HULTECH at the NTCIR-11 Temporalia Task: Ensemble Learning for Temporal Query Intent Classification

Mohammed Hasanuzzaman
Normandie University
GREYC UMR 6072
Caen, France
first.last@unicaen.fr

Gaël Dias
Normandie University
GREYC UMR 6072
Caen, France
first.last@unicaen.fr

Stéphane Ferrari
Normandie University
GREYC UMR 6072
Caen, France
first.last@unicaen.fr

ABSTRACT

This paper describes the HULTECH system of the NTCIR-11 Temporal Query Intent Classification (TQIC) subtask. Given a query string, the task is to assign one of four temporal classes i.e. Past, Recency, Future or Atemporal. In particular, we experimented an ensemble learning paradigm, which underlying idea is to reduce bias by combining multiple classifiers instead of a single one. We considered 11 types of features from three different information sources (TempoWordNet, Web snippets results, the query itself seen as a sentence) and used a subset of them for our submitted runs. Our system reaches average results but outperforms other participants for the temporal class Recency in terms of accuracy. These initial results open interesting issues for future works.

Team Name

HULTECH (Human Language Technology Team - GREYC UMR 6072)

Subtasks

Temporal Query Intent Classification (TQIC) subtask

Keywords

Ensemble Learning, TempoWordNet, Temporal Query Intent Classification

1. INTRODUCTION

Web is a dynamic information source in which the number and content of pages change continuously over time. Web search queries are dynamic in nature and temporally sensitive. Temporally sensitive implies that the intent of a given user for information changes over time. Many queries may only be answered accurately if their underlying temporal orientations are correctly judged. So, recognizing the temporal intent behind users' queries is a crucial part towards improving the performance of information access system and also diversifying the retrieved results from a search engine. For instance, a direct application would be to limit the returned search results pages from a search engine based on the publishing time belonging to the intent of the users.

Although, there has recently been an increased attention in investigating temporal characteristics of queries, very few works have been persued to understand users' queries temporal intents. The Temporal Information Access [5] is the first such challenge, which is organized to provide a common platform for designing and analyzing time-aware information access systems. It is hosted by the 11th NTCIR Workshop on Evaluation of Information Access Technologies (NTCIR-11)[1]. In particular, Temporalia involves two subtasks to address temporal information access technologies:

1. *Temporal Query Intent Classification subtask*,
2. *Temporal Information Retrieval subtask*.

In this paper, we present our participation to the Temporal Query Intent Classification (TQIC) subtask. In this subtask, given a query string, systems are required to classify it into one of four temporal categories, namely Past, Recency, Future and Atemporal. Given below are examples of queries from different temporal classes:

1. **Past:** *Who was eliminated on dancing with the stars.*
2. **Recency:** *Did the Pirates Win Today.*
3. **Future:** *10 Days Weather Forecast.*
4. **Atemporal:** *The Differences Between Republicans Democrats.*

Organizers released one hundred (100) queries along with their respective temporal class and issuing time as training data. Three hundred (300) queries along with their issuing time were released as test data. Participants were allowed to submit upto three runs for the NTCIR-11 TQIC subtask. Performance of the submitted systems/runs were measured by the number of queries with correct temporal classes divided by the total number of queries.

We implemented two systems (e.g. Run 1 and Run 2) based on several popular supervised machine learning algorithms. In particular, we combined the predictions of different algorithms under the ensemble learning paradigm. Ensemble learning employs multiple individual classifiers and combines their predictions to achieve better performance than a single classifier. It has been shown that the combination of multiple classifiers can be more effective compared to any individual ones [3]. Considering that different base classifier give different contributions to the final classification result, we assigned greater weights to the classifiers with better performance and adopted the weighted voting approach for our submitted runs. Apart from this, ensemble

[1]http://research.nii.ac.jp/ntcir/ntcir-11/

learning based systems can be, perhaps surprisingly, useful when dealing with huge amount of data. It can also be very useful when adequate data is not available. As for example, in our case only eighty (80) dry run queries are available for training/tuning and another twenty (20) for testing the performance of our systems. Therefore, we experimented ensemble learning with a very simple combining rule to obtain final results.

In the remainder of this paper, Section 2 describes the data acquisition process to compute the features used for classification. Section 3 briefly presents the adopted framework for our different experiments. Section 4 describes the submission runs and presents some results. Finally, we draw some conclusions in Section 5.

2. DATA ACQUISITION

In this section, we present the data used for our experiment. We also present the processing done in order to compute the eleven (11) independent features which helped us to learn different models to classify the queries.

2.1 Web Snippets Collection

We first collected the set of web snippets results returned by a commercial search engine API [2] for the hundred (100) dry run queries as well as for the three hundred (300) formal run queries provided by the task organizer.

