# Time Masking for Temporal Language Models

Guy D. Rosin
Technion
Haifa, Israel
guyrosin@cs.technion.ac.il

Ido Guy
Ben-Gurion University of the Negev
Beer-Sheva, Israel
idoguy@acm.org

Kira Radinsky
Technion
Haifa, Israel
kirar@cs.technion.ac.il

## ABSTRACT

Our world is constantly evolving, and so is the content on the web. Consequently, our languages, often said to mirror the world, are dynamic in nature. However, most current contextual language models are static and cannot adapt to changes over time. In this work, we propose a temporal contextual language model called TempoBERT, which uses time as an additional context of texts. Our technique is based on modifying texts with temporal information and performing time masking—specific masking for the supplementary time information. We leverage our approach for the tasks of semantic change detection and sentence time prediction, experimenting on diverse datasets in terms of time, size, genre, and language. Our extensive evaluation shows that both tasks benefit from exploiting time masking.

## CCS CONCEPTS

• **Computing methodologies → Lexical semantics**.

## KEYWORDS

temporal semantics; semantic change detection; language models

## 1 INTRODUCTION

The ability to learn from context is crucial for language understanding and modeling, as proved by the rise of contextual language models in recent years [6, 24]. But what is context? Context has always been textual, i.e., neighboring words [6, 27]. We argue there are other types of context that are valuable for language modeling, and propose to use time as context.

Our language is continuously evolving, especially for content on the web. The rate of change of the web, language, and culture have compressed the time in which critical changes happen [1]. Many important tasks in Natural Language Processing and Information Retrieval, such as web search, must scale not only to the number of documents, but also temporally [14, 17, 34, 36]. Language models

(LMs) are usually pretrained on corpora derived from a snapshot of the web crawled at a specific moment in time [6, 24]. This "static" nature of training prevents language models from adapting to time and generalizing temporally [7, 12, 23, 34].

At the heart of the masked language modeling (MLM) approach is the task of predicting a masked subset of the text given the remaining, unmasked text. In this work, we adapt this approach and propose a simple way to encode time into a language model. We explicitly model temporality by adding time information directly into texts. This modification only yields changes in the input representations, and is thus model agnostic. Most existing temporal models train separate models specialized to different time periods [11]. In contrast, we create a single model that is inherently temporal. In our approach, sequences in the training data are concatenated with a special token of the time period. The concatenation step enables the learning of word representations that are specific to each time period. For example, "<2021> Joe Biden is the President of the United States", where <2021> is a special time token added to the model's vocabulary by our framework to represent the year of 2021. By applying this modification contextual embedding methods can utilize the temporal information attached to the input text. We present several masking techniques to exploit this modification when pretraining LMs, as part of (or in addition to) the standard language modeling objective of BERT [6]. We refer to this process as *time masking*, and to this modification applied to BERT models as *TempoBERT* (Section 3). Figure 1 depicts the training process of TempoBERT, along with different ways to use it for inference.

It was previously argued that the embeddings generated by BERT [6] are by definition dependent on the time-specific context and therefore do not require diachronic fine-tuning [26]. In this work, we challenge this hypothesis by training the language model in a temporal fashion, using time masking. We experiment two tasks—semantic change detection and sentence time prediction.

Semantic change detection is the task of identifying which words undergo semantic changes and to what extent. Most existing contextual methods detect changes by first embedding the target words in each time period, and then either aggregating them to create a time-specific embedding [26], or computing a cluster of the embeddings for each time [9, 22, 25, 28]. The embeddings or clusters are compared to estimate the degree of change between different time periods. These approaches are similar to the traditional static word embedding models, where each word from the predefined vocabulary is presented as a unique vector, and to find the degree of change, the vectors of a word at different times are compared [11, 37]. In this work, we propose a new method to detect changes, exploiting time masking (Section 4.1). Through time masking, it is no longer needed to focus on the target word embedding, but rather learn the context of the time token. TempoBERT detects changes by analyzing the distribution of the times predicted by the transformers.
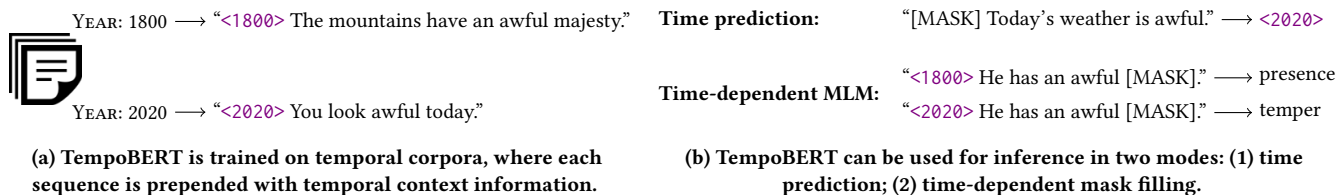
YEAR: 1800 ⟶ "<1800> The mountains have an awful majesty."

YEAR: 2020 ⟶ "<2020> You look awful today."

**(a) TempoBERT is trained on temporal corpora, where each sequence is prepended with temporal context information.**

Time prediction: "[MASK] Today's weather is awful." ⟶ <2020>

Time-dependent MLM: "<1800> He has an awful [MASK]." ⟶ presence
"<2020> He has an awful [MASK]." ⟶ temper

**(b) TempoBERT can be used for inference in two modes: (1) time prediction; (2) time-dependent mask filling.**

**Figure 1: Example of TempoBERT's time masking for training and inference. The word 'awful' changed its meaning in the last two centuries from marvelous to disgusting.**

We experiment with several diverse datasets in terms of time, size, genre, and language. Our empirical results show that TempoBERT outperforms state-of-the-art methods [5, 26, 28, 37].

