Learning Word Relatedness over Time

Guy D. Rosin[1], Eytan Adar[2], Kira Radinsky[1,3]
[1]Technion – Israel Institute of Technology, Haifa, Israel
[2]University of Michigan, Ann Arbor, USA
[3]eBay Research, Israel
{guyrosin,kirar}@cs.technion.ac.il, eadar@umich.edu

Abstract

Search systems are often focused on providing relevant results for the "now", assuming both corpora and user needs that focus on the present. However, many corpora today reflect significant longitudinal collections ranging from 20 years of the Web to hundreds of years of digitized newspapers and books. Understanding the *temporal intent* of the user and retrieving the most relevant historical content has become a significant challenge. Common search features, such as query expansion, leverage the relationship between terms but cannot function well across all times when relationships vary temporally. In this work, we introduce a temporal relationship model that is extracted from longitudinal data collections. The model supports the task of identifying, given two words, *when* they relate to each other. We present an algorithmic framework for this task and show its application for the task of query expansion, achieving high gain.

1 Introduction

The focus of large-scale Web search engines is largely on providing the best access to present snapshots of text – what we call the "Now Web". The system constraints and motivating use cases of traditional information retrieval (IR) systems, coupled with the relatively short history of the Web, has meant that little attention has been paid to how search engines will function when search must scale not only to the number of documents but also temporally. Most IR systems assume fixed language models and lexicons. They focus only on the leading edge of query behavior (i.e., what does the user likely mean today when they type "Jaguar"). In this context, features as basic as disambiguation and spelling corrections are fixed to what is most likely today or within the past few years (Radinsky et al., 2013), query expansions and synonyms are weighted towards current information (Shokouhi and Radinsky, 2012), and results tend to include the most recent and popular content. While this problem would seem to be speculative in that it will be years until we need to address it, the reality is the rate of change (Adar et al., 2009) of the Web, language, and culture have simply compressed the time in which critical changes happen.

The "Now Web" assumptions are entirely reasonable for temporally coherent text collections and allow users (and search engines) to ignore the complexity of changing language and concentrate on a narrower (though by no means simpler) set of issues. The reality is that this serves a significant user population effectively. There are nonetheless a growing number of both corpora and users who require access not just to what is relevant at a particular instant (e.g., Hathitrust (Willis and Efron, 2013), historical news corpora, the Internet Archives, and even fast changing Twitter feeds). Within such contexts, a search engine will need to vary the way it functions (e.g., disambiguation) and interacts (e.g., suggested query expansions) depending on the period and temporal scale of documents being queried. This, of course, is further complicated by the fact that Web pages are constantly evolving and replaced.

Take for example the query "Prime Minister Ariel Sharon". When fed into a news archive search engine, the likely intent was finding results about *Sharon's* role as Israel's *Prime Minister*, a role held from *2001 to 2006*. Singh et al. (2016) refer to this as a *Historical Query Intent*. However, most popular search engines return results about Sharon's death in 2011, when he was no longer

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1179–1189
Copenhagen, Denmark, September 7–11, 2017.

prime minister. The searcher may take the additional step of filtering results to a time period, but this requires knowing what that period should be. Other query features are also unresponsive to temporal context. For example, the top query suggestions for this query focus on more recent events of his death: "Former prime minister Sharon dies at 85", "Former prime minister Sharon's condition worsens", etc. While these *might* satisfy the searcher if they are looking for the latest results, or the results most covered by the press, there are clearly other possible needs (Bingham, 2010).

In this paper, we focus on the task of measuring word relatedness over time. Specifically, we infer whether two words (tokens) relate to each other during a certain time range. This task is an essential building block of many temporal applications and we specifically target *time-sensitive query expansion (QE)*. Our focus is on *semantic relatedness* rather than *semantic similarity*. Relatedness assumes many different kinds of specific relations (e.g. meronymy, antonymy, functional association) and is often more useful for computational linguistics applications than the more narrow notion of similarity (Budanitsky and Hirst, 2006).

We present several temporal word-relatedness algorithms. Our method utilizes a large scale temporal corpus spanning over 150 years (The New York Times archive) to generate *temporal deep word embeddings*. We describe several algorithms to measure word relatedness over time using these temporal embeddings. Figure 1a presents the performance of one of those algorithms on the words "Obama" and "President". Note that the highest relatedness score for the words appears during the presidential term of Barack Obama. Similarly, Figure 1b shows a high score for "Ariel Sharon" and "prime minister" only during his term.

