

Chapter 8

Structured representation of temporal document collections by diachronic linguistic periodization

Yijun Duan^a, Adam Jatowt^b & Masatoshi Yoshikawa^c


^aNational Institute of Advanced Industrial Science and Technology ^bUniversity of Innsbruck ^cKyoto University

Language is our main communication tool. Deep understanding of its evolution is imperative for many related research areas including history, humanities, social sciences, etc. as well as for effective temporal information retrieval. To this end, we are interested in the task of segmenting long-term document corpora into naturally coherent periods based on the embodied evolving word semantics. There are many benefits of such segmentation including better representation of content in long-term document collections and support for modeling and understanding semantic drift. We propose a two-step framework for learning time-aware word semantics and periodizing document archive. The effectiveness of our model is demonstrated on the New York Times corpus spanning from 1990 to 2016.

1 Introduction

Language is an evolving and dynamic construct. Awareness of the necessity and possibilities of large scale analysis of the temporal dynamics on linguistic phenomena has increased considerably in the last decade (Zhang et al. 2015, Yao et al. 2018, Tahmasebi et al. 2021). Temporal dynamics play an important role in many time-aware information retrieval (IR) tasks. For example, when retrieving documents based on their embeddings, one needs accurate representations of content by temporal embedding vectors.



Yijun Duan, Adam Jatowt & Masatoshi Yoshikawa. 2021. Structured representation of temporal document collections by diachronic linguistic periodization. In Nina Tahmasebi, Lars Borin, Adam Jatowt, Yang Xu & Simon Hengchen (eds.), *Computational approaches to semantic change*, 261–285. Berlin: Language Science Press. DOI: 10.5281/zenodo.5040316 

It is intuitive that if an IR system is required to effectively return information from a target time period T_a , it may fail to do so if it is unable to capture the change in context between T_a and the current time, or just another time period in the past T_b . To which extent is the context of T_a different from that of T_b ? Are there any turning points in the interval between T_a and T_b when a significant context change occurred, or do T_a and T_b belong to the same stage in the evolving process of language rather? The capability of answering such questions is crucial for effective IR systems when coping with time-aware tasks. However, to the best of our knowledge, the research problem of distinguishing key stages in the evolution's trajectory of language still remains a challenge in the field of temporal IR.

Traditionally, a language's diachrony is segmented into pre-determined periods (e.g., the "Old", "Middle" and "Modern" eras for English) (Schätzle & Booth 2019), which is problematic, since such an approach may yield results concealing the true trajectory of a phenomenon (e.g., false assumption on abrupt turning point about the data). Moreover, these traditional segments are very coarse and can be easily obscured and derived from arbitrary and non-linguistic features (Degaetano-Ortlieb & Teich 2018). Thanks to accumulated large amounts of digitized documents from the past, it is now possible to employ large scale data-driven analyses for uncovering patterns of language change. Thus, instead of blindly adopting a pre-determined periodization scheme, data-driven approaches, which reflect actual changes in the data, and which are able to achieve meaningful generalizations, can be applied. This can not only help with evolutionary linguistic studies by providing data-driven evidence, but could also support better understanding of variations in performance of diverse temporal IR systems on different periods of a temporal document collection. Furthermore, automatic periodization can be also beneficial for many less-researched languages for which there may not be a sufficient number of historical linguistics-oriented studies and findings.

In this study, we design a data-driven approach for segmenting a temporal document collection (e.g., a long-term news article archive) into natural, linguistically coherent periods, thanks to which we can both capture the features involved in diachronic linguistic change, as well as identify the time periods when the changes occurred. Our approach is generic and can be applied to any diachronic data set. The detected periods could then be applied in diverse temporal IR scenarios, such as temporal analog retrieval and archival document recommendation.

Our method is based on the computation of dynamic word embeddings needed to properly represent changing word semantics. Semantic senses of words are

subject to broadening, narrowing, or other kinds of shifts throughout time. For instance, *Amazon* originally referred to mythical female warriors (in ancient Greek mythology), while it assumed a new sense of a large e-commerce company since the mid 1990s.

Additionally, different words may become conceptually equivalent or similar across time. For example, a music device *Walkman* played a similar role of mobile music playing device 30 years ago as *iPod* plays nowadays. The phenomenon of evolving word semantics is however rarely considered in the existing corpus periodization schemes.

In this paper, we structure document collections by periodizing the evolving word semantics embodied in the corpus. Specifically, for a long-term document corpus, our goal is to split the entire time span into several consecutive periods, where within the same period most words do not undergo significant fluctuations in term of their senses, while linguistic shifts are on the other hand relatively prevalent across different periods. In other words, a word is represented by a constant vector within one period, while it may have fairly different representations in different periods (see Figure 8.1).

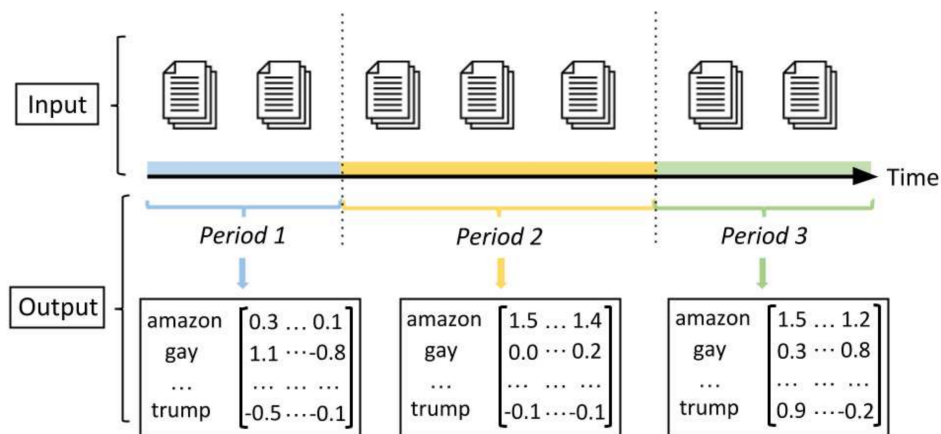


Figure 8.1: Conceptual view of our task. Our goal is to identify latent periods in the input document collection, such that word semantics are relatively stable within the same period (i.e., a word is represented by the same embedding vector), and major linguistic shifts exist between different periods (i.e., a word may be represented by fairly different vectors in different periods).

The problem of document collection periodization based on evolving word semantics is however not trivial. In order to solve this problem, we address the following two research questions:

- a. How to compute temporal-aware word embeddings?
- b. How to split the document collection based on learned word embeddings?

