock price" and "long term weather forecast" contain time-sensitive words.

With those obvious classification features, classification rules were built manually, which were described as following.

(1) Date distance: compare the date in query and query submitting date, if submitting date earlier than date in query, the query intent class is future, if submitting date later than date in query, the query intent class is *past*.

(2) Time-sensitive word dictionary: first, built a dictionary for time-sensitive words. In the dictionary, time-sensitive words were saved with class label. For example, word "tomorrow" was save with class label "future", "current" was save with class label "recency" in the dictionary. At the query intent classification step, just judge whether the query contains time-sensitive words in the dictionary, if the query contains time-sensitive word, used the word class label as query intent class [4].

(3) Combining date distance and verb tense, the query submitting date and date in query is in same year, if the verb in query is present tense or future tense, the query intent class is "future", otherwise, the class is "*recency*" or "*past*".

2.2 Machine Learning Method

Temporal query intent classification can be considered as text classification problem here, machine learning algorithms were used to classify temporal query intent. 2 aspects features were extracted for machine learning methods. First, different linguistic level features were extracted from user query directly. Every user query is short, only contain some keywords, the classification ability of the literal features of query is limited, so feature expanding based on user query was used as second aspect features. Expanding features were extracted from the search results downloaded from search engine *Bing* for each query.

2.2.1 Query literal feature extraction

Query feature was extracted based on the literal. Here 5 groups features were extracted shown as following.

(1) N-gram terms of query, extracted all the n-gram terms of each query as features;

(2) POS n-gram, extracted the n-gram string of the POS of words in query as features;

(3) Named entity, whether query contains named entity;

(4) Normalized date, whether query contains date string that can be normalized;

(5) Date distance, if the query contains date string that can be normalized, compute the date distance between date in query and query submitting date, used the distance as feature;

(6) Special word, whether the query contains words in the time-sensitive word dictionary.

2.2.2 Expanding feature extraction

To expanding classification features for user query, the search results from commercial search engine *Bing* for each query were downloaded to extract feature.

First, the top 50 search results were downloaded from commercial search engine *Bing* for each query. Second, the title and snippet for each search result were extracted. Third, n-gram terms were extracted from the title text and snippet text and were used as expanding features.

2.2.3 Feature selection

In feature set extracted for each query, some features appeared in only one query, some features appeared in every query. All those features were ineffective for query intent classification. So it is necessary to do feature selection to get out the features that are effective for query intent classification. Here *information gain* and *gain ratio* were used to feature selection.

2.3 Multi-Classifier Voting and Result Combining

The formal run query intent classification result is not sure before the answer public. But train set for classification models is too small for the task, here only used dry run query as train set. So the model performance on each formal run query cannot be sure whether effective. Here multi-classifier voting method was used to improve the final formal run query classification results. And different voting strategies were included: (a)same features different algorithms, (b)different features same algorithm, (c)different features different algorithms.

The accuracy of the rule based method is higher than other methods, but for the rules are designed manually, the coverage rate of each rule is low. And the recall ratio for rule based method is low in formal run query set. To use the advantage of rule based method and machine learning method both, the final results of rule based method and machine learning method margining is a best way: giving a query, if the rule based method can classify it, use the rule based method result as the submit class, if the rule based method cannot classify, used the machine learning method result as the submit class.

3. TEMPORAL INFORMATION RETRIEVAL

TIR subtask asked participants to submit top 100 relevant documents for each subtopic in every search topic. For *atemporal* subtopic, the result documents should be relevant to the subtopic in content. For temporal subtopic, the result documents should not only be relevant to the subtopic in content, but also meet the temporal requirement of the subtopic.

To get the relevant documents that meet the TIR subtask requirements, a correlation value that can show a document satisfy the content relevant and temporal requirement between a subtopic and a document need to be calculated. Steps were used here to calculate the relevant value: first, got a smaller candidate documents set for each subtopic by index searching; second, calculated temporal relevant score between candidate document and search subtopic; third, weight sum using the content relevant score and temporal relevant score.

3.1 Candidate Relevant Document Searching

Giving a subtopic, most documents in corpus are irrelevant in content. And it is time waste to calculate relevant score for every document. So quickly method to get the candidate relevant document for a search subtopic is necessary, and document indexing is an effective way.

First, building an index on the document corpus to do relevant document searching. Second, giving a search subtopic, search all the relevant documents based on the index and get the relevant documents list. Third, save the top N most relevant documents as candidate relevant documents and save the content relevant score for each document calculated by the index searching model, the relevant score will be used to calculate the final relevant score for the document.

3.2 Temporal Relevant Score Calculating

Giving a document for a subtopic, how to judge whether the document satisfy the temporal requirement is a core problem for TIR subtask.

