TUTA1 at the NTCIR-11 Temporalia Task

Hai-Tao Yu[‡][*]
yu-haitao@iss.tokushima-u.ac.jp

Xin Kang[†][‡][*]
xkang@tongji.edu.cn
kang-xin@iss.tokushima-u.ac.jp

Fuji Ren[‡]
ren@is.tokushima-u.ac.jp

[†]Electronics and Information, Tongji University
[‡]Faculty of Engineering, The University of Tokushima

ABSTRACT

This paper details our participation in the NTCIR-11 Temporalia task including Temporal Query Intent Classification (TQIC) and Temporal Information Retrieval (TIR). In the TQIC subtask, we explore the rich temporal information in the labeled and unlabeled search queries. Semi-supervised and supervised linear classifiers are learned to predict the temporal classes for each search query. In the TIR subtask, we perform temporal ranking based on the technique of learning-to-rank. Two classes of features are investigated for estimating the document relevance.

Team Name

TUTA1

Subtasks

Temporal Query Intent Classification (English)
Temporal Information Retrieval (English)

Keywords

Temporal feature extraction, Semi-supervised learning, Learn to rank

1. INTRODUCTION

The TUTA1 group at The University of Tokushima participated in both subtasks of the NTCIR-11 Temporalia task, i.e., the Temporal Query Intent Classification (TQIC) subtask and the Temporal Information Retrieval (TIR) subtask. For detailed introductory information of each subtask, please refer to the task design paper [4] and the overview paper [5]. Both subtasks focus on the identification of temporal information across the "past", "recent", "future", and "atemporal" categories for information retrieval. In the TQIC subtask, both search queries and query submission dates have been provided for predicting the users' temporal intents. Because the query strings are usually very short, e.g. 4.2 words in dry-run on average, to find useful temporal features in queries and to explore the background information (especially for named entities) seem to be prominent in this subtask. For those search queries with explicit temporal expression or predicate verbs, we extract the "time gap" and "verb tense" features, separately. We also investigate the word lemmas and named entities in search queries, which are very sparse but count the majority of the query contents. As an external resource, the AOL 500K User Session Collection[1][8] has been employed to further expand our knowledge of temporal features from the background, through a semi-supervised learning model. In the TIR subtask, we experiment with the learning-to-rank technique for temporal ranking. Based on the state-of-the-art algorithms for learning ranking models, we mainly focus on feature selection.

In the remainder of this paper, we outline the notations, the methods for intermediate natural language parsing in §2. §3 and §4 detail the approaches proposed for TQIC and TIR subtasks respectively. We conclude our work in §5.

2. PRELIMINARIES

In this section, we first outline the notations used throughout this paper, then the mechanisms we designed for intermediate document parsing are introduced.

The document collection "*LivingKnowledge news and blogs annotated subcollection*" released in Temporalia task is denoted as $\overline{D} = \{\overline{d}_1, ..., \overline{d}_n\}$. $\overline{d}$ denotes an annotated document. The meta-info of a document includes its unique *identifier*, *host name* where the document page was pulled from, *date* when the page was published, *url* where the document page was pulled from, *source css* that was accessed to retrieve the page, and *title* of document page. A document $\overline{d}$ is represented as a sequence of annotated sentences, i.e., $\overline{d} = \{\overline{se}_1, ..., \overline{se}_m\}$. The named entities and temporal expressions in a sentence are annotated. Correspondingly, d and se respectively denote the document and sentence, from which the tags of named entities and temporal expressions are removed.

In context of Temporal Query Intent Classification (TQIC) subtask, τ denotes a temporal class, "P", "R", "F", and "A" are the abbreviations for past, recent, future, and atemporal respectively. In context of the TIR subtask, t denotes a topic, st denotes a subtopic, $stStr$ denotes a subtopic string, $tit(t)$ and $des(t)$ denote the title and description of a topic respectively. For both subtasks TQIC and TIR, we use the concept of *search query* to refer to the input string that is used to perform retrieval. For the subtask of TIR, the *Stanford CoreNLP*[2] (version 3.3.1) is used to perform part-of-speech (POS) tagging. To determine the tense of a sentence, we make use of the verb's POS tag. Specifically, the tag "VBD" is used to identify past tense, the tags "VBP",

[*]These authors contributed equally to this work. Xin Kang is mainly responsible for the TQIC subtask; Hai-Tao Yu is mainly responsible for the TIR subtask.

