Temporal Information Retrieval

Shrinivas Kane

May 19, 2014

1 Introduction

Time is important factor in any domain and it has been studied in many areas ranging from text data mining to query log analysis.Crawling applications keeps on feeding enormous data to search application for analysis and information retrieval. Analyzing these document corpus using time based extraction techniques and Extracting time based information from corpus provides new insights. Time based data extraction will help to provide chronological order of events and may help to recreate the scenarios as they happened.Temporal information is virtually available in all documents as a timestamp in metadata or actual text content for e.g. Jane will be emailing documents by **May 8th 2014**. Thus extracting this information and presenting to users will improve search application quality.

2 Searching time in text documents

As we stated earlier time related information available in document implicitly or explicitly. But it has different representations. In this section we will identify Characteristics and representation of temporal Data in text documents

2.1 Characteristics of temporal Data

Temporal data has its own key Characteristics[4]

Well Defined

For temporal data is well defined means data can be analyzed or retrieved using starting point and end point which clearly defines boundaries.

Easy data normalization

Temporal data can be normalized with reference to specific semantics in specified format. For example in text "After Christmas day it's time to celebrate upcoming year 2014" can be normalized to *12-25-2013, 1/1/2014*

Data clustering

Data can be arranged in hierarchical clusters using different granularity for e.g. one can cluster data using granularity of day or in case of large corpus one may like to cluster data with years' worth granularity

2.2 Types of temporal information

Temporal information in documents can be categorized in four groups [6].

Date

This specifies specific date in text for e.g. we will be working on project since 12/2/2013. So *12/2/2013* constitutes to temporal information.

Time

This gives specific time in text for e.g. mission completed at 3.00 pm.So *3.00* pm constitutes to temporal information

Duration

These type of data items provide information about duration of event for e.g. three years gone and we are still working on the same problem. In this case *three years* will give temporal information

Set

This gives recurrence of events for example let's schedule meeting twice a week Thus twice a week provides recurrence information.

2.3 Temporal expressions

Time can be expressed in several ways as follows [2].

Implicate

Single or multiples values of this data can be mapped to single value and we may map it to timeline after data normalization for e.g. Day of Silence and Day of Dialogue can be mapped to *15 April*

Explicate

These expressions give explicit information at different granularity for e.g. *Dec 20 2014*

Relative

These expressions provide information in context with other expression(s) for example text in email states "meeting on Monday April 27 2014?" In this case *Monday* is relative expression which related to explicit expression *April 27 2014*

3 Current Research work

3.1 Storyline-based summarization [3]

In the world of streaming news we encounter enormous number of news feed It is not possible for each individual to go through all news articles so idea is to summarize these news feeds based on event, topic and present single line news summery in chronological order. Let's take example of presidential elections. In this case single line news summery will be presented from date of election declaration to declaration of results.

3.2 Temporal tagging

Major task is to extract temporal expression and try to normalize to single format. We have well established markup language called TimeML. It annotates documents text and applies following properties to text data.
Offset: start and end point of expression.

Type: setting expression type specified in section 2.2

Value: normalized value of expression

3.3 Temporal querying

In this area users are allowed to add temporal data as Additional constraint to text query for e.g. query string *world cup football 2014* query points to FIFA world cup in brazil this year.Search application should return top documents containing both terms football world cup and time data 2014

3.4 Temporal query re-ranking

Temporal information is not used while ranking query results in current search engines rather they use similarity matrix or link analysis techniques to rank document. In this area we add extra weight to each document based on its temporal data. We will explore more details about this research area in case study in next section

4 Case study : Learning Recurrent Event Queries for Web Search [1]

4.1 Introduction

We have some queries which reoccur with specific time interval for example world cup occurs every four years. Thus we have lots of query string world cup every four year. Current search engine returns results in chronological

order when user is interested in most recent edition of tournament. This case study tries to solve this problem by re-ranking documents based on recurring queries.

4.2 Feature generation

Basic terms :

Explicate time stamp

Contains at least one token containing time (World cup 2010). These queries are called time indicator queries.

Implicate timestamp

These queries can be generated by removing token containing timing data.

No time stamp

Other queries those can't be part of above two categories

REQ-query

Recurrent query

Queries for event(s) which occur at regular time interval

Non REQ-query

Non recurrent query

4.2.1 Query log analysis

Through query log analysis they created statical data about query terms. It includes number of unique queries, frequency of query terms and number of

queries with explicit time stamp. Furthermore they calculate ratio of number of queries with implicate time to no time stamped queries.

4.2.2 Query reformation

This analysis they were looking for how users narrowed their search queries by using time information. It include query terms switched from implicate timestamp to explicate timestamp. Finally they calculated ratio of the queries for switched queries to query frequency.

4.2.3 Search engine result set

For each implicit timestamp query they hit search engine and Retrieved results. They analyzed top 5,10,15 returned urls as a result and generated statistical data. it preliminary contains were number of titles/urls having temporal information.