Using the approach above, we present a specific application – producing temporally appropriate query-expansions. For example, consider the query: "Trump Businessman". Figure 2 shows the non-temporal query expansion suggestions which focus heavily on the first entity (i.e., "Trump") and his current "state" (i.e., a focus on Donald Trump as President, rather than presenting suggestions about Trump's business activity as implied by the query). We present an empirical analysis presenting the strengths and weaknesses of the different temporal query-expansion algorithms and comparing them to current word-embeddings-based QE

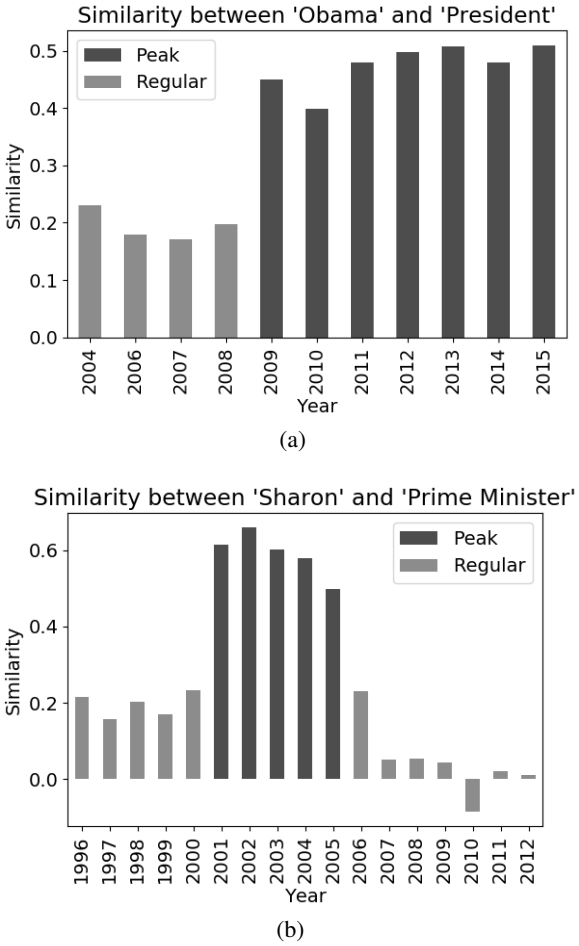

Figure 1: Similarity identified by our algorithms between words over time. Dark gray indicates high similarity whereas light gray indicates non-significant similarity.

algorithms (Kuzi et al., 2016; Diaz et al., 2016).

In this paper we describe a novel problem of evaluating word relatedness over time and contribute our datasets to evaluate this task to the community[1]. Second, we present novel representations and algorithms for evaluating this task and show high performance. We share our code with the community as well. Finally, we present the application of this task to query-expansion and present several methods built on top of the temporal relatedness algorithms that show high performance for QE.

2 Related Work

Understanding the semantic change of words has become an active research topic (Section 2.1). Most work has focused on identifying semantic

[1] https://github.com/guyrosin/learning-word-relatedness

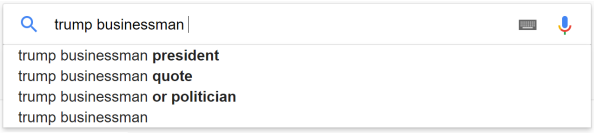

Figure 2: Query expansions for the terms "Trump Businessman". Most results are referring to his term as president and not to his business activity.

drifts and word meaning changes. A lot of effort has been made into analyzing texts temporally: several methods for temporal information extraction were recently proposed (Ling and Weld, 2010; Kuzey and Weikum, 2012; Talukdar et al., 2012), as well as publicly released knowledge bases, such as YAGO2 (Hoffart et al., 2013). These methods automatically extract temporal relational facts from free text or semi-structured data. In addition, Pustejovsky et al. (2003); UzZaman et al. (2012) and others annotated texts temporally, and extracted events as well as temporal expressions.

Work in Information Retrieval (IR, Section 2.2) has discussed the concept of 'time' as a contextual parameter for understanding user intent. Largely, this research utilizes query-log analysis with the time *of the query* as a context signal. In this work, we leverage the temporal variation in word relatedness to understand, and better accommodate, intent.

2.1 Word Dynamics

Continuous word embeddings (Mikolov et al., 2013) have been shown to effectively encapsulate relatedness between words. Radinsky et al. (2011) used temporal patterns of words from a large corpus for the task of word similarity. They showed that words that co-occur in history have a stronger relation. In our work, we focus on identifying *when* a relation holds. Numerous projects have studied the change of word meanings over time, and specifically focused on identification of the change itself. Sagi et al. (2009) used Latent Semantic Analysis for detecting changes in word meaning. Wijaya and Yeniterzi (2011) characterized 20 clusters to describe the nature of meaning change over time, whereas Mitra et al. (2014) used other clustering techniques to find changes in word senses. Mihalcea and Nastase (2012) identified changes in word usage over time by the change in their related part-of-speech. Others have investigated the use of word frequency to identify epochs (Popescu and Strapparava, 2013).

Jatowt and Duh (2014) represented a word embedding over the Google Books corpus (granularity of decades) and presented qualitative evaluation for several words. Hamilton et al. (2016) built Word2Vec embedding models on the Google Books corpus to detect known word shifts over 30 words and presented a dozen new shifts from the data. The authors presented two laws that govern the change of words – frequent words change more slowly and polysemous words change more quickly. Finally, Kenter et al. (2015) studied changes in meaning (represented by a few seed words), and monitored the changing set of words that are used to denote it.

In our work, we focus on learning relatedness of words over time. We evaluate the technique using a large scale analysis showing its prediction accuracy. Moreover, we define the task of identifying the temporality of relatedness between two words. We show that understanding the temporal behavior of entities improves performance on IR tasks.

