Sequential Modelling of the Evolution of Word Representations for Semantic Change Detection

Adam Tsakalidis[1,3] Maria Liakata[1,2,3]

[1]Queen Mary University of London, London, UK
[2]University of Warwick, Coventry, UK
[3]The Alan Turing Institute, London, UK
{atsakalidis,mliakata}@turing.ac.uk

Abstract

Semantic change detection concerns the task of identifying words whose meaning has changed over time. Current state-of-the-art approaches operating on neural embeddings detect the level of semantic change in a word by comparing its vector representation in two distinct time periods, without considering its evolution through time. In this work, we propose three variants of sequential models for detecting semantically shifted words, effectively accounting for the changes in the word representations over time. Through extensive experimentation under various settings with synthetic and real data we showcase the importance of sequential modelling of word vectors through time for semantic change detection. Finally, we compare different approaches in a quantitative manner, demonstrating that temporal modelling of word representations yields a clear-cut advantage in performance.

1 Introduction

Identifying words whose lexical meaning has changed over time is a primary area of research at the intersection of natural language processing and historical linguistics. Through the evolution of language, the task of "semantic change detection" (Tahmasebi et al., 2018; Tang, 2018; Kutuzov et al., 2018) can provide valuable insights on cultural evolution over time (Michel et al., 2011). Measuring linguistic change is also relevant to understanding the dynamics in online communities (Danescu-Niculescu-Mizil et al., 2013) and the evolution of individuals (McAuley and Leskovec, 2013). Recent years have seen a surge in interest in this area since researchers are now able to leverage the increasing availability of historical corpora in digital form and develop models that detect the shift in a word's meaning through time.

However, two key challenges in the field still remain. Firstly, there is little work in existing literature on model comparison (Schlechtweg et al., 2019; Dubossarsky et al., 2019; Shoemark et al., 2019). Partially due to the lack of (longitudinal) labelled datasets, existing work assesses model performance mainly in a qualitative manner, without quantitative comparisons against prior work. Therefore, it becomes difficult to assess *what constitutes an appropriate approach for semantic change detection*. Secondly, on a methodological front, a large body of related work detects semantically shifted words by pairwise comparisons of their representations in distinct time periods, *ignoring the sequential modelling aspect of the task*. Since semantic change is a time-sensitive process (Tsakalidis et al., 2019), considering consecutive vector representations through time – instead of two bins of word representations (Schlechtweg et al., 2018, 2020) – can be crucial to improving model performance (Shoemark et al., 2019).

Here we tackle both challenges by approaching semantic change detection as an anomaly identification task. Working on embedding representations of words in the English language, we learn their evolution through time via an encoder-decoder architecture. We hypothesize that once such a model has been successfully trained on temporally sensitive sequences of word representations, it will accurately predict the evolution of the semantic representation of any word through time. Words that have undergone semantic change will be those that yield the highest errors by the prediction model. Our work makes the following contributions:

- we develop three variants of an LSTM-based architecture to measure the level of semantic change of a word by tracking its evolution through time in a sequential manner: (a) a word representation autoencoder, (b) a future word representation decoder and (c) a hybrid approach combining (a) and (b);

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 8485–8497,
November 16–20, 2020.

- we show the effectiveness of our models under thorough experimentation with synthetic data;
- we compare our models against current practices and competitive baselines using real data, demonstrating important gains in performance and highlighting the importance of sequential modelling of word vectors through time;
- we release our code, to help set up a benchmark for model comparison within the domain in a quantitative fashion.[1]

2 Related Work

One can distinguish two directions within the literature on semantic change detection: (a) learning word representations over discrete time intervals (bins) and comparing the resulting vectors and (b) jointly learning word representations across time (Bamler and Mandt, 2017; Rosenfeld and Erk, 2018; Yao et al., 2018; Rudolph and Blei, 2018). Such representations can be generated via different approaches, such as topic- (Frermann and Lapata, 2016; Perrone et al., 2019), graph- (Mitra et al., 2014) and neural-based models (e.g., word2vec) – work by Tahmasebi et al. (2018) provides an overview of such approaches. In this work we focus on (a) due to scalability issues in learning diachronic representations from very large corpora, as in our case, and – without loss of generality – we utilise pre-trained, neural-based representations.

Related work in (a) derives word representations W_i ($i \in [0,..,|T-1|]$) across $|T|$ time intervals and performs pairwise comparisons for different values of i. Early work used frequency- or co-occurrence-based representations (Sagi et al., 2009; Cook and Stevenson, 2010; Gulordava and Baroni, 2011; Mihalcea and Nastase, 2012). However, leveraging word2vec-based representations (Mikolov et al., 2013) has become the common practice in recent years. Due to the stochastic nature of word2vec, Orthogonal Procrustes (OP) (Schönemann, 1966) is often applied to the resulting vectors, aiming at aligning the pairwise representations (Kulkarni et al., 2015; Hamilton et al., 2016; Del Tredici et al., 2019; Shoemark et al., 2019; Tsakalidis et al., 2019; Schlechtweg et al., 2019). Given two word matrices W_k, W_j at times k and j respectively, OP finds the optimal transformation matrix $R = \underset{\Omega;\Omega^T\Omega=I}{argmin} \|\Omega W_k - W_j\|_F$ and the semantic shift level of a word w during this time interval is defined as the cosine distance between the two aligned matrices (Hamilton et al., 2016). By operating in a linear pairwise fashion, such approaches ignore the time-sensitive and possibly non-linear nature of semantic change.