[1]http://www.gregsadetsky.com/aol-data/
[2]http://www-nlp.stanford.edu/software/corenlp.shtml

"VBZ" and "MD" are used to identify non-past tense. For a tagged token, it will be regarded as a noun term if its POS tag begins with "N" or "n".

3. TEMPORAL QUERY INTENT CLASSIFICATION

Temporal Query Intent Classification (TQIC) subtask challenges text classification in two aspects: to recognize temporal features in search queries, and to explore the background information in these queries. In this section, we describe the temporal information extraction for search queries, and illustrate the learning of supervised and semi-supervised classifiers, based on the labeled and unlabeled training examples, for Temporal Query Intent Classification (TQIC).

3.1 Temporal feature extraction for TQIC

1. **Time Gap Features**. The ideal temporal features in a query string should indicate the gap between the intended time point in the search query and the query submission time, in which case the Temporal Query Intent Classification problem falls back to evaluating this time gap. For example, a particular month of a year as in "June 2013 movie releases" becomes a strong indicator of the "future" class given the query submitting date of "May 28, 2013 GMT+0".

The time gap features are represented as "DIFF_past" and "DIFF_future", depending on the relative difference between the query submission time points and the intended query time points. We employ the SUTIME library[2] in Stanford CoreNLP pipeline to recognize and normalize the temporal expressions in search queries. For example, a particular date "June 2013" is normalized as "2013-06", while a less particular date "2013 winter" is normalized as a season "2013-WI". Time gaps are acquired by directly comparing the normalized time and the query submission time. Besides, some temporal expression which indicates the time offsets could also be properly recognized. For example, "5 years from now on" and "last week" are recognized as "OFFSET P5Y" and "THIS P1W OFFSET P-1W", indicating a 5 years offset in the future (5Y) and a 1 week offset in the past (-1W).

However, the time gap features are rare in search queries. Within 100 dry-run search query samples, we extracted 7 temporal features indicating the "past" time point and 5 indicating the "future" time point, and within 3.1M AOL query samples, we extracted 4.8K such "past" features and 0.6K such "future" features. And in some cases, the time gap features turn less indicative when the recognized time overlaps the query submission time. For example, in the search query "comet coming in 2013", the recognized year 2013 does not indicate a meaningful time gap relative to the query submitting data "Oct 28, 2013 GMT+0". We represent these time gap features as "DIFF_same_year", "DIFF_same_month", or "DIFF_same_day", to indicate that the recognized time points and the query submission time are within the same period.

2. **Verb Tense Features**. Another important temporal feature in search queries is the verb tense. For example, the verb "was" in "Who Was the Youngest President" turns to be a strong indicator of the "past" intension.

We employ the Stanford POS tagger library[12] in Stanford CoreNLP pipeline, which uses the Penn Treebank tag set [7], to recognize verbs and to distinguish their tenses in the search queries. Verb tenses represented by the Penn Treebank tag set include the past tense (VBD), the singular present tense (VBZ/VBP for 3rd person/non-3rd person), the present participle (VBG), the past participle (VBN), and the base form (VB). Because the 3rd and non-3rd person verb forms make little difference to temporal classification, we represent the present tense as VBZ for both verb forms. In our temporal feature recognition, the verb tense features are represented by the combination of the part-of-speech tag and the verb lemma. For example, "was" in the previous search query is represented as "VBD_be", implying the "past" temporal intent.

