Diachronic Linguistic Periodization of Temporal Document Collections for Discovering Evolutionary Word Semantics

Yijun Duan[1(✉)], Adam Jatowt[2], Masatoshi Yoshikawa[3], Xin Liu[1], and Akiyoshi Matono[1]

[1] AIST, Tsukuba, Japan
{yijun.duan,xin.liu,a.matono}@aist.go.jp
[2] Department of Computer Science, University of Innsbruck, Innsbruck, Austria
jatowt@acm.org
[3] Graduate School of Informatics, Kyoto University, Kyoto, Japan
yoshikawa@i.kyoto-u.ac.jp

Abstract. Language is our main communication tool. Deep understanding of its *evolution* is imperative for many related research areas including history, humanities, social sciences, etc. To this end, we are interested in the task of segmenting long-term document archives into naturally coherent periods based on the evolving word semantics. There are many benefits of such segmentation such as 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. Encouraging effectiveness of our model is demonstrated on the New York Times corpus spanning from 1990 to 2016.

Keywords: Dynamic word embedding · Temporal document segmentation · Knowledge discovery in digital history

1 Introduction

Language is an evolving and dynamic construct. The awareness of the necessity and possibilities of large scale analysis of the temporal dynamics on linguistic phenomena has increased considerably in the last decade [26,29,30]. 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.

It is intuitive that, if an IR system is required to effectively return information from a target time period T_a in the past, it may fail to do so if it is unable to capture the change in context between T_a and the current time T_b. To which extent is the context of T_a different from that of T_b? Are there any turning points

H.-R. Ke et al. (Eds.): ICADL 2021, LNCS 13133, pp. 3–17, 2021.
https://doi.org/10.1007/978-3-030-91669-5_1

in the interval between T_a and T_b when a significant context change occurred, or rather do T_a and T_b belong to the same stage in the evolving process of language? Being capable 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 and text mining.

Traditionally, a language's diachrony is segmented into pre-determined periods (e.g., the "Old", "Middle" and "Modern" eras for English) [24], 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 as well as can be easily obscured and derived from arbitrary and non-linguistic features [7]. Thanks to accumulated large amounts of digitized documents from the past, it is possible now to employ large scale data-driven analyses for uncovering patterns of language change. In this study, we propose a data-driven approach for segmenting a temporal document collection (e.g., a long-term news article archive) into natural, linguistically coherent periods. Based on our method, 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 be then applied in diverse time-aware downstream tasks, such as temporal analog retrieval, archival document recommendation, and summarization.

Our method is based on the computation of *dynamic word embeddings*. 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 rainforest in South Africa since around 16th century, and a large e-commerce company since middle 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. Such 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 we assume 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 an identical vector in the same period, while it may have fairly different representations in different periods* (see Fig. 1).

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:

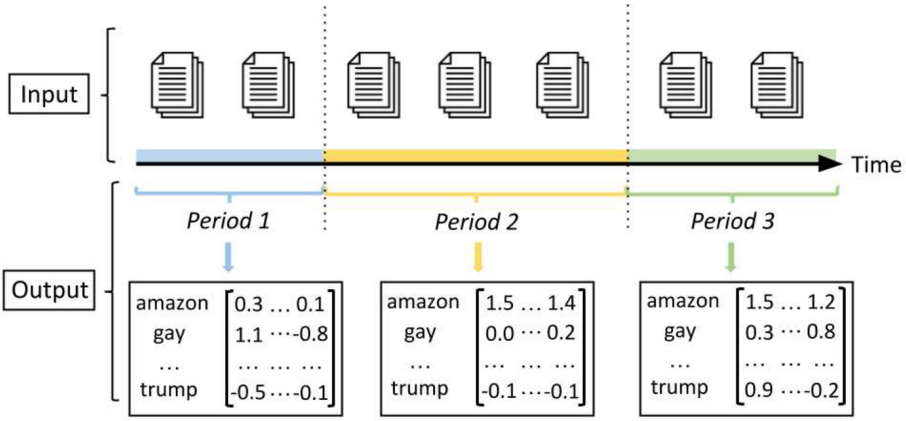

Fig. 1. Conceptual view of our task. Our goal is to identify *latent periods* in the input document collection, such that word semantics are relatively stable within the same period (i.e., a word is represented by the same embedding vector), and major linguistic shifts exist between different periods (i.e., a word may be represented by fairly different vectors in different periods).

a. How to compute temporal-aware word embeddings (**Task 1**)?
b. How to split the document collection based on learned word embeddings (**Task 2**)?