By contrast, Kim et al. (2014), Kulkarni et al. (2015), Dubossarsky et al. (2019) and Shoemark et al. (2019) derive time series of a word's level of semantic change to detect semantically shifted words. Even though these methods incorporate temporal modelling, they either rely heavily on the linear transformation R (Kulkarni et al., 2015; Shoemark et al., 2019) or focus primarily on the *generation* of temporally-sensitive representations as a means towards capturing semantic change (Kim et al., 2014; Dubossarsky et al., 2019). A key contribution of our work is that we do not base our methods on pre-defined transformations, but instead propose a model for learning how (any type of) pre-trained word representations vary across time, effectively exploiting the full sequence of a word's evolution.

Finally, the comparative evaluation of semantic change detection models is still in its infancy. Most related work assesses model performance based on artificial tasks (Cook and Stevenson, 2010; Kulkarni et al., 2015; Rosenfeld and Erk, 2018; Dubossarsky et al., 2019; Shoemark et al., 2019) or on a few hand-picked examples (Sagi et al., 2009), without cross-model comparison. The recently introduced shared tasks SemEval Task 1 (Schlechtweg et al., 2020) and DIACR-Ita (Basile et al., 2020) aim at bridging this gap; however, the respective datasets consist of documents split in two distinct time periods, thus not facilitating the study of the sequential nature of semantic change. Setting a benchmark for model comparison with real-world and sequential word representations is crucial in this field.

3 Methods

We formulate semantic change detection as an anomaly detection task in the evolution of pre-trained word embeddings. We assume that pre-trained word vectors $W_t \in [W_0, ..., W_{|T-1|}]$, where $W_t \in R^{|V|\times d}$ ($|V|$: vocabulary size; d: word representation size) in a historical corpus over $|T|$ time periods, evolve according to a non-linear function

[1] Code is available at: https://github.com/adtsakal/semantic_change_evolution

$f(W_t)$.[2] By approximating f, we obtain the level of semantic shift of a word w at time t by measuring the similarity between its word representation w_t against $f(w_t)$. A low similarity score for a given word implies an inaccurate model prediction (anomaly) and thus a high level of semantic change for the given word. Therefore, we can obtain a ranking of the words based on their semantic shift level by ordering them in ascending order of their similarity scores between w_t and $f(w_t)$. We approximate f via temporally sensitive deep neural models: (a) an autoencoder, which aims to reconstruct a word's trajectory up to a given point in time i $[w_0, ..., w_{|i|}]$ (section 3.1); and (b) a future predictor, which aims to predict future representations of the word $[w_{|i+1|}, ..., w_{|T-1|}]$ (section 3.2). The two models can be trained individually or (c) in a joint multi-task setting (section 3.3). These models benefit from accounting for sequential word representations across time $[W_0, ..., W_{|T-1|}]$, which is better suited for detecting semantically shifted words compared to the common practice of comparing only the first and last word representations $[W_0, W_{|T-1|}]$ (Shoemark et al., 2019).

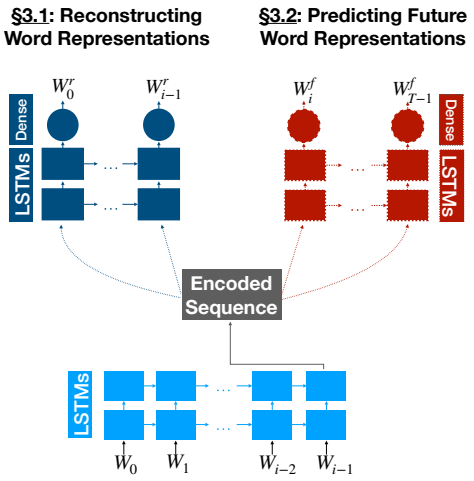


Figure 1: Overview of our proposed model: the sequence of the representation of a set of word vectors (vocabulary) over different time steps $W_{0:i-1}$ is encoded through two LSTM layers and then passed over to a reconstruction (3.1) decoder and a future prediction decoder (3.2). The model is trained by utilising **either decoder** in isolation, or **both** of them in parallel (3.3).

3.1 Reconstructing Word Representations

Given an input sequence of vectors representing the words in a vocabulary across i points in time $W_{0:i-1} = [W_0, W_1, ..., W_{i-1}]$, the goal of the autoencoder is to reconstruct the input sequence $W_{0:i-1}$. Since the task of semantic change includes a natural temporal dimension, we model our autoencoder via RNNs (see Figure 1). The encoder is composed of two LSTM layers (Hochreiter and Schmidhuber, 1997) with Dropout layers operating on their outputs, for regularisation (Srivastava et al., 2014). The first layer encodes the input sequence of $W_{0:i-1}$ and returns the hidden states as input to the second layer. The output of the second layer is the final encoded state, which is then copied $|i|$ times and fed as input to the decoder. The decoder has the same architecture as the encoder, albeit with additional dense layers on top of the second LSTM layer to make the final reconstruction $W^r_{0:i-1}$ on the $|i|$ time steps. The model is trained by minimising the mean squared error (MSE) loss function:

$$L_r = \frac{1}{i}\sum_{j=0}^{i-1} MSE(W_j, W_j^r). \quad (1)$$

After training, the words that yield the highest error rates in a given test set of word representations through time are considered to be the ones whose semantics have changed the most during the given time period. This is compatible with prior work based on word alignment (Hamilton et al., 2016; Tsakalidis et al., 2019), where the alignment error of a word indicates its level of semantic change.