For each query, we considered the top ten (10) web snippets returned by the search engine. We removed unwanted symbols and characters from the collected web snippets.

Some examples of the collected web snippet are presented in Table 1.

2.2 Associated Year Dates Collection

For the hundred (100) dry run queries and the three hundred (300) formal run queries, we also collected their most relevant associated year date along with their confidence values by using the freely accessible web service GTE [3] proposed by [1].

Given a query, GTE returns a similarity value calculated between the query and all the candidate year dates together with the corresponding contents (i.e. title, web snippet and url) where the set of candidate dates appear. Table 2 gives some examples of extracted dates and confidence values for some given queries.

Query	Date, Confidence Value
michael douglas cancer	2010, 0.8079
price of samsung galaxy note	2013, 0.8943
bruins game tonight live	2013, 0.8510
american eagle	1977, 0.8952

Table 2: Examples of extracted dates and confidence values for query # 2, 4, and 19 of the formal run.

2.3 Features

The most important step in building a classifier is deciding what features of the input instances are relevant and how to represent them. Therefore, choosing discriminating and independent features are keys to any machine learning algorithm being successful in classification.

We identified eleven (11) independent features and used them in the learning process of our selected classifiers. These features are computed from the information extracted from three different sources and resources. All the considered features are listed in Table 3.

Features	Description
D_b_Dates	Difference between dates.
C_o_Date	Confidence on date.
N_o_PaW	Number of Past words present in the query.
N_o_RW	Number of Recency words present in the query.
N_o_FW	Number of Future words present in the query.
N_o_PaS	Number of snippet classified as Past.
N_o_RS	Number of snippet classified as Recency.
N_o_FS	Number of snippet classified as Future.
N_o_AS	Number of snippet classified as Atemporal.
C_o_Q	Class of the query itself.
Q_S	Text of the query (unigrams).

Table 3: Overall features considered for temporal query intent classification.

1. **D_b_Dates (Feature 1):** It is calculated as the difference between the date explicitly mentioned inside a query string and the issue date of that query (given by the organisers). If there is no mention of a date inside a query string, we consider the date obtained from the GTE web service for calculation of this attribute.

2. **C_o_Date (Feature 2):** For this feature, a weight between 0 and 1 is assigned to the date associated to a particular query. It is set to 1 when there is explicit mention of a date inside the query string otherwise it is set to the returned confidence value by GTE web service. It is set to NULL when there is no date related to that particular query.

3. **N_o_PaW, N_o_RW, and N_o_FW (Feature 3-5):** For a given query string, these features represent the number words belonging to Past, Recency and Future categories respectively. These numbers are extracted from an external knowledge base called TempoWordNet [2]. TempoWordNet is a lexical knowledge base where each synset of WordNet has been assigned its intrinsic temporal value. Words present in a query string are looked up directly into TempoWordNet (without word sense disambiguation) and counted for each temporal class i.e Past, Recency and Future. Finally, these numbers are normalized based on the total number of words present on that particular query.

4. **N_o_PaS, N_o_RS, N_o_FS, and N_o_AS (Feature 6-9):** These features are the number of web snippets classified as Past, Recency, Future and Atemporal respectively for a query. In order to compute these features, we used a two steps classification process. First, collected web snippet results are classified as Atemporal or Temporal using the temporal text classifier proposed in [2]. In particular, we used the classification model proposed to classify WordNet as *atemporal* or *temporal* in TempoWordNet. Afterwards, the temporal set of the web snippets is classified using a 3

[2]In our experiment we used Bing Search API, however any other search API could be used.

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

Queries	Urls and Web Snippets
michael douglas cancer	http://www.cnn.com/2013/10/14/health/michael-douglas-tongue-cancer/index.htm. Michael Douglas never had throat cancer, as he told the press in 2010. He actually had tongue cancer http://www.people.com/people/article/0,,20745023,00.html The world was shocked when Michael Douglas announced he had stage four throat cancer in August 2010, but the Oscar winner now reveals that he
price of samsung galaxy note	http://mobiles.pricedekho.com/mobiles/samsung/samsung-galaxy-note-price-p4s8R.html Connectivity Offered. Samsung Galaxy Note offers a many connectivity options to the user, thus enabling them to get connected with other networks and devices within a ... http://www.mysmartprice.com/mobile/samsung-galaxy-note-msp1479 The best price of Samsung Galaxy Note in India is Rs. . The price has been sourced from 9 online stores in India as on 2014 15th June. The same price may be used to ...
i am a gummy bear	http://www.youtube.com/watch?v=astISOttCQ0 From the CD I Am Your Gummy Bear. Also from the DVD I Am A Gummy Bear Available on Amazon at: http://tinyurl.com/gummybeardvd ... http://www.youtube.com/watch?v=Z47EUaIFrdQ My version of a log Gummybear video. Im no pro at mixing music, hope you like.Ive just finished making Mix 2 in which Ive removed the POP and fixed the ....
madden 2014 release date	http://www.nflschedule2014.org/madden-release-date.htmll Madden 2015 Release Date The Madden NFL 15 release date is currently slated for August 30th 2014. Prelaunch information and developer leaks regarding Madden 15 will ... http://www.ign.com/articles/2014/04/28/madden-nfl-15-release-date-revealed EA announced today that Madden NFL 15 is slated for release August 26 in North America and August 29 in Europe. The release date was revealed in the first ...