2.2 Temporal Search

The temporal aspects of queries and ranking gained significant attention in IR literature. Some have focused on characterizing query behavior over time. For example, different queries change in popularity over time (Wang et al., 2003) and even by time of day (Beitzel et al., 2004). Jones and Diaz (2007) described queries to have three temporarilty patterns: atemporal, temporally unambiguous and temporally ambiguous. Others have leveraged temporal variance to look for indicators of query intent (Kulkarni et al., 2011). Several efforts (Shimshoni et al., 2009; Chien and Immorlica, 2005; M. Vlachos and Gunopulos, 2004; Zhao et al., 2006; Shokouhi, 2011; Radinsky et al., 2012) were done to not only characterize the temporal query behavior but also model it via time-series analysis. Radinsky and colleagues modeled changes in the frequency of clicked URLs, queries, and clicked query-URL pairs by using time-series analysis and show its application for improving ranking and query auto-suggestions (Radinsky et al., 2013; Shokouhi and Radinsky, 2012). Singh et al. (2016) focused on serving the specific needs of historians, and introduced the notion of a *Historical Query Intent* for this purpose.

Whereas prior work mainly focuses on query-log analysis and understanding the user's intent

based on the time the query was issued or the document was changed, our work focuses on understanding the subtle changes of language over time that indicate a temporal intent. We show that understanding the temporal relatedness between words is a building block needed for understanding better query intent. Given two words or entities we identify when their relatedness was the strongest to help produce better query expansions.

3 Temporal Relatedness Dynamics and Semantics

To address the task of understanding temporal relatedness, our approach consists of three main steps: (1) Represent relatedness using word embeddings over time (Section 3.1). (2) Model relatedness *change* over time as a time series (Section 3.2). (3) Combine these to identify when relatedness relations hold temporally (Section 3.3).

3.1 Representing Relatedness using Word Embeddings

In our work we leverage the distributed representation framework of Word2Vec (Mikolov et al., 2013) (specifically skip-grams). Intuitively, given a word w_t, skip-grams attempt to predict surrounding words, e.g. $w_{t-2}, w_{t-1}, w_{t+1}, w_{t+2}$ for a window size of $c = 2$.

Definition 3.1. Let C_i be the word context for a word w_i. In this work, we consider the context to be the surrounding words of a window size of n $C_i = \{w_{i-n}, w_{i-1}, w_{i+1}, w_{i+n}\}$. We define C_i^t to be the context for a word w_i in time period t, i.e. only in the documents written during time t.

Definition 3.2. We define the word context of a **specific embedding** of a year y for a word w_i, to be $\{w_c \in C_i^y\}$. Intuitively, a specific embedding represents the embedding of a word in a certain year. We denote the vector representation of a word w_i in a year y by v_i^y.

Definition 3.3. We define the word context of a **global embedding** for a word w_i to be C_i. Intuitively, a global embedding represents the embedding of a word over all time periods. We denote the vector representation of a word w_i by v_i.

Using Word2Vec, we approximate the semantic relatedness between two entities by the cosine similarity between their embeddings (Turney et al., 2010).

Definition 3.4. A **temporal relation** (e_1, e_2, t), where e_i are entities and t is a referenced time period, is said to be true if e_1, e_2 relate during t, i.e. their semantic relatedness during that time is relatively high. For example: *(Christopher Nolan, The Dark Knight, 2008)* is true, due to the fact that the movie, which was released in 2008, was directed by Nolan.

Definition 3.5. The **dynamics** of two entities e_1, e_2 is defined to be the time series of the semantic relatedness of e_1 and e_2:

$$Dynamics(e_1, e_2) = \left\langle cos(v_1^{t_1}, v_2^{t_1}), \ldots, cos(v_1^{t_n}, v_2^{t_n}) \right\rangle \quad (1)$$

where $t_1, \ldots, t_n$ are all the time periods.

We model entities' relatedness change over time by constructing their *dynamics*. Recall Figure 1a, which shows the semantic distance between the vector representations of *Barack Obama* and *President* over the years. Semantic distance is an accurate indicator of the time period when Obama was president. Therefore, our goal is to detect the time periods of high relatedness using peak detection.

3.2 Understanding Relatedness Dynamics

When considering a relationship between entities, detecting time periods of high relatedness enables us to reveal and identify what we call "periods of interest" – time periods in which the entities were the closest. In this section, we present an algorithm to identify these "periods of interest". Intuitively, when considering the *dynamics* of two entities, its peaks represent the lowest distances over time. More formally, given two entities e_1, e_2 we want to construct their *dynamics* and find peaks in it, i.e. the following sequence of time periods: $\{t_i \mid cos(v_1^{t_i}, v_2^{t_i})$ is relatively high, and $t_i < t_j$ if $i < j\}$.

Traditional peak detection algorithms focus on detecting peaks that are sharp and isolated (i.e. not too many surrounding points have similar values) (Palshikar et al., 2009). In our case, relations often imply continuous periods of peaks, e.g. Obama was president for eight years. Therefore, we need to detect peaks, as well as periods of continuous peaks (i.e., steps).