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 Sect. 3) as anchors for aligning word embeddings of all time to the same latent space. 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 get the optimal solution, we employ three classes of 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 [29], which contains a total of 99,872 articles published between January 1990 and July 2016. To evaluate the periodization effectiveness, we construct the test sets by utilizing New York Times article tags (see Sect. 5), and evaluate the analyzed methods based on two standard metrics: Pk [2] and WinDiff [22], which are commonly reported 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 techniques for document collection periodization.
– We perform extensive experiments on the New York Times corpus, by which the encouraging effectiveness of our approach is demonstrated.

2 Problem Definition

We start by presenting the formal problem definition.

Input: The input are documents published across time. Formally, let $D = \{D_1, D_2, ..., D_N\}$ denote the entire article set where D_i represents the subset of documents belonging to the time unit t_i. The length of a time unit can be at different levels of granularity (months, years, etc.)

Task 1: Our first task is to embed each word in the corpus vocabulary $V = \{w_1, w_2, ..., w_{|V|}\}$ [1] into a d-dimensional vector, for each time unit $t_i (i = 1, ..., N)$, respectively. Thus, the expected output is a tensor of size $N \times |V| \times d$, which we denote by A. A_i the embedding matrix for t_i, thus A_i is of size $|V| \times d$.

Task 2: Based on Task 1, our second goal is to split the text corpus D into m latent periods $\Theta = (P_1, P_2, ..., P_m)$ and compute their corresponding word embedding matrix $E_i, i = 1, ..., m$. Each period $P_i = [\tau_b^i, \tau_e^i]$ is expressed by two time points representing its beginning date τ_b^i and the ending date τ_e^i. Let $L(\Theta) = (\tau_b^1, \tau_b^2, ..., \tau_b^m)$ denote the list of beginning dates of all periods, notice that searching for Θ is equivalent to searching for $L(\Theta)$.

3 Temporal Word Embeddings

In this section, we describe our approach for computing dynamic word embeddings (solving **Task 1** in Sect. 2), which captures word semantic evolution across time.

3.1 Learning Static Embeddings

The distributional hypothesis [10] 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 expressed by the co-occurrence statistics of w_i. In this study, we compute the PPMI (positive pointwise mutual information) matrix for obtaining such inter-word co-occurrence information, following previous works [13,16,29]. Moreover, for word vectors v_i and v_j, we should have $\text{PPMI}[i][j] \approx v_i \cdot v_j$, thus static word vectors can be obtained through factorizing the PPMI matrix.

[1] The 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.

3.2 Learning Dynamic Embeddings

We denote PPMI_i as the PPMI matrix for time unit t_i, then word embedding matrix A_i at t_i should satisfy $\text{PPMI}_i \approx A_i \cdot A_i^T$.

However, if A_i is computed 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 units may not be placed in the same latent space. Some previous works [13,15,30] solved this problem by imposing an alignment before any two adjacent matrices A_i and A_{i+1}, resulting in $A_i \approx A_{i+1}, i = 1, ..., N-1$.

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*. Intuitively, if word w did not change its meaning across time (or change its meaning to very small extent), we desire its vector to be close among all temporal embedding matrices. Such words are regarded as *anchors* for aligning 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 in the long-term history, as their dominant meaning are commonly used in everyday life and used by so many people. This assumption is reasonable as it has been reported in many languages including English, Spanish, Russian and Greek [17,20]. 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 time unit t_i. Our joint matrix factorization framework for learning temporal word embeddings is then shown as follows (see Fig. 2 for an illustration):

$$\begin{aligned} A_1, ..., A_N = \arg\min \sum_{i=1}^{N} \left\| \text{PPMI}_i - A_i \cdot A_i^T \right\|_F^2 \\ + \alpha \cdot \sum_{i=1}^{N} \|A_i\|_F^2 + \beta \cdot \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left\| A_i^{SFT} - A_j^{SFT} \right\|_F^2 \end{aligned} \tag{1}$$

Here $\|\cdot\|_F$ represents the Frobenius norm. $\left\|A_i^{SFT} - A_j^{SFT}\right\|_F^2$ is the key smoothing term aligning *shared frequent terms* in all time units, 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. α and β are used to control the weight of different terms to achieve the best factorization performance. Finally, we iteratively solve for A_i by fixing other embedding matrices as constants, and optimizing Eq. (1) using the block coordinate descent method [27].

