Predicting the News of Tomorrow Using Patterns in Web Search Queries

Kira Radinsky, Sagie Davidovich and Shaul Markovitch
Computer Science Department
Technion–Israel Institute of Technology
{kirar, sagied, shaulm}@cs.technion.ac.il

Abstract

The novel task we aim at in this work is to predict top terms that will prominently appear in the future news. This is a difficult task that nobody attempted before, as far as we know. We present a novel methodology for using patterns of user queries to predict future events. Query history is obtained from web resources such as Google Trends. In order to predict whether a term will appear in tomorrow's news, we examine if the terms in today's queries indicated this term in the past. We provide empirical support for the effectiveness of our method by showing its prediction power on news archives.

1 Introduction

Many organizations invest significant efforts in trying to predict events that are likely to take place in the near future. Such predictions can be beneficial for various purposes, such as planning, resource allocation and identification of risks and opportunities. Predicting global events in politics, economics, society, etc. is a difficult task that is usually performed by human experts possessing extensive domain-specific and common-sense knowledge. Is it possible to design an algorithm that will automate this process?

Many events are hard to predict due to the fact that their occurrence is spread over a long time period, in a very tangled net of relations and mutual influence. However, some of them share a common pattern. Events that indicated other events in the past, might do so again as history repeats itself. Some of the events have preliminary signs. Identifying these signs may enable us to predict the events themselves.

The current state of the art in NLP does not allow us to perform deep analysis of news to identify events. However, a global event usually draws the attention of web users, triggering many of them to submit queries related to that event. We assume that the popularity of terms appearing in such queries peaks when the event occurs. These spikes can be used as supporting evidence for the occurrence of the event.

In this paper we present a novel method, PROFET, that mines large-scale web resources to predict terms that are likely to appear in the news of the near future. Specifically, we predict 100 terms that will prominently appear in the news up to one week from now. The main resource used by our method is a search-query history archive (specifically, in this work we use *Google Trends*). In order to predict whether an event will appear in tomorrow's news, we examine if the terms representing today's events (extracted from today's queries) indicated this event in the past. This is done by analysis of patterns in user queries for these terms.

We test our algorithm by examining if the terms it predicts indeed appear in the news. We compared its performance to a baseline method which assumes that the news of today will be the news of tomorrow and found our algorithm to be significantly better, especially for longer prediction periods.

The main contributions of this paper are threefold: First, we introduce a new method for prediction of global future events using their patterns in the past. Second, we present a novel usage of aggregated collection of search queries. Finally, we introduce a testing methodology for evaluating such news prediction algorithms.

2 The Prediction Algorithm

In this work we obtain history of user queries from two main sources:

1. **Google Trends** is a service that provides charts representing the popularity of given search terms over time.
2. **Google Hot Trends** is a service that presents the 100 top searched queries on a certain day, that deviate the most from their historic search pattern. The service also provides related searches for each of the top terms.

2.1 Formal framework

Let $W = \{w_1, w_2, ..., w_k\}$ be a set of terms characterizing events. Let $D = \langle d_1, ..., d_n \rangle$ be an ordered set of days.

Assume that for each term w_i we are given a binary vector $g(w_i) = \langle d_1^i, ..., d_n^i \rangle$. Intuitively, $d_j^i = 1$ indicates that on day d_j the term w_i appeared prominently.

We define prediction for a k days interval (events that will occur in k days):

Definition 1 *A term w_j* ***indicates term*** *w_i* ***in an interval k****, denoted by $w_j \xrightarrow{k} w_i$, if : $P(d_t^i = 1) + \Delta < P(d_t^i = 1 | d_{t-k}^j = 1)$, where t is uniformly drawn from the interval $[1, n]$ and $0 < \Delta < 1$.*

To make the formal setup more practical we make the following simplifications:

1. We conclude if a term w_i *prominently appeared* on day d_t ($d_t^i = 1$) by testing for peaks in its frequency of search, specifically the peaks from its corresponding *Google Trends*' chart.

2. Due to complexity considerations, we do not address the entire set $W \times W$ to find indications. We test for indications only from the subset $Salient \times Related(Salient) \subset W \times W$, specifically in this work we used *Google Hot Trends* to determine saliency, and *Google Related Trends* to determine relatedness.