Let $L = (t_1, v_1), (t_2, v_2), \ldots, (t_n.v_n)$ be a list of tuples, where t_i are time periods and v_i are values. Given L, the algorithm returns a list of peak periods, i.e. $\{t \in L \mid t$ contains a peak$\}$.

The first step is to find relative maxima: we scan L and look for pairs (t_i, v_i) such that $v_{i-1} < v_i > v_{i+1}$, i.e. v_i is a local maximum. We apply two minimum thresholds in order to filter insignificant points: An absolute threshold, which below it we consider the peak not to be significant enough, and a relative threshold which facilitates removing points that are much lower than the highest maximum. The final step of the algorithm is to find plateaus which will also be considered part of the peak. For each peak point, the algorithm considers the point's surrounding neighbors. Using a threshold relative to the current peak, those identified as close in their value to the current peak are added to the peak list.

3.3 Learning Temporal Relatedness

We define the task of learning temporal relatedness: given two entities e_1, e_2, identify whether they relate to each other during a certain year y, i.e., whether the temporal relation (e_1, e_2, y) is true. For example, in Figure 1a, Obama was related to President in 2010 but not in 2005.

3.3.1 Specific Classifier

The first method we present to classify word relatedness, employs a classifier that receives as inputs v_i corresponding to the entities e_i and an additional feature of the year. In our evaluation (Section 4) we will use about 40 years of data, i.e. the year feature will have 40 possible values. The classifier will predict, given two entities, whether they relate to each other during a referenced year.

Let $Cl\colon \mathbb{R}^n \rightarrow \{0, 1\}$ be a classifier mapping a vector of n features $F = (f_1, \ldots, f_n)$ to a label $L \in \{0, 1\}$. Let our feature vector be of the form:

$$F = (v_1^y \| v_2^y \| y) \tag{2}$$

where v_1^y is the first entity's specific embedding (i.e. at time y), v_2^y is the second entity's specific embedding and y is the year. As a preliminary step, we need to train Cl on a diverse dataset of positive and negative examples, i.e. true and false temporal relations. For this purpose, we utilize our temporal relations dataset (Section 3.4.2). From each relation, we extract a temporal relation (e_1, e_2, y), calculate its feature vector and use it for training.

Given a new temporal relation, we apply Cl to predict whether it is true, i.e. whether its entities relate during the referenced time.

3.3.2 Temporal Classifier

Here we use a classifier similar to the one described in Section 3.3.1 (training is performed the same way), combined with input from the entities *dynamics*. First, we build the *dynamics*, i.e. build specific word embeddings of e_1 and e_2 for every year y, and calculate their cosine similarity $cos(v_1^y, v_2^y)$. We then apply our peak detection algorithm (Section 3.2) on the *dynamics* and use its output as one of the classifier's features (denoted by *isPeak*). As a result, our feature vector is:

$$F = (v_1^y \| v_2^y \| y \| isPeak) \tag{3}$$

3.4 Leveraging World Knowledge

We apply our techniques to two corpora: the first is a temporal corpora (Section 3.4.1), which we used for creating word embeddings, and the second is a relational corpora (Section 3.4.2), which we used for training our models and for evaluation.

3.4.1 Temporal Corpora

For constructing the corpora, we used The New York Times archive[2], with articles from 1981 to 2016 (9GB of text in total). Specific word embeddings were generated for every time period (i.e., year) using Word2Vec. Each year's data was used to create word embeddings using Word2Vec's skip-gram with negative sampling approach (Mikolov et al., 2013), with the Gensim library (Řehůřek and Sojka, 2010). We trained the models with the following parameters: window size of 5, learning rate of 0.05 and dimensionality of 140. We filtered out words with less than 30 occurrences during that year.

We observe both *ambiguity* (Apple, the company and the fruit) and *variability* (different phrases referring to the same entity, e.g., *President Obama*, *Barack H. Obama*, *Obama*). While such 'noise' may be problematic, both the scale of the data and stylistic standards of The New York Times help. Additionally, we ensure a connection to an entity database (Wikipedia) and perform additional cleaning methods (lemmatization, removing stopwords).

3.4.2 Relational Corpora

In this work, we use YAGO2 (Hoffart et al., 2013) as our relational corpora due to its temporal focus. The YAGO2 knowledge base contains millions of facts about entities, automatically extracted from

[2] http://spiderbites.nytimes.com/

Relation Type	Count	%
Directed	37713	47.42
HoldsPoliticalPosition	2765	3.48
IsMarriedTo	2210	2.78
PlaysFor	4376	5.50
Produced	23567	29.63
HappenedIn	8899	11.19
Total	79530	100

Table 1: Relations Dataset Composition

Wikipedia and other sources. We use relations from YAGO2 to build our own dataset of temporal relations, which we use in all of our algorithms and evaluation – as a source for temporal relations.

The dataset consists of temporal relations in the following format: *(*$entity_1$*,* $entity_2$*, year, type, class)*, where $entity_1$ and $entity_2$ are entities, $type$ is a relation type, and $class$ is true if the relation holds on $year$. For example, *(Tim Burton, Batman Returns, 1992, Directed, true)* and *(Battle Mogadishu, Somalia, 2010, HappenedIn, true)*. Table 1 shows the exact dataset composition. We built our dataset on all the relation types that have a temporal dimension in YAGO2: *Directed, HoldsPoliticalPosition, IsMarriedTo, Produced, PlaysFor, HappenedIn*. The dataset contains 80K of such relations.

