IRISM @ NTCIR-12 Temporalia Task: Experiments with MaxEnt, Naive Bayes and Decision Tree classifiers

Jitendra Kumar
Dept. of Computer Science and Engineering,
Indian School of Mines,
Dhanbad, India
jitendrakumar@cse.ism.ac.in

Sudha Shanker Prasad
Dept. of Computer Science and Engineering,
Indian School of Mines,
Dhanbad, India
sudha.shanker@cse.ism.ac.in

Sukomal Pal*
Dept. of Computer Science and Engineering,
Indian Institute of Technology (BHU), Varanasi, India
spal.cse@iitbhu.ac.in

ABSTRACT

This paper describes our participation in Temporal Intent Disambiguation (TID), which is a subtask of the pilot task of NTCIR'12 Temporal Information Access (Temporalia-2) task [6]. We considered the task as a slight variation of supervised machine learning classification problem. Our strategy involves building models on different standard classifiers based on probabilistic and entropy models from MALLET, a Natural Language Processing tool. We focus on the feature engineering to predict the probability distribution of given temporal classes for search queries. We submitted three runs based on MaxEnt, Naive Bayes and C4.5 Decision Tree classifiers. Out of them, Decision Tree based runs exhibited our best performance while the other two were average.

Team Name

IRISM

Subtasks

Temporal Intent Disambiguation (English)

Keywords

Temporal IR, Information Retrieval, Natural Language Processing, Machine Learning

1. INTRODUCTION

Temporal Information Retrieval serves a crucial role in improving the effectiveness of information retrieval methods by better exploiting temporal information in user queries [5]. In recent times, substantial number of user queries on the web are having time-sensitive information needs. The identification of temporal information need and its presentation are very demanding problem. Research in this area usually focuses to meet the increasing demand of processing user generated web queries more effectively over temporal information.

The task attempts to foster research in temporal information extraction [16] [17] [18] which is pivotal for text summarization and other natural language processing applications. The Temporal Information Access (Temporalia-2) task [6] is the second such task, which is organized to promote the research in analysing temporal information in user web queries. It is hosted by the 12th NTCIR workshop on Evaluation of Information Access Technologies (NTCIR-12)[1].

In this paper, we present our participation to the Temporal Intent Disambiguation (TID) subtask. Given a query string, here a system is required to determine the probabilistic distribution of four given temporal classes, i.e. *Past, Recency, Future, Atemporal*.

Below are the examples of queries of training data from four different temporal classes:

- **Atemporal:** *How to lose weight*
- **Recency:** *Canadian dollar exchange rate*
- **Future:** *When to File 2014 Taxes*
- **Past:** *Beer Night 1974*

Rest of the paper is organized as follows. Section 2 briefly covers some of the works already done in this area. In Section 3, we describe the proposed system. Section 4 details the methodology and implementation. Section 5 reports our results. In Section 6 we discuss the results obtained along with limitations of our work and possible future extension. Finally we conclude in Section 7.

2. RELATED WORK

Good amount of work has been done in the area of Temporal Information Retrieval. One of the earliest is by Bruce [1972] [2] who presented a formal model for temporal references. Another formal approach is given by Allen[1983] [1], which describes a set of 13 possible temporal relationships between any two time intervals. More discussion about can be found in Kanhabua's Temporal Information Retrieval book chapter "Temporal Query Analysis" [9]. An exhaustive survey on Temporal Information Retrieval and Related Applications is done by Campos et al. [3].

Other works focus more on time-sensitive queries. Metzler et al. [8] developed the automatic detection of implicitly year qualified queries by analyzing query logs. Kanhabua and Nørvåg [10] proposed three different methods to determine the time of implicit temporal queries: (1) dating queries using only query keywords, (2) dating queries using the retrieved top-k documents, and (3) dating queries using the timestamp of the retrieved top-k documents. Work done by Jones and Diaz[2007] [7] includes temporal classification of queries into three classes *atemporal*, *temporally unambiguous* and *temporally ambiguous*.