2.2 Peak Detection

The raw data for our peak detection is $TermChart(w_i) = \langle f_1^i, ..., f_n^i \rangle$, where $f_j^i \in [0, 1]$ represents the normalized value of the search volume of the term w_i on day $d_j \in \langle d_1, ..., d_n \rangle$.

We assume that events occur when a "deep maximum" is present, that is, people search for the term abnormally more than usual. Therefore, we must define criteria that will separate "events' peaks" from local noise.

The algorithms first extracts local maximum and local minimum points from $TermChart(w_i)$. Each maximum point m has at most two neighboring minimum points. We consider a maximum point a peak if $m > \Delta_1$ and the difference between the max point and the lowest of its neighboring minimum points is above Δ_2.

After extracting the peaks of w_i, we can construct $g(w_i) = \langle d_1^i ... d_n^i \rangle$, where $d_j^i = 1$ if a peak was observed on day d_j.

2.3 The PROFET Algorithm

The goal of the PROFET algorithm is to predict which terms will peak in k days.

The algorithm returns a list of terms with their associated weights. A higher weight means that the algorithm has a stronger belief that it will appear in k days. The weight of each candidate term (which belongs to the union of the related terms) is computed as the sum of indication weights for all terms salient today. When $w_1 \xrightarrow{k} w_2$, the indication weight of w_2 given w_1, where w_1 is one of today's salient terms and w_2 is the prediction, is calculated based on:

- How many of the peaks of w_2 occurred k days after w_1, i.e. $P(d_t^2 | d_{t-k}^1)$.
- How salient the term w_1 is today. Where *Saliency* of a term is based on how strongly the frequency of search for this term today deviates from its historic search pattern.

We call this method of computing the strength of prediction of one term based on another *W-Prediction*. For a more formal description of the algorithm see Figure 1.

Procedure PREDICTION(k, day)
$W^S \leftarrow$ MOSTSALIENT(day)
$W^R \leftarrow \{\bigcup Related(w) | w \in W^S\}$
Return $\{\langle w, \sum_{t \in W^S} \text{W-PREDICTION}(t, w, k)\rangle | w \in W^R\}$

Procedure W-PREDICTION(w_1, w_2, k)
$N = |TermChart(w_1)|$
$g(w_1) \leftarrow \langle d_1^1, ..., d_n^1 \rangle)$, $d_j^1 = \begin{cases} 1 & j \in Peaks \\ 0 & otherwise \end{cases}$
$g(w_2) \leftarrow \langle d_1^2, ..., d_n^2 \rangle)$, $d_j^2 = \begin{cases} 1 & j \in Peaks \\ 0 & otherwise \end{cases}$
$p(w_2) \leftarrow |Peaks(g(w_2))|/N$
$hits \leftarrow |\{\ d\ |g(w_1)[d-k] = 1 \text{ and } g(w_2)[d] = 1, k \leq d \leq N\}|$
$p(w_2|w_1) \leftarrow hits/|Peaks(g(w_1))|$
Saliency$(w_1) \leftarrow \frac{TermChart(w_1)[today]}{Average(TermChart(w_1))}$
IndicationWeight $\leftarrow p(w_2|w_1) \cdot Saliency(w_1)$
Return $\begin{cases} IndicationWeight & p(w_2|w_1) - p(w_2) > \Delta \\ 0 & otherwise \end{cases}$

Figure 1. The Prediction algorithm

3 Experimental Evaluation

We have implemented the PROPHET algorithm and evaluated its performance. In the learning phase we trained seven predictors, $p_1, \ldots, p_7$, for prediction intervals of one to seven days. The training data included history of 5 years of user queries obtained from *Google Trends*.

For testing we used a very large dataset of news obtained from **Google News**. It contains around 5,000,000 items daily and covers most of the available news sources (4,500 news sources). Our learner did not have access to this data.

The task we set was to predict 100 terms that will *significantly appear* on a day:

Definition 2 *Let t be a given term. A term t* ***significantly appears*** *on day d if* $\frac{1}{365}\sum_{i=d-365}^{d-1} frequency(t,i) < \frac{1}{\mu} frequency(t,d)$.

In our tests, $\mu = 20$. Thus, a term *significantly appears* on a certain day if its frequency of appearance on that day is substantially bigger than its average daily appearance in the past year.

To test a predictor p_i, we run it on 25 consecutive days (June 6th, 2007 – July 1st, 2007). For each input day d, the task of the p_i was to output 100 terms it predicts to *significantly appear* in day $d + i$. We then used the real data to count what fraction of the 100 terms indeed *significantly appeared*. Thus, the principle performance measurement of our test was precision at 100.

