Generating Timelines by Modeling Semantic Change

Guy D. Rosin[1], Kira Radinsky[1,2]
[1]Technion – Israel Institute of Technology, Haifa, Israel
[2]eBay Research, Israel
{guyrosin,kirar}@cs.technion.ac.il

Abstract

Though languages can evolve slowly, they can also react strongly to dramatic world events. By studying the connection between words and events, it is possible to identify which events change our vocabulary and in what way. In this work, we tackle the task of creating timelines—records of historical 'turning points', represented by either words or events, to understand the dynamics of a target word. Our approach identifies these points by leveraging both static and time-varying word embeddings to measure the influence of words and events. In addition to quantifying changes, we show how our technique can help isolate semantic changes. Our qualitative and quantitative evaluations show that we are able to capture this semantic change and event influence.

1 Introduction

Languages respond to world events in many ways. New words, phrases, and named entities are created, new senses may develop, and valences may change. Various approaches support the study of historical linguistics (e.g., comparative linguistics, etymology, etc.). In this work, we focus on a specific process for tracking the progression of meaning over time in sense, semantics, and in relation to other words and concepts. By leveraging changing relationships in temporal corpora, we demonstrate a way of 'embedding' words and world events. Observing changes in this embedding allows us to construct timelines that support the study of evolving languages.

The timeline of scientific and technical discoveries, for example, can drive the emergence of new word senses as these discoveries are 'named'. Take the word "cell" which evolved from its 12^{th} century meaning (a small room or chamber) to a new sense in the 17^{th} century (a basic unit of an organism) to the 19^{th} century meaning (an electric battery) and most recently to a shorthand for a mobile phone[1]. Critically, the *dominant senses* of a word vary over time as some meanings become less commonly used while others gain in popularity. This dynamic need not be driven only by the addition of certain senses. The prevalence of a hyponym, for example, may also drive a change in the ranking of senses. The word 'disaster' may call to mind very different things depending on the latest *type* of disaster. Thus, the word may evoke 'nuclear disaster' in a reader in 2011 (e.g., driven by the *Fukushima* incident). However, in 2012 the 'storm' sense may be more salient (e.g., driven by *Superstorm Sandy*).

Evolution of senses is but one way a language can evolve. Broader semantic changes can also occur. For example, the valence of the word may move or even flip (e.g., *terrific* or *bully*). Of particular interest to us are those changes that are more immediate and precipitated by key world events. For example, a war may lead certain terms to take on a negative connotation as a country or people become the 'enemy'. Large collections of text from a given period can capture all of these language changes as reflected by evolving context. By mining this text, our goal is to support the study of evolving languages.

Etymological studies allow us to understand the origin of words and changes in meaning (Alinei, 1995). This work produces not only an accounting of change but also an explanation of the social, scientific, or other world events that drive language shifts. Conventional production of etymological analysis often requires a detailed and laborious manual close-reading of historical texts (Geeraerts et al., 1997). By applying computational methods, our focus is on detecting semantic changes of words and events and producing possi-

[1]https://www.etymonline.com/word/cell

Proceedings of the 23rd Conference on Computational Natural Language Learning, pages 186–195
Hong Kong, China, November 3-4, 2019.

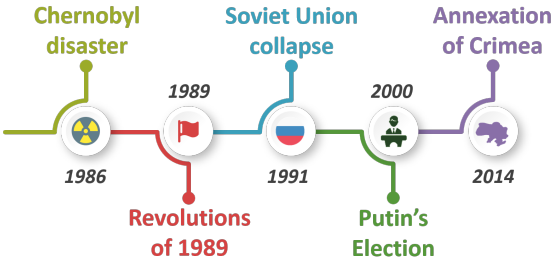

Figure 1: A timeline generated by our framework for *Russia*.

ble explanations from real-world drivers.

Problem Definition: In this work, we study the problem of timeline generation for a word or phrase. We use the term 'word' throughout the paper for convenience. Given a timeline of a word, a researcher should be able to understand the word's dynamics, i.e., the changes the word underwent over time. A timeline is defined as a sequence of time points and their descriptors (i.e., explanations of the changes the word underwent at that time point). A good timeline is such that enables the researcher to gain a better understanding of the word and its history (Althoff et al., 2015). It contains time points of significant changes, with relevant explanations of the changes, and with a minimal number of missing or redundant information.

We define several building blocks for constructing timelines. The first is the identification of time points during which the target word underwent significant semantic change (we refer to those as *Turning Points*, see Section 4). Second, we consider identifying associated descriptors of those changes. These descriptors are associated with a word's change at a particular time and can serve to explain its dynamics.