Our main technical contribution lies in a two-step framework for answering the above questions. First of all, we develop an anchor-based joint matrix factorization framework for computing time-aware word embeddings. More specifically, we concurrently factorize the time-stamped PPMI (positive pointwise mutual information) matrices, during which we utilize shared frequent terms (see Section 3) as anchors for aligning the word embeddings of each time frame to the same latent space. Furthermore, a block coordinate descent method is adopted to solve the learning model efficiently. Secondly, we formulate the periodization task as an optimization problem, where we aim to maximize the aggregation of differences between the word semantics of any two periods. To solve this problem, we employ three classes of optimization algorithms which are based on greedy splitting, dynamic programming and iterative refinement, respectively.

In the experiments, we use the crawled and publicly released New York Times dataset (Yao et al. 2018), which contains a total of 99,872 articles published between January 1990 and July 2016. We compare the performance of our models with existing competitive temporal word embedding methods, and corpus periodization methods, respectively. To demonstrate the quality of our learned temporal word embeddings, we focus on the task of searching for temporal analogs (see Section 5). To evaluate the periodization effectiveness, we construct the test sets by utilizing New York Times article tags (see Section 6), and evaluate the analyzed methods based on two standard metrics: Pk (Beeferman et al. 1999) and WinDiff (Pevzner & Hearst 2002) used in text segmentation tasks.

In summary, our contributions are as follows:

- From a conceptual standpoint, we introduce a novel research problem of periodizing diachronic document collections for discovering the embodied evolutionary word semantics. The discovered latent periods and corresponding temporal word embeddings can be utilized for many objectives, such as tracking and analyzing linguistic and topic shifts over time.
- From a methodological standpoint, we develop an anchor-based joint matrix factorization framework for computing time-aware word embeddings, and three classes of optimization techniques for document collection periodization.

- We perform extensive experiments on the New York Times corpus, which demonstrate the effectiveness of our approaches.

2 Problem definition

We start by presenting the formal problem definition.

2.1 Input

The input is a set of documents published across time. Each document is time-stamped and the whole text corpus spanning over a certain range of time is split into N basic time frames (t_1, t_2, \dots, t_N) . The length of a time frame can be on the order of months, years, decades or centuries. Formally, let $D = \{D_1, D_2, \dots, D_N\}$ denote the entire document set where $D_x, x = 1, \dots, N$ represents the subset of documents belonging to the time frame t_x .

2.2 Task 1

Our first task is to find a d -dimensional embedding vector for each term in the overall corpus vocabulary $V = \{w_1, \dots, w_{|V|}\}$ ¹ for each time unit $t_i, i = 1, \dots, N$, respectively. We denote by A_i the embedding matrix for t_i , whose j -th row represents the d -dimensional embedding vector of j -th term w_j in V . Thus A_i is of size $|V| \times d$.

2.3 Task 2

Based on Task 1, our second goal is to split the text corpus D into m contiguous, disjoint and coherent periods $\Theta = (P_1, P_2, \dots, P_m)$ and compute their corresponding word embedding matrices $E_i, i = 1, \dots, m$. Note that in this study the value of m is pre-defined. Each period $P_i = [\tau_b^i, \tau_e^i], i = 1, \dots, m$ is expressed by two time points representing its beginning date τ_b^i and the ending date τ_e^i , with $\tau_b^1 = t_1$ and $\tau_e^m = t_N$. Let $L(\Theta) = (\tau_b^1, \tau_b^2, \dots, \tau_b^m)$ denote the list of beginning dates of m periods, where $\tau_b^1 = t_1$. Notice that searching for Θ is equivalent to discovering $L(\Theta)$.

¹The overall vocabulary V is the union of vocabularies of each time unit, and thus it is possible for some $w \in V$ to not appear at all in some time units. This includes emerging words and dying words that are typical in real-world news corpora.

3 Temporal word embeddings

In this section, we describe our approach for computing dynamic word embeddings (solving Task 1 in Section 2), that captures lexical semantic dynamics across time.

3.1 Learning static embeddings

The distributional hypothesis (Firth 1957) states that semantically similar words usually appear in similar contexts. Let v_i denote the vector representing word w_i , then v_i can be embodied in the co-occurrence statistics of w_i . In this study we first factorize the PPMI (positive pointwise mutual information) matrix for constructing static (i.e., time-agnostic) word embeddings, following previous works (Yao et al. 2018, Levy & Goldberg 2014, Hamilton et al. 2016).

For a corpus D with vocabulary V , the i, j -th entry of PPMI matrix (of size $|V| \times |V|$) is given by

$$\begin{aligned} \text{PPMI}_{i,j} &= \max \left\{ \log_2 \left(\frac{p(w_i, w_j)}{p(w_i)p(w_j)} \right), 0 \right\} \\ &= \max \left\{ \log_2 \left(\frac{c(w_i, w_j) \cdot |D|}{c(w_i) \cdot c(w_j)} \right), 0 \right\} \end{aligned} \quad (1)$$

where $p(w_i, w_j)$ represents the probability of words w_i and w_j co-occurring within a fixed-size sliding window of text, $c(w_i, w_j)$ counts the number of times that w_i and w_j co-occur, and $|D|$ is the total number of word tokens. Discarding the PPMI values under zero offers much better numerical stability (Yao et al. 2018).

For word vectors v_i and v_j , we should have $\text{PPMI}_{i,j} \approx v_i \cdot v_j$, thus such word vectors can be obtained through factorizing the PPMI matrix.

3.2 Learning dynamic embeddings

In order to compute the embedding matrices $E = E_1, \dots, E_m$ for a given segmentation Θ on corpus D , we first construct the embedding matrix $A_i, i = 1, \dots, N$ for each time unit. We denote PPMI_i the PPMI matrix for time frame t_i , thus temporal word embeddings A_i should satisfy $\text{PPMI}_i \approx A_i \cdot A_i^T$.

However, if A_i is constructed separately for each time unit, due to the invariant-to-rotation nature of matrix factorization these learned word embeddings A_i are non-unique (i.e., we have

$$\text{PPMI}_i \approx A_i \cdot A_i^T = (A_i W^T) \cdot (W A_i^T) = \tilde{A}_i \tilde{A}_i^T$$

for any orthogonal transformation W which satisfies $W^T \cdot W = I$). As a byproduct, embeddings across time frames may not be placed in the same latent space. Some previous works (Kulkarni et al. 2015, Hamilton et al. 2016, Zhang et al. 2015) solved this problem by imposing an alignment before any two adjacent matrices A_i and A_{i+1} , resulting in $A_i \approx A_{i+1}, i = 1, \dots, N - 1$.