*During this submission the author relocated

[1] http://research.nii.ac.jp/ntcir/ntcir-12/

For explicit temporal intent, Shokouhi and Radinsky[2012] [14] proposed a time-sensitive approach for query auto completion by applying time series analysis. Cheng et al.[2013] [4] presented a language model that incorporates the timeliness factor to retrieve fresh recent results for nonspike timely queries.

3. SYSTEM DESCRIPTION

We describe here the details of our proposed scheme.

3.1 Dataset & Tool

The corpus available for training includes 393 search engine queries collected from Temporalia-1 challenge in which each one is a text query along with the Query Issue Time and there temporal annotation i.e atemporal, recency, past, future. Along with it, training corpus also contains Dry Run Queries that are later released by the task organizers with probability distribution of their temporal classes. The official test set for the Temporalia-2 challenge consisted of 300 unlabelled queries.

For the classification task we have used MALLET[2] in our system. MALLET that stands for "MAchine Learning for LanguagE Toolkit" is an integrated collection of Java-based package useful for natural language processing, classification, information extraction, and other machine learning applications.

3.2 Classifier Used

Maximum Entropy (MaxEnt)

The intuition of the Maxium Entropy (MaxEnt) model is to use a set of user-specified features and learn appropriate weights. The model uses search-based optimization to find weights for the features that maximize the likelihood of the training data.

Assuming only the word level features, we define a joint feature $f(w, c) = N$, for each word w and class cϵC, where, N is frequency of w occurs in a query in class C. With iterative optimization, we assign a weight to each joint feature to maximize the log-likelihood of the training data. The probability of class c given a query q and weights λ is

$$P\left(c|q,\lambda\right) = \frac{exp\sum_i \lambda_i f_i\left(c,q\right)}{\sum_{c'\,\epsilon C} exp\sum_i \lambda_i f_i\left(c',q\right)} \quad (1)$$

These parameters are learned to maximize the entropy of the distribution.

Naive Bayes

Naive Bayes classifier is a simple probabilistic classifier based on applying Baye's theorem with strong independence assumptions between the features.

Given a problem instance to be classified, represented by a vector $x = (x_1, ..., x_n)$ representing some n features (independent variables), it assigns to this instance probabilities $p(C_k|x_1, ..., x_n)$ for each of K possible outcomes or classes. Using Baye's theorem, the conditional probability can be decomposed as:

$$p\left(C_k|x\right) = \frac{p\left(C_k\right)p\left(x|C_k\right)}{p\left(x\right)} \quad (1)$$

[2] `http://mallet.cs.umass.edu`

where,

$p(C_k)$ = probability of class k,
$p(x|C_k)$ = probability of query x given class k,
$p(x)$ = probability of query x.

C4.5 Decision Tree

C4.5 is a classification algorithm based on decision tree approach that uses the information gain ratio evaluated by entropy [13]. The test feature at each node in the tree is selected using information gain ratio. The attribute with the highest information gain ratio is chosen as the test feature for the current node [11]. Let D be a set consisting of $(D_1...D_j)$ data instances. Suppose the class label attribute has m distinct values defining m distinct classes, C_i (for i= 1,..., m). Let D_j be the number of sample of D in class C_i. The expected information needed to classify a given sample is :

$$Splitinfo_A(D) = -\sum\left(\left(\frac{|D_j|}{|D|}\right) * \log\left(\frac{|D_j|}{|D|}\right)\right) \quad (1)$$

$$Gain\ ratio(A) = \frac{Gain(A)}{Splitinfo_A(D)} \quad (2)$$

where,

$$Gain(A) = Info(D) - Info_A(D) \quad (3)$$

$$Info(D) = -\sum P_i * log_2(P_i) \quad (4)$$

and

$$Info_A(D) = -\sum(\frac{|D_j|}{|D|}) * Info(D_j) \quad (5)$$

where,

P_i = probability of distinct class C_i,
D =data set,
A=sub-attribute from attribute,
$(\frac{|D_j|}{|D|})$=act as weight of j^{th} partition.

In other words, Gain(A) is the expected reduction in entropy caused by knowing the value of feature A.