3.2 Predicting Future Word Representations

Reconstructing the word vectors can reveal which words have changed their semantics in the past (i.e., up to time $i-1$, see 3.1). If we are interested in predicting changes in the semantics of future word representations (i.e., after time $i-1$), we can consider a future word representation prediction task: given the sequence of past word representations $W_{0:i-1} = [W_0, W_1, ..., W_{i-1}]$ over the first i time points, we predict the future representations of the words in the vocabulary $W_{i:T-1} = [W_i, W_{i+1}, ..., W_{T-1}]$, for a sequence of overall length $|T|$ (see Figure 1). We follow the same model architecture as in section 3.1, with the only difference being the number of time steps $(T-i)$ used in the decoder to make $|T-i|$ predictions. The model is trained using the loss function L_f:

$$L_f = \frac{1}{T-i}\sum_{j=i}^{T-1} MSE(W_j, W_j^f). \quad (2)$$

[2]Note: t in W_t indicates the time period from when the associated pre-trained word vectors are taken (e.g., year 2000).

3.3 Joint Model

The two models can be combined into a joint one, where, given an input sequence of representations of the vocabulary $W_{0:i-1}$ over i points in time, the goal is both to (a) reconstruct the input sequence and (b) predict the future word $|T-i|$ representations $W_{i:T-1}$. The complete model architecture is provided in Figure 1: the encoder is identical to the one used in 3.1 and 3.2. However, the bottleneck is now copied $|T|$ times and passed to the decoders of the reconstruction ($|i|$ times) and future prediction ($|T-i|$ times). The loss function L_{rf} here is the summation of the individual losses in Eq. 1 and 2:

$$L_{rf} = \frac{1}{i}\sum_{j=0}^{i-1} MSE(W_j, W_j^r) + \frac{1}{T-i}\sum_{j=i}^{T-1} MSE(W_j, W_j^f). \quad (3)$$

There are two main reasons for modelling semantic change in this multi-task setting. Firstly, we benefit from the finer granularity of the two decoders due to their handling of only part of the sequence in a more fine-grained manner, compared to the individual task models. Secondly, the joint model is insensitive to the value of i in Eq. 3 compared to Eq. 1 and 2, as discussed next.

3.4 Model Equivalence

The three models perform different operations; however, setting the operational time periods appropriately in Eq. 1-3 can result in model equivalence (i.e., performing the same task). Specifically, to detect the words whose semantics have changed during [0, $T-1$], the autoencoder (Eq. 1) needs to be fed and reconstruct the full sequence across [0, $T-1$] (i.e., i=T-1). Reducing this interval (reducing i) would limit its operational time period. On the contrary, an increase in the value of i in Eq. 2 of the future prediction model shortens the time period during which it can detect the words whose semantics have changed the most – to detect the words whose semantics have changed within the full sequence [1, $T-1$], it requires only the word representations W_0 in the first time interval. Therefore, setting the parameter i can be crucial for the performance of the individual models. By contrast, the joint model in section 3.3 is able to detect the words that have undergone semantic change, regardless of the value of i (see Eq. 3), since it is still able to operate on the full sequence – we showcase these effects in section 5.2.

4 Experiments with Synthetic Data

Tasks run on artificial data have been used for evaluation purposes in related work (Gale et al., 1992; Schütze, 1998; Cook and Stevenson, 2010; Kulkarni et al., 2015; Rosenfeld and Erk, 2018; Dubossarsky et al., 2019; Shoemark et al., 2019). In this section, we work with artificial data as a proof-of-concept of our proposed models – we compare against state-of-the-art and other baseline methods with real data in the next section. Here we employ a longitudinal dataset of word representations (4.1) and artificially alter the representations of a small set of words across time (4.2). We then train (4.3) our models and evaluate them on the basis of their ability to identify those words that have undergone (artificial) semantic change (4.4).

4.1 Dataset

We employ the UK Web Archive dataset (Tsakalidis et al., 2019), which contains 100-dimensional representations of 47.8K words for each year in the period 2000-2013. These were obtained by employing word2vec (i.e., skip-gram with negative sampling (Mikolov et al., 2013)) on the documents published in each year independently. Note that our models can be applied to any type of pre-trained embeddings. Each year corresponds to a time step in our modelling. The dataset contains 65 words whose meaning has changed within 2001-13, as indicated by the Oxford English Dictionary. These are removed for the purposes of this section, to avoid interference with the artificial data modeling. We use one subset (80%) of the remaining longitudinal word representations for training our models and the rest (20%) for evaluation purposes.