Time expressions in documents were annotated out in the corpus. And each time expression had been normalized [1]. So it is easy to get the temporal relation between time expression in document and search date of the subtopic. Firstly each time expression was classified to *past*, *recency* or *future* based on its relation with search date. Secondly, temporal relevant score for the document was calculated based on the class of the time expressions. Time expression was classified with following formula.

$$dis_i = Dq - DX_i$$

$$C_i = \begin{cases} future, & if\ dis_i < 0 \\ past, & if\ dis_i > B_p \\ recency, & if\ 0 \le dis_i \le B_r \end{cases}$$

Where Dq is search date of the topic, DX_i is normalized time expression in document, B_p is the classification boundary for *past* class time expression, B_r is the classification boundary for *recency* class time expression.

To calculate temporal relevant score between a document and a search subtopic, it needs to judge whether class of time expression in the document and class of search subtopic is same. If it exist at least one time expression which have same class with the search subtopic, the temporal relevant score *TR* for the document is 1, otherwise, the score *TR* is 0.

The Classification boundary time distance for *past* and *recency* was set to 300 days here.

3.3 Document Re-Ranking

In TIR subtask, temporal factor should be considered when ranking relevant documents. A document that is relevant to search subtopic in content and meets the temporal requirement of the search subtopic at same time should be more relevant the document that only meet the content relevant requirement or temporal requirement for the subtopic.

The candidate documents are the top *N* documents in the content relevant result list. Here it needs to re-rank the result documents based on content relevant and temporal relevant. Two document re-rank methods were designed: relevant score weight sum method and learning to rank method.

For the content relevant score produced by index searching is out of the range [0, 1], the content relevant score had been first normalized to value [0, 1] for each document in candidate list.

3.3.1 Relevant score weight sum

The content relevant and temporal relevant are both import for temporal query. Relevant score weight sum is an intuitive way to calculate the final relevant score. Here a linear combination of content relevant score and temporal relevant score is used to get final relevant score for a document.

$$R = \alpha R_c + (1-\alpha) R_t$$

Where R is the document final relevant score to the search subtopic, R_c is the content relevant score, R_t is the temporal relevant score, α is the weight coefficient and $\alpha \ge 0$, $\alpha \le 1$.

Different coefficient value was set for different subtopic class. If the subtopic class is *atemporal*, it become to content relevant problem, and content relevant score was used as final relevant score: $R = R_c$.

Table 1. Coefficient value for relevant score weight sum method

Subtopic Class	Coefficient
past	0.85
recency	0.73
future	0.76
atemporal	1

3.3.2 Learning to rank

It is difficult to get the best coefficient for the relevant score weight sum method. Each coefficient was selected by trying over and over. Temporal factor can be used as feature to train rank model. So learning to rank was used as another method to re-rank the candidate documents. Here the feature extraction method referenced [5].

The features were used to learning to rank including: similarity between search topic and document title, similarity between search topic and document content, similarity between search subtopic and document title, similarity between search subtopic and document content, BM25 relevant score between search topic and document, BM25 relevant score between search subtopic and document, temporal relevant score of a document. Document was transferred to feature vector for ranking model training and testing.

4. SUBMITTED RESULTS

4.1 Temporal Query Intent Classification

In TQIC subtask, we used Stanford CoreNLP [6] to extract N-gram POS feature, named entity feature and normalized date

feature. To get the expanding features, we downloaded the top 50 search results for each query from search engine Bing (www.bing.com) and used those search results as source text for expanding feature extraction. We used toolkit WEKA [7] to do the feature selection and classification model training and testing. All the parameters of algorithms used in WEKA were default setting.

In model training step, we used the dry run queries as training data to train classification model. There are only 100 queries in dry run dataset, 25 for each class. The training dataset is too small to get effective classification model. So in formal run step, we used multi-classifier voting and combination method on the results from different classifier algorithms and with different groups features to get best results to submit. The results used to vote were produced by the models whose performance was top 5 best on training data.

In formal run query set, there are 300 queries in total, 75 queries in each query class. We submitted three runs for TQIC subtask: HITSZ_PrW, HITSZ_PrWsQW and HITSZ_qRPrHNB. The results evaluation of formal runs is shown in table 2 and table 3.

Table 2. Results evaluation of TQIC formal runs

runID	Correct Number	Precision
HITSZ_PrW	207	**69 %**
HITSZ_PrWsQW	203	67.67%
HITSZ_qRPrHNB	201	67%

Table 3. Precision of each class in TQIC formal runs

runID	atemporal	future	past	recency
HITSZ_PrW	**70.67%**	64.00%	78.67%	**62.67%**
HITSZ_PrWsQW	69.33%	66.67%	77.33%	57.33%
HITSZ_qRPrHNB	57.33%	**68.00%**	**81.33%**	61.33%

Giving a query, all the three run first used rule based method to judge its class, if the rule based method cannot judge the query class, used the multi-classifier voting result as the final query class. In run HITSZ_PrW, the voting result was from the classifier trained only with expanding features. In HITSZ_PrWsQW, the voting result was from the classifier trained with both query literal features and expanding features. In HITSZ_qRPrHNB, the result was from the classifier trained only with query literal features.