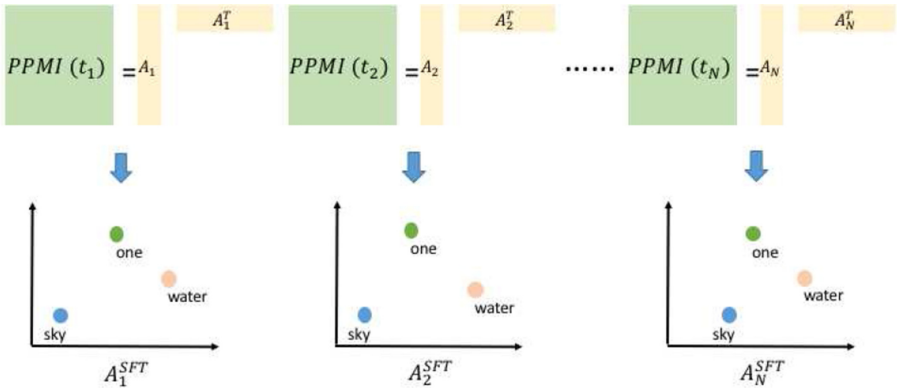

Fig. 2. Illustration of our joint matrix factorization model. *Shared frequent terms* (e.g., sky, one, water) in all time units are aligned to similar positions, which leads word embeddings across time in the same latent semantic space.

4 Document Collection Periodization

In this section, we explain how to split the document collection based on learned temporal word embeddings (solving **Task 2** in Sect. 2).

4.1 Scoring

In general, we prefer the embedding matrices of different periods to be characterized by *high inter-dissimilarity*. Thus, the objective $Obj(\Theta)$ for an overall segmentation is given by aggregating the dissimilarity between all pairs of period-specific embedding matrices, as follows:

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

Here E_i is measured as the average of embeddings in period P_i:

$$E_i = \frac{1}{\tau_e^i - \tau_b^i + 1} \sum_{t=\tau_b^i}^{\tau_e^i} A_t \tag{3}$$

The segmentation that achieves the highest score of Eq. (2) will be adopted.

4.2 Periodization

Greedy Algorithm Based Periodization. At each step, this algorithm inserts a new boundary (which is the beginning date of a new period) to the existing boundaries to locally maximize the objective function, until desired m periods are discovered. The process is formulated in Algorithm 1, where $L(\Theta)^i$ denotes the list of boundaries at the i-th step, and $L(\Theta)^0 = \{t_1\}$.

Algorithm 1: Greedy algorithm based periodization

input : $L(\Theta)^0$; m
output: $L(\Theta)^{m-1}$

for $i \leftarrow 0$ **to** $m-2$ **do**
  $max\_score \leftarrow 0$;
  $next\_boundary \leftarrow 0$;
  for $t_p \leftarrow t_1$ **to** t_N **do**
    ▷`Find the best local boundary`;
    if $t_p \in L(\Theta)^i$ **then**
      continue
    end
    $score \leftarrow Obj(L(\Theta)^i \cup \{t_p\})$;
    if $score > max\_score$ **then**
      $max\_score \leftarrow score$;
      $next\_boundary \leftarrow t_p$;
    end
  end
  $L(\Theta)^{i+1} \leftarrow L(\Theta)^i \cup \{next\_boundary\}$;
end

Dynamic Programming Based Periodization. The core idea of this algorithm is to break the overall problem into a series of simpler smaller segmentation tasks, and then recursively find the solutions to the sub-problems. Let Θ_k^l denotes 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 expressed in Algorithm 2, where $\Theta_1^l = [t_1, t_l]$ and $L(\Theta_1^l) = \{t_1\}, l = 1, ..., N$.