4.2 Artificial Examples of Semantic Change

We generate artificial examples of words with changing semantics, by following a paradigm inspired by Rosenfeld and Erk (2018). We uniformly at random select 5% of the words in the test set to alter their semantics. For every selected "source" word α, we select a "target" word β (details about the selection process of β are provided in the next paragraph). We then alter the representation $w_t^{(\alpha)}$ of the source word α at each point in time t so that it shifts towards the representation $w_t^{(\beta)}$ of the target word at this point in time as:

$$w_t^{*(\alpha)} = \lambda_t w_t^{(\alpha)} + (1 - \lambda_t) w_t^{(\beta)}. \quad (4)$$

Following Rosenfeld and Erk (2018), we model λ_t via a sigmoid function. λ_t receives values within [0, 1] and acts as a decay function that controls the speed of change in the source word's semantics towards the target. Thus, the semantic representation of α is not altered during the first time points and then it gradually shifts towards the representation of β (for middle values of t), where it stabilizes towards the last time points. Since the duration of the semantic shift of a word may vary, we experiment under different scenarios (see "Conditioning on Duration of Change" below). Alternative modelling approaches of artificial semantic change have been presented in Shoemark et al. (2019) – e.g., forcing a word to acquire a new sense while also retaining its original meaning. We opted for the "stronger" case of semantic shift (Eq. 4) as a proof-of-concept for our models. In section 5 we experiment with real-world data, without any assumptions about the form of the function underlying semantic change.

Conditioning on Target Words The selection of the target words should be such that they allow the representation of the source word to change through time (Dubossarsky et al., 2019). This will not be the case if we select a pair of $\{\alpha, \beta\}$ {source, target} words whose representations are very similar (e.g., synonyms). Thus, for each source word α we select uniformly at random a target word β s.t. the cosine similarity of their representations at the initial time point (i.e., in the year 2000) falls within a certain range $(c - 0.1, c]$. Higher values of c enforce a lower semantic change level for α through time, since its representation will be shifted towards a similar word β, and vice versa. To assess model performance across different levels of semantic change, we experiment with varying $c = \{0.0, 0.1, ..., 0.5\}$.

Conditioning on Duration of Change The duration of semantic change affects the value of λ_t in Eq. 4. We conventionally set $\lambda_{07} = 0.5$, s.t. the artificial word representation $w_{07}^{*(\alpha)}$ of a source word α in 2007 (i.e., the midpoint between 2001-2013) to be equal to $0.5(w_{07}^{(\alpha)} + w_{07}^{(\beta)})$. We then experiment with four different duration [start, end] ranges for the semantic change: (a) "Full" [2001-13], (b) "Half" [2005-10], (c) "OT" (One-Third) [2006-09] and (d) "Quarter" [2007-08]. A longer lasting semantic change duration implies a smoother transition of word α towards the meaning of word β, and vice versa (see Figure 2). By generating synthetic examples of varying semantic change duration we are able to measure model performance under different conditions.

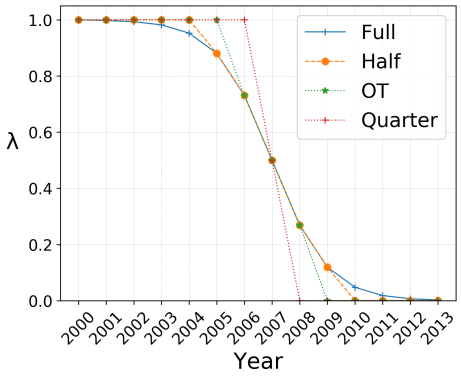


Figure 2: The different functions used to model λ_t in Eq. 4, indicating the speed and duration of semantic change of our synthetic examples (see section 4.2).

4.3 Artificial Data Experiment

Our task is to rank the words in the test set by means of their level of semantic change. We first train our three models on the training set and then we apply them on the test set. Finally, we measure the level of semantic change of a word by means of the average cosine similarity between the predicted and actual word representations at each time step of the decoder. Model performance is assessed via rank-based metrics (Basile and McGillivray, 2018; Tsakalidis et al., 2019; Shoemark et al., 2019).

Model Training We define and train our models as follows:

- `seq2seq`$_r$: the autoencoder (section 3.1) receives and reconstructs the full sequence of the word representations in the training set: $[W_{00}, ..., W_{13}] \rightarrow [W_{00}^r, ..., W_{13}^r]$.
- `seq2seq`$_f$: the future prediction model (section 3.2) receives the representation of the words in the training set in the year 2000 and learns to predict the rest of the sequence: $[W_{00}] \rightarrow [W_{01}^f, ..., W_{13}^f]$.
- `seq2seq`$_{rf}$: the multi-task model (section 3.3) is fed with the first half of the sequence of the word representations in the training set and jointly learns to (a) reconstruct the input sequence and (b) predict the word representations in the future: $[W_{00}, ..., W_{06}] \rightarrow \{[W_{00}^r, ..., W_{06}^r], [W_{07}^f, ..., W_{13}^f]\}$.