Detail steps to get HITSZ_PrW: giving a user query, first, judge its class using the PRISM rule set produced by classifier Prism trained with query literal features; second, if the query cannot be classified, use multi-classifier voting method to classify. The multi-classifier voting method here includes 2 voting levels: the first level includes 5 classifiers: John Platt's sequential minimal optimization algorithm (SMO), HyperPipe, Hidden Naive Bayes (HNB), Naive Bayes, logistic regression; the second level includes 3 classifiers: HyperPipe, HNB, SMO. The features used for multi-classifier voting method are expanding features.

Steps to HITSZ_PrWsQW: for query, first, judge its class using the PRISM rule set; second, if the query cannot be classified, use multi-classifier voting method to classify. The multi-classifier voting includes 2 levels: irst level includes 5 classifiers: logistic regression, HyperPipe, SMO, HNB, MultilayerPerceptron; second level includes 3 classifiers: MultilayerPerceptron, HyperPipe, AODEsr. The features used include query literal features and expanding features.

Steps for HITSZ_qRPrHNB: for user query, first, judge its class using Rule set collected manually; second, if the query cannot be classified, use the PRISM rule set; third, if the query still cannot be classified, use classifier HNB. The features used for HNB here are only query literal features.

The results shows that the combination rule based method and voting method only using expanding features is most effective. But the distribution of formal run results is different to dry run results. The precisions of all dry run results are above 0.9, but the precision of all the formal run result are less than 0.7. This may be caused by the shortage of training data for machine learning methods. The precision in different class is different for all the runs. The result is imbalance for different class. In the 3 runs, precision of class *past* is higher than other class. Precision of class *recency* is lower than other class. This shows that the features we extracted are more effective for class *past*. So extracting more effective features for *recency* class query is a way to improve classification precision for TQIC task.

4.2 Temporal Information Retrieval

In TIR subtask, we used Lucene to build index for the "LivingKnowledge news and blogs annotated sub-collection" corpus [2]. In candidate documents search step, we used BM25 [8] model to search candidate relevant documents for each search subtopic.In learning to rank method step, we used the LambdaMART algorithm in RankLib tool kit to training rank model. The train data used in this step was the dry run search subtopic.

TIR subtask required participant not use the class information of search subtopic. But our ranking method was designed based on subtopic class, for different class using different parameters. When re-ranking the candidate documents, the subtopic class is necessary. Here the classification methods developed for TQIC subtask were used to classify the search subtopic. In ranking step, different parameters were chosen based on the search subtopic classified results.

There are 50 search topics in formal run topic set, each class 1 subtopic, and 200 search subtopics in total. 3 formal runs were submitted: HITSZ_BW, HITSZ_BWCC and HITSZ_LTRNC2. The results evaluation of formal runs submitted is shown in table 4 and table 5.

Table 4. Results evaluation of TIR subtask runs

runID	nDCG@20	AP@20	P@20	nERR@20
HITSZ_BW	0.4544	0.4587	0.5895	0.6056
HITSZ_BWCC	0.4554	0.4599	0.5902	0.6064
HITSZ_LTRNC2	0.4768	0.483	0.6018	0.6313

Table 5. nDCG@20 of each class in TIR formal runs

runID	atemporal	future	past	recency
HITSZ_BW	0.4669	0.4607	0.4005	0.4897
HITSZ_BWCC	0.4678	0.4593	0.403	0.4915
HITSZ_LTRNC2	**0.5092**	**0.4804**	**0.4227**	**0.495**

HITSZ_BWCC and HITSZ_LTRNC2 did not use the original class information of search subtopic in formal run search topics. HITSZ_BW used the original class information of search subtopic. HITSZ_BW and HITSZ_BWCC used relevant score weight sum method to re-rank candidate documents. HITSZ_LTRNC2 used the rank model trained with LambdaMART algorithm to re-rank the candidate documents. All the three runs used search topic and subtopic as the search string in candidate documents retrieval step.

The results of the 3 runs were evaluated by the NTCIR evaluation tool NTCIREVAL [9]. Table 4 shows the average nDCG@20, AP@20, P@20 and nERR@20 values of the 3 runs. Table 5 shows the detail nDCG@20 value of each subtopic class in the results of the 3 runs. The evaluation results show that the learning to rank method is most effective for TIR subtask in the 3 runs. HITSZ_BW and HITSZ_BWCC have little difference. This shows that if the precision for search subtopic classification is accurate enough, the negative influence to candidate documents re-ranking would become less.

