L3S at the NTCIR-12 Temporal Information Access (Temporalia-2) Task

Tom A. Cruise
Leibniz University Hannover
toma@cruise.com

Tom B. Cruise
Leibniz University Hannover
tomb@cruise.com

Tom C. Cruise
Leibniz University Hannover
tomc@cruise.com

ABSTRACT

This paper describes our participation in the NTCIR-12 Temporalia-2 task including Temporal Intent Disambiguation (TID) and Temporally Diversified Retrieval (TDR). In the TID subtask we extract linguistic features from the query, time distance features and language modeling of the query n-grams which are then combined using a rule based voting method to estimate a probability distribution over the temporal intents. In the TDR subtask we perform temporal ranking based on two approaches, linear combination of textual and temporal relevance method and learning to rank method. Three classes of features comprising of linguistic, topical and temporal features were used to estimate document relevance in the learning to rank approach.

Team Name

L3S

Subtasks

Temporal Intent Disambiguation Task (English), Temporally Diversified Retrieval Task (English)

Keywords

temporal information retrieval, learning to rank, temporal intent disambiguation, text classification

1. INTRODUCTION

The L3S team participated in the Temporal Intent Disambiguation (TID) and Temporally Diversified Retrieval (TDR) subtask of the NTCIR-12 Temporalia-2 Task [4]. This minority report describes our approach to solving the CDT problem and discusses the official results.

2. TEMPORAL INTENT DISAMBIGUATION

This subtask was related to estimating the probability distribution across four temporal intent classes: past, recency, future and atemporal for a given query [4]. In this section we first describe the features extracted from the queries and then the approach of how the rule based voting method was used to build the required distribution across the intent classes.

2.1 Features extracted for TID

- **Time Distance Features**: A temporal mention in a query is an ideal feature to indicate the distance between the intended time in the query and the query submission date. For example, "French Open 2012" becomes a strong indicator of past intent given the query submission date is "May 1, 2013 GMT+0". We use the SUTime Library [2] available part of the Stanford CoreNLP pipeline to recognize and normalize temporal expressions in the queries. A particular date "December 2015" is normalized to "2015-12", a less explicit date such as "Winter 2013" is represented as a season "2013-WI". The distance of the normalized time expressions from the query hitting time is measured to provide an estimate of the intent. It can also identify relative time expressions, such as, "two weeks in the future" which is normalized to "offset P2W". It also recognizes simple temporal words such as "now" and "the future" indicating it as present or future reference respectively, but this leads to false indicators like in the case of "the power of now" or "future shop".
In the dry run queries only 16 out of 100 contained time expressions which could be used to estimate the time distance, this reflects how rare it is to find explicit or implicit temporal expressions in a search query. Thus in order to obtain more candidate temporal expressions for the formal and dry run queries, we used the freely accessible GTE[1] web service detailed in [1]. Given a query, the GTE web service returns a set of candidate years extracted from the top k web snippets returned by the Bing Search Api.

- **Linguistic Features**: Verb tense is a strong temporal indicator for search queries. For example, the verb "was" in "When was the first Olympics held" is a strong indicator of "past" intent. We use the Stanford Part-Of-Speech Tagger (POS Tagger) library [9] that recognizes verbs along with the tenses in a search query using the Penn Treebank tag set [7]. Verb tenses included in the Penn Treebank set include past tense (VBD), past participle (VBN), present tense (VBP/VBZ), present participle (VBG) and base verb form (VB). Since the Penn Treebank set doesn't include any tags for future tense, we consider a verb (VB) that is preceded by a modal verb (MD) such as will/shall to be an indicator of future tense. A query can include multiple verbs with different tenses. Thus we do a syntactic parsing of the query using the Stanford Parser Library [8] to determine the main predicate by selecting the uppermost verb in the parse tree.