4 Evaluation

4.1 Experimental Methodology

We compare the methods described in Section 3.3, where for Cl we chose to use a Support Vector Machine (SVM)[3], with an RBF kernel and C=1.0 (chosen empirically). Two baselines were used for comparison. The first is the common non-temporal model, i.e. a classifier that uses the global (all-time) word embeddings and the following features: the two entities' global embeddings, and a year. More formally,

$$F = (v_1||v_2||y) \tag{4}$$

Given a new temporal relation, the classifier predicts whether it is true during the referenced year, and we output the classifier's prediction. The second baseline we compare against is a standard text classifier that uses the global word embeddings as its only features, i.e. $F = (v_1||v_2)$.

The dataset on which we perform the evaluation is described in Section 4.2. The dataset is not balanced: it contains more negative examples than positive ones. Therefore, for evaluating the methods that involve a classifier we use stratified 10-fold cross validation. We remove relations from consideration if there is insufficient data in the corpora for that year (i.e., one of the entities was filtered out due to low incidence).

4.2 Dataset Construction

Recall that our relational corpora consists of 80K temporal relations in the following format: *(*$entity_1$*,* $entity_2$*, year, type, class)*, where $type$ is a relation type, and $class$ is true if the relation holds on $year$.

For training and evaluating our classifiers we need negative examples as well as positive examples. We generate negative examples in the following way: for every relation in the corpora, we randomly sample 10 negative examples. We exclude the years of the true examples from the dataset's year range, and then randomly choose years for the negative examples. To illustrate, let us observe the case of *Obama, President*: Obama was president from 2009-2016, so we sample negative examples from 1981 to 2008, such as *(Obama, President, 1990, HoldsPoliticalPosition, false)*. The resulting dataset contains 420K relations. We refer to it as the *Temporal Relations Dataset*[4].

4.3 Main Results

Table 2 presents the results of our experiments.

Baselines: The Global and Global+Year baselines produced an AUC of 0.55 and 0.57, respectively. They both performed much worse compared to our methods, with F1 of 0.13.

Specific Classifier produced an AUC of 0.72. It has the highest recall score of all methods (0.88), but its other scores are relatively low.

Temporal Classifier produced an AUC of 0.83. As reported in Table 2, it performed significantly better compared to all other methods, with $p < 0.05$. We applied the Wilcoxon signed-rank test to calculate statistical significance.

[3]We used the implementation by the scikit-learn library (Pedregosa et al., 2011).

[4]https://github.com/guyrosin/learning-word-relatedness

Algorithm	Acc.	Rec.	Pr.	F1	AUC
Global	0.67	0.26	0.08	0.13	0.57
Global+Year	0.52	0.39	0.08	0.13	0.55
Specific	0.52	**0.91**	0.32	0.47	0.73
Temporal	**0.81**	0.67	**0.58**	**0.62**	**0.84**

Table 2: Relatedness Learning Evaluation Results (Accuracy, Recall, Precision, F1, AUC)

4.4 Performance Analysis

We tuned Word2Vec parameters by empirically testing on a random subset of our dataset: we set the vector size to be 140 and used a minimum threshold of 30 occurrences (per year). We found that this balanced the removal of noisy data while ensuring that key entities were retained. For constructing the Word2Vec models, the amount of data is crucial or it may lead to unreliable (Hellrich and Hahn, 2016) or inaccurate results. We saw a clear correlation between accuracy and number of occurrences of a participating word. That drove our decision to evaluate our algorithms only on New York Times articles from 1981 onwards – where the number of articles per year is sufficiently large.

5 Task Example: Query Expansion

Temporal relatedness learning can be used for various NLP and IR-related tasks. For example, it is a common practice in IR to expand user queries to improve retrieval performance (Carpineto and Romano, 2012). Our technique allows us to produce temporally appropriate expansions. Specifically, given a query of n entities $Q = \{e_1, e_2, \ldots, e_n\}$, our task is to expand Q with additional search terms to add to it to improve retrieval of relevant documents.

For example, consider the query "Trump Businessman" (Figure 2). Current QE methods, which do not have a temporal aspect, focus on Donald Trump as President of the United States, a potentially erroneous result depending on the temporal focus of the searcher. A reasonable temporal expansion, might contain terms that relate to Donald Trump's business activity, such as "billionaire" or "real estate". Using our technique, the temporal focus of a query can be identified and appropriate expansions offered to the end-user. Specifically, we can analyze the relationship between the query entities to identify the "most relevant" time period – when those entities were strongly connected. Intuitively, the QE algorithms will identify the most relevant time period t for the query entities, and find semantically related terms from that time, to expand the query with.

To tackle this task, we use the algorithms described in Section 3.3. Several different algorithms can utilize the temporal relation models for the task of query expansion. Let us introduce the following definitions for our QE algorithms:

Definition 5.1. Let $NN_K^t(e)$ be the set of K terms that are the closest to an entity e in time t.