Precision at k documents retrieved is a common measure of information retrieval performance, when recall on the set of relevant documents is hard to estimate. In our case, retrieving all news on a certain day was indeed hard. This measure also reflects the utility of retrieval results more directly.

We had difficulties finding appropriate baseline algorithms to compare with, as this task was not tried before. The baseline method we eventually chose is the "today = tomorrow" algorithm. This method predicts that the same terms that are prominent today will be also prominent tomorrow (or in k days, when the prediction interval is k). The baseline method works quite well, since events mentioned in the news tend to last more than one day. An additional method we suggested is one that randomly chooses terms from the related terms, rather than use the indication mechanism.

Before we started the experiments, we had preformed parameter tuning of Δ_1 and Δ_2 using a time range different than the one used for testing (1-18 of January, 2008). The performance peaks with $\Delta_1 = 0.2$ and $\Delta_2 = 0.1$, which we therefore select as the default values.

3.1 The performance of PROFET

Graph 2(a) shows the performance of PROFET and the baseline on the test set with prediction interval (k) of 1 to 7. Each point in the graph represents the average of the precision (at 100) over the 25 test days for the given prediction interval. In Figures 2(b) and 2(c), we give a more focused view on a 1 day prediction and a 7 days prediction respectively. The X-axis and Y-axis represent the precision-at-100 of the baseline method and PROFET respectively. Each point represents the precision of each algorithm on one test day.

The pattern of difference in performance is quite clear. As the prediction interval increases, the advantage of PROFET becomes more apparent. Indeed, a paired t-test shows that the advantage of PROFET is significant ($p < 0.05$) for all values of k, except $k = 1$. The baseline algorithm has shown to be a strong competitor to PROFET for a prediction of one day. This is no surprise, as news does not fade away quickly. However, for prediction for longer periods, we observe a significant advantage of our algorithm.

3.2 Qualitative results

To have a better understanding of the prediction abilities of PROFET, we list here several predictions it made.

1. **Hurricane and Gas Prices** – On September 9th 2008, several terms related to Galveston Hurricane which hit Texas appeared as salient. One of the related terms this day was "gas prices" (was related to the search term "gas buddy"). In the past, queries about "Hurricane" were followed 3 days later by queries about "gas prices", probably due to the fact that hurricanes in the past, such as Katrina and Rita in 2005, affected the gas prices dramatically, and people search queries conveyed this situation. Due to the fact that 9 salient terms from that day indicated the term "gas prices" this result got a high weight. PROFET predicted that this term will appear on September 12th on the news. And indeed, it appeared in the news, due to a record in gasoline prices.

2. **Apple Share** – On January 15th, terms related to Steve Jobs keynote on Apple's new products were salient in the search queries that day. PROFET revised the related terms, one of which was "AAPL" (Apple share). In the past queries about "Steve Jobs" indicated the term "AAPL" in an interval of 7 days, and thus PROFET predicted it would appear in the news a week later. Indeed, a week later, "AAPL" has been unusually dominant in the news in several economy reviews about Apple.

3.3 Random indicator vs PROFET

One potential critique of PROFET could be that the main power of the algorithm is embedded in Google's related terms, and the indication mechanism is redundant. To evaluate the importance of the indication mechanism to the performance of PROFET, we tested a version of PROFET without it. This *No-Indication* algorithm will not conduct any checking of indications, but rather randomly output 100 of the related terms. Figures 3(a) and 3(b) compare the performance of the *No-Indication* algorithm with that of PROFET and show that indeed the indication process is crucial. All the results were found to be statistically significant ($p < 0.05$).

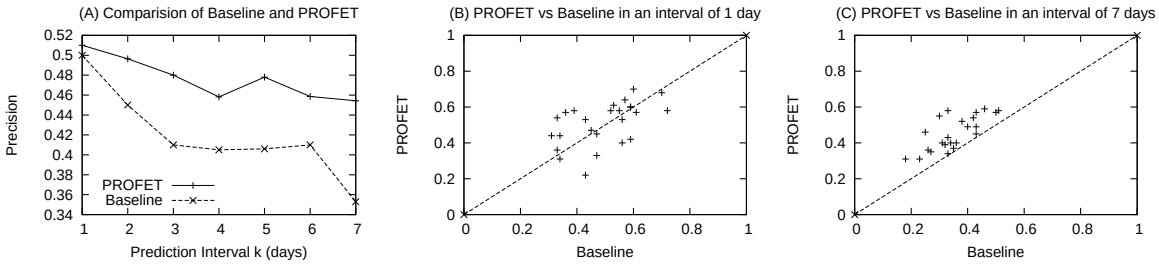

Figure 2. (a) Comparison of the performance of Baseline and PROFET on several prediction intervals. (b), (c) Performance on a prediction interval of 1 and 7 days