In a search query, there could be multiple verbs with different verb tenses. For example, in the query "an experience that you enjoyed while learning something new", we get 2 verb tense features "VBD_enjoy" and "VBG_learn". These verb tense features are not equally important in determining the temporal class of a search query. In most cases, we can perform a meaningful syntactic parsing to a search query, by using the Stanford Parser library[11] in Stanford CoreNLP pipeline, and find the main predicate by picking the uppermost verb in the parse tree. The tense of the main predicate (uppermost verb) then determines the tense of the whole search query, and therefore implies the temporal intent of the search query. In our temporal feature recognition, we use the Uppermost Verb Tense "UVT_VB*" to represent the tense of the entire search query. Lemmas of the uppermost verbs are not attached to make this feature more distinguishable in a linear classifier. In this example, the uppermost verb tense feature is "UVT_VBD", which indicates the "past" class for the search query. Within 100 dry-run search queries, we extract 28 UVT features, and within 3.1M unlabeled search query samples, we extract 664K UVT features.

3. **Lemmas and Named Entities**. For search queries without indicative temporal expressions or verb tense information, words and recognized named entities could also indicate the temporal intents. For example, the word "history" in search query "history of slavery" indicates a query of the "past" events, and the recognized person name "Yuri Gagarin" in search query "Yuri Gagarin Cause of Death" suggests a "past" query if we possess the background information about this person, e.g. Yuri Gagarin is a historical people.

We employ the Stanford Named Entity Recognizer[3] in Stanford CoreNLP pipeline to recognize the named entities in search queries. Unlike the time gap and verb tense features, words and named entities count the majority of search query texts. The number of distinct word and named entity features is much bigger than the number of time gap and verb tense features, and these features are also much sparser in the search queries. We use word lemma instead of raw words to make these feature less sparse. However, to understand the meanings in lemmas and named entities or to reconstruct their functions in predicting the temporal intents in search queries, could still be difficult with limited resources.

To expand our knowledge in existing temporal features within a small number of labeled examples to the unknown features in other unlabeled examples, we employ a semi-supervised Support Vector Machine classifier SVMlin[3]. For a detailed description of SVMlin, please refer to [9, 10]. We explore the lemma and named entity features as well as the

[3]http://vikas.sindhwani.org/svmlin.html

time gap and verb tense features in the raw search queries from AOL 500K User Session Collection[8].

3.2 Experiments

3.2.1 Experimental Setup

The Temporal Query Intent Classification (TQIC) subtask provides a dry-run dataset of 100 search query samples for developing the temporal classification models, and a formal-run dataset of 300 search query samples for examining the results. Both datasets comprise 4 fields, including an "id" for each query example, a "query_string" of the text content, a "query_issue_time" recording the query submission date in the form of "Month Day, Year GMT+0", and a "temporal_class" representing the temporal intent of a query, consisting of "past", "recent", "future", and "atemporal". Because the "temporal_class" field for last 20 search query examples in the dry-run dataset was empty, we only use the first 80 examples in our experiment.

Besides, the AOL 500K user Session Collection [8] has been employed as the extra unlabeled training corpus for feeding the semi-supervised SVM classifier. Totally 5 fields are recorded in this dataset, including "AnonID" as an anonymous user ID, "Query" as the query text, "QueryTime" as the submission time in the "YYYY-MM-DD hh:mm:ss" format, "ItemRank" and "ClickURL" which record the user clicks (if any) in the search sessions. From 36M search query samples in the AOL dataset, we choose those samples which consist of at least 3 words in the query text and at the same time have been clicked ("ItemRand" and "ClickURL" are not empty) during the search session. The filtered AOL dataset contains 3.1M unique search queries, with the "Query" and "QueryTime" fields extracted, which are analogous to the "query_string" and "query_issue_time" fields in the TQIC datasets.

Features described in Section 3.1 are extracted for both the TQIC dataset and the AOL dataset. We build two feature lists, based on the 80 labeled search query examples in dry-run and the queries in both dry-run and AOL, for the supervised and semi-supervised training respectively.