The second task is sentence time prediction. A textual unit has two main temporal dimensions: its timestamp, or creation time, and its focus time [3]. Both dimensions are important to meaningfully answer temporal queries in Information Retrieval systems. Knowledge of the creation time of texts is essential for many important tasks, such as temporal search, summarization, event extraction, or temporally-focused information extraction. While existing approaches for these tasks assume accurate knowledge of the creation time, it is not always available, especially for arbitrary documents from the Web. For most of the documents on the Web, the timestamp metadata is either missing or cannot be trusted [18]. Thus, predicting the time of textual units (e.g., documents, posts, or sentences) based on their content is an important task [4, 20, 36, 40]. We tackle this task as a multiclass classification problem, with classes defined as time intervals (e.g., years or decades). In this work, we focus on dating sentences, which is a harder task than dating documents, and is a building block of the document dating task [36]. We use time masking to directly predict the writing time of sentences (Section 4.2). We experiment with two datasets of different granularities (i.e., years and decades) and find that TempoBERT outperforms classifiers based on GloVe [30] or BERT [6] embeddings.

Overall, our work offers the following contributions: (1) We propose a simple modification to pretraining that facilitates the acquisition of temporal knowledge through time masking; (2) We conduct evaluations on the tasks of semantic change detection and sentence time prediction that demonstrate the impact of temporally-scoped pretraining; (3) On semantic change detection, we reach state-of-the-art performance on various datasets; (4) We perform a quantitative analysis of time masking for the two evaluation tasks. We also publish our code and data.[1]

## 2 RELATED WORK

*Temporal Language Models.* Language models are usually pretrained on corpora derived from a snapshot of the web crawled at a specific moment in time [6, 24]. This "static" nature of training prevents LMs from adapting to time and generalizing temporally [7, 12, 23, 34]. Hombaiah et al. [12] performed incremental training to better handle continuously evolving web content. Dhingra et al. [7] experimented with temporal language models for question answering. The authors focused on probing LMs for factual knowledge that changes over time, and showed that conditioning

the temporal LM on the temporal context of the text data improves memorization of facts. Others focused on document classification by using word-level temporal embeddings [14] and adapting pretrained BERT models to domain and time [34].

In this work, we focus on the novel concept of temporal contextual representation. We leverage our proposed model for the tasks of semantic change detection and sentence time prediction, which were not studied in the above papers. Furthermore, we learn *temporal contexts* for word embeddings, introduce the concept of time masking, and demonstrate its benefits for temporal LMs.

*Semantic Change Detection.* Semantic change detection is the task of determining whether and to what extent the meaning of a set of target words has changed over time, with the help of time-annotated corpora [21, 39]. This task is often addressed using distributional semantic models: time sensitive word representations are learned and then compared between different time periods [2, 8, 11]. Gonen et al. [10] used a simple nearest-neighbors-based approach to detect semantically-changed words. Del Tredici et al. [5] measured changes of word meaning in online Reddit communities by employing an incremental fine-tuning approach. Montariol et al. [28] used statistical tools to detect semantic change in a scalable way using clusters. Others attempted to simultaneously learn time-aware embeddings over all time periods and resolve the alignment problem using regularization [41], modeling word usage as a function of time Rosenfeld and Erk [33], Bayesian skip-gram model with latent time series [2], and exponential family embeddings [35].

In all of these methods, each word has a single representation for each time period, which limits their sensitivity and interpretability. Recent contextualized architectures allow for overcoming this limitation by taking sentential context into account when inferring word token representations. Indeed, such architectures were applied to diachronic semantic change detection [9, 13, 22, 25, 26]. While all these studies used language models by aggregating the information from the set of the token embeddings, we directly exploit their contextual properties by integrating time *directly* into their training process.

*Sentence Time Prediction.* In the last decade, the exploitation of temporal information to improve document search and exploration received a notable attention from the Information Retrieval community. Specifically, in this context, Temporal Information Extraction is crucial to support retrieval systems [3, 17]. Predicting the time of textual units (e.g., documents, posts, or sentences) based on their content is an important task [4, 20, 36, 40] with numerous approaches. Most existing work uses documents as textual units: One

---

[1]https://github.com/guyrosin/tempobert

of the first studies to model temporal information for the automatic dating of documents is de Jong et al. [4], who used a statistical LM based on word usage statistics over time. Several papers extended this model with temporal entropy [18], KL divergence [20], and n-grams [15]. Niculae et al. [29] tackled this task as a ranking problem. As for deep neural models for document dating, Vashishth et al. [40] used a Graph Convolution Network together with the syntactic and temporal structure of the document, and Ray et al. [32] used an attention-based neural model. Recently, Savov et al. [36] used Ordinal Classification to predict the time of each sentence individually and then aggregated the scores to document level.

In this work, we focus on dating sentences. Sentences are often significantly shorter than documents, making this task potentially harder than dating documents, due to the limited context. In addition, sentence dating can be used as a building block of document dating, as was done in [36]. We use time masking to directly predict the writing time of sentences.

## 3 TIME MASKING

In this work, we use bidirectional language models (also called "contextual language models"). These models represent words as a function of the entire context of a unit of text (e.g., sentence or paragraph), and not only conditioned on previous words as in traditional unidirectional models. Formally, given an input sequence of tokens $w = [w_1, w_2, \ldots, w_n]$ and a position $1 <= i <= n$, bidirectional LMs estimate $p(w_i) = p(w_i \mid w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n)$ using the left and right context of that word. To estimate this probability, BERT [6] samples positions in the input sequence randomly and learns to fill the word at the masked position. To this end, a Transformer architecture is employed.

To adapt language models to time, we propose to directly encode temporality information in the training process. We design a modification to the training process designed specifically for contextual representations: rather than including time information for each word as in existing approaches (e.g., temporal referencing [8]), we concatenate a special time token to each textual sequence. For example, "<2021> Joe Biden is the President of the USA", where "<2021>" is a special time token added to the model's vocabulary by our framework to represent the year of 2021. By performing this modification at the sequence level instead of at the token level, we avoid adding time-specific variants of each token to the vocabulary, thus increasing its size significantly. In addition, contextual embedding methods can utilize this temporal information attached to the input text. The time token denotes the time at which the sequence was written or a point in time at which its assertion is valid.