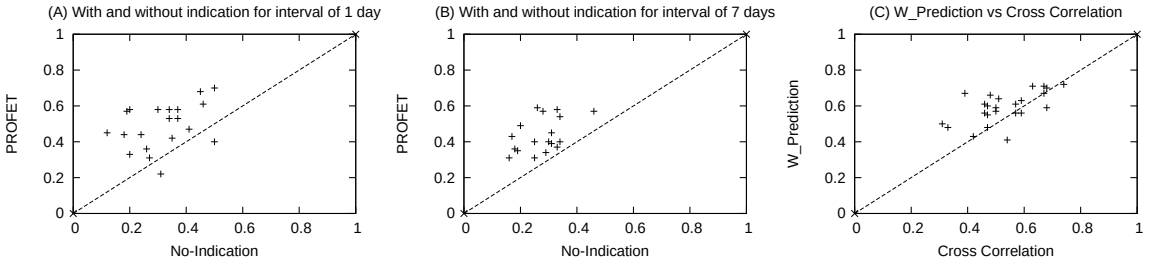

Figure 3. (a), (b) Performance with and without the indication mechanism on a prediction interval of 1 day and 7 days. (c) W-Prediction compared with Cross-Correlation on a prediction interval of 1 day. Each point represents the precision of the algorithm on each of the days of the test period.

3.4 W-Prediction vs Cross-Correlation

The motivation for the development of our *W-prediction* method was the lack of sufficient statistics for using more traditional correlation methods such as cross correlation. For example, Chien and Immorlica ([4]) have investigated the idea of finding semantically-related search-engine queries based on the correlation coefficient of their frequency functions. In this subsection we experiment with a version of PROFET where the w-prediction component is replaced by cross correlation (for a fixed delta). Figure 3(c) compares the results of this method with that of PROFET for $k = 1$. PROFET shows a much better performance and its advantage was found to be statistically significant ($p = 0.0015$). While w-prediction was found to perform better on this data, it is quite possible that with much larger quantities of data (perhaps spread over longer periods) cross correlation would have been a better method.

3.5 Related Work

Time-series analysis has been extensively used for prediction tasks. Some of the most common approaches in time series prediction include Autoregressive, Moving Average, ARMA, ARIMA, ARCH [16] and extensions of these algorithms (such as seasonal ARIMA [3] which deals with seasonable time series that display repetitive behavior or periodic patterns). Some neural network algorithms, such as MLP, SOM and recurrent network have been also applied to time series forecasting [10, 17, 1]. These techniques have shown good results, but they are not necessarily applicable for the task of predicting global events. Most of them predict numerical values, rather than the semantic representation of the events. Global events, such as elections or social events, are hard to represent intuitively by numeric values.

In this work we use query logs to produce future predictions. Query log mining deals with extraction of knowledge from logging user activities on the web. This data is mostly used for personalization [7, 11], related queries suggestion [12, 20], query extension [5], improvement of search [18], prediction of user activity [19], extracting semantic relations [2] and identifying similarity between queries [4]. Facca and Lanzi ([8]) provide a comprehensive survey of query log mining. Here we use query logs in a novel way for a task that has never been attempted before in those means.

In the field of sentiment analysis, researchers have devel-

oped methods for using trends in user activities to identify users' preferences. Mostly, the methods employed include combinations of basic NLP methods and machine learning, and achieve good accuracy [6, 14, 15]. In particular, some of the works in sentiment analysis try to use blogs to predict events such as book sales [9] and movie views [13]. These works are related to ours as they also use web data to predict events. The work of Gruhl et. al. ([9]) even uses peaks in book references (analogously to our usage of peaks in query terms). These methods, however, are aimed at predicting specific types of events, while ours is aimed at predicting global events that are not domain specific.