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 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 partially retain its computational benefit. Let $L(\Theta)_G^i[j]$ denote the j-th element in $L(\Theta)^i$ after the i-th greedy search step, the refinement process for finding $L(\Theta)^i[j]$ is shown in Algorithm 3:

Algorithm 2: Dynamic programming based periodization

input : $\mathrm{L}(\Theta_1^l)$, $l = 1, ..., N$; m
output: $L(\Theta_m^N)$

for $row \leftarrow 2$ **to** m **do**
  for $col \leftarrow row$ **to** N **do**
    ▷`Recursively find the solutions to the sub-problems`;
    $max\_score \leftarrow 0$;
    $next\_boundary \leftarrow 0$;
    $subtask \leftarrow 0$;
    for $j \leftarrow row - 1$ **to** $col - 1$ **do**
      $score \leftarrow Obj(L(\Theta_{row-1}^j) \cup \{t_{j+1}\})$;
      if $score > max\_score$ **then**
        $max\_score \leftarrow score$;
        $next\_boundary \leftarrow t_{j+1}$;
        $subtask \leftarrow j$;
      end
    end
    $L(\Theta_{row}^{col}) \leftarrow L(\Theta_{row-1}^{subtask}) \cup \{next\_boundary\}$
  end
end

Algorithm 3: Iterative refinement based periodization

input : $L(\Theta)^0$; m
output: $L(\Theta)^{m-1}$

for $i \leftarrow 0$ **to** $m - 2$ **do**
  $next\_boundary, max\_score \leftarrow Greedy(L(\Theta)^i)$;
  $L(\Theta)^{i+1} \leftarrow L(\Theta)^i \cup \{next\_boundary\}$;
  for $j \leftarrow 1$ **to** i **do**
    ▷`Iteratively refine the previous boundaries`;
    $new\_boundary \leftarrow L(\Theta)^{i+1}[j]$;
    $t_{begin} \leftarrow L(\Theta)^{i+1}[j-1]$;
    $t_{end} \leftarrow L(\Theta)^{i+1}[j+1]$;
    for $t_p \leftarrow t_{begin}$ **to** t_{end} **do**
      $score \leftarrow Obj(L(\Theta)^{i+1} - L(\Theta)^{i+1}[j] \cup \{t_p\})$;
      if $score > max\_score$ **then**
        $max\_score \leftarrow score$;
        $next\_boundary \leftarrow t_p$;
      end
    end
    $L(\Theta)^{i+1} \leftarrow (L(\Theta)^{i+1} - L(\Theta)^{i+1}[j]) \cup \{new\_boundary\}$;
  end
end

4.3 Analysis of Time Complexity

For greedy periodization, it requires $m-1$ steps and the i-th step calls scoring function Eq. (2) $N-i$ times. In total, it is $O(Nm-N-m^2+m/2)$. 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 Eq. (2). 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 Periodization Effectiveness

5.1 Datasets

News corpora, which maintain consistency in narrative style and grammar, are naturally advantageous to studying language evolution [29]. We thus perform the experiments on the New York Times Corpus, which has been frequently used to evaluate different researches on temporal information processing or extraction in document archives [4]. The dataset we use [29] 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 units, setting the length of time unit to be 1 year. Stopwords and rare words (which have less than 200 occurrences in entire corpus) were removed beforehand, following the previous work [29,30]. The basic statistics of our dataset are shown in Table 1.

Table 1. Summary of New York Times dataset.

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

5.2 Experimental Settings

For the construction of PPMI matrix, the length of sliding window and the value of embedding dimension is set to be 5 and 50, respectively, following [29]. During the training process of learning dynamic embeddings, the values of parameters α and β (see Eq. (1)) are set to be 20 and 100, respectively, as the result of a grid search. The selection of *shared frequent terms* used as anchors is set to be the top 5% most frequent words in the entire corpus, as suggested by [30].

5.3 Analyzed Methods

Baseline Methods. We test four baselines as listed below.

- **Random:** The segment boundaries are randomly inserted.
- **VNC** [12]**:** A bottom-up hierarchical clustering periodization approach.
- **KLD** [7]**:** An entropy-driven approach which calculates the Kullback-Leibler Divergence (KLD) between term frequency features to segment.

– **CPD** [15]**:** An approach which uses statistically sound change point detection algorithms to detect significant linguistic shifts.

Proposed Methods. We list three proposed methods below (see Sect. 4.2).

– **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.

5.4 Test Sets

As far as we know, there is no standard testsets for New York Time Corpus, we then manually create test sets. The collected news articles dataset is associated with some metadata, including title, author, publish 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.