In addition to encoding time into the model and thus making the model temporal, time masking enables predicting the writing time of texts. The same way prompts [16, 31] exploit the masked language model to fill mask tokens for the task of question answering, we can fill mask time tokens to predict the time of the sequence. We exploit this feature in both tasks we experiment with in this work: semantic change detection and sentence time prediction. For example, to predict the time of the sentence "Joe Biden is the President of the USA", we prefix it with a mask time token: "[MASK] Joe Biden is the President of the USA" and feed it into the model. The model is likely to fill the mask token with a time token, since

it was pretrained on such sentences. Thus, the expected token is the relevant time token to Biden's presidency, e.g., "<2021>".

Formally, given a sequence $w$ and a time $t$, we denote $w_0 = t$ for clarity and prepend the time token $t$ to $w$. Now the inputs to the model are a sequence $w = [t, w_1, w_2, \ldots, w_n]$ and a position $0 <= i <= n$. Let us analyze the two options for $i$: (1) If $1 <= i <= n$, then $w_i$ is one of the original tokens in the sentence. In this case, $p(w_i) = p(w_i \mid t, w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n)$, i.e., $p(w_i)$ is computed based on both the textual context and the temporal context. (2) If $i = 0$, then $w_i$ is $t$. In this case, $p(w_i) = p(t) = p(t \mid w_1, \ldots, w_n)$, i.e., we predict the time $t$ by the whole input text.

As for the granularity of $t$, different values can be used, according to the use case. In this work, for the sentence time prediction task (Section 4.2), we experiment with granularity of a year and of a decade. For the semantic change detection task (Section 4.1), the corpora are divided into two time buckets, so we use two time tokens, one for each bucket (i.e., <1> and <2>).

Throughout this paper, we use pretrained BERT-base [6] (with 12 layers, 768 hidden dimensions, and 110M parameters)[2], and then post-pretrain it on the temporal corpora with time masking, as described above. We refer to the resulting model as *TempoBERT*.

Finally, time masking can be performed in two ways: The straightforward one is to use the standard MLM, where time tokens are treated like any other token for the masking process. But we can also use a *separate* masking process for time tokens. In this case, we choose time tokens with a custom probability $p_{TM}$; a chosen time token is then either masked (i.e., replaced with the mask token [MASK]) 80% of the time, replaced by a random time token 10% of the time, or kept unchanged 10% of the time (we follow Devlin et al. [6] for the 80-10-10 probabilities). See Section 7.1.2 for an analysis of the different settings.

## 4 APPLICATIONS FOR TEMPORAL CONTEXTUAL REPRESENTATIONS

### 4.1 Semantic Change Detection

Words can change their meaning, gain or lose senses over time. Knowing whether, and to what degree, a word has changed is key to various tasks, e.g., aiding in understanding historical documents, searching for relevant content, or historical sentiment analysis. In semantic change detection [21, 39], the task is to rank a set of target words according to their degree of semantic change between two time periods $t_1$ and $t_2$.

We propose two scoring functions that utilize TempoBERT for this task (see Section 7.1.1 for an analysis of them): given a target word $w$, they approximate the semantic change it underwent between $t_1$ and $t_2$.

*temporal-cosine.* We generate time specific representations of words and compare them to detect semantic changes. This method was used by Martinc et al. [26]. It is model agnostic and does not explicitly utilize time masking. In detail, we sample $n$ sentences containing the target word $w$ from each time period $t$ and feed them into the model. For each of the sentences, we create a sequence embedding by extracting their last $h$ hidden layers and averaging

---

[2]We use Hugging Face's implementation of BERT: https://github.com/huggingface/transformers.

them. We then extract the embedding of the specific target word. This is the contextual token embedding. Note that these representations vary according to the context in which the token appears, meaning that the same word has a different representation in each specific context (sequence). Following, the resulting embeddings are aggregated at the token level and averaged, in order to get a time specific representation for $w$ for each time $t$, denoted by $e_t(w)$. Finally, we estimate the semantic change of $w$ by measuring the cosine distance ($cos\_dist$) between two time specific representations of the same token:

$$score(w) = cos\_dist(e_{t_1}(w), e_{t_2}(w))$$

*time-diff.* This method is tailored to TempoBERT. We use our model's ability to predict the time of sentences and identify semantic change by looking at the distribution of the predicted times. Specifically, we sample $n$ sentences containing the target word $w$ from each time period and add them all to a set of sentences for $w$, denoted by $S_w$. For each sentence $s \in S_w$, we prepend a mask token and feed it into the model for the task of MLM. The model fills the mask token with the most appropriate token, which in our case is a time token (since during our post-pretraining, the model was trained on sentences starting with a time token). This produces a predicted time for the sentence. Let $p_t(s)$ be the probability of time $t$ to be predicted for $s$. Consider the distribution of these predicted probabilities (for each of the time periods). Intuitively, a uniform distribution means those sentences could have been written at any time period, which means there was no semantic change. A non-uniform distribution implies there was a change that caused the sentences to be written in a specific time period. In our case, since there are just two time periods, we simply use the absolute difference of the probabilities $|p_{t_1}(s) - p_{t_2}(s)|$, so the semantic change score of the target word is set to the average difference of the predicted time probabilities:

$$score(w) = \frac{1}{|S_w|} \sum_{s \in S_w} |p_{t_1}(s) - p_{t_2}(s)|$$

### 4.2 Sentence Time Prediction

As another evaluation of TempoBERT's generalization potential, we examine its use for the task of sentence time prediction; it is a multiclass classification problem, where the goal is to predict the time of a sentence [4, 20, 36, 40]. This task is a building block of the document dating task [36]. Tackling this task with TempoBERT is straightforward: given an input sentence, we prepend a mask token and predict it. In more detail, we feed the prepended sentence into the model for the task of MLM. The model fills the mask token with the most appropriate token, which in our case is a time token (since during our post-pretraining, the model was trained on sentences starting with a time token). This produces a predicted time for the sentence.

## 5 DATASETS

### 5.1 Semantic Change Detection Datasets

For semantic change detection, we experiment with several diachronic data sources, covering a variety of genres, time periods, languages, and sizes. We use the LiverpoolFC corpus [5] and the SemEval-2020 Task 1 corpora [38] (English and Latin). Each dataset is split into two time periods; see Table 1 for the datasets' statistics.