We experiment with two types of descriptors. The first involves *words* associated with the target word or affected by it (Section 5). The above 'cell' and 'disaster' examples can serve as examples of timelines with word descriptors. The second type is *events* (Section 6). One can explain changes the target word underwent based on significant world events. As an example, consider the timeline generated by our framework for the word "Russia" (Figure 1).

To identify events that are strongly associated with the change, we utilize time-varying language embeddings on both static snapshots (e.g., Wikipedia) and historical texts (35 years of the *New York Times*), allowing us to capture both syntactic and semantic variation of words (Section 3). We develop a mechanism for simultaneously embedding words and events in the same space (Section 6.1). We present several methods to leverage those embeddings for key historical events detection by evaluating the distance between words and events (Sections 6.2, 6.3).

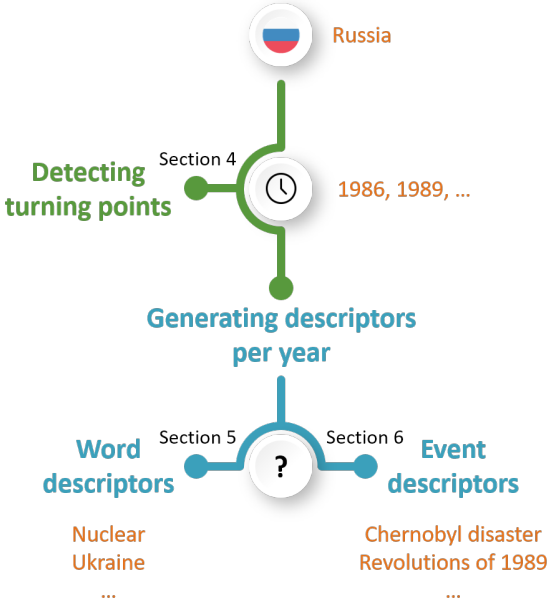

Figure 2: Flow diagram of timeline generation. The two basic building blocks are detecting turning points and generating descriptors, where the descriptors can be either words or events.

Figure 2 presents the flow of the paper through an example. Consider the word 'Russia'. First, we identify its turning points (Section 4), and then generate descriptors – either words (Section 5) or events (Section 6) – to construct its timeline. We contribute several algorithms for identifying significant events leveraging various types of embeddings, including a supervised learning approach.[2]

2 Related Work

Semantic Change: Most work on language evolution has focused on identifying semantic drifts and word meaning changes (see Kutuzov et al. (2018) for a recent survey). Various approaches have pursued the task of detecting changes in word meaning (Sagi et al., 2009; Mitra et al., 2014; Wijaya and Yeniterzi, 2011; Mihalcea and Nastase, 2012; Popescu and Strapparava, 2013; Jatowt and Duh, 2014; Kenter et al., 2015; Hamilton et al., 2016;

[2]Code and data available at https://github.com/guyrosin/generating_timelines

Azarbonyad et al., 2017). Specific approaches include: dynamic embedding models using a probabilistic Bayesian version of Word2Vec (Bamler and Mandt, 2017), pointwise mutual information (PMI) (Yao et al., 2018), and exponential family embeddings (Rudolph and Blei, 2018). Relatedness over time between words has also been studied. Radinsky et al. (2012) showed that words that co-occur in history have a stronger relation, Rosin et al. (2017) introduced the supervised task of temporal semantic relatedness, and Orlikowski et al. (2018) studied diachronic analogies. In our work, we focus on the world events behind the semantic changes—and isolate those events that co-occur with significant language change.

Change Detection for Semantic Shift Analysis: Detecting major changes involves detecting continuous peaks in time series. Kulkarni et al. (2015) and Basile et al. (2016) offered a mean shift model, and Rosin et al. (2017) used a threshold-based method for this task. We utilize the latter approach as it is simpler and more computationally effective.

Timeline Generation: Past work focused on generating timelines by leveraging information retrieval methods. Examples include the use of Facebook data (Graus et al., 2013), Twitter (Li and Cardie, 2014), and Wikipedia (Tuan et al., 2011) to generate context-aware timelines by ranking related entities by co-occurrence with the main timeline entity. Althoff et al. (2015) created timelines by mining a knowledge base, based on submodular optimization and web-co-occurrence statistics. Shahaf and Guestrin (2010) generated a chain of events connecting two news articles. Our work differs from the prior work in several ways. First, we consider the semantic changes a word undergoes and detect the events that influenced them. We study several word embeddings to measure change and relatedness between words and events over time and construct a timeline.

3 Event and Temporal Word Embedding

We consider several methods to represent events and words. These are used to identify semantic changes and generate timeline descriptors. To capture both words and events in the same space we consider *global embeddings*, which are created upon the English Wikipedia (see Section 7.1).