3.3 Feature Engineering

In classification task, for successful result, selecting independent and discriminating features are pivotal with supervised machine learning algorithm. In this section, we provide the details of the features we formulate to accomplish our task.

Temporal Feature

Queries come with two kind of temporal information i.e Query Issue Time and Date Field within query (present in some queries). We compare both the year fields in such queries and compute it's difference.

i.e

$$diff = Q_{it} - Q_{df} \quad (1)$$

where,

Q_{it}=Query issue time
Q_{df}= Date field within query.

Based on the diff value, we assign the probability to the temporal queries as shown in Table 1.

Past	Recency	Future	Atemporal	*diff*
1.000	0.000	0.000	0.000	>0
0.000	1.000	0.000	0.000	= 0
0.000	0.000	1.000	0.000	<0

Table 1: probablistic distribution based on temporal feature

Linguistic Feature

A list of ascendant temporal keywords is maintained [12] for calculating final probabilistic distribution.Table 2 shows snippet from it.

Let P_{past}, $P_{recency}$, P_{future}, $P_{atemporal}$ represents the probability distribution for past, recency, future, atemporal classes respectively truncated upto 3 decimal places. The residual value can be computed as

$$Value_{residue} = 1 - (P_{past} + P_{recency} + P_{future} + P_{atemporal})$$

For each query, the system count the number of temporal keywords for each classes. $Value_{residue}$ is added to the probability distribution of the class having maximum count. In case of any ambiguity, the class having maximum probability score is chosen.

Temporal Class	Set of Words
Future	future, should, forecast, will, would, shall, next
Recent	recent, present, latest, current, live
Past	history, origin, were, was, past, did, start
Atemporal	discipline, sport, system, office

Table 2: Snippet from temporal keywords

4. METHODOLOGY

We treat the task as a variant of 4-class classification problem although we are asked to find the probabilistic distribution for each search query instead of finding a concrete temporal class. We attempt to identify appropriate features and apply them to a few supervised machine learning techniques to obtain membership probabilities for each class. In this section, we will provide the details of our approach and submitted runs. Figure 1 illustrates a flowchart of steps involved in our approach.

4.1 Pre-processing

All the search queries from both the training and test data are pre-processed in the following manner.

1. Tokenization: identify individual terms from the query
2. Case folding: computing its lower-case version
3. Extraction of Query Issue Time: From each query the field is extracted and stored in a separate file along with its query id.
4. Selection of concrete classes for dry run queries from Temporalia-2 task is done by choosing the class having maximum probability distribution.

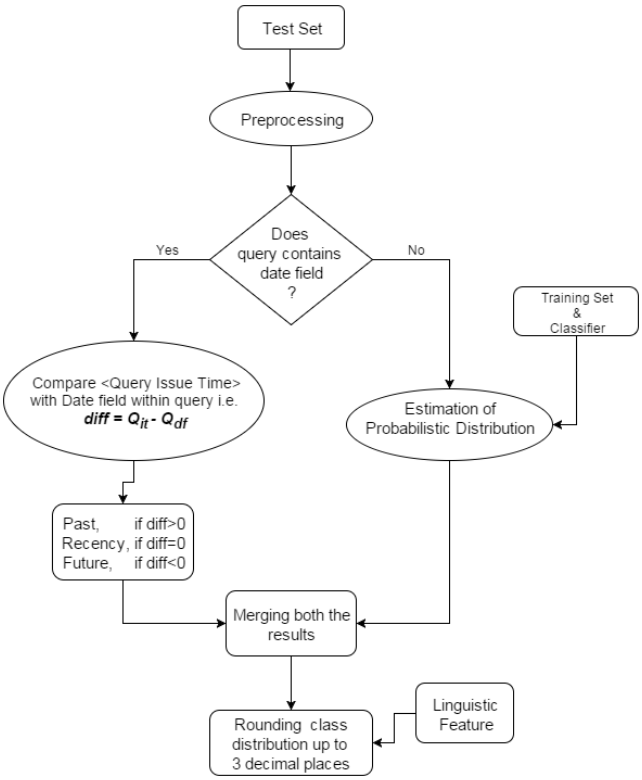

Figure 1: flowchart illustrating our approach

5. Format Conversion: This step deals with conversion into a pre-defined format i.e (<Query-Id> <Temporal Class> <Search Query>) required for training using MALLET.