To learn the supervised models for TQIC, we employ the Logistic Regression Classifier in the scikit-learn 0.15.0[4], which is a machine learning package in Python. Model parameters "C" and "penalty" are selected through a 5-fold cross validation on 80 labeled training examples. We use the overall Precision, which counts the proportion of search queries with correctly predicted temporal labels among the validation set, to evaluate the model performance. To further explore the lemma and named entity features for TQIC with large amount of unlabeled query samples, we employ the SVMlin package[9, 10], which implements the linear SVM classifier with the semi-supervised extension. Rather than utilizing all the unlabeled samples in AOL dataset for semi-supervised training, we randomly choose a specified number (N) of search queries from AOL, and combine them with the labeled examples in the TQIC training set to learn a classifier. The SVMlin model parameters[5] A, W, U, R, and the unlabeled query amount N for semi-supervised training are selected through a 5-fold cross-validation, based on the overall Precision scores.

3.2.2 Temporal Query Intent Classification Runs

We submitted 3 runs for the TQIC subtask. To examine the effectiveness of the time gap and verb tense features in TQIC, we add a 4th run as the baseline:

- TUTA1-TQIC-RUN-1. Temporal labels are predicted by a Logistic Regression classifier, trained on the 80 labeled search query examples in the dry-run dataset, based on all temporal features introduced in Section 3.1. Model parameters $C = 30$, and $penalty = l_1$ have been selected through a 5-fold cross-validation.

- TUTA1-TQIC-RUN-2. Temporal labels are predicted by a Logistic Regression classifier, trained through a similar process as in TUTA1-TQIC-RUN-1, except that the second best parameter combination $C = 300$, and $penalty = l_1$ has been selected.

- TUTA1-TQIC-RUN-3. Temporal labels are predicted by an SVMlin classifier, which has been trained on the 80 labeled search query examples in the dry-run dataset and another $N = 10$K unlabeled search query samples randomly drawn from the AOL dataset. Model parameters $A = 2$, $W = 0.03$, $U = 3$, and $R = 0.03$ have been selected through a 5-fold cross-validation.

- TUTA1-TQIC-RUN-4. Temporal labels are predicted by a Logistic Regression classifier, trained through a similar process as in TUTA1-TQIC-RUN-1, except that only the lemma and named entity features are employed. Model parameters $C = 3$, and $penalty = l_2$ have been selected through a 5-fold cross-validation.

3.2.3 Experimental Results

We report the evaluation of 4 TQIC runs with the Precision scores. As defined in the TQIC subtask, Precision for each temporal class τ is calculated as

$$P(\tau) = \frac{correct(\tau)}{total(\tau)}, \qquad (1)$$

in which $correct(\tau)$ is the number of correctly predicted search queries in temporal class τ and $total(\tau)$ counts the total number of search queries in τ. And the overall Precision $\bar{P}$ is just the Micro-average of $P(\tau)$ for each τ:

$$\bar{P} = \frac{\sum_\tau correct(\tau)}{\sum_\tau total(\tau)}. \qquad (2)$$

Besides, we compare the TQIC results for each pair of runs through the Wilcoxon signed-rank test[6], to examine if a significant difference exists in the prediction for each class τ. The detailed evaluations are shown in Table 1.

RUN-1 achieves the highest overall Precision and the highest Precisions for the "future" and "atemporal" classes, while RUN-3 yields the highest Precisions for the "past" and "recent" classes. All the submitted runs RUN-1 to RUN-3 outperform the baseline RUN-4 in in the overall Precision. This result suggests that by exploring the time gap and verb tense features in search queries, TQIC results could be significantly improved (8.33% overall Precision increment in RUN-1). Precision improvement for "past" and "recent" in

[4]http://scikit-learn.org/stable/index.html

[5]A for SVM algorithm selection, W and U specify the regularization parameters λ and λ_u respectively, R specifies the positive class fraction in unlabeled data.

[6]http://en.wikipedia.org/wiki/Wilcoxon_signed-rank_test

RUN-3 implies that by exploring the lemma and named entity features through semi-supervised learning, the TQIC result could be improved.