*LiverpoolFC [5].* The LiverpoolFC corpus contains 8 years of Reddit posts from the Liverpool Football Club subreddit. It was created for the task of short-term meaning shift analysis in online communities. The corpus is in English and split into two time periods from 2011-2013 and 2017. We follow Martinc et al. [26] and only conduct light text preprocessing on the LiverpoolFC corpus, where we remove URLs.

Semantic change evaluation dataset: Del Tredici et al. [5] published a set of 97 words from the LiverpoolFC corpus, manually annotated with semantic shift labels by the members of the LiverpoolFC subreddit. Each word was assigned a graded label (between 0 and 1) according to its degree of semantic change. The average of these judgements is used as a gold standard semantic shift index.

*SemEval-2020 Task 1 datasets [38].* The SemEval-2020 Task 1 deals with the semantic change detection task. It introduced datasets in four different languages. In this work, we use the English and Latin datasets. They are all long-term, in contrast to the LiverpoolFC dataset: the English dataset spans two centuries, and the Latin dataset spans more than 2000 years.

Semantic change evaluation dataset: The task contains two subtasks: binary classification of whether a word sense has been gained or lost (subtask 1) and ranking a list of words according to their degree of semantic change (subtask 2). We use the data for subtask 2, which is a set of target words and their graded labels. The target words are balanced for part of speech (POS) and frequency. For the English dataset, we remove POS tags from both the corpus and the evaluation set (as done in previous work [28]).

### 5.2 Sentence Time Prediction Datasets

We use the New York Times archive with 40 years of news articles (from 1981 to 2020, 11 GB of text in total). We use two variants of this corpus (Table 2); one with a granularity of decades (*NYT-decades*) and the other with a granularity of individual years (*NYT-years*). For both of them, we downsample the corpus to 10k sentences per year (1.4M tokens in average, around 1.4 MB of text) to ease computation. The year-granularity dataset contains 40 years, so we have 40 classes for our classification task. To create the decade-granularity dataset, we merge each decade. The resulting dataset contains 4 decades (i.e., four classes), each is composed of 14 MB of text. Each row in the two datasets is composed of a sentence and its associated time (year or decade). For the evaluation, we randomly split both datasets to train and test sets using an 80-20 ratio.

## 6 EXPERIMENTAL SETUP

### 6.1 Semantic Change Detection Setup

*Baseline methods.* We use five baseline methods for semantic change detection: (1) Del Tredici et al. [5] was the first paper to address the notion of short-term semantic change detection. They use a Skip-gram with Negative Sampling model (SGNS) [27] and employ an incremental model fine-tuning approach; the model is first trained on a large Reddit corpus of around 900M tokens (a random crawl from Reddit in 2013) for the initialization of the word vectors, and then fine-tuned on the LiverpoolFC corpus. (2) Schlechtweg

**Table 1: Semantic change detection datasets composition. Each dataset is split into two time periods, denoted by C1 and C2.**

| Dataset | Labelled Words | C1 Source | C1 Time Period | C1 Tokens | C2 Source | C2 Time Period | C2 Tokens |
|---|---|---|---|---|---|---|---|
| LiverpoolFC | 97 | Reddit | 2011–2013 | 8.5M | Reddit | 2017 | 11.9M |
| SemEval-English | 37 | CCOHA | 1810–1860 | 6.5M | CCOHA | 1960–2010 | 6.7M |
| SemEval-Latin | 40 | LatinISE | -200–0 | 1.7M | LatinISE | 0–2000 | 9.4M |

**Table 2: Sentence time prediction datasets composition.**

| Dataset | Source | Time Period | Classes | Avg. Tokens per Period | Avg. Tokens per Sentence |
|---|---|---|---|---|---|
| NYT-decades | NYT Archive | 1981–2020 | 4 | 14M | 22.2 |
| NYT-years | NYT Archive | 1981–2020 | 40 | 1.4M | 22.2 |

et al. [37] is the state-of-the-art semantic change detection method employing non-contextual word embeddings: the Skip-gram with Negative Sampling model is trained on two periods independently and aligned using Orthogonal Procrustes. Cosine distance is used to compute the semantic change. (3) Gonen et al. [10], who use SGNS embeddings along with a simple detection method. For each period, a word is represented by its top nearest neighbors according to cosine distance. Semantic change is then measured as the size of the intersection between the nearest neighbors lists of two periods. (4) Martinc et al. [26] were one of the first to use BERT for semantic change detection, by creating time-specific embeddings of words, and comparing between different times by calculating the cosine distance between the averaged embeddings. Notice, unlike our work they consider time-specific context rather than capture temporal context. They focused on short-term semantic change as well. Their model performs slightly worse than Del Tredici et al. [5], but it is more general and scalable. (5) Montariol et al. [28] generate a set of contextual embeddings using BERT for each word. These representations are clustered, and the derived cluster distributions are compared across time slices. Finally, words are ranked according to the distance measure, assuming that the ranking resembles a relative degree of usage shift. We use their best performing method as reported in [28], which uses affinity propagation for clustering word embeddings and Wasserstein distance as a distance measure between clusters. For the LiverpoolFC dataset, we experiment with all the variants proposed in [28] and choose the best performing one, which uses k-means with $k = 7$ for clustering embeddings, and Wasserstein distance as a distance measure between clusters.

Finally, existing methods for this task that concatenate time information to individual words (i.e., Temporal Referencing [8] and Word Injection [37]) were found [37, 38] to be outperformed by Schlechtweg et al. [37], so we leave them out of this evaluation.

*Our method.* For each dataset, we train a TempoBERT model using a pretrained case-insensitive BERT-base model.[3] Since the vocabulary of this model does not contain all the target words in the evaluation datasets, we add all target words to TempoBERT's vocabulary (to avoid the tokenizer splitting any occurrences of the target words to subwords).