Definition 5.2. Let $NN_K(e)$ be the set of "globally" (all-time) closest terms to an entity e .

Definition 5.3. Mutual closeness between an entity x and a query Q is defined by the sum of cosine similarities between x and every $e \in Q$, i.e.

$$M_{cos}(x, Q) := \sum_{e \in Q} cos(x, e) \tag{5}$$

5.1 Query Expansion Algorithms

We describe alternative strategies to provide temporal query expansion ranging from a generic baseline to algorithms that leverage our embedding and classifiers. As a running example, we use the query: "Steven Spielberg, Saving Private Ryan" (Spielberg directed the movie in 1998). 'Reasonable' (temporally relevant) expansions for this query might include: actors who played in this movie, other Spielberg films or similar films from the same time and genre, etc.

5.1.1 Baseline

Following the results of Roy et al. (2016), we consider a baseline method that expands each entity separately, based on global Word2Vec similarity. We define the set of candidate expansion terms as

$$C = \bigcup_{e \in Q} NN_K(e) \tag{6}$$

i.e., for each entity, we choose the closest K global terms. For each $c \in C$, we compute the mutual closeness $M_{cos}(c, Q)$ and sort the terms in C on the basis of this value. The top K candidates are selected as the actual expansion terms. The baseline (poorly) expands the query "Steven Spielberg, Saving Private Ryan" with "Inglourious [Basterds], George Lucas". In some sense, one can see the relation – both are war movies, and

Lucas and Spielberg have worked together. However, Inglourious Basterds was created at 2009 and was directed by Quentin Tarantino.

5.1.2 Globally-Based Classifier

We use a heuristic and assume that the most relevant time period t for the entities of the query is the time when the entities were the closest. We use the classifier from our baseline method in Section 4.1, whose goal is to estimate how relevant a year is to a given set of n entities. Its features are the global word embeddings, as well as a year: $F = (v_1 \| v_2 \| \dots \| v_n \| y)$.

We apply the classifier to every year y, and choose the one with the highest returned probability of the true label as the most relevant time t.

$$t = \arg\max_y \{Cl(v_1, v_2, \dots, v_n, y)\} \quad (7)$$

We take as candidate-expansion terms the K closest terms to each entity from that year, separately:

$$C = \bigcup_{e \in Q} NN_K^t(e) \quad (8)$$

C is then filtered as described in the baseline.

To train the classifier, for each temporal relation in our temporal relations dataset, we calculate its feature vector and use it for training. Considering our example, this method wrongly chooses $t = 2004$. In that year the entity *Saving Private Ryan* does not exist, so we end up with a wrong expansion of "Francis Ford Coppola film".

5.1.3 Temporal Classifier

As we have seen in the previous subsection, the globally-based classifier is limited in cases where time-specific knowledge might yield better results. Thus, in this method we use the specific classifier from Section 3.3.1. Its features are the entities' specific embeddings, and a year: $F = (v_1^y \| v_2^y \| \dots \| v_n^y \| y)$. We then continue as described in the previous method (find t, choose candidate terms and filter).

For our example, this method chooses correctly $t = 1998$, which is exactly the year of *Saving Private Ryan* release. Its expansion is "Tom Hanks, Movie". Since Tom Hanks had a lead role in the movie, the expansion is reasonable. The next algorithm produces the same expansion as well.

5.1.4 Temporal Model Classifier

This method uses the temporal classifier from Section 3.3.2. Its feature vector is: $F = (v_1^y \| v_2^y \| \dots \| v_n^y \| y \| isPeak)$. The rest is the same as described in Section 5.1.3.

5.2 Query Expansion Evaluation

Dataset. To evaluate temporal query expansion we use our temporal relations dataset, which will be made publicly available (described in Section 3.4.2). First, we evaluate on queries consisting of two entities ($n = 2$): for each relation, we create a distinct query that consists of its two entities concatenated. We search The New York Times corpus with this query[5]. We compare search performance when applying the various QE methods described in Section 5.1. To evaluate the methods that involve a classifier, we use stratified 10-fold cross validation, as the previous task was evaluated (Section 4.1). We use $K = 2$ for all methods, i.e. we generate two expansion terms per query.

In addition, we evaluate on queries consisting of three entities ($n = 3$): we created a new dataset, which contains triplets of entities instead of pairs, by merging every two related (true) relations from our relations dataset. Two relations are considered related if they share an entity, and their time periods overlap. We then generate negative relations as described in Section 4.2. In this new dataset, each temporal relation consists of three entities, a year and a binary classification.

Evaluation Metrics. Though a complete evaluation of QE is beyond the scope of this paper we describe here an evaluation suited for the temporal case. It should be noted that the technique we propose here would likely be used alongside established QE techniques (e.g., log mining).

First, when providing query expansions and suggestions we would like for them to not only retrieve relevant content, but *temporally*-relevant content. To test the latter we say that given a temporal relation, a retrieved article is considered "true" if it were published within the referenced time, and "false" otherwise. Additional manual validation was done to evaluate its relevance to the query. This metric, while not the most accurate one, allows us to distinguish between results from the most relevant time period and others. Precision of the top 10 retrieved documents (P@10) is used to evaluate the retrieval effectiveness.