Instead of solving a separate alignment problem for circumventing the non-unique characteristic of matrix factorization, we propose to learn the temporal embeddings across time concurrently. Note that for a word, we desire its vector to be close among all temporal embedding matrices, if it did not change its meaning across time (or change its meaning to very small extent). Such words are regarded as “anchors” for connecting various embedding matrices, in our joint factorization framework.

Essentially, we assume that very frequent terms (e.g., *man*, *sky*, *one*, *water*) did not experience significant semantic shifts as their dominant meanings are commonly used in everyday life and by many people. This assumption is reasonable as it has been reported in many languages including English, Spanish, Russian and Greek (Lieberman et al. 2007, Pagel et al. 2007). We refer to these words as SFT, standing for SHARED FREQUENT TERMS. Specifically, we denote by A_i^{SFT} the $|V| \times d$ embedding matrix whose i -th row corresponds to the vector of word w_i in A_i , if w_i is a shared frequent term, and corresponds to zero vector otherwise, for a given time unit t_i . Our joint matrix factorization framework for discovering temporal word embeddings is then presented as follows (see Figure 8.2 for an illustration):

$$\begin{aligned} A_1, \dots, A_N = \underset{A}{\operatorname{argmin}} \sum_{i=1}^N \left\| \text{PPMI}_i - A_i \cdot A_i^T \right\|_F^2 \\ + \alpha \cdot \sum_{i=1}^N \|A_i\|_F^2 + \beta \cdot \sum_{i=1}^{N-1} \sum_{j=i+1}^N \|A_i^{SFT} - A_j^{SFT}\|_F^2 \end{aligned} \quad (2)$$

where the key smoothing term $\|A_i^{SFT} - A_j^{SFT}\|_F^2$ aligns shared frequent terms in all years, thus places word embeddings across time in the same latent space. The regularization term $\|A_i\|_F^2$ is adopted to guarantee the low-rank data fidelity for overcoming the problem of overfitting. Parameters α and β are used to control the weight of different terms to achieve the best factorization.

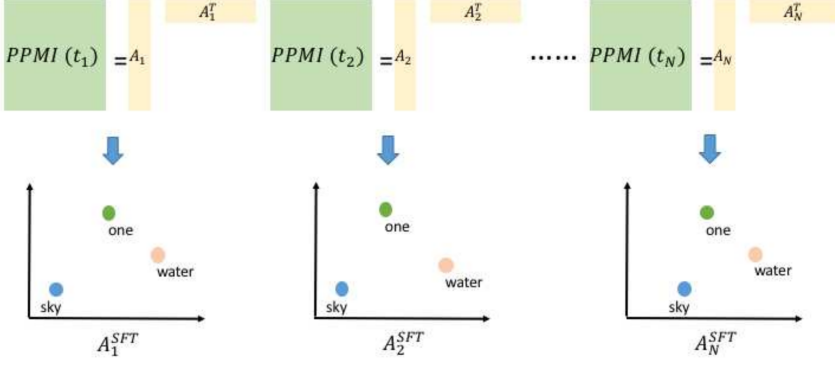


Figure 8.2: Illustration of our joint matrix factorization model. Shared frequent terms in all time frames (t_1, t_2, \dots, t_N) are aligned to similar positions, which places word embeddings across time in the same latent semantic space.

3.3 Optimization

The optimization problem in Equation (2) is not jointly convex to $A_i, i = 1, \dots, N$, we first decompose the objective across periods, and solve for A_i by fixing other embedding matrices as constants at each step. The problem of optimizing A_i can be then formulated as follows:

$$A_i = \underset{A}{\operatorname{argmin}} \Omega(A_i) = \underset{A}{\operatorname{argmin}} \left\| \text{PPMI}_i - A_i \cdot A_i^T \right\|_F^2 + \alpha \cdot \|A_i\|_F^2 + \beta \cdot \sum_{j=1}^N \|A_i^{SFT} - A_j^{SFT}\|_F^2 \quad (3)$$

Notice that $\Omega(A_i)$ is quartic in A_i , thus Equation (3) can not be optimized analytically. We then adopt the block coordinate descent (Tseng 2001) for iteratively minimizing $\Omega(A_i)$. Specifically, the gradient of $\Omega(A_i)$ with regard to A_i is given by

$$\frac{\partial \Omega(A_i)}{\partial A_i} = 2(A_i \cdot A_i^T - \text{PPMI}_i + \alpha) \cdot A_i + 2\beta \cdot \sum_{j=1}^N (A_i^{SFT} - A_j^{SFT}) \quad (4)$$

where each above computation is of the order $O(\text{nnz}(\text{PPMI}_i)d + d^2V)$ where $\text{nnz}(\text{PPMI}_i)$ is the number of non-zeros in the matrix.

4 Document collection periodization

In this section, we prescribe how to obtain the final periods (solving Task 2 in Section 2). We first introduce the scoring objective of linguistic periodization, then we study the effectiveness of three optimization approaches: (1) greedy algorithm based periodization, which searches for the best available boundary at each step; (2) dynamic programming based periodization, which is able to discover the optimal periods in a dynamic programming manner; (3) an iterative refinement scheme, which iteratively refines the boundaries for improving the performance of the greedy strategy.

4.1 Scoring

To frame the periodization problem as a form of optimization, having built a particular segmentation Θ , we now specify the way to quantify the quality of Θ , and then adopt different classes of techniques to optimize that scoring objective. In general, we prefer the embedding matrices of different periods to be characterized by high inter-dissimilarity. More explicitly, the objective $\text{Obj}(\Theta)$ for an overall segmentation is given by aggregating the dissimilarity (expressed by the squared F-norm of the difference of two embedding matrices) between all pairs of period-specific embedding matrices, as follows:

$$\text{Obj}(\Theta) = \text{Obj}(L(\Theta)) = \sum_{i=1}^{m-1} \sum_{j=i+1}^m \|E_i - E_j\|_F^2 \quad (5)$$

Here, m is the pre-defined number of periods. E_i is measured as the average of embeddings A_t for time unit t in period $P_i = [\tau_b^i, \tau_e^i]$, as follows:

$$E_i = \frac{1}{\tau_e^i - \tau_b^i + 1} \sum_{t=\tau_b^i}^{\tau_e^i} A_t \quad (6)$$

The segmentation that achieves the highest score of Equation (5) will be adopted.