*Metrics.* Semantic change detection performance is measured by the correlation between the semantic shift index (i.e., the ground truth) and the model's semantic shift assessment for each word in the evaluation set. Previous work used different correlation measures for different datasets; for the LiverpoolFC dataset, Pearson's $r$ was used [5, 26], while for the SemEval-2020 datasets it was Spearman's rank-order correlation coefficient $\rho$ [28, 38]. The difference between them is that Spearman's $\rho$ only considers the order of the words, while Pearson's $r$ considers the actual predicted values. In our evaluation, we make an effort to evaluate our methods and the baselines using both Pearson and Spearman correlation, to make the evaluation as comprehensive as possible. There were some cases where we could not reproduce the original authors' results; in such cases we opted to report only the original result.

### 6.2 Sentence Time Prediction Setup

*Baseline Methods.* We use the following baselines: (1) CNN [19]: a convolutional neural network loaded with pretrained GloVe [30] vectors. (2) Bi-LSTM + Attention [43]: an attention-based bidirectional Long Short-Term Memory Network loaded with pretrained GloVe vectors. (3) Linear GloVe: a linear model with one hidden layer of 10 units and a softmax, loaded with pretrained GloVe vectors. (4) Boosting methods based on BERT embeddings (extracted using BERT's last four hidden layers): XGBoost and CatBoost.[4]

*Our method.* For each dataset, we train a TempoBERT model using a pretrained uncased BERT-base model with time masking probability $p_{TM} = 0.9$.

*Metrics.* As sentence time prediction is a multiclass classification task, we report accuracy and macro-F1 scores on the test sets.

### 6.3 Implementation Details

Due to limited computational resources, we train our models with a maximum input sequence length of 128 tokens. To train TempoBERT models, we tune the following hyperparameters for each dataset: learning rate $l \in \{$1e-7, 1e-6, 1e-5, 1e-4, 2e-4, 3e-4$\}$ and number of epochs $e \in \{1, 2, 5, 10, 20\}$. To use TempoBERT for semantic change detection, we use a maximum number of sentences

---

[3]For English: bert-base-uncased from the Hugging Face library, and for Latin: latin-bert from https://github.com/dbamman/latin-bert.

[4]https://github.com/dmlc/xgboost, https://github.com/catboost/catboost

$n = 200$, to ease computation (previous work used all available sentences [26]). Specifically for the temporal-cosine method, we tune the number of last hidden layers to use for embedding extraction $h \in \{1, 4, 12\}$. All experiments were run using four NVIDIA GeForce RTX 2080 Ti GPUs. Runtime varied by experiment; training our best TempoBERT models on each of the datasets took between 30 minutes and 4 hours.

For fine-tuning BERT on labelled data, we train for three epochs with a batch size of 32, which corresponds to the default settings recommended by Devlin et al. [6].
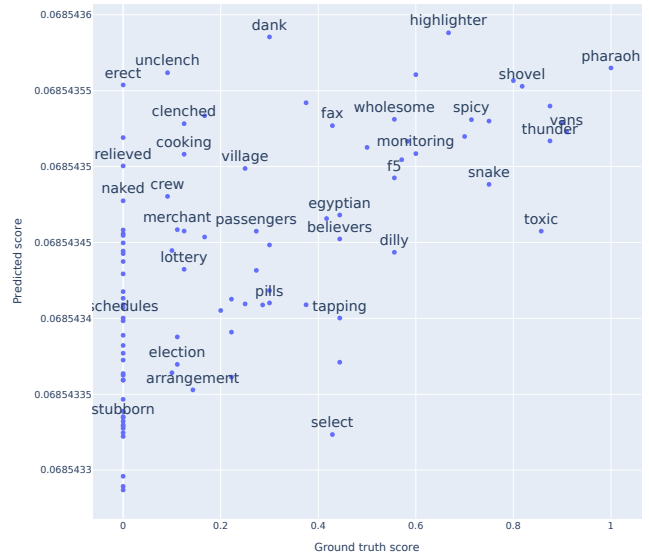
## 7 RESULTS

### 7.1 Semantic Change Detection Results

Table 3 shows the results for semantic change detection on the LiverpoolFC and SemEval datasets. As can be seen, TempoBERT outperforms all the baselines using both Pearson and Spearman correlation with significant correlations ($p < 0.002$), except for the SemEval-English dataset, where it is outperformed by [28] by 0.028, although TempoBERT got a higher Spearman correlation by 0.03. For the LiverpoolFC dataset, we observe a strong correlation (Pearson's $r = 0.637$ and Spearman's $\rho = 0.620$) along with a very large performance gap compared to the baselines (which achieve around 0.490 for both measures, at max). For the SemEval corpora, we observe a moderate correlation (around 0.47–0.54) and a relatively smaller (but still considerable) gap compared to the baselines. The difference originates from the fact that the LiverpoolFC corpus is short-term, i.e., it contains texts from just a few years, in contrast to the SemEval corpora, which span centuries. We further analyze this phenomenon in Section 7.1.1.

Figure 2 shows the Pearson correlation between TempoBERT's scores and the ground truth scores for the LiverpoolFC dataset. The correlation is strong (0.637), so the number of false positives and false negatives is considered small; there are around 10 false positive words (top-left corner), while only one false negative (bottom-right corner). Thus, while we successfully detect changed words, future work should focus on better detecting unchanged words.

*7.1.1 Short-term versus long-term semantic change.* As shown in Table 3, TempoBERT is particularly effective on short-term corpora. TempoBERT encodes time information at the sentence level. As a result, its strength lies in distinguishing between sentences of different times (see the evaluation on sentence time prediction in Section 4.2). For long-term corpora, the writing style between different times (e.g., centuries) is very different; this makes the time information encoded in TempoBERT less valuable.

In long-term periods, there are major difference in writing style, which makes the prediction relatively easier to make. In short time spans, the temporal information encoded into the texts by TempoBERT helps it predict the time accurately and without direct supervision. To demonstrate the difference between short and long term corpora, we include in our evaluation two recent methods that were originally evaluated only on long-term corpora [28, 37]. As showed in Table 3, these two methods achieved lower results compared to both TempoBERT and the two existing methods for short-range semantic change detection [5, 26].