Results. The results of the QE evaluation are reported in Table 3. We observe a consistent be-

[5] the archive was indexed using Solr (https://lucene.apache.org/solr/)

Method	P@10	
	$n = 2$	$n = 3$
Baseline (Roy et al., 2016)	14.0%	17.7%
Globally-Based Classifier	18.0%	22.7%
Temporal Classifier	27.1%	38.5%
Temporal Model Classifier	**29.4%**	**39%**

Table 3: Results of QE Algorithms Evaluation

havior for different query sizes ($n = 2, 3$). For our temporal classifiers, for $n = 3$ there is a 30% increase in precision, compared to $n = 2$.

All of our methods performed significantly better compared to the baseline (statistical significance testing has been performed using paired t-test with $p < 0.05$). This establishes our claim that utilizing temporal knowledge yields more temporal–promising results. The *Temporal Model Classifier* showed the best performance of all. This, too, suits our claim and fits to the results from the previous task (Section 4.3).

5.3 Textual Relevance

To validate results, we compared our query expansion algorithms' performance on different relation types and found big differences. On the relations *HoldsPoliticalPosition, HappenedIn* and *IsMarriedTo*, the temporal algorithms achieved around 50% accuracy, while on *Directed* and *Produced* they got only 20%. This difference is reasonable, as our models were built upon a news corpus.

Let us observe an example of using the QE algorithms with the query "Vicente Fox President" (Fox was president of Mexico from 2000 to 2006). The baseline expands with Mexico's two previous presidents (Zedillo and Salinas). This makes sense as the baseline doesn't take time into account. The globally-based classifier expands with Roh Moohyun, who was president of Korea during the same time period. *Temporal Classifier* expands with "Ricardo Lagos, National Action Party" (Lagos was president of Chile during that time. The latter is Fox's political party). The *Temporal Model Classifier* expands with 'presidential' and Francisco Labastida (the candidate who lost the elections to Fox).

Figure 3 shows the similarity between Apple and its top products since 1990. We can infer which products were the most significant at each time. Take as example the query "Apple Steve Jobs". Using our technique, we can find the most relevant time period for this query, which is from Apple's foundation in 1976 until Jobs' death in 2011. Leveraging Figure 3, we can expand this query focusing on the most popular products of that time.

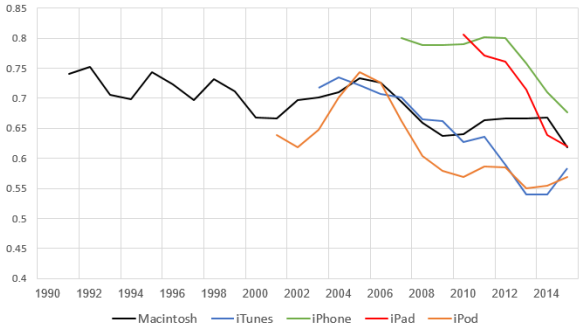

Figure 3: Similarity between Apple and its top products over time. The y-axis is cosine similarity.

6 Conclusions

We believe that as corpora evolve to include temporally-varying datasets, new techniques must be devised to support traditional and new IR methods. In this paper, we introduced a novel technique for extracting relations in temporal datasets. The technique is efficient at large scales and works in an unsupervised manner. Our experiments demonstrate the viability of the extraction technique as well as describing ways that it can be used in downstream applications. We specifically demonstrate a number of query expansion algorithms that can benefit from this technique.

References

Eytan Adar, Jaime Teevan, Susan T. Dumais, and Jonathan L. Elsas. 2009. The web changes everything: understanding the dynamics of web content. In *WSDM'09*.

S. M. Beitzel, E. C. Jensen, A. Chowdhury, D. Grossman, and O. Frieder. 2004. Hourly analysis of a very large topically categorized web query log. In *Proceedings of the Special Interest Group on Information Retrieval (SIGIR)*.

Adrian Bingham. 2010. the digitization of newspaper archives: Opportunities and challenges for historians. *Twentieth Century British History*, 21(2):225–231.

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating wordnet-based measures of lexical semantic

relatedness. *Computational Linguistics*, 32(1):13–47.

Claudio Carpineto and Giovanni Romano. 2012. A survey of automatic query expansion in information retrieval. *ACM Comput. Surv.*, 44(1).

S. Chien and N. Immorlica. 2005. Semantic similarity between search engine queries using temporal correlation. In *WWW*, pages 2–11.

Fernando Diaz, Bhaskar Mitra, and Nick Craswell. 2016. Query expansion with locally-trained word embeddings. In *ACL'16*.

William L. Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. In *ACL'16*.

Johannes Hellrich and Udo Hahn. 2016. Bad company - neighborhoods in neural embedding spaces considered harmful. In *COLING*.

Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61.

Adam Jatowt and Kevin Duh. 2014. A framework for analyzing semantic change of words across time. In *JCDL '14*.

Rosie Jones and Fernando Diaz. 2007. Temporal profiles of queries. *ACM Transactions on Information Systems*, 25.