We also utilize the work of Rosin et al. (2017) for *temporal word embeddings*. Timeline construction requires modeling changes in words and events over time. When looking at a specific word, we wish to focus on its relevant meaning at a particular time. Thus, the *temporal word embeddings* are created using data from a large temporal corpus. Specifically, we leverage the New York Times (NYT) archive. The embeddings are generated for every time period (i.e., year) and enable us to investigate how words meanings and relatedness between words change over time (see Section 7.1).

Finally, in order to compare vectors of the same word in different, independently-trained, vector space models, we align every pair of models using Orthogonal Procrustes (Hamilton et al., 2016).

We use the following notations throughout the paper:

Notation 3.1. v_w is the vector representation of a word w.

Notation 3.2. v_w^t is the vector representation of a word w during time t.

Notation 3.3. $NN_k(w)$ is the set of k-nearest neighbors (kNN) of a word w.

Notation 3.4. $NN_k^t(w)$ is the set of k-nearest neighbors (kNN) of a word w during time t.

Notation 3.5. cos is cosine similarity, which we use as a similarity function between embeddings.

4 Timeline Turning Points

A timeline is composed of time points that identify the changes a word underwent. We refer to those as *Turning Points*, and experiment with several methods to identify them. Formally, let w be a target word, and t be a time point. Each method approximates the probability $d_t(w)$ of t to be a turning point of w. The turning points are then selected by performing peak detection (Rosin et al., 2017) on the series of $d_t(w)$ for every t. We experiment with two methodologies for turning point detection, leveraging the embeddings we introduce in Sections 3, 6.1:

(1) Neighborhood: Changes in the neighborhood of a word over time can be used to capture semantic changes of the word. This method measures the difference between the similar words sets of w between two consecutive years. Formally:

$$d_t(w) = 1 - \frac{\left|NN_k^t(w) \cap NN_k^{t-1}(w)\right|}{k} \quad (1)$$

where $NN_k^t(w)$ is the set of k-nearest neighbors (kNN) of w during time t.

(2) EmbeddingSimilarity: Employing word embeddings, we can also look at the change in the embedding vectors of w:

$$d_t(w) = 1 - cos\left(v_w^t, v_w^{t-1}\right) \quad (2)$$

5 Word Descriptors

The second building block of a timeline is *explaining* what triggered the semantic changes. We refer to such explanations as *descriptors*. One can explain semantic changes with words—significant associated words or terms that correlate to the target word's meaning drift. Letting w be the target word, and t be a time point, we are looking for words that became closer to w in vector space during t. Specifically, we look at the nearest neighbors of w at times t and $t-1$, and denote the set of descriptors by D_t:

$$D_t(w) = NN_k^t(w) \setminus NN_k^{t-1}(w) \quad (3)$$

For example, using this method for 'Russia' results in *Soviet Union* and *Soviet* for 1989, when the Soviet Union was dissolving, and *Ukraine*, *Kyrgyzstan*, and *Latvia* for 1990, when these countries attempted to gain independence from the Union.

6 Event Descriptors

As an alternative for word descriptors, we consider event descriptors. To generate these, given a target word w, our task is to identify its change in time t by a set of significant events E that likely affected w. In this section, we first describe the embeddings we use (Section 6.1) and then present several methods for detecting significant events (Sections 6.2, 6.3).

6.1 Projected Embedding for Events

Temporal word embeddings (Section 3) are created for every time period. They enable us to investigate how word meaning and relatedness between words change over time. However, as it may take some time until an event's name is determined and referred to in newspapers, the paper's text may not have meaningful embeddings for those events. For example, the name "World War I" was used only after WWII started. As a result, we are not able to compare events and words.

To address this problem, we leverage the global embeddings (Section 3). Since Wikipedia articles typically contain balanced descriptions of events, they can be a proper basis for event embeddings. Recall that the way these embeddings are created enables creating a common latent space for *both* words and concepts (and specifically, events), allowing us to compare both at the same time. Our solution involves projecting the global model on each temporal one. This way, we create a joint vector space for words and events, which represents a specific time period. We refer to these embeddings as "*Projected Embeddings*".