We have a different number of timesteps for $\texttt{seq2seq}_r$ and $\texttt{seq2seq}_f$ in their input, so that the decoder in each model operates on the maximum possible output sequence, thus exploiting the semantic change of the words over the whole time period (see section 3.4). $\texttt{seq2seq}_{rf}$ is expected to be insensitive to the number of input time steps, therefore we conventionally set it to half of the overall sequence. We keep 25% of our training set for validation purposes and train our models using the Adam optimiser (Kingma and Ba, 2015). We select the best parameters after 25 trials using the Tree of Parzen Estimators algorithm of the hyperopt module (Bergstra et al., 2013), by means of the maximum average (i.e., per time step) cosine similarity in the validation set.[3]

Testing and Evaluation After training, each model is applied to the test set, yielding its predictions for every word through time.[4] The level of semantic change of a word is then calculated via the average cosine similarity between the actual and the predicted word representations through time, with higher values indicating a better model prediction (thus, a lower level of semantic change). The words are ranked on ascending order of their average cosine similarity, with the first ranks indicating words whose representations have changed the most (low cosine similarity). For evaluation, similarly to Tsakalidis et al. (2019), we employ the average rank across all of the semantically changed words (in %, denoted here as μ_r), with lower scores indicating a better model. We prefer μ_r to the mean reciprocal rank, because the latter emphasises the first rankings. Since semantic change detection is an under-explored task in quantitative terms, we aim at getting better insights on model performance by working with an averaging metric. For the same reason we avoid using classification-based metrics that are based on a cut-off point (e.g., recall at k (Basile and McGillivray, 2018)). We do make use of such metrics in the cross-model comparison with real data (section 5.2).

4.4 Results

Model Comparison Figure 3 presents the results of the three models on our synthetic data across all (c, λ) combinations. $\texttt{seq2seq}_{rf}$ performs consistently better than the individual ($\texttt{seq2seq}_r$, $\texttt{seq2seq}_f$) models in μ_r, showing that combining the two models under a multi-task setting benefits from the joint and finer-grained parameter tuning of the two components. $\texttt{seq2seq}_r$ performs slightly better than $\texttt{seq2seq}_f$, probably due to the autoencoder having to output a longer sequence (i.e., due to W^r_{00}), which helps explore the temporal variation of the words more effectively.

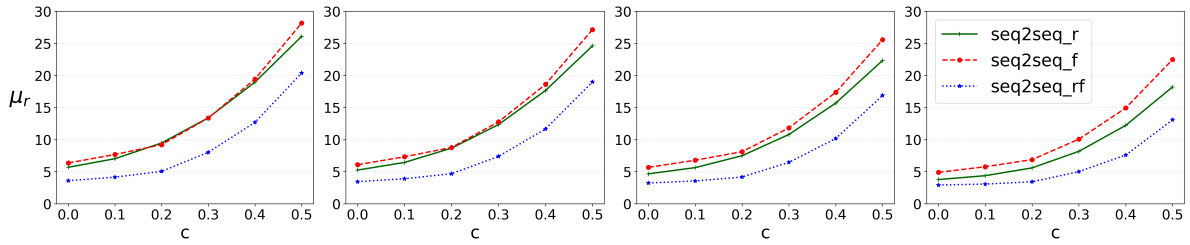
Figure 4 shows the cosine similarity between the predicted and actual representation of each synthetic word per time step for the "Full" case when c=0.0 (highest level of change, see section 4.2). $\texttt{seq2seq}_r$ reconstructs the input sequence of synthetic examples more accurately than the future prediction component (average cosine similarity per year (avg_cos): .65 vs .50). It particularly manages to reconstruct the synthetic word representations *during* the years 2006-2008 ($avg\_cos_{06:08}$=.75), which are the points when λ_t varies more rapidly (see Figure 2); however, it fails to reconstruct equally well their representations before ($avg\_cos_{00:05}$= .65) and after ($avg\_cos_{09:13}$= .59) this sharp change. On the contrary, $\texttt{seq2seq}_f$ predicts more accurately the synthetic word representations during the first years ($avg\_cos_{01:05}$ = .74), when the change in their semantics is minor, but (correctly) fails after the semantic change is almost complete (i.e., when $\lambda_t \leq .25$, $avg\_cos_{09:13}$= .24). $\texttt{seq2seq}_{rf}$ benefits from the individual components' advantage: it appropriately reconstructs the artificial examples in the first years ($avg\_cos_{00:05}$ = .85) so that their semantic shift is highlighted more clearly during ($avg\_cos_{06:08}$= .62) and after the process is almost complete ($avg\_cos_{09:13}$= .26). Finally, avg_cos in $\texttt{seq2seq}_{rf}$ highly correlates with λ_t (ρ=.987), potentially providing insights on how to measure the speed of semantic change of a word.

Effect of Conditioning Parameters Regardless of the duration of the semantic change process, an increase in the value of c results in model performance degradation. This is expected, since the increase of c implies that the level of semantic change of the source words is lower, as discussed in 4.2, thus making the task of detecting them more difficult. Nevertheless, our worst performing model in the most challenging setting (c=0.5, Full, seq2seq$_f$) achieves μ_r=28.17, which is clearly better than the μ_r expected by a random baseline (μ_r=50.00).

The decrease of the duration of semantic change

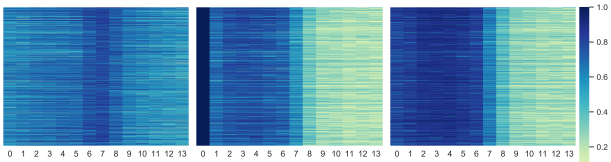
[3] For the complete list of parameters that were tested in all models/baselines in our work, refer to Appendix A.

[4] Note that the future prediction model does not make a prediction for the first time step (year 2000).