**Figure 2: Semantic change detection on the LiverpoolFC dataset: ground truth scores vs. TempoBERT scores (Pearson's $r = 0.637$).**

Finally, Table 4 shows the comparison results of the two flavors of TempoBERT for semantic change detection, i.e., time-diff and temporal-cosine, for all the corpora, measured by both Pearson and Spearman correlation. We can clearly see that time-diff outperforms temporal-cosine on the short-term LiverpoolFC corpus, while it is outperformed on the long-term SemEval corpora. This further strengthens our hypothesis, as time-diff is based on sentence time prediction, while temporal-cosine is focused on the target word and is thus less affected by context differences (e.g., writing style).

*7.1.2 Time masking experiments.* As described in Section 3, time masking can be performed in two ways: (1) As part of the standard MLM (i.e., the time tokens are treated like any other token); (2) As a separate masking process, where time tokens are masked with a custom masking probability $p_{TM}$.

In this section, we experiment with these time masking techniques: we train TempoBERT models on the LiverpoolFC corpus, each with a different method, and compare the results for the task of semantic change detection. We train all models using a 3e-4 learning rate for just 5 epochs (39K steps), due to computational resource limits.

Figure 3 shows the comparison results, where the points show the Pearson and Spearman correlation for different time masking probabilities ($p_{TM} \in \{0.0, 0.1, \ldots, 0.9, 1.0\}$), and the horizontal lines show the correlations for time masking performed as part of the standard MLM. We observe a positive trend from $p_{TM} = 0$ to 0.3. In particular, for $p_{TM} >= 0.1$, TempoBERT benefits from custom time masking compared to standard MLM. In addition, when training the model without time masking at all (i.e., $p_{TM} = 0$), performance is much lower, as expected, with correlation around 0.2–0.26.
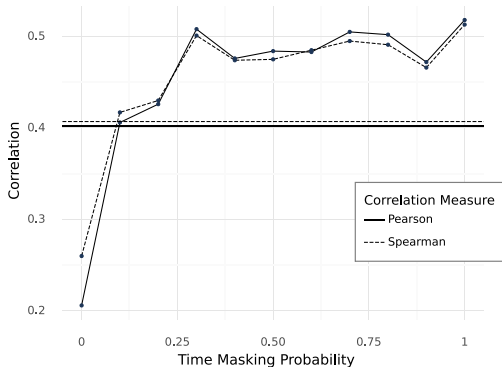
*7.1.3 Comparison of BERT model sizes.* TempoBERT is based on a pretrained BERT model with 768 hidden dimension size and 12

Table 3: Semantic change detection results on LiverpoolFC, SemEval-English, and SemEval-Latin.

| Method | LiverpoolFC | | SemEval-Eng | | SemEval-Lat | |
|---|---|---|---|---|---|---|
| | Pearson | Spearman | Pearson | Spearman | Pearson | Spearman |
| Del Tredici et al. [5] | 0.490 | – | – | – | – | – |
| Schlechtweg et al. [37] | 0.428 | 0.425 | 0.512 | 0.321 | 0.458 | 0.372 |
| Gonen et al. [10] | – | – | 0.504 | 0.277 | 0.417 | 0.273 |
| Martinc et al. [26] | 0.473 | 0.492 | – | 0.315 | – | 0.496 |
| Montariol et al. [28] | 0.378 | 0.376 | **0.566** | 0.437 | – | 0.448 |
| TempoBERT | **0.637** | **0.620** | 0.538 | **0.467** | **0.485** | **0.512** |

Table 4: Semantic change detection results comparing the two TempoBERT methods time-diff and temporal-cosine on Pearson's $r$ and Spearman's $\rho$, on LiverpoolFC, SemEval-English, and SemEval-Latin.

| Method | LiverpoolFC | | SE-Eng | | SE-Lat | |
|---|---|---|---|---|---|---|
| | $r$ | $\rho$ | $r$ | $\rho$ | $r$ | $\rho$ |
| time-diff | **0.637** | **0.620** | 0.208 | 0.381 | 0.385 | 0.425 |
| temporal-cosine | 0.371 | 0.451 | **0.538** | **0.467** | **0.485** | **0.512** |

Table 5: Results for semantic change detection for models with different sizes on LiverpoolFC (measured using Pearson) and SemEval-English (measured using Spearman).

| Method | LiverpoolFC | SemEval-Eng |
|---|---|---|
| Tiny baseline | 0.300 | 0.376 |
| Best baseline | 0.490 | 0.436 |
| TempoBERT-tiny | 0.561 | 0.427 |
| TempoBERT | **0.637** | **0.467** |



Figure 3: Time masking probability vs correlation for semantic change detection. The horizontal lines show the correlations for time masking as part of the standard MLM.

transformer layers. This is the most commonly used setting of BERT and is called BERT-base. In this section, we train TempoBERT-tiny, a variant of TempoBERT that is based on a much smaller pretrained variant of BERT, called BERT-tiny. BERT-tiny has just 4% of the parameters of BERT-base: its hidden dimension is 128, and it contains only 2 transformer layers.

Table 5 shows the comparison results, where we compare TempoBERT with TempoBERT-tiny, the best baseline for each corpus (i.e., Del Tredici et al. [5] for LiverpoolFC, and Montariol et al. [28] for SemEval-English), and a tiny variant of the best BERT-based baselines (i.e., Martinc et al. [26] for LiverpoolFC, and Montariol et al. [28] for SemEval-English). We evaluate only on the English

datasets because a pretrained BERT-tiny model is currently available only for English. As expected, TempoBERT-tiny achieves significant ($p < 1e\text{-}5$) but lower results compared to the standard TempoBERT, but the interesting result is that even TempoBERT-tiny outperforms the best baseline on the LiverpoolFC dataset. On SemEval-English it is outperformed but by a small margin (0.09). In addition, we observe relatively worse performance by the two tiny baselines: Martinc et al. [26] deteriorated from 0.473 to 0.300, and Montariol et al. [28] declined from 0.437 to 0.376. Compared to them, the performance decline from TempoBERT to TempoBERT-tiny is much smaller (0.637 to 0.561, and 0.467 to 0.427). To conclude, the tiny version of TempoBERT shows comparable or better results to the best baselines. We hypothesize that to understand time there is no need to use extremely large models. This is encouraging for both researchers possessing limited computational resources and for Earth. In addition, the tiny version of TempoBERT is relatively stronger compared to tiny versions of the baselines.
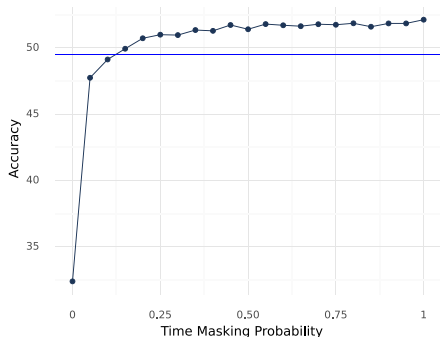
## 7.2 Sentence Time Prediction Results