We assume that most words' meanings do not change over time and learn a transformation of one embedding space onto another, minimizing the distance between pairs of points. Let us define $W_{wiki} \in \mathbb{R}^{|V_{wiki}| \times d_{wiki}}$ as the matrix of embeddings learned from Wikipedia, where d_{wiki} is the embedding size and $|V_{wiki}|$ is the vocabulary size of Wikipedia. Similarly, $W_{nyt}^{(t)} \in \mathbb{R}^{|V_{nyt}^{(t)}| \times d_{nyt}^{(t)}}$ is the matrix of embeddings learned from the NYT at time t, where $d_{nyt}^{(t)}$ is the embedding size and $|V_{nyt}^{(t)}|$ is the vocabulary size of the NYT at time t. We seek a matrix $W^{(t)} \in \mathbb{R}^{|V_{wiki}| \times d_{nyt}^{(t)}}$ that will contain the transformation of W_{wiki} to $W_{nyt}^{(t)}$ for time t. By making an additional simplifying assumption that the vector spaces are equivalent under a linear transformation, we are able to find $W^{(t)}$ by optimizing the following linear regression model:

$$\arg\min_T \sum_{w_i \in V_{wiki} \cap V_{nyt}^{(t)}} \left\| W_{wiki}(w_i)T - W_{nyt}^{(t)}(w_i) \right\|_2^2 \quad (4)$$

where $T \in \mathbb{R}^{d_{wiki} \times d_{nyt}^{(t)}}$. We then obtain the projected matrix:

$$W^{(t)} = W_{wiki}\widehat{T}^{(t)} \quad (5)$$

where $\widehat{T}^{(t)}$ is the result of Equation 4. Similar methods were used in the field of temporal semantics to align embeddings of different time periods to a unified coordinate system (Szymanski, 2017; Kulkarni et al., 2015).

6.2 Similarity-Based Event Detection

We hypothesize that the events closest in vector space to a word should be the most significant to its timeline. We experiment with two score functions that are based on semantic similarity. The descriptors are chosen as the top-scoring events.

ByWord: Given a target word w, we look for its closest events in a specific time. We define a score function of an event e: $score(e) = cos(v_w, v_e)$ where cos is cosine similarity, and v_w and v_e are the respective embeddings of w and e.

ByKNN: We wish to extend the "impact circle" of the event, as events sometimes do not affect a word directly but through other words. We look for events that are closest not only to the target word but also to its neighbors. We calculate the following score for every possible event e:

$$score(e) = avg(\{cos(v_n, v_e) : n \in \{w\} \cup NN_k(w)\}) \quad (6)$$

6.3 Supervised Event Detection

The previous methods discuss only the semantic similarity of words and events, where we are actually interested in the probability of events to affect words. There are often multiple possible events that can act as explanations. Consider the following example. In 2010, several events related to Russia happened: New START (nuclear arms reduction treaty between the USA and Russia) was signed, there was a Winter Olympics, and the ROKS Cheonan (a South Korean warship) sunk. All relate to Russia, but only one appears to indicate a meaningful change to it—the New START event. Identifying the right events is a highly challenging task, as most usually all the candidate events are impactful events, related to the target word, and have an impact on various words due to their significance. As another example, Nelson Mandela's death is semantically the closest event to the word 'Clinton' during 2013, based on our projected embeddings. Though it is surely a powerful event and one that is related to many world leaders, it is hard to describe it as a turning point for either Bill or Hillary Clinton. Therefore, we attempt to *learn* which are the most relevant events for a given word. We present a classifier that receives an event and a word, and outputs the probability this event affected that word. This classifier functions as a predictor of the probability of an event e to cause a semantic change of a word w.

Training Data

To create training data for the classifier, we can use any embedding model that embeds both events and words, namely the global or the projected embeddings (Section 3 and 6.1). Given an event, we find terms affected by it and terms that are not.

Affected Terms: We limit the set of possible affected terms by an event and consider semantically similar terms to the event (we consider cosine similarity > 0.3). A term is considered to be affected by an event if the term's meaning changed during the time of the event, and did not change in the year before the event.

Unaffected Terms: Given an event, we sample terms from its semantically similar terms that were changed in the years *after* the event, and were not changed during the year it happened. This way, we try to capture terms that are related to the event but were changed due to other reasons.

Machine Learning Approach

We consider several supervised machine learning approaches, experimenting with random forest, SVM, neural networks, etc. We also devise several features leveraged by our classifiers:

- v_e and v_w, i.e., the embeddings of the event e and the word w. Any embedding model (Sections 3, 6.1) can be used.
- The semantic similarity between v_e and v_w.
- Categories of the event e, taken from DBpedia and represented using bag-of-words. Each event is associated with one or more categories in DBpedia (e.g., social event, sports, military conflict). The bag-of-words vector comprises the top 150 categories.
- Features that indicate the event's popularity: number of internal and external links in e's Wikipedia page, pageviews count of e's Wikipedia page[3]. Intuitively, a popular event might be impactful.

7 Experimental Setup

We briefly describe our dataset and embeddings before focusing on the evaluation.