(a) Full (b) Half (c) OT (d) Quarter

Figure 3: μ_r of our models on the synthetic dataset for different values of the threshold c (x-axis) and the different periods of duration of semantic change (one per chart, see 4.2). Lower μ_r values indicate a better performance.



(a) $\texttt{seq2seq}_r$ (b) $\texttt{seq2seq}_f$ (c) $\texttt{seq2seq}_{rf}$

Figure 4: Cosine similarity between the actual and the predicted vectors of the synthetic words that have undergone artificial semantic change (rows), per year (columns). Light colours indicate inaccurate model predictions of the word vectors – i.e., indicating that the associated words have undergone semantic change.

has a positive effect on our models (see Figure 3). This is more evident in the cases of high value of c, where $\texttt{seq2seq}_r$ (μ_r: 26.09-18.21 in the Full-to-Quarter cases), $\texttt{seq2seq}_f$ (μ_r: 28.17-22.48) and $\texttt{seq2seq}_{rf}$ (μ_r:20.38-13.09) show clear gains in performance. This indicates that our models can capture the semantic change in small subsequences of the time-series. Studying this effect in datasets of longer duration is an important future direction.

5 Model Comparison with Real Data

5.1 Experimental Setting

We approach the task in a rank-based manner, as in section 4. However, here we are interested in detecting real-world examples of semantic change in words and comparing our models against strong baselines and current practices.

Data and Task We employ the UK Web Archive dataset (see section 4.1). We keep the same 80/20 train/test split as in section 4 and incorporate in the test set the 65 words with known changes in meaning according to the Oxford English Dictionary. We train our models as in section 4.3, aiming at detecting the 65 words in the test set. We use μ_r (as in section 4) and recall at k (Rec@k, k=5%, 10%, 50%) as our evaluation metrics. We refrain from using precision at k, since Oxford English Dictionary is not expected to have a full coverage of the semantically shifted words. Lower μ_r and higher Rec@k scores indicate better models.

Models We compare the three variants from section 3 against four types of baselines:

– *A random word rank generator* (RAND). We report average metrics after 1K runs on the test set.

– *Variants of Procrustes Alignment*, as a common practice in past work: Given word representations in two different years $[W_0, W_i]$ centered around the origin and s.t. $tr(W_k W_k^T) = 1$, PROCR transforms W_i into W_i^* s.t. the squared differences between W_0 and W_i^* are minimised. We also use the $\texttt{PROCR}_k$ and $\texttt{PROCR}_{kt}$ variants (Tsakalidis et al., 2019), which first detect the k most stable words across either $[W_0, W_i]$ ($\texttt{PROCR}_k$) or $[W_0, ..., W_{T-1}]$ ($\texttt{PROCR}_{kt}$) to learn the alignment and then transform W_i into W_i^*. Words are ranked based on the cosine distance between $[W_0, W_i^*]$.

– *Models leveraging the first and last word representations only*. We use a Random Forest (Breiman, 2001) regression model (RF) that predicts W_i, given W_0. We also use the same architectures presented in sections 3.1-3.2, trained on $[W_0, W_i]$ (ignoring the full sequence): $\texttt{LSTM}_r$ reconstructs the sequence $[W_0, W_i]$; $\texttt{LSTM}_f$ predicts W_i, given W_0, similarly to RF. Words are ranked in ascending order of the (average, for $\texttt{LSTM}_r$) cosine similarity between their predicted and actual representations.

– *Models operating on the time series of distances*. Given a sequence of vectors $[W_0, ..., W_i]$, we construct the time series of cosine distances that result by PROCR. Then, we use two global trend models (Shoemark et al., 2019): $\texttt{GT}_c$ ranks the words by means of the absolute value of the Pearson correlation of their time series; $\texttt{GT}_\beta$ fits a linear regression model for every word and ranks the words by the absolute value of the slope. Finally, we

		μ_r		Rec@5		Rec@10		Rec@50	
		'00-'13	avg±std	'00-'13	avg±std	'00-'13	avg±std	'00-'13	avg±std
Past Work/Baselines	RAND	49.97	50.01±0.04	5.00	4.99±0.03	10.01	9.98±0.04	50.02	49.97±0.08
	PROCR	30.63	28.51±2.68	18.46	14.32±5.00	27.69	29.94±4.64	78.46	**80.47±3.79**
	PROCR$_k$	31.01	28.67±2.73	21.54	14.91±4.75	27.69	30.18±4.42	75.38	79.53±4.50
	PROCR$_{kt}$	31.91	28.47±2.85	20.00	14.32±4.23	27.69	28.88±4.45	70.77	80.00±4.53
	RF	30.01	30.45±4.15	10.77	15.62±4.30	21.54	27.46±7.16	78.46	77.63±6.42
	LSTM$_r$	27.87	**27.83±2.65**	12.31	15.98±5.94	29.23	30.30±6.39	80.00	80.12±4.72
	LSTM$_f$	28.62	28.61±3.47	16.92	**17.40±5.60**	32.31	**31.83±6.07**	76.92	78.82±4.83
	GT$_c$	47.87	44.04±1.54	7.69	7.41±2.26	16.92	14.13±3.76	52.31	57.90±2.94
	GT$_\beta$	38.09	36.16±1.74	13.85	14.83±4.14	24.62	23.36±3.94	66.15	69.37±3.26
	PROCR$^*_*$	25.01	27.99±3.03	21.54	15.15 ±4.52	32.31	28.40±3.75	81.54	80.24±3.49
Ours	seq2seq$_r$	24.75	28.36±3.38	21.54	19.05±4.47	38.46	29.94±6.64	**84.62**	81.42±4.64
	seq2seq$_f$	**23.86**	27.17±4.16	26.15	22.01±6.72	**46.15**	34.32±10.13	**84.62**	81.18±5.07
	seq2seq$_{rf}$	24.28	**24.29±0.67**	**29.23**	**25.77±2.28**	36.92	**39.49±2.11**	**84.62**	**85.00±1.16**