Naturally, 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 $<w, s, t>$ triplet. A boundary of w is registered if it is assigned to different sections in two adjacent years (i.e., both triplet $<w, s, t>$ and $<w, s', t+1>$ hold and $s \neq s'$). Some examples of words changing their associated section in adjacent years are shown in Table 2.

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 [2] of the New York Times, following previous work [29].

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.

Table 2. Example words changing their associated section for evaluating *periodization effectiveness*.

Word	Year	Section	Year	Section
cd	1990	Arts	1991	Technology
seasoning	2002	Home and Garden	2003	Fashion and Style
zoom	2008	Fashion and Style	2009	Technology
roche	2009	Business	2010	Health
viruses	2009	Health	2010	Science
uninsured	2014	Health	2015	U.S.

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

5.5 Evaluation Metrics

We evaluate the performance of analyzed methods with respect to two standard metrics commonly used in text segmentation tasks: Pk [2] and WinDiff [22]. Both metrics use a sliding window over the document and compare the machine-generated segments with the reference ones. Within each sliding window, if the machine-generated boundary positions are not the same as the reference, Pk will register an error. If the number of boundaries are different, WinDiff will register an error. Both Pk and WinDiff are scaled to the range [0, 1] and equal to 0 if an algorithm assigns all boundaries correctly. The **lower** the scores are, the better the algorithm performs.

5.6 Evaluation Results

Table 3 and Table 4 summarize the Pk and WinDiff scores for each method, respectively. Based on the experimental results we make the following analysis.

- The proposed methods exhibit the overall best performance regarding both Pk and WinDiff. More specifically, they outperform the baselines under 7 of 9 predefined numbers of periods in terms of Pk, and 6 of 9 in terms of WinDiff. Such encouraging observations demonstrate the effectiveness of our proposed periodization frameworks.
- Regarding baseline methods, Random achieves the worst performance. 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 certain computational efficiency for the globally optimal splitting.
- The operation of iterative refinement indeed improves the performance of greedy periodization in some cases, though many results generated by IR-WSE and by G-WSE are the same.

6 Related Work

6.1 Text Segmentation

The most related task to our research problem is text segmentation. Early text segmentation approaches include TextTiling [14] and C99 algorithm [5], which are based on some heuristics on text coherence using a bag of words representation. Furthermore, many attempts adopt topic models to tackle the segmentation task, including [9,23]. [1] is a segmentation algorithm based on time-agnostic

Table 3. Performance comparison by each method using Pk.

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 4. Performance comparison by each method using WinDiff.

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

semantic word embeddings. Most text segmentation methods are unsupervised. However, neural approaches have recently been explored for domain-specific text segmentation tasks, such as [25]. Many text segmentation algorithms are greedy in nature, such as [5,6]. On the other hand, some works search for the optimal splitting for their own objective using dynamic programming [11,28].

6.2 Temporal Word Embeddings

The task of representing words with low-dimensional dense vectors has attracted consistent interest for several decades. Early methods are relying on statistical models [3,18], while in recent years neural models such as word2vec [19], GloVE [21] and BERT [8] have shown great success in many NLP applications. Moreover, it has been demonstrated that both word2vec and GloVE are equivalent to factorizing PMI matrix [16], which primarily motivates our approach.

The above methods assume word representation is static. Recently some works explored computing time-aware embeddings of words, for analyzing linguistic change and evolution [13,15,29,30]. 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 [15,30], imposing an orthogonal transformation [13] or jointly smoothing every pair of adjacent time slices [29]. 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.

7 Conclusion

This work approaches a novel and challenging research problem - *diachronic linguistic periodization of temporal document collections*. 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 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 three effective embedding-based periodization algorithms. We perform extensive experiments on the commonly-used New York Times corpus, and show that our proposed methods exhibit superior results against diverse competitive baselines.

In future, we plan to detect correlated word semantic changes. We will also consider utilizing word sentiments in archive mining scenarios.