- **NGram Features**: As a set of baseline features, we

[1]http://www.ccc.ipt.pt/ ricardo/software.html

extract the unigram and bigram terms of the queries from the training data, which is 73 dry run queries. We model the per class multinomial distribution of the *ngrams* by using the ngrams overall frequency (*ngTotCount*) and the per class ngram count (*ngClassCount*). The per class ngram count is computed by counting the ngrams per class (like past) generated from those queries which had a past probability > 0.0 in the training data.

$$p(ng, class) = \frac{ngClassCount}{ngTotCount} \quad (1)$$

Each formal run query (Q) is representated as a set of all possible permutations of the unigram and bigram terms that are extracted from the query. Then the probability distribution of a query across the temporal classes is calculated using the following equation.

$$p(Q, class) = \arg\max_{q \in Q}(\prod_{ng \in q} p(ng, class)) \quad (2)$$

*** Rule Based Voting Method to be described ****

2.2 Experiments

2.2.1 Temporal Intent Disambiguation Runs

2.2.2 Results and Discussion

3. TEMPORALLY DIVERSIFIED RETRIEVAL

This subtask requires a set of relevant documents to be retrieved for each temporal intent class (past, present, future and atemporal) plus a diversified set which is diverse across the above temporal classes for a given topic. Documents provide temporal information in two forms, publication date and temporal expressions in the content. In this section we first describe the preliminaries required to compute the temporal relevance score of a document using the temporal expressions in the content. We then shortly describe how this temporal relevance score is combined with the topical relevance score in a parameterized sum. Then we focus on the features extracted from the document that will be used in learning-to-rank approaches to build different ranking models for each temporal intent. Finally we describe the approach for producing the diversified set of documents across all temporal intents that uses the set of documents considered relevant for each temporal intent as candidates.

3.1 Subtopic Classification

For each search topic, the workload also contains subtopics which are indicative search questions for each of the temporal classes. As we can't use the class information provided for each subtopic, we used a multiclass SVM classifier[2] to classify subtopics using the features extracted from it.

- We extract the tense of the subtopic similar to the approach used for the TID subtask.
- We identified certain words from the dry run queries which were very common for certain temporal intents and built a word dictionary. We also included synonyms of the above identified words and the classes as well.
  - Future: future, forecast, will, would, should, shall, next, expected, soon, projected, possibility, scheduled
  - Past: past, history, were, origin, did, been, previous, earlier, former, historical
  - Recent: recent, present, current, latest, recently, trendy, now, today
- We compute the average expected distance (3.2) for the subtopic from the top 20 pseudo relevant documents that were retrieved.

[2]https://www.cs.cornell.edu/people/tj/svm_light/

The multiclass SVM classifier sometimes classifies two subtopics belonging to the same topic to be of the same temporal intent, which is not ideal. So we use the confidence score returned from the SVM classifier to carry out a joint classification where each subtopic will have a unique intent. We take all possible combinations of unique (subtopic, intent) pairs and select the combination that gives the maximum confidence.

3.2 Temporal Relevance Score of Document

Given a document and a topic, we have to determine the relevance of the document to each of the temporal intents along with the topical relevance. Each document in the corpus was annotated with normalized time expressions ($TE(d)$) [5]. We then map each time expression te in the document d to a time interval $[b, e]$ at day granularity(e.g., May 2014 is mapped to [01/05/2014, 31/05/2014]). We find the temporal distribution of time references in a document at month granularity, thus each document d is represented by a set $TE_m(d)$ of monthly time intervals $[t_{mb}, t_{me}]$. The weight of each monthly time interval is calculated as follows

$$w([t_{mb}, t_{me}]) = \sum_{te \in TE(d)} \begin{cases} \frac{te_{count}}{|TE(d)|}, & te \in [t_{mb}, t_{me}] \\ \frac{te_{count}}{|TE(d)|} * \frac{1}{12}, & te \in \{t_{year}\} \& [t_{mb}, t_{me}] \in te \end{cases} \quad (3)$$

where te_{count} is the count of the number of occurences of a te in the document. The constant t_{year} is used to describe if a time interval is at a year granularity. For intent specific filters (recency, past, future) we use an exponential distribution to model it as follows