The Precision evaluation in Table 1 also suggests that temporal classes “past” and “future” are relatively easier to predict than “recent” and “atemporal”. This could be because our temporal features are less distinguishable for the latter 2 classes. It is also important to notice that RUN-3 achieves significant improvement (over 10% compared to RUN-1) in recognizing “recent” search queries, which implies that the semi-supervised model could learn useful features for the “recent” class.

To look into the results generated from different runs, we calculate the confusion matrix for each run prediction and plot them in the gray-scale color maps as shown in Fig. 1. The subplots in the upper-left, upper-right, lower-left, and lower-right corresponds to the confusion matrices from RUN-1 to RUN-4 respectively. In each subplot, the cell located at row i and column j corresponds to the number of observations known to be in class i while predicted as class j. For all runs, the mis-prediction of “recent” to “future” counts the largest number of errors, while the mis-predictions of “atemporal” and “future” to “recent” count a significant part of the classification errors. Compared to the baseline RUN-4, none of our submitted runs have effectively reduce these errors.

We list some of the common errors generated through RUN-1 to RUN-4 in Table 2. All search queries in the formal-run were submitted in “May 1, 2013 GMT+0”. For “recent” to “future” errors, we find 10 samples contain the year “2013”, in which 3 contain the month “May”. As discussed in Section 3.1, the time gap features “DIFF_same_year” and “DIFF_same_month” which have been recognized for these temporal expressions turn to be less indicative of the temporal classes, since they cannot suggest a useful time gap. 11 search queries about “weather” have been recognized as “future”. Because in dry-run 5 out of 6 “weather” queries have been labeled as “future”, this error could reflect an over-fitting problem occurred in the model learning phase. 5 search queries about the “tonight” events are labeled as “recent” but predicted as “future”. Before we analyze this error, it is also interesting to find the same query string[7] “bruins game tonight time” from dry-run and formal-run, although submitted in different dates, has been labeled with “future” and “recent”, separately. We believe that in most cases “tonight” indicates a time point in the near “future”, and because the time gap is not very long, the interpretation of “recent” is also reasonable. This observation implies that the distinction between some temporal classes could become very vague, which may cause strong disagreements among different annotators. For the “atemporal” and “future” to “recent” errors, we find the over-fitting problem on some specific features, e.g. “cost”, “exchange rate”, and “score”, has been a major cause. And the failure of understanding named entities, e.g. “belmont stakes 2013” and “voice 2013”, is also responsible for some of the errors.

4. TEMPORAL INFORMATION RETRIEVAL

In this section, we investigate the effectiveness of learning-to-rank algorithms in Temporal Information Retrieval (TIR). For a detailed description of learning-to-rank technique, please refer to [1, 6]. Here we focus on feature selection, which is a fundamental issue in learning-to-rank.

[7]id 078 in dry-run and id 194 in formal-run

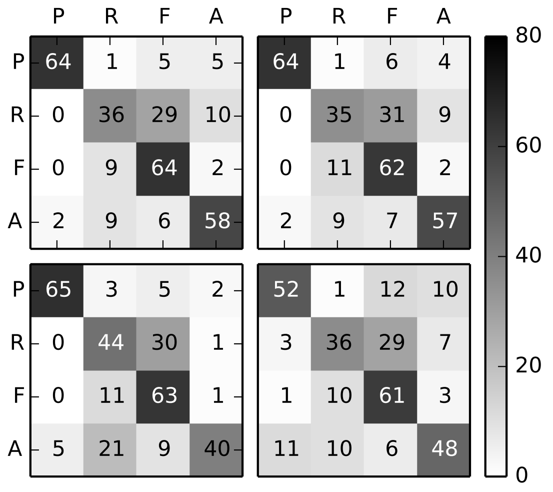

Figure 1: Confusion matrices plotted in the gray-scale color map for TQIC subtask. The subplots in the upper-left, upper-right, lower-left, and lower-right corresponds to the confusion matrices from RUN-1 to RUN-4.