There is big gap between the nDCG values of different search topics in the 3 runs results. For example, nDCG value of *recency* subtopic in search topic "Air pollution and its health effects" (topic 036) and "Social media impact" (topic 015) is 0.9159 and 0.8887 in HITSZ_LTRNC2 results, nDCG value of "Coffee: its advantages and disadvantages" (topic 050) and "Trends of popular movies" (topic 042) is 0.0772 and 0. The nDCG value of the result is dependent on the number of relevant documents of each subtopic in the formal run *qrels* file. The number of relevant document for subtopic 036r, 015r, 050r and 042r is 141, 173, 68 and 24, which shows that the more number of relevant documents in answer file, the higher nDCG value for each subtopic.

This phenomenon causes by 2 reasons: first, the result of candidate relevant document retrieval based on Lucene toolkit is not accurate enough, some candidate document has high content relevant score based on BM25 in Lucene toolkit, but it is not relevant to the search topic in content; second, the boundary between class *recency* and *past* on date has no adaptive ability to the document corpus, for example, the query date for each search topic is in year 2013, but some L2 relevant documents for *recency* class search subtopic in *qrels* file is in year 2011, our method used those documents as *past* class. For this error propagation reason in the 2 steps, a relevant document becomes less chance to be retrieved. The issue on candidate document retrieval and date boundary setting is the room for further improving.

5. COCLUSION AND DISCUSSION

This paper presents the methods HITSZ-ICRC group used for Temprolia task at the NTCIR-11.We participate in TQIC subtask and TIR subtask. For TQIC subtask, the final results were produced by combining results of rule based method and results of multi-classifier voting based on different machine learning algorithms. For TIR subtask, relevant score weight sum method and learning to rank method were used to do temporal information retrieval. Results evaluation demonstrates that the methods proposed were effective for TQIC and TIR subtask.

There is much room left to further improve both TQIC result and TIR result for our methods. In TQIC subtask, shortage of training cases for classification model training is main problem. There are two ways to improve the classification methods in further research: first, getting more training cases for model training by annotating more user queries extract from SE user log as training data; second, designing more classification rules based on the query syntax features and query expanding features.

In TIR subtask, the temporal relevant score used here is binary, 0 or 1. The score cannot indicate how relevant a document to a subtopic in term of temporal side. In further research, we plan to use float data between 0 and 1 to indicate the temporal relevant. For learning to rank method, the training cases shortage also is import element to improve rank model. It is necessary to get more training data to train more effective rank model. Documents in the "LivingKnowledge news and blogs annotated sub-collection" corpus have been tagged named entity and time expression. In our methods time expression tag in document was used to judge temporal relevant for the document, but the named entity tag have not been used. The named entity may be useful for judging content relevant and finding the important time express in a document. So in further research we plan to use the named entity tag in the document to do TIR subtask.

6. ACKNOWLEDGMENTS

This work is supported in part by the National Natural Science Foundation of China (61173075, 61203378 and 61272383), the Strategic Emerging Industry Development Special Funds of Shenzhen (ZDSY20120613125401420 and JCYJ20120613151940045) and the Key Basic Research Foundation of Shenzhen (JC201005260118A).

7. REFERENCES

[1] H. Joho, A. Jatowt, and B. Roi. NTCIR Temporalia: A Test Collection for Temporal Information Access Research. In *Proceedings of the 4th Temporal Web Analytics Workshop (TempWeb 2014)*, ACM Press, Seoul, Korea, pages 845-849, 2014.

[2] H. Joho, A. Jatowt, and B. Roi. Overview of NTCIR-11 Temporal Information Access (Temporalia) Task. In *Proceedings of the 11th Evaluation of Information Access Technologies (NTCIR)*, 2014.

[3] H. Joho, A. Jatowt, and B. Roi. A survey of temporal web search experience. In P*roceedings of the 22nd International Conference on World Wide Web Companion*, WWW '13 Companion, pages 1101-1108, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.

[4] Y. Hou, Y. Zhang, X. Wang, Q. Chen, Y. Wang and B. Hu. Recognition and Retrieval of Time-sensitive Question in Chinese QA System [J]. Journal of Computer Research and Development, 2013, 50(12), pages 2612-2620

[5] J. Xu, H. Li. AdaRank: A Boosting Algorithm for Information Retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 391-398, New York(NY), USA: ACM, 2007.

[6] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55-60, 2014.

[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. Witten. The WEKA Data Mining Software: An Update. SIGKDD Explorations, Volume 11, Issue 1, 2009.

[8] S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, CIKM '04, pages 42-49, New York, NY, USA, 2004. ACM.

[9] T. Sakai, R. Song. Evaluating diversified search results using per-intent graded relevance. In *Proceedings of the 34st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '11, 2011.