7.1 Implementation Details

Embeddings: The global embeddings were created based on the Wikipedia dump of May 2016, using Word2Vec's skip-gram with negative sampling, with a window size of 10. Following Sherkat and Milios (2017), we perform a preprocessing step necessary for the embedding process to capture Wikipedia concepts and not just words: each inner link in the text (i.e., a link to a Wikipedia article) is replaced by an ID, so that every link to the same page is replaced by this page's ID. After filtering low-frequency words and concepts, we find 3.2M unique embeddings, of which 1.7M are concepts.

[3]We used Wikimedia Foundation's API to get pageviews of a single month (specifically October 2017).

For constructing temporal embeddings, we used the NYT archive[4], with articles from 1981 to 2016 (9GB of text in total). For each year of content, we created embeddings using Word2Vec's skip-gram with negative sampling, with a window size of 5 and a dimensionality of 140, using the Gensim library (Rehurek and Sojka, 2010). We filtered out words with less than 50 occurrences.

Events Data: We used DBpedia and Wikipedia as sources for events. First, we mined entities from DBpedia whose type is 'event' (i.e., `yago/Event100029378`, `Ontology/Event`), and that have an associated Wikipedia page and associated year of occurrence. Second, we mined 50K events from Wikipedia's monthly events pages[5] and retained the 30K events corresponding to our focus time period of 1981 to 2016. In this work, we focus on large, significant events, since these have the most potential to affect language (Chieu and Lee, 2004). Thus, we retained events with over 6000 monthly views (in October 2017) and over 15 external references. Our final dataset contains 1233 events, most related to armed conflicts, politics, disasters, and sports.

Classifier Dataset: To construct a "ground truth" dataset for our classifier, we find pairs of events (from the events dataset) and their affected and unaffected terms, as described in Section 6.3. For each event, we use either the global or projected embeddings to find 20 affected terms (there may be less, depending on the sensitivity of the change detection algorithm) and 20 unaffected terms. The dataset contains 21K pairs in total.

7.2 Experimental Methodology

We perform both qualitative and quantitative evaluations. First, we conduct user studies to capture the utility of our algorithms as they might be used in practice (e.g., in search engine results). Twenty evaluators participated using real events data (Section 7.1). We select a set of target words based on two criteria: popularity—so that most evaluators would know them and preferably parts of their history; and the number of significant related events—so that meaningful timelines would be produced. In practice, we selected 30 target words based on popularity (as expressed in the number of page views of the corresponding Wikipedia page).

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

[5] For example, https://en.wikipedia.org/wiki/Category:February_1992_events

For a given target word, evaluators were presented with several timelines created by our algorithms and baseline methods (see below). While we evaluated both word and event descriptors for timelines, our evaluation is focused on events, as they proved to be more meaningful and interesting to the evaluators (see Section 8.2). Each timeline was accompanied by detailed descriptions and references. Our evaluators were asked to indicate whether an event was correct ('true') in its placement in the timeline and whether it was likely to have an impact on the target word. See Appendix A for a screenshot of the questionnaire used in the evaluation. Overall timeline quality was evaluated as described below.

Evaluation Metrics

Each timeline is evaluated using the following metrics (timelines with word descriptors are evaluated similarly—replacing 'event' with 'word'):

Accuracy: Fraction of events that are relevant to the target word (i.e., marked as true by the evaluators). $\frac{\#\text{true events}}{\#\text{true events}+\#\text{false events}}$

Relevance: How relevant the timeline is to the word. This is meant to approximate the precision metric. Relevance was indicated as a rank score on a scale from 1 to #timelines (#timelines being the number of timelines presented for the given word) and normalized as: $\frac{\text{relevance score}}{\#\text{timelines}}$

Missing Events: Evaluators were asked how many events they believed were missing from the timeline. We then normalize by the total number of events in the timeline. Intuitively, this measure is meant to approximate *1-recall*. $\frac{\#\text{missing events}}{\#\text{events in timeline}}$

Redundancy: Evaluators were asked how many events in the timeline are redundant (normalized by the total number of events in the timeline): $\frac{\#\text{redundant events}}{\#\text{events in timeline}}$

Ranking: Evaluators were asked to subjectively rank presented timelines from best to worst.

Effectiveness: Evaluators were asked to indicate their familiarity with the event history both before and after seeing the timeline. The difference between the two scores indicated the 'effectiveness' (as a soft measure of whether the timeline contained anything surprising or novel). The pre-evaluation score also served to measure the evaluator's familiarity with the topic.

Methods Compared

We perform experiments for the two building blocks of a timeline. First, we compare methods

of identifying turning points (Section 4). We approximate a gold standard for turning points by having evaluators mark years in which there is a turning point. This is determined by looking at all the events that took place during a specific year and approximating whether any of them is significant to the target word. We compare the identified years of our methods to this gold standard.