4.2 Estimation of Probabilistic Distribution

The system uses a feature extractor that extracts both Query Issue Time and Date field within query. In order to find probabilistic distribution of a given query, the system carries out the following steps.

1. Feature extractor is used to extract the temporal feature from the queries and effectiveness is verified on training data.
2. Queries and along with there temporal labels are fed into supervised machine learning algorithm to create the model.
3. During prediction, the same feature extractor along with model from previous step is used to estimate probabilistic distribution of queries from test data.
4. Probabilistic distribution is rounded upto 3 decimal places along with the help of linguistic feature.

Effectiveness of temporal feature is verified on training data.

4.3 TID Submitted Runs

IRISM-TID-E-1

MaxEnt Classifier is used in our proposed methodology to predict the probabilistic distribution for this run. Table 3 shows the training accuracy of this run.

Class	Precision	Recall	F1-Score
Atemporal	0.990	0.943	0.966
Past	0.989	0.929	0.958
Recency	0.939	0.920	0.929
Future	0.861	0.989	0.920

Table 3: Training accuracy by class based on MaxEnt

IRISM-TID-E-2

Naive Bayes Classifier is used in our proposed methodology to predict the probabilistic distribution for this run. Table 4 shows the training accuracy of this run.

Class	Precision	Recall	F1-Score
Atemporal	0.938	0.849	0.891
Past	0.877	0.868	0.873
Recency	0.862	0.880	0.871
Future	0.804	0.886	0.843

Table 4: Training accuracy by class based on Naive Bayes

IRISM-TID-E-3

C4.5 Decision Tree is used in our proposed methodology to predict the probabilistic distribution for this run. Table 5 shows the training accuracy of this run.

Class	Precision	Recall	F1-Score
Atemporal	0.968	0.566	0.714
Past	0.892	0.747	0.813
Recency	0.403	0.348	0.373
Future	0.917	0.678	0.779

Table 5: Training accuracy by class based on C4.5 Decision Tree

5. RESULT

The evaluation of the 3 submitted runs are done with Averaged Per-Class Absolute Loss and Average Cosine Similarity.

As defined in the TID task[3], for a specific query q, let $P = \{p_1, p_2, p_3, p_4\}$ denote its standard temporal class distribution, and W= $\{w_1, w_2, w_3, w_4\}$ denote the temporal class distribution from a participant. The classification loss for a single query will be measured using the following two ways.

Metric-1: Averaged per-class absolute loss, i.e.,

$$\frac{1}{4}\sum_{i=1}^{4}|w_i - p_i|$$

Metric-2: Cosine similarity between the two probability vectors P and W, i.e.,

$$cos\Theta = \frac{P.W}{|P|\,|W|} = \frac{\sum_{i=1}^{4}|p_i * w_i|}{\sqrt{\sum_{i=1}^{4} p_i^2} * \sqrt{\sum_{i=1}^{4} w_i^2}}$$

[3] `http://ntcirtemporalia.github.io/NTCIR-12/taskdescription.html`

Figure 2 & 3 depicts Avg. Cosine Similarity & Averaged per-class absolute loss respectively calculated across 3 runs.