Table 1: Examples of urls and web snippets for given queries.

class sentence temporal classifier (Past, Recency and Future). For that purpose, we used the semantic vector space representation of web snippets where each web snippet is augmented with the synonyms of any temporal word contained in it as shown in [2].

5. **C_o_Q (Feature 10):** This attribute-value is one of the 4 predefined temporal classes. It is computed using a semi-supervised learner, which temporally tags any text substring. Each query is represented as a bag of words and time tagged as Atemporal or Temporal using the model proposed in [2]. To fine tune temporal queries, a three class temporal classifier is used to tag queries as Past, Recency and Future as shown in [2].

6. **Q_S (Feature 11):** Each query string is represented as a bag of unigrams where the presence of a word is associated to the value 1 and 0 when it is not present.

3. METHODOLOGY

Combining classifiers is already a tested and proven research. It is familiar under different names in the literature: committees of learners, mixtures of experts, classifier ensembles, multiple classifier systems, consensus theory to name but a few.

Ensemble learning is the technique by which diverse models, such as classifiers or experts, are tactically built and combined to find a solution for a particular computational intelligence problem. The intuition behind classifier ensemble is that if each classifier makes different errors, then a strategic combination of these classifiers can reduce the total error.

Ensemble learning is largely used to augment the capabilities (e.g. classification, prediction, function approximation) of a model, or cut down the likelihood of a fateful selection of a poor one. Apart from these, the application of ensemble learning includes assigning a confidence to the decision made by the model, selecting optimal (or near optimal) features, data fusion, incremental learning, non-stationary learning and error-correcting.

Several sophisticated algorithms for combining classifiers are already available in which they have their own built in combination rules. For example, simple majority voting for bagging, weighted majority voting for AdaBoost, a separate classifier for stacking. In [6], authors mention three types of base model outputs to be used for classifier combination: (1) Abstract level, where each classifier provides a unique label for individual input pattern, (2) Rank level, where every single classifier provides a list of ranked class labels for each input pattern and (3) Measurement level, where each classifier outputs a vector of continuous valued measures.

Voting based combination methods are the most simple and popular combination rules for classifier ensemble. Voting based methods mainly functions on class labels where, $d_{t,j}$ is 1 or 0 depending on whether classifier t chooses j, or not, respectively. The ensemble then chooses class J that receives the largest total vote based on a given strategy:

Majority Voting.

$Class(\mathbf{X}) = argmax_{j=1,..,C} \sum_{t=1}^{T} d_{t,j}$

Weighted Voting.

$Class(\mathbf{X}) = argmax_{j=1,..,C} \sum_{t=1}^{T} w_t d_{t,j}$

Classifiers	SMO	NB	Multi.LP	Lazy.LWL	Logit.B	DT	NB.Tree	Ran.F
Run 1								
Precision for Past	69.6	60.0	73.1	56.4	66.7	90.0	48.7	63.6
Precision for Recency	47.8	40.0	54.5	45.8	45.5	29.4	43.5	45.0
Precision for Future	58.3	61.9	56.0	55.0	50.0	50.0	65.2	55.6
Precision for Atemporal	43.3	42.1	59.3	70.6	60.9	26.5	33.3	48.4
Overall Accuracy	55.0	50.0	61.0	56.0	56.0	39.0	49.0	53.0
Run 2								
Precision for Past	47.2	42.9	55.6	57.9	68.0	48.6	-	55.0
Precision for Recency	44.4	31.9	48.3	53.3	51.9	50.0	-	42.9
Precision for Future	75.0	68.8	53.8	81.3	63.0	73.9	-	60.0
Precision for Atemporal	29.4	44.4	72.2	75.0	66.7	37.5	-	32.4
Overall Accuracy	49.0	42.0	56.0	63.0	62.0	54.0	-	46.0

Table 4: 10-fold cross validation results of Run 1 and Run 2.

For our experiments, we adopted weighted voting as in [4]. In weighted voting, weights of votes of classifiers variy from classifier to classifier. Therefore, it is very important to select appropriate weights of votes for all the participant classifiers in an ensemble method. For that purpose, we adopted a very simple approach where the votes of a particular classifier are weighted by its confidence of prediction (hereby its accuracy). Higher weights of votes are assigned for a particular classifier which performs relatively well.