Error	Search query
R-F	2013 printable calendar susan miller may 2013 weather for chicago local forecast the weather channel bruins game tonight time tv schedule tonight
A-R	cost of living comparison exchange rate calculator free credit score
F-R	belmont stakes 2013 winner voice 2013 winner

Table 2: Some common errors found through RUN-1 to RUN-4 for TQIC subtask.

4.1 Features

A total of 15 features are derived for learning a ranking model. To capture the temporal features, we resort to the temporal expressions and tenses of a document. A temporal expression in a document is a sequence of tokens that indicates a point in time, a duration or a frequency. For example, two pre-annotated temporal expressions are extracted as: *<t val=“201006”> June 2010</t>* and *<t val=“201105”>the end of May</t>*. Tense is a specific mechanism built into a specific language, which grammatically indicates when a situation takes place. English has only two morphological tenses characterized by the presence of a verb, i.e, *past* and *non-past*, the latter covering both present and future in one verb form. In our work, we take a sentence as a single unit. Given the noun terms of a search query, we classify sentences in a document into two types: (1) se^y, it refers to a sentence that includes one or more noun terms of a search query, (2) se^n, it refers to a sentence

Run	past	recent	future	atemporal	all
TUTA1-TQIC-RUN-1	0.8533	0.4800	**0.8533**	**0.7733**	**0.7400**
TUTA1-TQIC-RUN-2	0.8533	0.4667^{1}	0.8267^{1}	0.7600	0.7267
TUTA1-TQIC-RUN-3	$\mathbf{0.8667}^{1,2}$	$\mathbf{0.5867}^{1,2}$	$0.8400^{1,2}$	$0.5333^{1,2}$	0.7067
TUTA1-TQIC-RUN-4	$0.6933^{1,2,3}$	$0.4800^{1,2,3}$	$0.8133^{1,2,3}$	$0.6400^{1,2,3}$	0.6567

Table 1: Precision scores for TQIC subtask. The best results are indicated in bold for each separate temporal class and the overall precision. We use the Wilcoxon signed-rank test with $p < 0.05$ for testing statistical significance: the superscripts 1, 2, 3 indicate statistically significant differences to RUN-1, RUN-2, and RUN-3 respectively.

that includes no noun terms of a search query. Besides the temporal features, the topical feature is also used.

1. **Tense-centric Features**. 5 tense-centric features are designed. They are: (1) the ratio of past tense w.r.t. sentences of the type se^y, (2) the ratio of non-past tense w.r.t. sentences of the type se^y, (3) the ratio of past tense w.r.t. sentences of the type se^n, (4) the ratio of non-past tense w.r.t. sentences of the type se^n, (5) the ratio of tense of the target subtopic string w.r.t. all sentences.

2. **Features Based on Temporal Expressions**. We collect the temporal expressions from the two types of sentences respectively, then 6 features are designed based on two lists of temporal expressions (i.e., L^y corresponding to sentences of the type se^y and L^n corresponding to sentences of the type se^n). (1) Year variance based on L^y, (2) Month variance based on L^y, (3) Day variance based on L^y, (4) Year variance based on L^n, (5) Month variance based on L^n, (6) Day variance based on L^n. Based on the query issue time and document publication time, another 3 features are derived using an exponential decay function $DecayRate^{|val_1 - val_2|}$, where val_1 and val_2 are two variables of the same temporal unit. The intuition is to generate a score that decreases proportional to the time span between val_1 and val_2. The less time span, the more temporally similar they are. Specifically, we calculate decay scores using (7) year, (8) month and (9) day respectively, where the $DecayRate$ is set as 0.5.

3. **Topical Feature**. The topical feature is used to indicate the topical relevance between a search query and a document. In our work, the relevance score of the selected retrieval model is directly used, e.g., the score for a query-document pair with the *TFIDF* model.