4.2 Periodizing

4.2.1 Greedy algorithm based periodization

The greedy periodization algorithm is not guaranteed to reach the optimal splitting, however it offers significant computational benefit. At each step, it greedily

inserts a new boundary (which is the beginning date of a new period) to the existing segmentation to locally maximize the objective function, until desired m periods are discovered. The process of greedy periodization is formulated as follows:

$$L(\Theta)^{i+1} = \arg \max_{t_p \in [t_1, t_N], t_p \notin L(\Theta)^i} \text{Obj}(L(\Theta)^i \cup \{t_p\}) \quad (7)$$

where $L(\Theta)^i$ denotes the list of boundaries (or the beginning dates of periods) at the i -th step, and $L(\Theta)^0 = \{t_1\}$. The process of greedy algorithm based periodization is shown in Algorithm 1.

Algorithm 2: Greedy algorithm based periodization

```

input  :  $L(\Theta)^0; m$ 
output :  $L(\Theta)^{m-1}$ 
1 for  $i \leftarrow 0$  to  $m - 2$  do
2    $max\_score \leftarrow 0$ ;
3    $next\_boundary \leftarrow 0$ ;
4   for  $t_p \leftarrow t_1$  to  $t_N$  do
5     ▷ Find the best local boundary;
6     if  $t_p \in L(\Theta)^i$  then
7       continue
8     end
9      $score \leftarrow \text{Obj}(L(\Theta)^i \cup \{t_p\})$ ;
10    if  $score > max\_score$  then
11       $max\_score \leftarrow score$ ;
12       $next\_boundary \leftarrow t_p$ ;
13    end
14  end
15   $L(\Theta)^{i+1} \leftarrow L(\Theta)^i \cup \{next\_boundary\}$ ;
16 end

```

4.2.2 Dynamic programming based periodization

The core idea of dynamic programming based periodization is to break the overall problem into a series of simpler smaller segmentation tasks, and then recursively find the solutions to the sub-problems. By recursively solving the sub-problems optimally, the dynamic programming approach yields the globally optimal value

of Equation (5). Let Θ_k^l denote the segmentation of the first l time slices of the entire time span into k periods. The computational process of dynamic programming based periodization is then expressed as follows:

$$L(\Theta_k^N) = \arg \max_{l < N} \text{Obj}(L(\Theta_{k-1}^l) \cup t_{l+1}) \quad (8)$$

where $\Theta_1^l = [t_1, t_l]$ and $L(\Theta_1^l) = \{t_1\}, l = 1, \dots, N$. In practice, though each of those sub-problems can be solved in one pass by storing their solutions in a memory-based data structure (array, map, etc), the dynamic programming approach can be costly to compute, compared to the greedy splitting, as shown below in Section 4.3. The process of dynamic programming based periodization is shown in Algorithm 2.

Algorithm 3: Dynamic programming based periodization

```

input :  $L(\Theta_1^l), l = 1, \dots, N; m$ 
output :  $L(\Theta_m^N)$ 
1 for row  $\leftarrow 2$  to  $m$  do
2   for col  $\leftarrow$  row to  $N$  do
3      $\triangleright$  Recursively find the solutions to the sub-problems;
4     max_score  $\leftarrow 0$ ;
5     next_boundary  $\leftarrow 0$ ;
6     subtask  $\leftarrow 0$ ;
7     for  $j \leftarrow$  row  $- 1$  to col  $- 1$  do
8       score  $\leftarrow \text{Obj}(L(\Theta_{row-1}^j) \cup \{t_{j+1}\})$ ;
9       if score  $>$  max_score then
10        max_score  $\leftarrow$  score;
11        next_boundary  $\leftarrow t_{j+1}$ ;
12        subtask  $\leftarrow j$ ;
13      end
14    end
15     $L(\Theta_{row}^{col}) \leftarrow L(\Theta_{row-1}^{subtask}) \cup \{next\_boundary\}$ 
16  end
17 end

```

4.2.3 Iterative refinement based periodization

The iterative refinement framework starts with the greedy segmentation. At each step after the best available boundary is found, a relaxation scheme which tries to adjust each segment boundary optimally while keeping the edges (i.e. adjacent boundaries) to either side of it fixed, is applied. This method can improve the performance of the greedy scheme, while at the same time retain its computational benefit to some extent. Let $L(\Theta)_G^i[j]$ denote the j -th element in $L(\Theta)^i$ after the i -th greedy search step, the iterative refinement process for finding $L(\Theta)^i[j]$ is shown as follows:

$$L(\Theta)^i[j] = \arg \max_{t_p \in (L(\Theta)^i[j-1], L(\Theta)^i[j+1])} \text{Obj}((L(\Theta)^i \setminus L(\Theta)_G^i[j]) \cup \{t_p\}) \quad (9)$$

The process of this method is shown in Algorithm 3 below.

Algorithm 4: Iterative refinement based periodization

```

input  :  $L(\Theta)^0; m$ 
output :  $L(\Theta)^{m-1}$ 
1 for  $i \leftarrow 0$  to  $m - 2$  do
2    $next\_boundary, max\_score \leftarrow Greedy(L(\Theta)^i);$ 
3    $L(\Theta)^{i+1} \leftarrow L(\Theta)^i \cup \{next\_boundary\};$ 
4   for  $j \leftarrow 1$  to  $i$  do
5      $\triangleright$  Iteratively refine the previous boundaries;
6      $new\_boundary \leftarrow L(\Theta)^{i+1}[j];$ 
7      $t_{begin} \leftarrow L(\Theta)^{i+1}[j - 1];$ 
8      $t_{end} \leftarrow L(\Theta)^{i+1}[j + 1];$ 
9     for  $t_p \leftarrow t_{begin}$  to  $t_{end}$  do
10       $score \leftarrow \text{Obj}(L(\Theta)^{i+1} - L(\Theta)^{i+1}[j] \cup \{t_p\});$ 
11      if  $score > max\_score$  then
12         $max\_score \leftarrow score;$ 
13         $next\_boundary \leftarrow t_p;$ 
14      end
15    end
16     $L(\Theta)^{i+1} \leftarrow (L(\Theta)^{i+1} - L(\Theta)^{i+1}[j]) \cup \{new\_boundary\};$ 
17  end
18 end

```

4.3 Analysis of time complexity

For greedy periodization, it requires $m - 1$ steps and the i -th step calls the scoring function Equation (5) $N - i$ times. In total, it is $Nm - N - m^2 + m/2$. In the case of $N \gg m$, the greedy periodization algorithm takes $O(Nm)$. For dynamic programming based periodization, it requires $O(Nm)$ states and evaluating each state involves an $O(N)$ calling of Equation (5). Then the overall algorithm would take $O(N^2m)$. Finally, for iterative refinement based periodization, an upper bound on its time complexity is $O(\sum_{i=1}^{m-1} (N - i) * i) = O(Nm^2)$.