Second, we evaluated different ways to produce descriptors[6]:

WordDescriptors (Section 5): We use words as descriptors. For every year t, we select words that were added to the kNN of the target word during t (with $k = 20$, chosen empirically).

BaseEvents: A baseline method for event descriptors. We select events 'close' to the target word as descriptors—as measured by the frequency of the target word's appearance on the event's Wikipedia page.

WikiTimelines: Finally, we extract timelines from crowd-created Wikipedia timeline pages[7].

These baselines were compared to our algorithmic techniques:

ByWord and **ByKNN** are described in Section 6.2.

ByKNNGlobCls: We first use *ByKNN* to find 30 close events (determined empirically), and then use the events classifier (Section 6.3) to predict each event's influence. We rank the events by a combination of this prediction and the score of *ByKNN*. The classifier is trained on a dataset that was created using the global embeddings.

ByKNNCls: Similar to *ByKNNGlobCls*, with one difference: here the classifier's training set was created using the projected embeddings.

8 Results

8.1 Turning Point Evaluation

Comparing the two methods for turning point detection, we observed that 23% of the years detected by *Neighborhood* are false, compared to 15% by *EmbeddingSimilarity*. We believe this difference is due to the *Neighborhood* method capturing only the local neighborhood of the word, while an embedding can be more meaningful. For example, a word can change semantically not by altering its neighbors, but by moving in space towards other meanings, together with its neighbors.

8.2 Timeline Evaluation

We present the results of the timeline evaluation (Table 1), where the turning points are detected using the *EmbeddingSimilarity* method, as it reached the highest empirical performance (Section 8.1).

The *WordDescriptors* method achieves poor results, as expected. It is inaccurate and contains many redundant descriptors. For example, it generated the following descriptors for the word "Terror" during 2012-2014: Islamic terror, brutality, genocidal. They are all related to terror but do not help us deduce why the word 'Terror' was impacted, or what happened. Alternatively, the *ByWord* method performs much better. Looking at its generated timeline for 'Terror', we observe that it successfully identifies highly relevant events: the Benghazi attack (a terror attack against US government facilities in Libya), the mass shooting at Westgate Shopping Mall in Kenya, and the international military intervention against ISIL (the Islamic State organization). We find that *ByKNN* results and performance are similar to *ByWord*. As both methods consider the similarity between an event and a target word, we conjecture that the similarity function has a less impact on the timeline generation process. We empirically observe that in most cases a word close to an event would also be close to its neighbors.

The *WikiTimelines* method has the top accuracy. Given these timelines are manually created by domain experts, this is unsurprising. Nonetheless, the *ByKNNCls* method wins the three most important metrics: relevance, ranking, and effectiveness. Thus in the eyes of the evaluators, this method gives the most relevant timelines compared to all others, and maybe most importantly, provides novel information. As an example, the timeline created by *ByKNNCls* for 'Russia' contains several significant events that are missing from the same timeline created by *WikiTimelines*, such as Chernobyl disaster, the Revolutions of 1989 and the Dissolution of the Soviet Union.

The *ByKNNCls* method is more accurate than the other embedding-based methods, likely because it can filter out events that are identified by other methods but are in fact not impactful for the particular target word. However, it has a higher *Missing Events* score—suggesting true events are occasionally filtered out as well.

[6]Refer to https://github.com/guyrosin/generating_timelines for the source code.

[7]For example, https://en.wikipedia.org/wiki/Timeline_of_Russian_history

Method	Accuracy	Relevance	Missing	Redundancy	Ranking	Effectiveness
WordDescriptors	0.43	0.32	0.65	0.3	0.28	0.03
BaseEvents	0.49	0.76	0.04	**0.03**	0.72	0.23
WikiTimelines	**0.81**	0.67	**0.03**	**0.03**	0.65	0.07
ByWord	0.60	0.86	0.09	0.06	0.78	**0.33**
ByKNN	0.61	0.81	0.12	0.08	0.80	0.31
ByKNNGlobCls	0.63	0.62	0.38	0.1	0.53	0.17
ByKNNCls	0.67	**0.89**	0.20	0.09	**0.86**	**0.33**

Table 1: Timelines evaluation results. Methods and metrics described in Section 7.2.

To measure the correspondence between evaluators' answers, we calculated Kendall's Tau, which resulted in an average value of 0.6.