4.2.4 Time series analysis and recurrent events seed-list

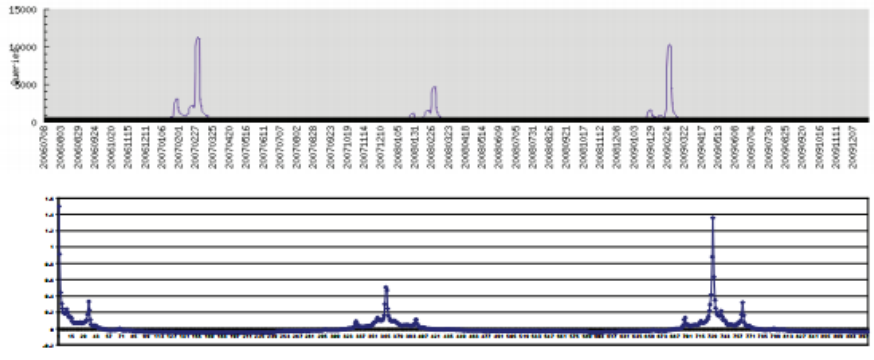

Figure 1: Term frequency plot for Recurrent query *Oscar* [1]

They plotted graph of query frequency related to Oscar for 2007, 2008 and 2009. As we can observe from graph there is one small spike when nominations were declared and at the time of actual ceremony. So using this data they calculated feature query period.

4.3 Query classification

4.3.1 Algorithms

Classifying text query as recurrent or non-recurrent is classical machine learning problem. There are lot of machine learning algorithms to solve this problem. They chose three different machine learning methods to implement classification problem

Naïve based algorithm

This method does not count relation between two features and treat them as independent entity. They used single Gaussian function for each independent feature. Mean and variance is used as evaluation parameters. Classification formula used is as follows

$p(y = c|x) = \frac{1}{z} * p(c) * \sum_{i=1}^{n} p(x_i|c)$

where x is each entity in feature vector.

Gradient boosted decision tree algorithm

This is decision tree based logistic regression model which uses additive regression algorithm.Probability of classifying query as req query is as follows

$p(y = c|x) = \frac{1}{1+e^{-f(x)}}$

where f(x) is learning function

They used 20 trees, 20 nodes and 0.8 shrinkage as input parameters for decision tree classification algorithm

SVM

They used LIBSVM which uses sequential minimal optimization algorithm for classification. They specifically used C-SVC linear function and 1.0 as shrinkage.

4.3.2 Algorithm Evaluation

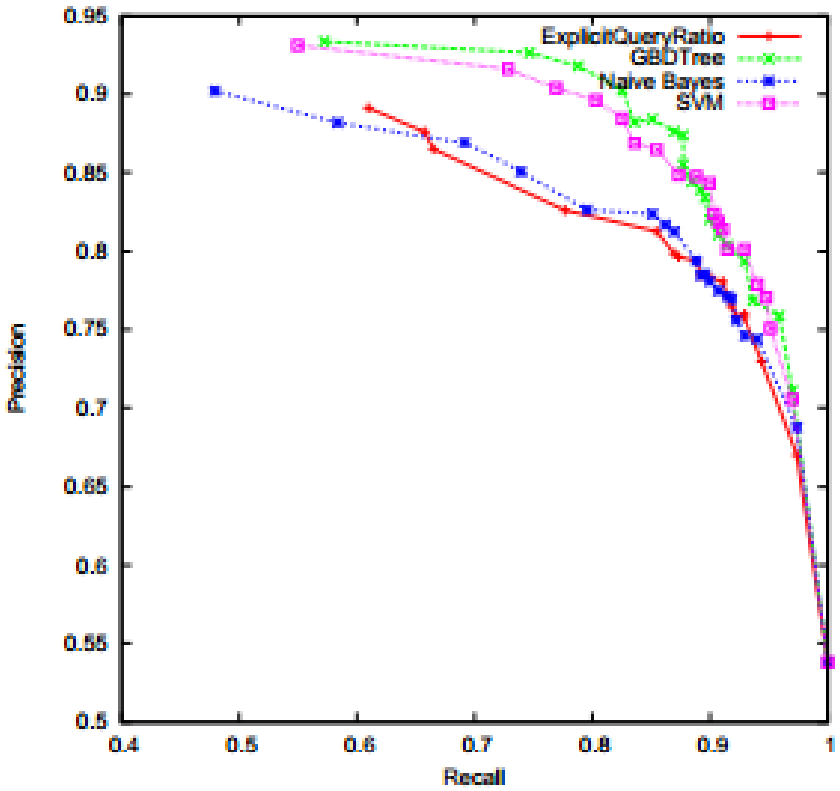

Figure 2: Precision recall rate for each method [1]

For training evaluation they collected 6000 queries. Out of 6000 queries

5000 were used to train data set and rest 1000 were used to test it. They computed features for each queries specified in section 3.2. Precision is defined as correctly classified req-queries to total number of queries classified as req-queries.

Recall is defined as correctly classified req-queries to total number of queries As we can observe from Fig 2 SVM and GBDT performed well over Naïve Bayes as they do account relations between feature(s).