5 Embedding effectiveness

5.1 Datasets

News corpora, which maintain consistency in narrative style and grammar, form a good basis for studying language evolution. We perform the experiments on the New York Times Corpus, which has been frequently used to evaluate different researches that focus on temporal information processing or extraction in document archives (Campos et al. 2014). The dataset we use (Yao et al. 2018) is a collection of 99,872 articles published by the New York Times between January 1990 and July 2016. For the experiments, we first divide this corpus into 27 frames, setting the length of time unit to be 1 year. Stopwords and rare words (which have less than 200 occurrences in the entire corpus) were removed before experiments, following previous work (Zhang et al. 2015). The statistics of our dataset are shown in Table 8.1.

Table 8.1: Summary of the New York Times dataset

| #Articles | #Vocabulary | #Word Co-occurrences | #Time units | Range |
|-----------|-------------|----------------------|-------------|---------------------|
| 99,872 | 20,936 | 11,068,100 | 27 | Jan. 1990–Jul. 2016 |

5.2 Experimental settings

We describe next the parameters used in the experiments. For the construction of the PPMI matrix, the length of the sliding window and the embedding dimensions is set to be 5 and 50, respectively, following (Yao et al. 2018). During the training process, the values of parameters α and β (see Equation (2)) are set to be 20 and 100, respectively, as the result of a grid search. The selection of shared

frequent terms (see Section 3.2) used as anchors is set to be the top 5% most popular words in the entire corpus excluding stopwords, as suggested by (Zhang et al. 2015).

5.3 Compared methods

We describe here the analyzed methods for learning temporal word embeddings.

Without transformation (Non-Tran): This method directly compares the vectors in different time without performing any transformation.

Linear transformation (LT) (Zhang et al. 2015): The embeddings are first trained separately for each year, and then are transformed by optimizing a linear transformation between adjacent years.

Orthogonal transformation (OT) (Hamilton et al. 2016): The embeddings are first trained separately for each year, and then are aligned by optimizing an orthogonal transformation between adjacent years.

Dynamic Word2Vec (DW2V) (Yao et al. 2018): The embeddings are trained based on PPMI matrices by minimizing the distance between embeddings in only adjacent years, without using SFTs.

The proposed model (this paper): The embeddings are jointly learned, by minimizing the difference between embeddings of shared frequent terms within the entire period.

We use the publicly available source code released by (Yao et al. 2018) for all baseline methods.²

5.4 Test sets

To demonstrate the effectiveness of our model, we focus on the task of searching for temporal analogs. We utilize 2 testsets (Yao et al. 2018) containing queries in the base time (e.g., *obama* in 2012) and their analogs in target time (e.g., *bush* in 2002). Testset 1 includes publicly recorded knowledge that for each year lists different names for a particular role (e.g., U.S. president),³ and testset 2 consists

²<https://github.com/yifan0sun/DynamicWord2Vec>

³Note that we find several mistakes in this testset, such as (*pistons-1990, knicks-1999*) (the correct pair should be (*pistons-1990, spurs-1999*)). Then we manually correct them and use the corrected version for all analyzed methods in experiment.

of interesting concepts such as emerging technologies, brands and major events (e.g., *app* in 2012 can correspond to *software* in 1990: Yao et al. 2018). In total, there are 11,473 pairs of terms (query and its analog) used in our experiments.

5.5 Evaluation metrics

The MEAN RECIPROCAL RANK (MRR) is used for evaluating the search results for each learning model, which is computed as follows:

$$\text{MRR} = \frac{1}{N} \sum_{i=1}^N \frac{1}{\text{rank}_i} \quad (10)$$

where rank_i is the rank of a correct temporal analog at the i -th test, and N is the number of test pairs.

In addition, precisions @1, @5, @10 and @20 are also reported. Those metrics refer to the rates of tests in which the correct temporal analog was included in the top 1, 5, 10 and 20 results, respectively. All the values of used metrics fall into [0,1]. The higher the values are, the more effectively a model works.

5.6 Experimental results

Table 8.2: Performance of all analyzed models for learning dynamic word embeddings.

| | Testset 1 | | | | | Testset 2 | | | | |
|-----------|-----------|-------|-------|-------|-------|-----------|-------|-------|-------|-------|
| | MRR | P@1 | P@5 | P@10 | P@20 | MRR | P@1 | P@5 | P@10 | P@20 |
| Non-Tran | 0.012 | 0.020 | 0.034 | 0.042 | 0.064 | 0.005 | 0.000 | 0.000 | 0.018 | 0.025 |
| LT | 0.137 | 0.118 | 0.232 | 0.267 | 0.355 | 0.038 | 0.021 | 0.065 | 0.146 | 0.219 |
| OT | 0.158 | 0.106 | 0.224 | 0.295 | 0.373 | 0.050 | 0.023 | 0.079 | 0.142 | 0.185 |
| DW2V | 0.422 | 0.331 | 0.549 | 0.619 | 0.703 | 0.144 | 0.076 | 0.220 | 0.382 | 0.487 |
| Our model | 0.454 | 0.348 | 0.563 | 0.651 | 0.740 | 0.157 | 0.082 | 0.255 | 0.406 | 0.520 |

Table 8.2 shows the scores for all the methods averaged on all the tested queries on testset 1 and testset 2, respectively. We first notice that the performance is extremely poor without transforming the contexts of queries. The correct answers in the Non-Tran approach are usually found at ranks $> 1\text{k}$ which is in line with observations made by Zhang et al. (2015). On the other hand, both transformation-based methods LT and OT are helpful since they exhibit significantly better effectiveness compared to Non-Tran. This observation suggests little overlap in

the contexts of news articles which are separated by long time gaps, and that the task of temporal analog identification is quite difficult. Moreover, it is evident that learning the temporal embeddings across time by enforcing a global alignment is superior to following the “separately learning-and-aligning” pattern, since both DW2V and our approach outperform LT and OT significantly. Therefore, enforcing a global alignment is more effective for solving the temporal analog detection task.

Lastly, a closer look at Table 8.2 reveals that regardless of the type of evaluation metric, our model improves upon the performance of the state-of-the-art DW2V model. Specifically, our method improves DW2V model by 9.0% and 7.6% when measured using the main metric MRR on testset 1 and testset 2, respectively. The plausible reason is that DW2V does not differentiate words with stable meanings from words whose semantics are evolving, while such assumption may lead to a less precise learned representation of words. By injecting additional knowledge of shared frequent terms as anchors, our approach allows for only aligning embeddings of such stable words, and keeping the representation of other words exactly as their diachronic contexts express.

6 Periodization effectiveness

6.1 Datasets

We use the same news article datasets as described in Section 5.1.