8.3 Events Classifier Evaluation

We experiment with several supervised approaches for the events classifier (Section 6.3) and evaluate their performance using stratified 10-fold cross-validation. In this evaluation, the projected embeddings were used to create the training and test sets and for creating the classifier's features, since this configuration was found to result in the best performance (Section 8.4). Specifically, the parameters (optimized using grid search) and the AUC are as follows:

Logistic regression produced an AUC of 0.75. **SVM** with RBF kernel and C=1.0 produced 0.97. **Random Forest** classifier with 800 trees produced 0.97 as well. **Neural Network** with a single hidden layer of 100 neurons and Adam as the optimization algorithm achieved the best performance, with an AUC score of 0.98.

8.4 Projection Contribution

We measure the embeddings projection's contribution (Section 6.1) to the tasks of timeline generation and learning influence of events on words.

Timeline Generation Performance

We refer the reader to Table 1 to discuss the comparison between *ByKNNCls* and *ByKNNGlobCls*. These methods are almost identical—both use our classifier for detecting significant events. They differ in how the classifier is trained. *ByKNNGlobCls*'s training set is created using global embeddings, while *ByKNNCls*'s training set is created using projected embeddings. We observe a significant difference in the performance of these two methods. *ByKNNCls* achieves the best performance of all methods. It has far fewer false negatives than *ByKNNGlobCls* and higher accuracy. The other important metrics—relevance, ranking, and effectiveness—show improved performance as well. We conclude that representing events using the projected embedding brings high performance boosts for this task.

Embeddings	Acc.	Rec.	Prec.	F1	AUC
Glob/Glob	0.63	0.66	0.62	0.64	0.69
Glob/Proj	0.74	0.73	0.74	0.74	0.81
Proj/Glob	0.76	0.76	0.75	0.76	0.86
Proj/Proj	**0.94**	**0.94**	**0.94**	**0.94**	**0.98**

Table 2: Classifier performance using global and projected embeddings for creating its features/dataset.

Events Classifier Performance

To better understand the embedding features on the events classifier's performance (Section 6.3) we compared the impact of global and projected embeddings. Additionally, the training data of the classifier can be generated using any embedding. Thus, we perform an empirical evaluation comparing all combinations of embeddings for representing the *features* and the *training data* employed by the classifier (Table 2). Using global embeddings, we find an AUC of 0.69. Using global embeddings for the features and projected embeddings for the dataset, or vice versa, resulted in AUC of around 0.84. Using the projected embeddings for both yielded an AUC of 0.98—significantly better than any other combination—with $p < 0.05$ (using a Wilcoxon signed-rank test).

8.5 Events Classifier Contribution

Our main goal in developing the classifier (Section 6.3) was to enable us to identify the relevant events that affect a given word. As presented in Table 1, the main drawback of *ByWord* and *ByKNN* is a high false positive ratio given

this task. The *ByKNNCls* method is more accurate than the other embedding-based methods, probably due to the classifier filtering out many events that are identified by the other methods but are in fact not impactful for the particular target word. For example, we observe several false events that appear in the 'Israel' timeline that was generated by the *ByKNN* method and are all ignored by *ByKNNCls*. Each one of them is related to Israel, but not significant enough to make an impact on it, e.g., the *anthrax letters* (2001) had "death to Israel" written inside them. *Hurricane Katrina* (2005) brought Israel to send a humanitarian aid delegation to New Orleans. Furthermore, the decrease in the false positive ratio results in high ratings of the *ByKNNCls* timelines. Looking at the results in Table 1, we observe significant differences in multiple metrics: relevance, ranking, and effectiveness (tested using paired t-test with $p < 0.05$).

8.6 Discussion

Three main factors seem to affect the performance of our approach. First, ambiguity harms performance. For creating timelines for ambiguous words, a contextual embedding approach would be necessary. We leave that for future work. For example, ambiguous target words such as 'Oil' had worse scores than other similar words (e.g., 'Tsunami' and 'Disaster') and than expected. Second, a sufficient amount of significant events relevant to the target word is crucial, otherwise, the timelines would be too short in the eyes of the evaluators or the end users. For example, 'ISIS' which is a relatively new organization, has a few significant relevant events and therefore had weak results in our evaluation. Third, the available amount of data about the target word makes a difference. The more the better, as the temporal embeddings would then be rich and meaningful. We observed worse performance for relatively rare words, such as 'Bombing', compared more common ones (e.g., 'Attack' and 'Russia').

9 Conclusions

In this work, we develop methods to model the evolution of language in relation to world events. We introduced the task of timeline generation, which is composed of two components: identifying turning points when semantic changes occur, and representing descriptors (i.e., words or events in our case). We presented several embeddings for the task and studied their effect. We find that our proposed method of projecting embeddings from a large, static model to a temporal one (i.e., from *Wikipedia* to the *New York Times*) yielded the best performance. Given several baselines we determined that a supervised approach leveraging the projected embeddings yields the best results. Using our method, high quality timeline generation can be done automatically and at scale.