4.2 Experiments

4.2.1 Experimental Setup

Given the provided document collection "*LivingKnowledge news and blogs annotated subcollection*" (denoted as $\overline{D}$), the script *CheckSyntax.class* is used to turn the original document collection files into sanitised XML format. The script *temporalia_solrify.pl* is used to strip all tags from the documents, and the obtained collection is denoted as D. Furthermore, $\overline{D}$ and D are indexed separately using *Solr*[8] (version 4.8.1). D is used to perform retrieval with different models. For a retrieved document, its tagged version is retrieved against $\overline{D}$ via its unique identifier. The temporal features (i.e., tense-centric features and features based temporal expressions) are generated based on tagged documents.

The 15 dry-run topics and their *rels* file[9] are used to generate datasets for learning ranking models. Each row in a dataset is a query-document pair: the fist column is the relevance judgement, the second column is query id, the subsequent columns are features, and the end of a row is comment about this pair (e.g., document identifier). For each temporal class (*past*, *recency*, *future* and *atemporal*), a dataset is generated as follows: for each subtopic t (w.r.t. a dry-run topic T) of the same temporal class, we perform the initial retrieval (the search query consists of the topic string, topic description and the subtopic string) using a specific retrieval model (e.g., TFIDF), and the top-50 documents are used. If a document is a really relevant document (i.e., indicated by the rels file), its relevance judgement is 1, otherwise 0. The temporal features are extracted based on the tagged version. For one subtopic, 50 rows of query-document pairs are generated, which are ranked in an decreasing order of relevance judgement. Each class-specific dataset can be used to learning a ranking model so as to predict document ranking for a formal-run subtopic of the same temporal class. Besides, we also combined the 4 datasets corresponding to 4 temporal classes into an entire dataset. We use this dataset to learn a ranking model, and predict document ranking for formal-run subtopics of each temporal class. Based on RankLib[10], we compared different learning algorithms included in this toolkit via 5-fold cross validation using the dry-run topics. Finally, the LambdaMART[13] learning algorithm is used.

4.2.2 Temporal Information Retrieval Runs

We submitted 3 runs for the TIR subtask. Here the 4th run is added as a baseline:

- TUTA1-TIR-RUN-1. The retrieval model LM is used to perform retrieval for generating training datasets and formal runs. The ranking model is learnt based on the entire dataset.

- TUTA1-TIR-RUN-2. The retrieval model LM is used to perform retrieval for generating training datasets and formal runs. The ranking model is learnt based on class-specific datasets.

- TUTA1-TIR-RUN-3. The retrieval model TFIDF is used, and the ranking model is learnt based on the entire dataset.

- TUTA1-TIR-RUN-4. As an unofficial baseline, this run merely relies on the retrieval model LM.

4.2.3 Experimental Results

[8]http://lucene.apache.org/solr/

[9]NTCIR-11TIRTopicsDryRunSupplementaryDocuments.docx

[10]http://sourceforge.net/p/lemur/wiki/RankLib/

Run	P@20					nDCG@20				
	past	recency	future	atemporal	all	past	recency	future	atemporal	all
TIR-RUN-1	0.4890	0.6080	0.5920	0.5840	0.5683	0.3717	0.4742	0.4536	0.4511	0.4377
TIR-RUN-2	**0.5130**	**0.6300**	0.5720	**0.6020**	0.5792	0.3803	**0.4920**	0.4241	**0.4673**	0.4409
TIR-RUN-3	0.5120	0.6090	**0.6180**	0.5940	**0.5832**	**0.3968**	0.4832	0.4673^{2}	0.4425	0.4474
TIR-RUN-4	$0.4190^{2,3}$	0.5790	0.5530^{3}	0.5150^{2}	$0.5165^{1,2,3}$	0.3876	0.4893	**0.4819**	0.4496	**0.4521**

Table 3: Results for TIR subtask. For each temporal class plus "all", the best result is indicted in bold. We used the Wilcoxon signed-rank test with $p < 0.05$ for testing statistical significance: the superscript $1, 2, 3$ indicate statistically significant differences to RUN-1, RUN-2, RUN-3 respectively

Table 3 shows the performance of the 4 runs, where $P@l$ and $nDCG@l$ are used as the evaluation metrics with a cut-off value of 20. $P@l$ represents the precision at l, $nDCG@l$ represents the normalised discounted cumulative gain.