Query	Probability
ncaa men's basketball tournament	0.999
bmw 328i sedan reviews	0.999
new apple iphone release	0.932
sigir	0.920
new york weather in april	0.717
academy awards reviews	0.404
google ipo	0.120
adidas jp	0.082

Figure 3: Probabilities for classified queries [1]

Figure 3 depicts results of classification algorithm. As we can observe *ncaa men's basketball tournament* is classified as recurrent query. This tournament reoccurs every year thus categorized correctly as req-query on the hand text query adidas jp is IPO release event for adidas japan which is going to happen once thus classifying query *adidas jp* as non-req

4.4 Re-ranking documents

They adjust document ranking by adding extra weight to recent document(s) if query was classified as REQ. They used ranking function F(q,d) to adjust documnet weight as below

$F(q,d) = R(q,d) + [e(d_0, d_n) + k]e^{\lambda\alpha(q)}$

Where R(q,d) base scoring function

$\lambda\alpha(q)$ is confidence score of REQ query

$[e(d_0, d_n) + k]e^{\lambda\alpha(q)}$ is difference between oldest page and newest page

They used GBDT algorithm for classification as it yielded best results for classifying queries into req and non-req classes.Furthermore they divided queries into tem buckets considering its probability as req or non-req with interval of 0.1. Clearly req queries goes into top buckets with Probability of 1.0 to 0.5 and non req queries reside in lower buckets i.e 0.5 to 0.0

Evaluating document re-ranking model was two-step process

Step 1 They ask humans to rate output results for query on grade of perfect match to bad match.

Step 2 They used discounted cumulative gain (DCG) for top and top 5 results. To measure retrieve performance. DCG function can be expressed mathematically as follows

$DCG@k = \sum_{i=1}^{k} \frac{2*r^i-1}{ln(1+i)}$

where r(i) is relevance grade of document.

		DCG@5			DCG@1		
bucket	#(query)	Organic	Our's	% over Organic	Organic	Ours	% over Organic
[0.0,0.1]	59	6.87	6.96	1.48(-2.3)	4.08	4.19	2.69(-1.07)
[0.1,0.2]	76	5.86	6.01	2.52(0.98)	2.88	2.91	1.14(1.69)
[0.2,0.3]	85	6.33	6.41	1.24(2.12)	3.7	3.7	0.0(0.8)
[0.3,0.4]	75	5.18	5.24	1.18(-0.7)	2.92	2.95	1.14(1.37)
[0.4,0.5]	78	4.96	4.82	-2.84(-1.35)	2.5	2.42	-3.06(0)
[0.5,0.6]	84	5.4	5.37	-0.45(-0.3)	2.82	2.85	1.05(-1.5)
[0.6,0.7]	78	4.78)	5.19)	8.42(3.64)	2.56	2.83	10.75(4.1)
[0.7,0.8]	80	4.45	4.60	3.41(3.19)	2.21	2.26	1.98(2.8)
[0.8,0.9]	78	4.81	4.96	3.15(4.79)	2.32	2.33	0.55(0.65)
[0.9,1.0]	107	5.08	5.50	**8.41***(4.41)	2.64	3.09	**16.78***(1.36)
[0.0,1.0]	800	5.33	5.47	**2.74***(2.17)	2.83	2.93	**3.6***(1.26)

Figure 4: Re-ranking document model results [1]

As we can observe form figure 4 query re-ranking certainly improved results considering time factor in re-ranking documents. They achieved 3 percent gain on average for all queries. It suggest successful improvement in document re-ranking with additional factor of time.

4.5 Learnings form case study

Case study demonstrated importance of time factor in document re-ranking for queries which point to reoccurring events. Machine learning approach suggests dynamic adjustments based on time so in future some events stops occurring this approach will slowly remove those queries from REQ class dynamically. Finally this case study shows results by successful implementation of problem with excellent results.

5 Challenges and future work

For time bound temporal information retrieval start time and end time is defined. In microblogging and social networking era it's really difficult to recognize start point and end point of event which makes very difficult to define time line of events. Furthermore it's still challenge to detect life of any event.

Runtime analysis of current events and mapping on timeline or optimizing index cache based on temporal query analysis can be extension of this term project

6 Conclusion

We started discussion from answering questions like what is time, how time is expressed in data, how time is classified . Then we explained current research trends. We analyzed case study where some of the questions mentioned answered and finally, we conclude with challenges and future work.

References

[1] Ruiqiang Zhang ,Yuki Konda .*Learning Recurrent Event Queries for Web Search.* Association for Computational Linguistics , Oct 11, 2010

[2] F. Schilder and C Habel *From temporal expressions to temporal information.* TASIP ,2001

[3] James Allan, Rahul Gupta. *Temporal summaries of new topics.* – SIGIR , 2001

[4] Omar Alonso,Jannik Strotgen. *Temporal information retrieval challenges and opportunities.* TWAW , 2011

[5] J F Allen. *Maintaining knowledge about temporal intervals.* communications of ACM 1983

[6] James Pustejovsky and José Castaño and Robert Ingria and Roser Saurí and Robert Gaizauskas and Andrea Setzer and Graham Kat *Timeml: Robust specification of event and temporal expressions in text.* IWCS-5, 2003