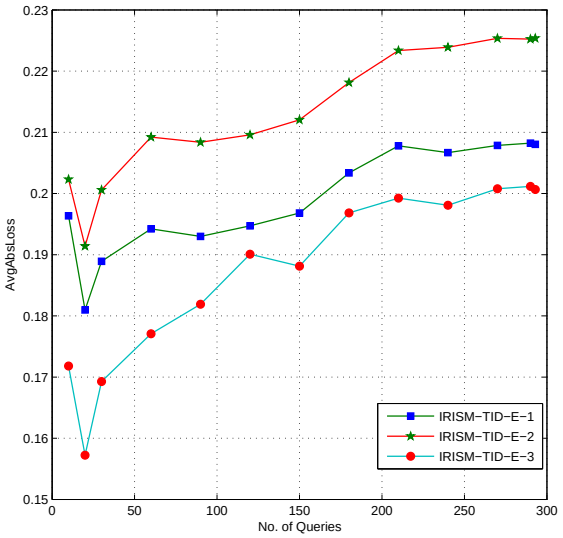

Figure 2: Averaged per-class absolute loss across 3 runs

Table 6 shows the final performance of the 3 submitted runs averaged across all test queries.

Run-ID	AvgAbsLoss	AvgCosin
IRISM-TID-E-1	0.20803852739726025	0.7613244466614848
IRISM-TID-E-2	0.225380993150685	0.6867669932113489
IRISM-TID-E-3	0.20065496575342454	0.7702563581979542

Table 6: Performance Results of TID subtask

We found that *IRISM-TID-E-3* that uses C4.5 Decision Tree algorithm shows best result in terms of Averaged per-class absolute loss & Averaged Cosine Similarity followed by *IRISM-TID-E-1* and *IRISM-TID-E-2*.

6. DISCUSSION

With the experiments and results obtained, we found that Temporal Intent Disambiguation (TID) task is quite challenging. We believe that 393 queries from both (Temporalia-1 & Temporalia-2) tasks that serve as a training data is not enough for building a robust machine learning classifier. Along with this, identification and establishing effective features is not trivial. Although manual verification of the result serves as a baseline for error analysis and further improvement, in this case, manual verification for the obtained result is nearly impossible for human. Hence, our dependency increases on the classifier and features for accuracy of the result. In addition to this, presence of short and highly ambiguous queries in the dataset is one of the major issues that reduces our result accuracy. Table 7 shows a snippet of such queries. All search queries in the formal-run were submitted in "May 1, 2013 GMT+0".

During our manual analysis, we also found that some of the queries in training data requires additional information

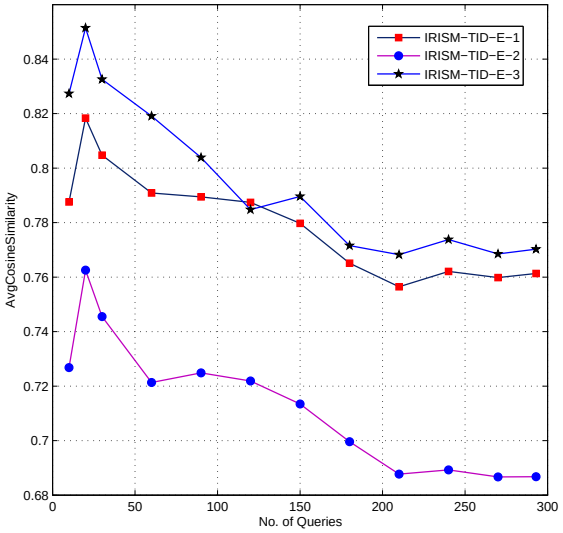

Figure 3: Averaged Cosine similarity across 3 runs

Query String
Bohemian Rhapsody
office2007 office2010
December Calendar
un drugs control programme

Table 7: Queries from dataset

that could help in better training of the system. An example is "father's day 2013 date" which is difficult for system to analyse. Some other have typos. Some of the queries are temporally inappropriate such as "newsday". Due to these, the model may get inconsistent training. In the Test data, some of the queries are easy to estimate distribution (for example "December 17 2010") while some are hard (for example "December Calendar") even if the query is containing the temporal information. The possible reason behind is different way to convey temporal information. Some of the queries are a little bit confusing due to the presence of more temporal information. An example is "office2007 office2010". Query such as "Bohemian Rhapsody" requires some extra information before estimating its probability distribution. Some queries contain words having two possible interpretation such as "un drugs control programme". Here 'un' may stand for United Nation or just the English prefix *un*. Due to the points discussed above, our model sometimes failed to correctly estimate the probability distribution.