Table 6 shows the results for sentence time prediction on the two NYT datasets. The two corpora show similar results: TempoBERT outperforms all the baselines. On NYT-decades, the accuracy gap between TempoBERT and the baselines is 4%–8%, while on NYT-years the gap is around 2%–3%. The difference between the two corpora is reasonable, since NYT-years is a much harder task (with 40 classes compared to 4 classes of NYT-decades): we observe x5–x8 higher accuracy for the evaluated methods on NYT-decades compared to NYT-years.

*7.2.1 Time masking experiments.* In this section, we experiment with various ways to perform time masking (Section 3): we train

**Table 6: Sentence time prediction results on NYT (4 decades) and NYT (40 years).**

| Method | NYT-decades | | NYT-years | |
|---|---|---|---|---|
| | Acc | Mac-F1 | Acc | Mac-F1 |
| CNN | 46.73% | 15.53% | 6.44% | 0.39% |
| Bi-LSTM + Attn | 48.62% | 16.01% | 6.69% | 1.13% |
| Linear GloVe | 47.18% | 15.71% | 7.35% | 0.56% |
| XGBoost BERT | 44.83% | 43.75% | 5.95% | 5.77% |
| CatBoost BERT | 46.41% | 45.14% | 6.53% | 6.30% |
| TempoBERT | **52.71%** | **51.28%** | **9.24%** | **8.85%** |

**Table 7: Comparison of different fine-tuned BERT variants for sentence time prediction on NYT (4 decades) and NYT (40 years).**

| Method | NYT-decades | | NYT-years | |
|---|---|---|---|---|
| | Acc | Mac-F1 | Acc | Mac-F1 |
| Fine-tuned BERT (10%) | 49.81% | 48.91% | 6.38% | 5.33% |
| Fine-tuned BERT (25%) | 51.43% | 50.71% | 7.14% | 6.47% |
| Fine-tuned BERT (50%) | 52.98% | 52.37% | 8.71% | 8.40% |
| Fine-tuned BERT (100%) | **53.08%** | **53.07%** | **9.67%** | **9.43%** |
| TempoBERT | 52.71% | 51.28% | 9.24% | 8.85% |



**Figure 4: Accuracy vs time masking probability for sentence time prediction. The horizontal line shows the accuracy for time masking as part of the standard MLM.**

TempoBERT models on the NYT-decades corpus, each with a different time masking method, and compare the results for the task of sentence time prediction. We trained all models for 10 epochs (16K steps), using a 2e-4 learning rate.

Figure 4 shows the comparison results, where the points show the accuracy for different time masking probabilities ($p_{TM} \in \{0.0, 0.05, \ldots, 1.0\}$), and the horizontal line shows the accuracy for time masking performed as part of the standard MLM. First, we observe a positive trend, i.e., TempoBERT improves for sentence time prediction as it performs more time masking. This is reasonable, as the time masking itself is some kind of supervision for this task. Second, when training the model without time masking at all (i.e., $p_{TM} = 0$), performance is much lower, as expected, with 32.38% accuracy. This is reasonable, as the training process in this case does not involve predicting time tokens at all, i.e., the model does not get any supervision for this task (i.e., zero-shot). Finally, when performing time masking as part of the standard MLM, the model achieves 49.50% accuracy. This is worse than performing time masking as a custom masking process with $p_{TM} >= 0.1$.

*7.2.2 Comparison to Fine-tuned BERT.* The sentence time prediction task is a supervised task. As such, our masking approach can be compared to fine-tuning. It has been argued that various types of masking can be equivalent to fine-tuning in terms of performance for supervised tasks, while being more efficient [42]. We wish to examine the different parameters of fine-tuning on the performance

of the sentence time prediction task to shed more light on when masking should be applied compared to fine-tuning. Specifically, we compare TempoBERT to a BERT model fine-tuned for the task of sentence time prediction. Note that the supervision TempoBERT receives in its training process is much limited compared to fine-tuned BERT; fine-tuned BERT is trained directly on the sentence time prediction task, whereas TempoBERT is a general temporal language model that is trained on token time prediction as a small part of its pretraining process.

We experiment with different fine-tuned BERT classifiers, trained on different sizes of data. Table 7 shows the comparison results. TempoBERT reaches similar performance as the fine-tuned BERT at 50% of the data. This shows most of the information of the time can be obtained from learning on only half the corpus but continues improving as data is available. When reaching 100% of the data as compared to 50% of it, we see a relatively smaller improvement.

## 8 CONCLUSION

In this paper, we presented a temporal contextual language model called TempoBERT which uses time as an additional context of texts. Our technique is based on modifying texts with time information and performing time masking—specific masking for the added time information. Using time masking, we model time as part of the context of texts. This enables us to both fill mask tokens while conditioning on time, and to predict the writing time of sentences. We experimented with two tasks: semantic change detection and sentence time prediction, and showed that both benefit from time masking. For semantic change detection, we proposed a method exploiting time masking that is especially strong for short-term corpora and outperformed existing state-of-the-art methods on diverse datasets, in terms of time, size, genre, and language. In addition, we experimented with small-sized TempoBERT models and showed their relatively strong performance on this task. For sentence time prediction, time masking allows us to directly predict the time of sentences without additional fine-tuning. We showed strong performance compared to various baselines and studied different time masking settings to demonstrate the contribution of time masking to this task.