6.2 Compared methods

We implemented below two types of periodization models as analyzed methods (proposed methods and baselines) in order to compare the periods they generate with the reference periods.

6.2.1 Baseline methods

We test four baselines as listed below.

Random: The segment boundaries are randomly inserted.

VNC (Gries & Hilpert 2012): A bottom-up hierarchical variability-based neighbor clustering (VNC) approach to periodization.

KLD (Degaetano-Ortlieb & Teich 2018): An entropy-driven approach which calculates the Kullback-Leibler Divergence (KLD) between term frequency features in text from temporally adjacent time periods to identify stages of language change.

CPD (Kulkarni et al. 2015): An approach which uses statistically sound change point detection (CPD) algorithms to detect significant linguistic shifts based on mean shift model.

6.2.2 Proposed methods

We list three proposed methods below (see Section 4.2).

These proposed methods adopt different strategies to optimize Equation (5), based on the temporal word embeddings obtained in Section 3.

G-WSE: Greedy periodization based on word semantic evolution.

DP-WSE: Dynamic programming periodization based on word semantic evolution.

IR-WSE: Iterative refinement based on word semantic evolution.

6.3 Test sets

As far as we know there are no standard testsets for New York Time Corpus. We therefore had to create test sets. Note that the collected news articles dataset is associated with some metadata, including title, author, publication time, and topical section label (e.g., *Science*, *Sports*, *Technology*) which describes the general topic of news articles. Such section labels could be used to locate the boundaries of word meanings.

Intuitively, if a word w is strongly related to a particular section s in year t , we associate w , s and t together and construct a $\langle w, s, t \rangle$ triplet. A boundary of w is registered if it is assigned to different sections in two adjacent years (i.e., both triplet $\langle w, s, t \rangle$ and $\langle w, s', t + 1 \rangle$ hold and $s \neq s'$).

More specifically, for each word w in the corpus vocabulary V we compute its frequency in all sections for each year t , and w is assigned to the section in which w is most frequent. Note that this word frequency information is not used in our learning model. In this study we utilize the 11 most popular and discriminative sections of the New York Times,⁴ following previous work (Yao et al. 2018).

⁴These sections are *Arts*, *Business*, *Fashion & Style*, *Health*, *Home & Garden*, *Real Estate*, *Science*, *Sports*, *Technology*, *U.S.*, *World*.

Recall that parameter m denotes the number of predefined latent periods. For each different m , we first identify the set of words S_m characterized by the same number of periods. Then for each method and each value of m , we test the performance of such method by comparing the generated periods with the reference segments of each word in S_m , and then take the average. In this study, we experiment with the variation in the value of m , ranging from 2 to 10.

6.4 Evaluation metrics

We evaluate the performance of the analyzed methods with respect to two standard metrics: Pk (Beeferman et al. 1999) and WinDiff (Pevzner & Hearst 2002) used in text segmentation tasks. Both metrics use a sliding window of fixed size k over the document and compare the newly generated segments with the reference ones. Here k is generally set as follows (Beeferman et al. 1999):

$$k = \left\lfloor \frac{\#time\ units}{2 \cdot \#periods} \right\rfloor - 1 \quad (11)$$

Specifically, the Pk metric counts the number of disagreements on the probe elements as follows:

$$Pk = \frac{1}{N - k} \sum_{i=1}^{N-k} [P_{hyp}(i, i + k) \neq P_{ref}(i, i + k)] \quad (12)$$

where N indicates the number of elements (in our case, the number of time units) and $P(i, i + k)$ is equal to 1 or 0 according to whether or not both element i and $i + k$ are recognized as being in the same segment in hypothesized segmentation P_{hyp} and reference segmentation P_{ref} . Since Pk metric has the disadvantage that it penalizes false positives more severely than false negatives (Aleami & Ginsparg 2015), the WinDiff metric was introduced. It is defined as follows:

$$WinDiff = \frac{1}{N - k} \sum_{i=1}^{N-k} [W_{hyp}(i, i + k) \neq W_{ref}(i, i + k)] \quad (13)$$

where $W_{hyp}(i, i + k)$ and $W_{ref}(i, i + k)$ each count the number of boundaries between the time units i and $i + k$ in generated and reference segments, respectively. An error is registered if they are different. Both Pk and WinDiff give values in the range $[0, 1]$. They are equal to 0 if and only if an algorithm assigns all boundaries correctly. The lower the scores are, the better the algorithm performs.

6.5 Evaluation results

Tables 8.3 and 8.4 summarize the Pk and WinDiff scores for each method, respectively. Based on the experimental data we make the following observations.

- The proposed methods exhibit the overall best performance regarding both Pk and WinDiff metrics. More specifically, they outperform the best baseline under 7 of 9 predefined numbers of periods in terms of Pk, and 6 of 9 in terms of WinDiff. This demonstrates the effectiveness of our proposed periodization frameworks.
- Regarding baseline methods, Random achieves the worst performance as expected. CPD and KLD show competitive performance under certain settings. CPD gets two wins in terms of Pk, and KLD obtains three wins in terms of WinDiff.
- DP-WSE is the best performer among all three proposed periodization algorithms. It contributes 6 best performance in terms of Pk, and 5 in terms of WinDiff. Moreover, when compared to G-WSE and IR-WSE, DP-WSE shows a 3.79% and 3.24% increase in terms of Pk, and a 7.77% and 6.46% increase in terms of WinDiff, respectively. This observation is in good agreement with the theoretical analysis, which states that dynamic programming based segmentation sacrifices computational efficiency for the optimal splitting.
- The operation of iterative refinement indeed improves the performance of greedy periodization. However, the improvement is marginal: many results generated by IR-WSE are similar or identical to those from G-WSE.