7. CONCLUSION

In this paper, we described our participation to Temporal Intent Disambiguation (TID), which is a subtask of the pilot task of NTICIR-12 Temporal Inforamation Access (Temporalia-2) task [6]. This paper shows the performance of different classifiers along with the Temporal and Linguistic features. One of the reasons behind our not-so-well performance is insufficient number of training data. Estimation of probability distribution of temporal classes is more challenging compared to identifying a concrete class as there are very limited resources available with such probability distribution. In addition to this, manual verification of the probability distribution precisely is a difficult task.

The performance can be improved after analyzing the substantial gap in accuracy estimates between training and test data. Possibly choosing more appropriate features and supervised machine learning techniques could further improve our results.

8. REFERENCES

[1] J. F. Allen. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843, 1983.

[2] B. C. Bruce. A model for temporal references and its application in a question answering program. *Artificial intelligence*, 3:1–25, 1972.

[3] R. Campos, G. Dias, A. M. Jorge, and A. Jatowt. Survey of temporal information retrieval and related applications. *ACM Comput. Surv.*, 47(2):15:1–15:41, Aug. 2014.

[4] S. Cheng, A. Arvanitis, and V. Hristidis. How fresh do you want your search results? In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 1271–1280. ACM, 2013.

[5] L. Derczynski, J. Strötgen, R. Campos, and O. Alonso. Time and information retrieval: Introduction to the special issue. *Information Processing and Management*, 51(6):786–790, 2015.

[6] H. Joho, A. Jatowt, R. Blanco, H. Yu, and S. Yamamoto. Overview of NTCIR-12 temporal information access (temporalia-2) task. In *Proceedings of the 12th NTCIR Conference on Evaluation of Information Access Technologies*, 2016.

[7] R. Jones and F. Diaz. Temporal profiles of queries. *ACM Transactions on Information Systems (TOIS)*, 25(3):14, 2007.

[8] R. Jones, D. Metzler, and F. Peng. Identifying and expanding implicitly temporally qualified queries, Apr. 10 2012. US Patent 8,156,111.

[9] N. Kanhabua, R. Blanco, and K. Nørvåg. Temporal information retrieval. *Foundations and Trends in Information Retrieval*, 9(2):91–208, 2015.

[10] N. Kanhabua and K. Nørvåg. Determining time of queries for re-ranking search results. In *Research and Advanced Technology for Digital Libraries*, pages 261–272. Springer, 2010.

[11] S. S. Prasad, J. Kumar, D. K. Prabhakar, and S. Pal. Sentiment classification: An approach for indian language tweets using decision tree. In *Mining Intelligence and Knowledge Exploration*, pages 656–663. Springer, 2015.

[12] A. Shah, D. Shah, and P. Majumder. Andd7@ ntcir-11 temporal information access task. In *NTCIR*, 2014.

[13] S. Sharma, J. Agrawal, and S. Sharma. Classification through machine learning technique: C4. 5 algorithm based on various entropies. *International Journal of Computer Applications*, 82(16), 2013.

[14] M. Shokouhi and K. Radinsky. Time-sensitive query auto-completion. In *Proceedings of the 35th international ACM SIGIR conference on Research and*

development in information retrieval, pages 601–610. ACM, 2012.

[15] A. Styskin, F. Romanenko, F. Vorobyev, and P. Serdyukov. Recency ranking by diversification of result set. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, pages 1949–1952. ACM, 2011.

[16] N. UzZaman, H. Llorens, J. Allen, L. Derczynski, M. Verhagen, and J. Pustejovsky. Tempeval-3: Evaluating events, time expressions, and temporal relations. *arXiv preprint arXiv:1206.5333*, 2012.

[17] M. Verhagen, R. Gaizauskas, F. Schilder, M. Hepple, G. Katz, and J. Pustejovsky. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 75–80. Association for Computational Linguistics, 2007.

[18] M. Verhagen, R. Sauri, T. Caselli, and J. Pustejovsky. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 57–62. Association for Computational Linguistics, 2010.