## 9 ACKNOWLEDGEMENTS

# REFERENCES

[1] Eytan Adar, Jaime Teevan, Susan T Dumais, and Jonathan L Elsas. 2009. The web changes everything: understanding the dynamics of web content. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*. 282–291.

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

[3] Ricardo Campos, Gaël Dias, Alípio M Jorge, and Adam Jatowt. 2014. Survey of temporal information retrieval and related applications. *ACM Computing Surveys (CSUR)* 47, 2 (2014), 1–41.

[4] Franciska de Jong, Henning Rode, and Djoerd Hiemstra. 2005. Temporal language models for the disclosure of historical text. *Humanities, Computers and Cultural Heritage* (2005), 161.

[5] Marco Del Tredici, Raquel Fernández, and Gemma Boleda. 2019. Short-Term Meaning Shift: A Distributional Exploration. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2069–2075.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 4171–4186.

[7] Bhuwan Dhingra, Jeremy R Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W Cohen. 2021. Time-Aware Language Models as Temporal Knowledge Bases. *arXiv preprint arXiv:2106.15110* (2021).

[8] Haim Dubossarsky, Simon Hengchen, Nina Tahmasebi, and Dominik Schlechtweg. 2019. Time-Out: Temporal Referencing for Robust Modeling of Lexical Semantic Change. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 457–470.

[9] Mario Giulianelli, Marco Del Tredici, and Raquel Fernández. 2020. Analysing Lexical Semantic Change with Contextualised Word Representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 3960–3973.

[10] Hila Gonen, Ganesh Jawahar, Djamé Seddah, and 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.

[11] 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)*, Vol. 1. 1489–1501.

[12] Spurthi Amba Hombaiah, Tao Chen, Mingyang Zhang, Michael Bendersky, and Marc Najork. 2021. Dynamic Language Models for Continuously Evolving Content. In *KDD 2021*.

[13] Renfen Hu, Shen Li, and Shichen Liang. 2019. Diachronic sense modeling with deep contextualized word embeddings: An ecological view. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 3899–3908.

[14] Xiaolei Huang and Michael Paul. 2019. Neural temporality adaptation for document classification: Diachronic word embeddings and domain adaptation models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 4113–4123.

[15] Adam Jatowt and Ricardo Campos. 2017. Interactive system for reasoning about document age. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 2471–2474.

[16] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics* 8 (2020), 423–438.

[17] Nattiya Kanhabua and Avishek Anand. 2016. Temporal information retrieval. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*. 1235–1238.

[18] Nattiya Kanhabua and Kjetil Nørvåg. 2008. Improving temporal language models for determining time of non-timestamped documents. In *International conference on theory and practice of digital libraries*. Springer, 358–370.

[19] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1746–1751. https://doi.org/10.3115/v1/D14-1181

[20] Abhimanu Kumar, Matthew Lease, and Jason Baldridge. 2011. Supervised language modeling for temporal resolution of texts. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. 2069–2072.

[21] 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*. 1384–1397.

[22] Severin Laicher, Sinan Kurtyigit, Dominik Schlechtweg, Jonas Kuhn, and Sabine Schulte im Walde. 2021. Explaining and Improving BERT Performance on Lexical Semantic Change Detection. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*. 192–202.

[23] Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Sebastian Ruder, Dani Yogatama, et al. 2021. Pitfalls of Static Language Modelling. *arXiv preprint arXiv:2102.01951* (2021).

[24] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[25] Matej Martinc, Syrielle Montariol, Elaine Zosa, and Lidia Pivovarova. 2020. Capturing Evolution in Word Usage: Just Add More Clusters?. In *Companion Proceedings of the Web Conference 2020*. 343–349.

[26] Matej Martinc, Petra Kralj Novak, and Senja Pollak. 2020. Leveraging Contextual Embeddings for Detecting Diachronic Semantic Shift. In *Proceedings of the 12th Language Resources and Evaluation Conference*. 4811–4819.

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

[28] Syrielle Montariol, Matej Martinc, and Lidia Pivovarova. 2021. Scalable and Interpretable Semantic Change Detection. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 4642–4652.

[29] Vlad Niculae, Marcos Zampieri, Liviu P Dinu, and Alina Maria Ciobanu. 2014. Temporal text ranking and automatic dating of texts. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*. 17–21.

[30] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[31] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language Models as Knowledge Bases?. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 2463–2473.

[32] Swayambhu Nath Ray, Shib Sankar Dasgupta, and Partha Talukdar. 2018. AD3: Attentive Deep Document Dater. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 1871–1880.

[33] Alex Rosenfeld and Katrin Erk. 2018. Deep neural models of semantic shift. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 474–484.

[34] Paul Röttger and Janet B Pierrehumbert. 2021. Temporal Adaptation of BERT and Performance on Downstream Document Classification: Insights from Social Media. *arXiv preprint arXiv:2104.08116* (2021).

[35] Maja Rudolph and David Blei. 2018. Dynamic Embeddings for Language Evolution. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 1003–1011.

[36] Pavel Savov, Adam Jatowt, and Radoslaw Nielek. 2021. Predicting the Age of Scientific Papers. In *International Conference on Computational Science*. Springer, 728–735.

[37] Dominik Schlechtweg, Anna Hätty, Marco Del Tredici, and Sabine Schulte im Walde. 2019. A Wind of Change: Detecting and Evaluating Lexical Semantic Change across Times and Domains. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 732–746.

[38] Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. 1–23.

[39] Nina Tahmasebi, Lars Borin, and Adam Jatowt. 2018. Survey of computational approaches to lexical semantic change. *arXiv preprint arXiv:1811.06278* (2018).

[40] Shikhar Vashishth, Shib Sankar Dasgupta, Swayambhu Nath Ray, and Partha Talukdar. 2018. Dating Documents using Graph Convolution Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1605–1615.

[41] 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*. ACM, 673–681.

[42] Mengjie Zhao, Tao Lin, Fei Mi, Martin Jaggi, and Hinrich Schütze. 2020. Masking as an Efficient Alternative to Finetuning for Pretrained Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2226–2241.

[43] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*. 207–212.