$$f(te) = \begin{cases} \lambda * e^{-\lambda * |T_q - te|}, & \lambda \in [0, 1],\ recency \\ \lambda * e^{\lambda * |T_q - te|}, & \lambda \in [0, 1],\ past\ or\ future \end{cases} \quad (4)$$

where T_q is the query hitting time and λ is a tunable parameter which we set to 0.03 in our experiments. The distance between the query hitting time and time expression is measured in months. We use these intent specific filters to transform the temporal distribution of time references in a document. Then the expected distance of the document with respect to the query hitting time, i.e., temporal relevance score is computed as follows

$$E(d) = \frac{1}{|TE_m(d)|} \sum_{te \in TE_m(d)} \begin{cases} w(te) * f(te) * |T_q - te|, & past\ \&\ future \\ w(te) * f(te) * \frac{1}{|T_q - te|}, & recency \end{cases} \quad (5)$$

3.3 Parameterized Sum Method

For all our experiments, we determine a set R of pseudo-relevant documents ($|R|$=1000) by employing a unigram language model with Dirichlet smoothing [12] (with $\mu = 2000$). We then re-rank the documents using scores obtained from

the linear combination of the temporal relevance and topical relevance score, defined as follows

$$R = \lambda E(d) + (1 - \lambda)R_c, \quad 0 \leq \lambda \leq 1 \tag{6}$$

where λ can be used to determine if we want to give more weightage to temporal relevance or topical relevance score.

3.4 Learning-To-Rank Features

In learning-to-rank approaches a ranking model is built by training a set of query-document pairs using a learning algorithm. The learned model is a weighted coefficient w of a feature vector x, which can then be used rank unseen query-document pairs. For a detailed description of the different approaches, refer to [6]. Feature selection is critical for learning-to-rank approaches, so in this section we describe the various features that we extract from the query-document pairs.

- **Verb Tense Features**: We take the noun terms of the search query into consideration and split the document into two sentence types: se_{noun} those sentences that contain atleast a noun search term and $se_{non-noun}$ those that don't contain any noun search term. We determine if a sentence talks about the past, present or future by using the same approach as the linguistic feature for TID subtask. We thus get 6 verb tense features: the ratio of past, present and future tense w.r.t se_{noun}, and ratio of past, present and future tense w.r.t $se_{non-noun}$. These features help determine the language of the document, how much of the text talks about the past, present or future which in turn helps match it to a particular temporal intent.

- **Topical Features**: These features include similarity features based on the jaccard similarity measure. We compute the similarity between search topic and document title, search topic and document content, search subtopic and document title, search subtopic and document content. These features help determine the topical similarity between the document and search topic and subtopic. We also use the topical relevance score between the search query and the document as a feature. The relevance score obtained from the unigram language model with Dirichlet smoothing is directly used.

- **Temporal Features**: This feature directly uses the temporal relevance score computed for a document as described in 3.2. It helps determine documents that are relevant to past, recent and future temporal intents. We also include temporal density feature which is the ratio of the number of temporal expressions to the length of the document. This helps differentiate between atemporal and temporal documents.

3.5 Earth Mover's Distance for Diversification

The earth mover's distance (EMD) is a measure of distance between two probability distributions, it is the minimum cost required to transform one probability distribution to the other. In our case we would measure the EMD between the temporal distribution of time references from one document to another. We consider sets R_i containing candidate documents from the top 100 documents retrieved for the specific intent ($i \in$ {atemporal, recency, past, future}) using the above ranking approaches. We represent the diversified set of results as DR, to which we add documents from sets R_i which have maximum EMD from the documents already present in DR. During the initial step we add $rank$ 1 document from one of the sets R_i at random, then we compute the EMD between two temporal distributions A and B as follows

$$\begin{aligned} EMD_0 &= 0.0 \\ EMD_{i+1} &= (w_A(te_i) + EMD_i) + w_B(te_i) \\ TotalEMD &= \sum_{i=1}^{|TE_m(A) \cup TE_m(B)|} EMD_i \end{aligned} \tag{7}$$

3.6 Experimental Setup

We used Lucene[3] to build the index for the "LivingKnowledge news and blogs annotated subcollection" corpus [4]. We use the unigram language model with Dirichlet smoothing implementation of lucene to retrieve the top 1000 pseudo-relevant documents. The query is constructed from the title of the topic and the subtopic, and then searched against the title and content fields of the documents. The features in 3.4 are extracted from the tagged version of a pseudo-relevant document.