Table 1: Model comparison when operating on the entire time sequence (2000-13) and averaged across time (2000-01, ..., 2000-13). Past work and baseline models shown in the table are defined in section 5.1 ("Models").

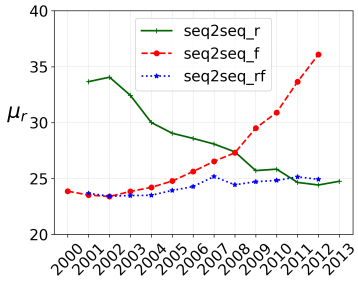


Figure 5: μ_r of our models for varying value of i (Eq. 1–3).[5] For the complete results, refer to Appendix B.

employ PROCR$_*$, ranking words based on the average cosine distance within $[0, i]$; this is similar to the "Mean Distances" model used in Rodina et al. (2019), with the difference that the distances at time point i are calculated by measuring the cosine distance resulting from the alignment against the initial time point 0 and not against $i-1$.[6]

We report the performance of the models (a) when they operate on the full interval [2000-13] and (b) averaged across all intervals [2000-01, ..., 2000-13]. In (b), our models use additional (future) information compared to our baselines; when seq2seq$_f$ is fed with the word sequences of [2000, 2001], it makes a prediction for the years [2002, ..., 2013] – such information cannot be leveraged by the baselines. Thus, for (b), we only perform intra-model (and intra-baseline) comparisons.

5.2 Results

Our models *vs* baselines Results are shown in Table 1. The three proposed models consistently achieve the lowest μ_r and highest Rec@k when working on the whole time sequence ('00-'13 columns). The comparison between {seq2seq$_r$, LSTM$_r$} and {seq2seq$_f$, LSTM$_f$} in the years 2000-13 showcases the benefit of modelling the full sequence of the word representations across time, compared to using the first and last representations only. Our models provide a relative boost of 4.6% in μ_r and [35.7%, 42.8%, 5.8%] in Rec@k (for k=[5, 10, 50]) compared to the best performing baseline. seq2seq$_f$ and seq2seq$_{rf}$ models outperform the autoencoder (seq2seq$_r$) in most metrics, while seq2seq$_{rf}$ yields the most stable results across all runs. We explore these differences in detail in the last paragraph of this section.

Intra-baseline comparison Models operating only on the first and last word representations fail to confidently outperform the Procrustes-based baselines, demonstrating again the weakness of operating in a non-sequential manner. The LSTM models achieve low μ_r on the 2000-13 experiments; however, the difference with the rest of the baselines in μ_r across all years is negligible. The intra-Procrustes model comparison shows that the benefit of selecting a few anchor words to learn a better alignment (PROCR$_k$, PROCR$_{kt}$) shown in Tsakalidis et al. (2019) in examining semantic change over two consecutive years might not apply when examining a longer time period. Finally, contrary to Shoemark et al. (2019), we find that time sensitive models operating on the word distances across time (GT$_c$, GT$_\beta$) perform worse than the baselines that leverage only the first and last word representations. This difference is attributed to the low number of time steps in our dataset that does not allow the GT models to exploit long-term correlations (i.e., considering the average distance across time (PROCR$_*$) performs better), but also highlights the importance of leveraging the full word sequence across time.

Effect of input/output lengths Figure 5 shows the μ_r of our three variants when we alter the length of the input and output (see section 3.4). The performance of seq2seq$_r$ increases with the input size since by definition the decoder is able to detect words whose semantics have changed over a longer period of time (i.e., within $[2000, i]$, with i increasing), while also modelling a longer se-

[5] Example (2005 in x-axis): The sequence of the word representations until 2005 is the input to all of our models. Then, seq2seq$_r$ reconstructs the word representations up to 2005, seq2seq$_f$ predicts the future representations (2006, ..., 2013) and seq2seq$_{rf}$ performs both tasks jointly.

[6] We refrain from evaluating the GT models when $i \leq 2$, due to the very short time interval that does not allow for correlations to appear in the data, leading to very poor performance.