7 Related work

7.1 Text segmentation

The most similar task to the document collection periodization is text segmentation. The task of text segmentation is formulated as splitting a chunk of text into meaningful sections based on their topic continuity, and it has many useful applications in information retrieval, text summarization, etc. Early text segmentation approaches include TextTiling (Hearst 1997) and the C99 algorithm (Choi 2000), which are based on some heuristics on text coherence using a bag

Table 8.3: Performance comparison using Pk (Lower scores indicate better performance).

| Acronym | Number of periods | | | | | | | | |
|---------|-------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Random | 0.467 | 0.474 | 0.545 | 0.522 | 0.542 | 0.480 | 0.480 | 0.480 | 0.539 |
| VNC | 0.385 | 0.253 | 0.249 | 0.290 | 0.282 | 0.302 | 0.302 | 0.294 | 0.303 |
| KLD | 0.385 | 0.278 | 0.244 | 0.270 | 0.276 | 0.278 | 0.284 | 0.290 | 0.304 |
| CPD | 0.238 | 0.234 | 0.246 | 0.260 | 0.282 | 0.263 | 0.249 | 0.299 | 0.338 |
| G-WSE | 0.115 | 0.201 | 0.248 | 0.282 | 0.300 | 0.310 | 0.312 | 0.292 | 0.303 |
| DP-WSE | 0.115 | 0.230 | 0.236 | 0.251 | 0.271 | 0.290 | 0.291 | 0.286 | 0.296 |
| IR-WSE | 0.115 | 0.201 | 0.244 | 0.279 | 0.300 | 0.304 | 0.312 | 0.292 | 0.303 |

Table 8.4: Performance comparison using WinDiff (Lower scores indicate better performance).

| Acronym | Number of periods | | | | | | | | |
|---------|-------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Random | 0.467 | 0.474 | 0.545 | 0.478 | 0.542 | 0.480 | 0.480 | 0.480 | 0.500 |
| VNC | 0.417 | 0.346 | 0.396 | 0.416 | 0.426 | 0.434 | 0.439 | 0.435 | 0.388 |
| KLD | 0.417 | 0.343 | 0.383 | 0.384 | 0.428 | 0.437 | 0.434 | 0.430 | 0.384 |
| CPD | 0.414 | 0.386 | 0.387 | 0.394 | 0.430 | 0.430 | 0.430 | 0.432 | 0.385 |
| G-WSE | 0.383 | 0.430 | 0.435 | 0.449 | 0.456 | 0.449 | 0.447 | 0.432 | 0.387 |
| DP-WSE | 0.383 | 0.336 | 0.387 | 0.403 | 0.423 | 0.422 | 0.430 | 0.431 | 0.388 |
| IR-WSE | 0.383 | 0.405 | 0.428 | 0.449 | 0.456 | 0.449 | 0.447 | 0.421 | 0.387 |

of words representation. Furthermore many attempts adopt topic models to inform the segmentation task, including Riedl & Biemann (2012), Du et al. (2013). Alemi & Ginsparg (2015) is a segmentation algorithm based on time-agnostic semantic word embeddings. Most text segmentation methods are unsupervised. However, neural approaches have also been explored for domain-specific text segmentation tasks, such as Sehikh et al. (2017). Many text segmentation algorithms are greedy in nature, such as Choi (2000), Choi et al. (2001). Moving beyond the greedy approach, some works search for the optimal splitting for their own objective using dynamic programming (Utiyama & Isahara 2001, Fragkou et al. 2004).

Apart from computer scientists, social scientists also have proposed a variety of methods to break a corpus into coherent sections. Related frameworks include

those of Ruef (1999), Gries & Hilpert (2008), Alsudais & Tchalian (2016). Some studies are investigating the temporal topics in various corpora including news (Allan et al. 2001), historical documents (Duan et al. 2017) or scientific archives (Blei & Lafferty 2006, Wang & McCallum 2006).

7.2 Temporal word embeddings

How to best represent words with low-dimensional dense vectors has attracted consistent interest for several decades. Early methods are relying on statistical models (Lund & Burgess 1996, Blei et al. 2003), while in recent years neural models such as word2vec (Mikolov et al. 2013) and GloVe (Pennington et al. 2014) have shown great success in many NLP applications. Moreover, it has been demonstrated that both word2vec and GloVe are equivalent to factorizing the PMI matrix (Levy & Goldberg 2014), which motivates our approach.

The above methods assume word representation is time-agnostic. Recently some works explored computing time-aware embeddings of words, for analyzing linguistic change and evolution (Yao et al. 2018, Zhang et al. 2015, Hamilton et al. 2016, Kulkarni et al. 2015, Azarbondy et al. 2017, Gonen et al. 2020). In order to compare word vectors across time most works ensure the vectors are aligned to the same coordinate axes, by solving the least squares problem (Zhang et al. 2015, Kulkarni et al. 2015), imposing an orthogonal transformation (Hamilton et al. 2016) or jointly smoothing every pair of adjacent time slices (Yao et al. 2018). Different from the existing methods, in this study we inject additional knowledge by using shared frequent terms as anchors to simultaneously learn the temporal word embeddings and circumvent the alignment problem.

8 Conclusion

This work approaches a novel task – diachronic document collection periodization. The special character of our task allows capturing evolutionary word semantics. The discovered latent periods can be an effective indicator of linguistics shifts and evolution embodied in analyzed diachronic textual corpora. To address the introduced problem we propose a two-step framework which consists of a joint matrix factorization model for learning dynamic word embeddings, and a well-defined optimization formulation for corpus periodization. For solving the resulting optimization problem we develop a series of effective algorithms. We perform extensive experiments to evaluate generated periods on the New York Times corpus spanning from 1990 to 2016, and show that our proposed methods perform favorably against diverse competitive baselines.

In the future, we plan to incorporate causal analysis for detecting correlated word semantic changes. We will also consider utilizing word sentiments in corpora periodization scenarios.

Acknowledgements

This research has been partially supported by MEXT Kakenhi grants (#17H01828, #18K19841 and #19H04215).

Abbreviations

| | |
|----------|---|
| CPD | change point detection |
| DP-WSE | dynamic programming based on word semantic evolution |
| DW2V | dynamic word2vec |
| G-WSE | greedy periodization based on word semantic evolution |
| IR-WSE | iterative refinement based on word semantic evolution |
| KLD | Kullback-Leibler Divergence |
| LT | linear transformation |
| Non-Tran | without transformation |
| OT | orthogonal transformation |
| VNC | variability-based neighbor clustering |

References

- Alemi, Alexander A & Paul Ginsparg. 2015. Text segmentation based on semantic word embeddings.
- Allan, James, Rahul Gupta & Vikas Khandelwal. 2001. Temporal summaries of new topics. In *Proceedings of the 24th annual international ACM SIGIR conference on research and development in information retrieval*, 10–18. DOI: 10.1145/383952.383954.
- Alsudais, Abdulkareem & Hovig Tchalian. 2016. Corpus periodization framework to periodize a temporally ordered text corpus. In *Twenty-second Americas conference on information systems*. Red Hook, NY: Curran Associates, Inc.
- Azarbonyad, Hosein, Mostafa Dehghani, Kaspar Beelen, Alexandra Arkut, Maarten Marx & Jaap Kamps. 2017. Words are malleable: Computing semantic shifts in political and media discourse. In *Proceedings of CIKM 2017*, 1509–1518. Singapore: ACM. DOI: 10.1145/3132847.3132878.