Acknowledgement. This paper is based on results obtained from a project, JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

References

1. Alemi, A.A., Ginsparg, P.: Text segmentation based on semantic word embeddings. arXiv preprint arXiv:1503.05543 (2015)
2. Beeferman, D., Berger, A., Lafferty, J.: Statistical models for text segmentation. Mach. Learn. **34**(1–3), 177–210 (1999)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. J. Mach. Learn. Res. **3**(Jan), 993–1022 (2003)
4. Campos, R., Dias, G., Jorge, A.M., Jatowt, A.: Survey of temporal information retrieval and related applications. ACM Comput. Surv. (CSUR) **47**(2), 1–41 (2014)
5. Choi, F.Y.: Advances in domain independent linear text segmentation. arXiv preprint cs/0003083 (2000)
6. Choi, F.Y., Wiemer-Hastings, P., Moore, J.D.: Latent semantic analysis for text segmentation. In: Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing (2001)
7. Degaetano-Ortlieb, S., Teich, E.: Using relative entropy for detection and analysis of periods of diachronic linguistic change. In: Proceedings of the Second Joint SIGHUM Workshop, pp. 22–33 (2018)
8. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL 2019, pp. 4171–4186 (2019)

9. Du, L., Buntine, W., Johnson, M.: 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, pp. 190–200 (2013)
10. Firth, J.R.: Papers in Linguistics 1934–1951: Repr. Oxford University Press (1961)
11. Fragkou, P., Petridis, V., Kehagias, A.: A dynamic programming algorithm for linear text segmentation. J. Intell. Inf. Syst. **23**(2), 179–197 (2004)
12. Gries, S.T., Hilpert, M.: Variability-based neighbor clustering: a bottom-up approach to periodization in historical linguistics (2012)
13. Hamilton, W.L., Leskovec, J., Jurafsky, D.: 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), Berlin, pp. 1489–1501. Association for Computational Linguistics, August 2016. https://doi.org/10.18653/v1/P16-1141. https://www.aclweb.org/anthology/P16-1141
14. Hearst, M.A.: TextTiling: segmenting text into multi-paragraph subtopic passages. Comput. Linguist. **23**(1), 33–64 (1997)
15. Kulkarni, V., Al-Rfou, R., Perozzi, B., Skiena, S.: Statistically significant detection of linguistic change. In: Proceedings of the 24th International Conference on World Wide Web, pp. 625–635 (2015)
16. Levy, O., Goldberg, Y.: Neural word embedding as implicit matrix factorization. In: Advances in Neural Information Processing Systems, pp. 2177–2185 (2014)
17. Lieberman, E., Michel, J.B., Jackson, J., Tang, T., Nowak, M.A.: Quantifying the evolutionary dynamics of language. Nature **449**(7163), 713 (2007)
18. Lund, K., Burgess, C.: Producing high-dimensional semantic spaces from lexical co-occurrence. Behav. Res. Methods Instrum. Comput. **28**(2), 203–208 (1996)
19. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv:1301.3781 (2013)
20. Pagel, M., Atkinson, Q.D., Meade, A.: Frequency of word-use predicts rates of lexical evolution throughout Indo-European history. Nature **449**(7163), 717 (2007)
21. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
22. Pevzner, L., Hearst, M.A.: A critique and improvement of an evaluation metric for text segmentation. Comput. Linguist. 19–36 (2002)
23. Riedl, M., Biemann, C.: Text segmentation with topic models. J. Lang. Technol. Comput. Linguist. 47–69 (2012)
24. Schätzle, C., Booth, H.: 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, pp. 126–135 (2019)
25. Sehikh, I., Fohr, D., Illina, I.: Topic segmentation in ASR transcripts using bidirectional RNNs for change detection. In: 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pp. 512–518. IEEE (2017)
26. Tahmasebi, N., Borin, L., Jatowt, A.: Survey of computational approaches to diachronic conceptual change. arXiv preprint arXiv:1811.06278 (2018)
27. Tseng, P.: Convergence of a block coordinate descent method for nondifferentiable minimization. J. Optim. Theory Appl. **109**(3), 475–494 (2001)
28. Utiyama, M., Isahara, H.: A statistical model for domain-independent text segmentation. In: Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics, pp. 499–506 (2001)

29. Yao, Z., Sun, Y., Ding, W., Rao, N., Xiong, H.: Dynamic word embeddings for evolving semantic discovery. In: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, pp. 673–681 (2018)
30. Zhang, Y., Jatowt, A., Bhowmick, S., Tanaka, K.: Omnia Mutantur, Nihil Interit: connecting past with present by finding corresponding terms across time. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pp. 645–655 (2015)