Acknowledgements

We thank Prof. Eytan Adar for early feedback and comments.

References

Mario Alinei. 1995. Thirty-five definitions of etymology or: etymology revisited. In *On languages and Language-The Presidential Addresses of the 1991 Meeting of the Societas Linguae Europaea*, pages 1–26. Mouton de Gruyter, Berlin, New York.

Tim Althoff, Xin Luna Dong, Kevin Murphy, Safa Alai, Van Dang, and Wei Zhang. 2015. Timemachine: Timeline generation for knowledge-base entities. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 19–28. ACM.

Hosein Azarbonyad, Mostafa Dehghani, Kaspar Beelen, Alexandra Arkut, Maarten Marx, and Jaap Kamps. 2017. Words are malleable: Computing semantic shifts in political and media discourse. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1509–1518. ACM.

Robert Bamler and Stephan Mandt. 2017. Dynamic word embeddings. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 380–389. JMLR. org.

Pierpaolo Basile, Annalina Caputo, Roberta Luisi, and Giovanni Semeraro. 2016. Diachronic analysis of the italian language exploiting google ngram. *CLiC it*, page 56.

Hai Leong Chieu and Yoong Keok Lee. 2004. Query based event extraction along a timeline. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 425–432. ACM.

Dirk Geeraerts et al. 1997. *Diachronic prototype semantics: A contribution to historical lexicology*. Oxford University Press.

David Graus, Maria-Hendrike Peetz, Daan Odijk, Ork de Rooij, Maarten de Rijke, et al. 2013. yourhistory–semantic linking for a personalized

timeline of historic events. In *Workshop: LinkedUp Challenge at OKCon*.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1489–1501.

Adam Jatowt and Kevin Duh. 2014. A framework for analyzing semantic change of words across time. In *Proceedings of the 14th ACM/IEEE-CS Joint Conference on Digital Libraries*, pages 229–238. IEEE Press.

Tom Kenter, Melvin Wevers, Pim Huijnen, and Maarten De Rijke. 2015. Ad hoc monitoring of vocabulary shifts over time. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 1191–1200. ACM.

Vivek Kulkarni, Rami Al-Rfou, Bryan Perozzi, and Steven Skiena. 2015. Statistically significant detection of linguistic change. In *Proceedings of the 24th International Conference on World Wide Web*, pages 625–635. International World Wide Web Conferences Steering Committee.

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic word embeddings and semantic shifts: a survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397.

Jiwei Li and Claire Cardie. 2014. Timeline generation: Tracking individuals on twitter. In *Proceedings of the 23rd international conference on World wide web*, pages 643–652. ACM.

Rada Mihalcea and Vivi Nastase. 2012. Word epoch disambiguation: Finding how words change over time. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 259–263.

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 *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1020–1029.

Matthias Orlikowski, Matthias Hartung, and Philipp Cimiano. 2018. Learning diachronic analogies to analyze concept change. In *Proceedings of the Second Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 1–11.

Octavian Popescu and Carlo Strapparava. 2013. Behind the times: Detecting epoch changes using large corpora. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 347–355.

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 the 21st international conference on World Wide Web*, pages 599–608. ACM.

Radim Rehurek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *In Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Citeseer.

Guy D Rosin, Eytan Adar, and Kira Radinsky. 2017. Learning word relatedness over time. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1168–1178.

Maja Rudolph and David Blei. 2018. Dynamic embeddings for language evolution. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 1003–1011. International World Wide Web Conferences Steering Committee.

Eyal Sagi, Stefan Kaufmann, and Brady Clark. 2009. Semantic density analysis: Comparing word meaning across time and phonetic space. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 104–111. Association for Computational Linguistics.

Dafna Shahaf and Carlos Guestrin. 2010. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–632. ACM.

Ehsan Sherkat and Evangelos E Milios. 2017. Vector embedding of wikipedia concepts and entities. In *International Conference on Applications of Natural Language to Information Systems*, pages 418–428. Springer.

Terrence Szymanski. 2017. Temporal word analogies: Identifying lexical replacement with diachronic word embeddings. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 448–453.

Tran Anh Tuan, Shady Elbassuoni, Nicoleta Preda, and Gerhard Weikum. 2011. Cate: context-aware timeline for entity illustration. In *Proceedings of the 20th international conference companion on World wide web*, pages 269–272. ACM.

Derry Tanti Wijaya and Reyyan Yeniterzi. 2011. Understanding semantic change of words over centuries. In *Proceedings of the 2011 international workshop on DETecting and Exploiting Cultural diversiTy on the social web*, pages 35–40. ACM.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic word embeddings for evolving semantic discovery. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 673–681. ACM.