Tom Kenter, Melvin Wevers, Pim Huijnen, and Maarten de Rijke. 2015. Ad hoc monitoring of vocabulary shifts over time. In *CIKM '15*.

Anagha Kulkarni, Jaime Teevan, Krysta M. Svore, and Susan T. Dumais. 2011. Understanding temporal query dynamics.

Erdal Kuzey and Gerhard Weikum. 2012. Extraction of temporal facts and events from wikipedia. In *2nd Temporal Web Analytics Workshop*, pages 25–32. ACM.

Saar Kuzi, Anna Shtok, and Oren Kurland. 2016. Query expansion using word embeddings. In *CIKM'16*.

Xiao Ling and Daniel S Weld. 2010. Temporal information extraction. In *AAAI*, volume 10, pages 1385–1390.

Z. Vagena M. Vlachos, C. Meek and D. Gunopulos. 2004. Identifying similarities, periodicities and bursts for online search queries. In *SIGMOD*.

Rada Mihalcea and Vivi Nastase. 2012. Word epoch disambiguation: Finding how words change over time. In *ACL'12*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Sunny Mitra, Ritwik Mitra, Martin Riedl, Chris Biemann, Animesh Mukherjee, and Pawan Goyal. 2014. That's sick dude!: Automatic identification of word sense change across different timescales. In *ACL'14*.

G Palshikar et al. 2009. Simple algorithms for peak detection in time-series. In *Proc. 1st Int. Conf. Advanced Data Analysis, Business Analytics and Intelligence*, pages 1–13.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Octavian Popescu and Carlo Strapparava. 2013. Behind the times: Detecting epoch changes using large corpora. In *IJCNLP'13*.

James Pustejovsky, José M Castano, Robert Ingria, Roser Sauri, Robert J Gaizauskas, Andrea Setzer, Graham Katz, and Dragomir R Radev. 2003. Timeml: Robust specification of event and temporal expressions in text. *New directions in question answering*, 3:28–34.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A word at a time: Computing word relatedness using temporal semantic analysis. In *Proceedings of International World Wide Web Conference (WWW)*.

Kira Radinsky, Krysta Svore, Susan Dumais, Jaime Teevan, Alex Bocharov, and Eric Horvitz. 2012. Modeling and predicting behavioral dynamics on the web. In *Proceedings of International World Wide Web Conference (WWW)*.

Kira Radinsky, Krysta M. Svore, Susan T. Dumais, Milad Shokouhi, Jaime Teevan, Alex Bocharov, and Eric Horvitz. 2013. Behavioral dynamics on the web: Learning, modeling, and prediction. *ACM Transactions on Information Systems*, 31(3):16.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modeling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA.

Dwaipayan Roy, Debjyoti Paul, Mandar Mitra, and Utpal Garain. 2016. Using word embeddings for automatic query expansion. *arXiv preprint arXiv:1606.07608*.

Eyal Sagi, Stefan Kaufmann, and Brady Clark. 2009. Semantic density analysis: Comparing word meaning across time and phonetic space. In *Workshop on Geometrical Models of Natural Language Semantics*.

Yair Shimshoni, Niv Efron, and Yossi Matias. 2009. On the predictability of search trends. In *Technical Report*.

Milad Shokouhi. 2011. Detecting seasonal queries by time-series analysis. In *Proceedings of the Special Interest Group on Information Retrieval (SIGIR)*.

Milad Shokouhi and Kira Radinsky. 2012. Time-sensitive query auto-completion. In *Proceedings of the Special Interest Group on Information Retrieval (SIGIR)*.

Jaspreet Singh, Wolfgang Nejdl, and Avishek Anand. 2016. History by diversity: Helping historians search news archives. In *CHIIR'16*, pages 183–192.

Partha Pratim Talukdar, Derry Wijaya, and Tom Mitchell. 2012. Coupled temporal scoping of relational facts. In *5th ACM international conference on Web search and data mining*, pages 73–82. ACM.

Peter D Turney, Patrick Pantel, et al. 2010. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188.

Naushad UzZaman, Hector Llorens, James Allen, Leon Derczynski, Marc Verhagen, and James Pustejovsky. 2012. Tempeval-3: Evaluating events, time expressions, and temporal relations. *arXiv preprint arXiv:1206.5333*.

P. Wang, M. W. Berry, and Y. Yang. 2003. Mining longitudinal web queries: trends and patterns. *Journal of the American Society for Information Science and Technology (JASIST)*, 54:743–758.

Derry Tanti Wijaya and Reyyan Yeniterzi. 2011. Understanding semantic change of words over centuries. In *2011 International Workshop on DETecting and Exploiting Cultural diversiTy on the Social Web*, DETECT '11.

Craig Willis and Miles Efron. 2013. Finding information in books: Characteristics of full-text searches in a collection of 10 million books. *Proceedings of the American Society for Information Science and Technology*, 50(1):1–10.

Qiankun Zhao, Steven C. H. Hoi, and Tie yan Liu. 2006. Time-dependent semantic similarity measure of queries using historical click-through data. In *Proceedings of International World Wide Web Conference (WWW)*.