- Beeferman, Doug, Adam Berger & John Lafferty. 1999. Statistical models for text segmentation. *Machine Learning* 34. 177–210. DOI: 10.1023/A:1007506220214.
- Blei, David M. & John D. Lafferty. 2006. Dynamic topic models. In *Proceedings of the 23rd international conference on machine learning*, 113–120.
- Blei, David M., Andrew Y. Ng & Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research* 3. 993–1022.
- Campos, Ricardo, Gaël Dias, Alípio M. Jorge & Adam Jatowt. 2014. Survey of temporal information retrieval and related applications. *ACM Computing Surveys (CSUR)* 47(2). 1–41.
- Choi, Freddy Y. Y. 2000. Advances in domain independent linear text segmentation. *arXiv preprint 0003083*.
- Choi, Freddy Y. Y., Peter Wiemer-Hastings & Johanna D. Moore. 2001. Latent semantic analysis for text segmentation. In *Proceedings of the 2001 conference on empirical methods in Natural Language Processing*.
- Degaetano-Ortlieb, Stefania & Elke Teich. 2018. Using relative entropy for detection and analysis of periods of diachronic linguistic change. In *Proceedings of the second joint SIGHUM workshop on computational linguistics for cultural heritage, social sciences, humanities and literature*, 22–33.
- Du, Lan, Wray Buntine & Mark Johnson. 2013. Topic segmentation with a structured topic model. In *Proceedings of the 2013 conference of the North American chapter of the Association for Computational Linguistics: Human language technologies*, 190–200. ACL.
- Duan, Yijun, Adam Jatowt & Katsumi Tanaka. 2017. Discovering typical histories of entities by multi-timeline summarization. In *Proceedings of the 28th ACM Conference on Hypertext and Social Media*, 105–114. DOI: 10.1145/3078714.3078725.
- Firth, John R. 1957. *Papers in linguistics 1934–1951*. London: Oxford University Press.
- Fragkou, Pavlina, Vassilios Petridis & Ath Kehagias. 2004. A dynamic programming algorithm for linear text segmentation. *Journal of Intelligent Information Systems* 23(2). 179–197.
- Gonen, Hila, Ganesh Jawahar, Djamé Seddah & Yoav Goldberg. 2020. Simple, interpretable and stable method for detecting words with usage change across corpora. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, 538–555.
- Gries, Stefan Th. & Martin Hilpert. 2008. The identification of stages in diachronic data: Variability-based neighbour clustering. *Corpora* 3(1). 59–81.

- Gries, Stefan Th. & Martin Hilpert. 2012. Variability-based neighbor clustering: A bottom-up approach to periodization in historical linguistics. In Terttu Nevalainen & Elizabeth Closs Traugott (eds.), *The Oxford handbook of the history of English*. Oxford: Oxford University Press.
- Hamilton, William L., Jure Leskovec & Dan Jurafsky. 2016. Diachronic word embeddings reveal statistical laws of semantic change. In *Proceedings of ACL 2016 (Volume 1: Long papers)*, 1489–1501. Berlin: ACL. DOI: 10.18653/v1/P16-1141.
- Hearst, Marti A. 1997. TextTiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics* 23(1). 33–64.
- Kulkarni, Vivek, Rami Al-Rfou, Bryan Perozzi & Steven Skiena. 2015. Statistically significant detection of linguistic change. In *Proceedings of the 24th international conference on the World Wide Web*, 625–635. Florence: ACM. DOI: 10.1145/2736277.2741627.
- Levy, Omer & Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, 2177–2185.
- Lieberman, Erez, Jean-Baptiste Michel, Joe Jackson, Tina Tang & Martin A. Nowak. 2007. Quantifying the evolutionary dynamics of language. *Nature* 449(7163). 713.
- Lund, Kevin & Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers* 28(2). 203–208.
- Mikolov, Tomas, Kai Chen, Greg Corrado & Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint 1301.3781*.
- Pagel, Mark, Quentin D. Atkinson & Andrew Meade. 2007. Frequency of word-use predicts rates of lexical evolution throughout Indo-European history. *Nature* 449(7163). 717.
- Pennington, Jeffrey, Richard Socher & Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP 2014*, 1532–1543. Doha: ACL. DOI: 10.3115/v1/D14-1162.
- Pevzner, Lev & Marti A. Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics* 28(1). 19–36.
- Riedl, Martin & Chris Biemann. 2012. Text segmentation with topic models. *Journal for Language Technology and Computational Linguistics* 27. 47–69.
- Ruef, Martin. 1999. Social ontology and the dynamics of organizational forms: Creating market actors in the healthcare field, 1966–1994. *Social Forces* 77(4). 1403–1432.
- Schätzle, Christin & Hannah Booth. 2019. DiaHClust: An iterative hierarchical clustering approach for identifying stages in language change. In *Proceedings*

- of the 1st international workshop on computational approaches to historical language change, 126–135.
- Sehikh, Imran, Dominique Fohr & Irina Illina. 2017. Topic segmentation in ASR transcripts using bidirectional RNNs for change detection. In *2017 IEEE automatic speech recognition and understanding workshop (ASRU)*, 512–518. IEEE.
- Tahmasebi, Nina, Lars Borin & Adam Jatowt. 2021. Survey of computational approaches to lexical semantic change detection. In Nina Tahmasebi, Lars Borin, Adam Jatowt, Yang Xu & Simon Hengchen (eds.), *Computational approaches to semantic change*, 1–91. Berlin: Language Science Press. DOI: 10.5281/zenodo.5040302.
- Tseng, Paul. 2001. Convergence of a block coordinate descent method for non-differentiable minimization. *Journal of optimization theory and applications* 109(3). 475–494.
- Utiyama, Masao & Hitoshi Isahara. 2001. A statistical model for domain-independent text segmentation. In *Proceedings of the 39th annual meeting of the Association for Computational Linguistics*, 499–506.
- Wang, Xuerui & Andrew McCallum. 2006. Topics over time: A non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD international conference on knowledge discovery and data mining*, 424–433. ACM. DOI: 10.1145/1150402.1150450.
- Yao, Zijun, Yifan Sun, Weicong Ding, Nikhil Rao & Hui Xiong. 2018. Dynamic word embeddings for evolving semantic discovery. In *Proceedings of the eleventh ACM international conference on web search and data mining*, 673–681.
- Zhang, Yating, Adam Jatowt, Sourav Bhowmick & Katsumi Tanaka. 2015. Omnia mutantur, nihil interit: Connecting past with present by finding corresponding terms across time. In *Proceedings of ACL/IJCNLP 2015 (Volume 1: Long papers)*, 645–655. Beijing: ACL. DOI: 10.3115/v1/P15-1063.