From Table 3, we can see that: (1) Based on the same LM retrieval model, RUN-2 that uses a class-specific ranking model performs better than RUN-1 (past, recency, atemporal), whose ranking model is learnt by combining the training datasets of each temporal class. The probable reason is that the adopted features perform inconsistently with respect to different temporal classes, thus class-specific ranking model is better. (2) Based on the TFIDF retrieval model and a ranking model learnt using the entire training dataset, RUN-3 is able to retrieve more relevant documents for the subtopic of future. (3) All learning-to-rank models perform better in terms of rank-insensitive metric P@20 than the baseline method. But in terms of $nDCG$@20, not all learning-to-rank models outperform the baseline method. For the future temporal class, the baseline method even outperforms the 3 learning-to-rank models. The overall results demonstrate that the technique of learning-to-rank can improve the temporal retrieval performance.

5. CONCLUSIONS

In this paper, we described our approaches to solving the TQIC and TIR subtasks in the NTCIR-11 Temporalia task.

For TQIC, 3 temporal features, i.e. the time gap feature, the verb tense feature, and the lemma and named entity feature, were employed to recognize the temporal information in search queries. Besides, an external resource AOL 500K User Session Collection, was employed to explore the temporal information from the background, with a semi-supervised model. Evaluation of 3 submitted runs and a baseline run indicated that our temporal features could significantly improve the TQIC precision. However, the error analysis also suggested that for some specific mis-predictions such as "R-F" and "A-R", the submitted runs achieved very little or no improvements. We thoroughly analyzed these errors and summarized the possible causes. Our future work for TQIC will focus on investigating the temporal information in lemmas and named entities. Meanwhile, other methods for preventing the learning algorithms from over-fitting will also be employed.

For TIR, we investigated the learning-to-rank technique for temporal ranking. The experimental results show that: the technique of learning-to-rank can improve the temporal retrieval performance. There are also some limitations of our current work. For example, (1) the training datasets generated using 15 dry-run topics are in fact very small. (2) the editorial relevance is just binary judgement. In the future work, we will investigate the effectiveness of exploring more reasonable features.

References

[1] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 23rd ICML*, pages 129–136, 2007.

[2] A. X. Chang and C. D. Manning. Sutime: A library for recognizing and normalizing time expressions. In *LREC*, pages 3735–3740, 2012.

[3] J. R. Finkel, T. Grenager, and C. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[4] H. Joho, A. Jatowt, and R. Blanco. Ntcir temporalia: a test collection for temporal information access research. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 845–850. International World Wide Web Conferences Steering Committee, 2014.

[5] H. Joho, A. Jatowt, R. Blanco, H. Naka, and S. Yamamoto. Overview of NTCIR-11 temporal information access (Temporalia) task. In *Proceedings of NTCIR-11 Workshop*, 2014.

[6] T.-Y. Liu. *Learning to Rank for Information Retrieval.* Springer, 2011.

[7] M. P. Marcus, M. A. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of english: The penn treebank. *Computational linguistics*, 19(2):313–330, 1993.

[8] G. Pass, A. Chowdhury, and C. Torgeson. A picture of search. In *InfoScale*, volume 152, page 1. Citeseer, 2006.

[9] V. Sindhwani and S. S. Keerthi. Large scale semi-supervised linear svms. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 477–484. ACM, 2006.

[10] V. Sindhwani and S. S. Keerthi. Newton methods for fast solution of semi-supervised linear svms. *Large scale kernel machines*, pages 155–174, 2007.

[11] R. Socher, J. Bauer, C. D. Manning, and A. Y. Ng. Parsing with compositional vector grammars. In *In Proceedings of the ACL conference*. Citeseer, 2013.

[12] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.

[13] Q. Wu, C. J. Burges, K. M. Svore, and J. Gao. Adapting boosting for information retrieval measures. *Journal of Information Retrieval*, 13(3):254–270, 2010.