The 10 dry run topics and the 50 formal run topics of last year along with their *qrels*[4] are used to generate the training data for learning the ranking models. Each row in the training data is a query-document pair: the first column is the relevance judgement, the second is query id (qid) used to restrict the generation of constraints, the subsequent columns are feature/value pairs ordered by increasing feature number. We created separate training data for each temporal class (*past, recency, future* and *atemporal*), the data is prepared as follows: for each subtopic of a given topic we retrieve using the query (containing topic and subtopic), the top-1000 pseudo relevant documents using a language model (LM). The relevance judgements in the *qrels* are of the order 2 (*really relevant*), 1 (*relevant*) and 0 (*irrelevant*). From the pseudo relevant documents, we select relevant and irrelevant documents in the ratio 1:2 for preparing the training data for a particular temporal class. Each temporal class-specific training data is then 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 experimented by combining all the class-specific training data into one large training set, but the performance of the results were lower than the class-specific training data. We use RankLib[5] library to compare different *listwise* learning to rank approaches such as AdaRank [11], RankBoost [3] and LambdaMART [10]. We use the default learning algorithm specific parameters and optimize for the measure $NDCG$@20. For the final runs we use the AdaRank learning algorithm.

3.6.1 Temporal Diversified Retrieval Runs

We submitted 3 runs for the TDR subtask. For all the runs the diversification of the results across all temporal intents is carried out using the earth mover's distance measure.

[3]https://lucene.apache.org/core/

[4]http://research.nii.ac.jp/ntcir/permission/ntcir-11/perm-en-Temporalia.html

[5]https://sourceforge.net/p/lemur/wiki/RankLib/

- *L3S-TDR-E-1*: Manual run with manually crafted queries using the topic and subtopic. The training and test data (formal runs) is generated from the pseudo-relevant documents retrieved using LM. The ranking model is learnt based on the class-specific datasets.

- *L3S-TDR-E-2*: Automatic run in which the subtopics are classified using the joint classifier described in 3.1. The pseudo relevant documents are retrieved using LM and then reranked using the parameterized sum method (3.3). The parameter λ is set to 0.3, giving more weightage to the textual relevance score.

- *L3S-TDR-E-3*: Automatic run in which the subtopics are classified using the joint classifier described in 3.1. The training and test data (formal runs) is generated from the pseudo-relevant documents retrieved using LM. The ranking model is learnt based on the class-specific datasets.

3.6.2 Results and Discussion

4. CONCLUSIONS

5. REFERENCES

[1] R. Campos, G. Dias, A. Jorge, and C. Nunes. Gte: A distributional second-order co-occurrence approach to improve the identification of top relevant dates in web snippets. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM 2012)*, pages 2035–2039, 2012.

[2] A. X. Chang and C. D. Manning. Sutime: A library for recognizing and normalizing time expressions. In *Proceedings of 8th International Conference on Language Resources and Evaluation*, 2012.

[3] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969, 2003.

[4] H. Joho, A. Jatowt, R. Blanco, H. Yu, and S. Yamamoto. Overview of the NTCIR-12 Temporal Information Access (Temporalia-2) Task. In *Proceedings of NTCIR-12*, 2016.

[5] 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)*, pages 845–849, 2014.

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

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

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

[9] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of HLT-NAACL*, pages 252–259, 2003.

[10] Q. Wu, C. Burges, K. Svore, and J. Gao. Adapting boosting for information retrieval measures. *Journal of Information Retrieval*, 2007.

[11] J. Xu and H. Li. AdaRank: a boosting algorithm for information retrieval. In *Proceedings of SIGIR*, pages 391–398, 2007.

[12] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to Ad Hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '01)*, pages 334–342, 2001.