4 Conclusions

In this paper we tackle a very interesting problem: Can we predict what will appear in the news tomorrow, next week, or later? We presented a novel methodology that uses query popularity today, and historical patterns of what queries tended to follow other queries to predict terms in future news.

Empirical evaluation shows precision of up to 0.8 (mean of 0.52) for 1-day prediction and up to 0.6 (mean of 0.46) for 7-day prediction. The results for the longer prediction intervals are significantly better than those for the baseline method. Our experiments confirmed that our indication method is essential to the success of the algorithm, and that using the related terms alone, without this mechanism, is not sufficient. We have also shown that our *W-prediction* method is more suitable for this task than Cross-Correlation.

While we used *Google Trends* for the experiments described in this paper, we could have used other data sources available on the web. For example, **Technorati** (a large blog engine providing term popularity in blogs) and top terms from Google News. A preliminary implementation looks rather promising. For example, using the above, our algorithm was successful in predicting the terms "oil" and "stock", day after the term "dollar drop".

We believe that the novel method described in this work is a good demonstration for the potential usage of the knowledge that can be extracted from web resources. Such rich source of knowledge can be utilized for many complex tasks.

References

[1] A. Atiya and A. Parlos. Identification of nonlinear dynamics using a general spatio-temporal network. *Math.Comput. Modeling*, 21:53–71, 1995.

[2] R. Baeza-Yates and A. Tiberi. Extracting semantic relations from query logs. In *Proceedings of KDD2007*, pages 76–85, 2007.

[3] G. E. P. Box and G. Jenkins. *Time Series Analysis, Forecasting and Control*. Holden-Day, Incorporated, 1990.

[4] S. Chien and N. Immorlica. Semantic similarity between search engine queries using temporal correlation. In *Proceedings of WWW2005*, New York, NY, USA, 2005.

[5] H. Cui, J.-R. Wen, J.-Y. Nie, and W.-Y. Ma. Probabilistic query expansion using query logs. In *Proceedings of WWW2002*, pages 325–332, 2002.

[6] D. Dave and S. Lawrence. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. In *Proceedings of WWW2003*, 2003.

[7] Z. Dou, R. Song, and J.-R. Wen. A large-scale evaluation and analysis of personalized search strategies. In *Proceedings of WWW2007*, pages 581–590, 2007.

[8] F. M. Facca and P. L. Lanzi. Mining interesting knowledge from weblogs: a survey. *Data and Knowledge Engineering*, 53:225–241, 2005.

[9] D. Gruhl, R. Guha, R. Kumar, J. Novak, and A. Tomkins. The predictive power of online chatter. In *Proceedings of KDD2005*, pages 78–87, New York, NY, USA, 2005.

[10] H. R. J. Walter and K.Schulen. Non linear prediction with self-organizing maps. In *Proceedings of ICNN*, volume 1, pages 589–594, 1990.

[11] T. Joachims, L. Granka, B. Pan, H. Hembrooke, and G. Gay. Accurately interpreting clickthrough data as implicit feedback. In *Proceedings of SIGIR2005*, pages 154–161, 2005.

[12] N. Liu, S. Nong, J. Yan, B. Zhang, Z. Chen, and Y. Li. Similarity of temporal query logs based on arima model. In *Proceedings of ICDMW2006*, pages 366–370, 2006.

[13] G. Mishne and N. Glance. Predicting movie sales from blogger sentiment. In *Proceedings of AAAI2006*, 2006.

[14] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79–86, 2002.

[15] T. Peter. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL-02*, pages 417–424, 2002.

[16] E. E. Peters. *Fractal Market Analysis: Applying Chaos Theory to Investment and Economics*. Wiley, 1994.

[17] A. S. Weigend and N. A. Gershenfeld. Time series prediction: Forecasting the future and understanding the past. In *Proceedings of NATO Comparative Time Series*, 1994.

[18] G.-R. Xue, H.-J. Zeng, Z. Chen, Y. Yu, W.-Y. Ma, W. Xi, and W. Fan. Optimizing web search using web click-through data. In *Proceedings of CIKM*, pages 118–126, 2004.

[19] Q. Yang, H. Wang, and W. Zhang. Web-log mining for quantitative temporal-event prediction. *IEEE Computational Intelligence*, 1:10–18, 2002.

[20] Z. Zhang and O. Nasraoui. Mining search engine query logs for query recommendation. In *Proceedings of WWW2006*, pages 1039–1040, 2006.