Most of the times when training data is too small, resampling techniques like bootstrapping or bagging are used to learn different classifiers using different samples of the data in order to achieve classifier diversity. However, we have not followed such methods to resample our training data. Instead, we used the same training data to learn our models. After training, no parameter tuning for individual classifiers was performed as it is not required by weighted voting.

We used eight different classifiers, namely Support Vector Machines (SMO), Naïve Bayes (NB), Multilayer Perceptron (Multi.LP), Locally Weighted Learning (Lazy.LWL), LogitBoost (Logit.B), Decision Tables (DT), Hybrid Learning (NB.Tree) and Random Forests (Ran.F) to build classification models depending upon the various representations of the features listed in Table 3. This technique is a very general one and its performance may further be improved depending upon the choice and/or the number of classifiers as well as the use of more complex features.

4. SUBMISSION RUNS AND RESULTS

The experiments have been conducted using the Weka platform[4] with default parameters.

4.1 Run 1

A supervised learning strategy has been used to learn Past, Recency, Future and Atemporal classes with eight 8 different classifiers over the training data set released by the task organizer. All 11 independent features extracted from different sources have been used in this learning process and the results of the 10-fold cross validation process are presented in Table 4 based on the dry run.

For the formal run, these 8 learned models have been applied to tag the 300 test queries. For that purpose, we combined the predictions of the different classifiers using a weighted voting scheme where votes of classifiers are weighted by their accuracies obtained from the dry run. Performances of this run are illustrated in Table 5. In particular, this run performed moderately and achieved overall accuracy of 68.00%. Two hundred and four queries (204) out of three hundred (300) were correctly tagged by our system. Please note that our system outperforms all the participants in terms of accuracy for the temporal class Recency. We also achieved high precision (81.33%) for the temporal class Past, although all major competitors also achieved high results.

[4]http://www.cs.waikato.ac.nz/ml/weka/

	Past	Recency	Future	Atemporal	Accuracy
Run1	81.33	**65.33**	62.66	62.66	68.00
Run2	78.66	**65.33**	46.66	50.66	60.33

Table 5: Class wise precision and overall accuracy achieved by our submitted runs

4.2 Run 2

We followed the same learning strategy for our Run 2 submission but with some slight changes. All 8 base classifiers were learned on the training data using features ranging from 1 to 10. The underlying idea was to withdraw from the decision all unigrams. As such, we would be able to test how much unigrams contribute to the correct classification. Results of the 10-fold cross validation process are presented in 4 and only predictions from 7 classifiers were then combined using weighted voting scheme. NBTree was barred from voting due to its very low accuracy level.

Performances of our system for Run 2 on three hundred (300) formal run queries are illustrated in Table 5. Overall accuracy of 60.33% is achieved by our system and worst results are obtained compared to Run 1, which indicates great influence of the vocabulary used in queries to learn the temporal intent.

5. CONCLUSIONS

In this paper, we report on our works as part of our participation on the Temporal Query Intent Classification (TQIC) subtask proposed in NTCIR-11 Temporal Information Access (Temporalia) Task. We submitted two runs for TQIC subtask. Both of our submitted runs are based on ensemble learning where weighted voting is used as combination technique. We made use of a set of features which could easily be extracted from three different freely available sources in the web to allow reproducibility. Our submitted runs out-

performed all other runs in terms of precision for temporal class Recency, although our best performing run achieved moderate accuracy for the other classes.

Initial results are interesting to us and open numerous directions for future research such as discovering additional features which could boost query intent classification results based on the same resources. We also plan to investigate how machine learning optimization techniques could further improve the classification results.

6. 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] G. Dias, M. Hasanuzzaman, S. Ferrari, and Y. Mathet. Tempowordnet for sentence time tagging. In *Proceedings of the 4th ACM Temporal Web Analytics Workshop (TEMPWEB 2014) associated to 23rd International World Wide Web Conference (WWW 2014)*, pages 833–838, 2014.

[3] T. Dietterichl. Ensemble learning. *The Handbook of Brain Theory and Neural Networks*, pages 405–408, 2002.

[4] A. Ekbal and S. Saha. Weighted vote-based classifier ensemble for named entity recognition: a genetic algorithm-based approach. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(2):9, 2011.

[5] H. Joho, A. Jatowt, B. R., H. Naka, and S. Yamamoto. Overview of ntcir-11 temporal information access (temporalia) task. In *Proceedings of the NTCIR-11 Conference*, 2014.

[6] L. Xu, A. Krzyzak, and C. Suen. Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on Systems, Man and Cybernetics (SMC)*, 22(3):418–435, 1992.