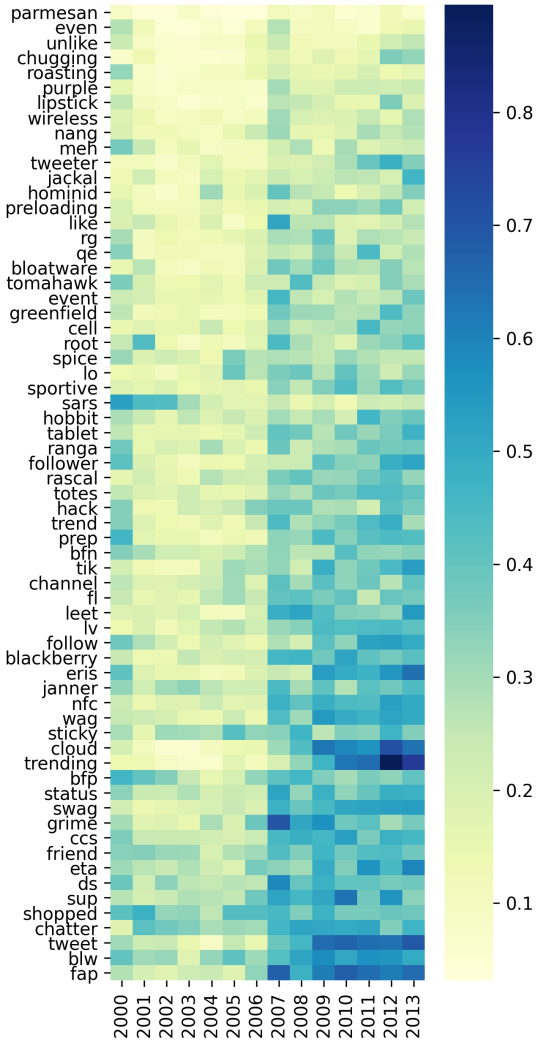


Figure 6: Cosine distances (actual *vs* predicted vectors) of each semantically shifted word (as indicated by the Oxford English Dictionary), per year. Lighter colours indicate better model performance – thus, lower level of semantic change predicted by our joint model.

quence of a word's representation through time. On the contrary, the performance of $\texttt{seq2seq}_f$ increases alongside the decrease of the number of input time steps. This is expected since, as i decreases, $\texttt{seq2seq}_f$ encodes a shorter input sequence and the decoding (and hence the semantic change detection) is applied on the remaining (and increased number of) time steps within $[i{+}1, 2013]$. These findings provide empirical evidence that both models can achieve better performance if trained over longer sequences of time steps. Finally, the stability of $\texttt{seq2seq}_{rf}$ showcases its input length-invariant nature, which is also clearly evident in all of the averaged results (standard deviation in avg$\pm$std columns) in Table 1: in its worst performing setting, $\texttt{seq2seq}_{rf}$ still manages to achieve results that are close to the best performing model (μ_r=25.17, Rec@k=[21.54, 36.92, 83.08] for the three thresholds) and always better (or equal to) the best performing baseline shown in Table 1 in Rec@k. This is a very attractive aspect of the model as it removes the need to manually define the number of time steps to be fed to the encoder.

Words with shifted meaning Figure 6 shows the cosine distances between the actual and predicted vectors of the 65 words that acquired a new meaning between 2001-2013. The distances are calculated by applying the $\texttt{seq2seq}_{rf}$ model (trained as in section 4.3) on the test set. The words are ranked based on their average cosine distance throughout the years such that the words in the first rows form more challenging examples for detecting their semantic shift. Despite that some of these words have acquired an additional meaning in the context of social networks (e.g., "like", "unlike"), this is not effectively captured by their vectors. Utilising contextual representations (Giulianelli et al., 2020) in our models can be more effective for capturing such cases in future work.

6 Conclusion and Future Work

We introduce three sequential models for semantic change detection that effectively exploit the full sequence of a word's representations through time to determine its level of semantic change. Through extensive experiments on synthetic and real data we showcase the effectiveness of the proposed models under various settings and in comparison to state-of-the-art on the UK Web Archive dataset. Importantly, we show that their performance increases alongside the duration of the time period under study, confidently outperforming competitive models and common practices on semantic change.

Future work can use anomaly detection approaches operating on our model's predicted word vectors to detect anomalies in a word's representation across time. We also plan to investigate different architectures, such as Variational Autoencoders (Kingma and Welling, 2014), and incorporate contextual representations (Devlin et al., 2019; Hu et al., 2019) to detect new senses of words. A limitation of our work is that it has been tested on a single dataset, where 65 words have undergone semantic change; testing our models in datasets of different duration and in different languages will provide clearer evidence of their effectiveness.

Acknowledgements

This work was supported by The Alan Turing Institute (grant EP/N510129/1) and by a Turing AI Fellowship to Maria Liakata, funded by the Department of Business, Energy & Industrial Strategy. The authors would like to thank Stephen Clark, Mihai Cucuringu, Elena Kochkina, Barbara McGillivray, Federico Nanni, Nicole Peinelt and the anonymous reviewers for their valuable feedback.

References

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

Pierpaolo Basile, Annalina Caputo, Tommaso Caselli, Pierluigi Cassotti, and Rossella Varvara. 2020. Overview of the EVALITA 2020 Diachronic Lexical Semantics (DIACR-Ita) Task. In *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*, Online. CEUR.org.

Pierpaolo Basile and Barbara McGillivray. 2018. Exploiting the Web for Semantic Change Detection. In *International Conference on Discovery Science*, pages 194–208. Springer.

James Bergstra, Daniel Yamins, and David Daniel Cox. 2013. Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. pages 115–123.

Leo Breiman. 2001. Random Forests. *Machine Learning*, 45(1):5–32.

Paul Cook and Suzanne Stevenson. 2010. Automatically Identifying Changes in the Semantic Orientation of Words. In *Proceedings of the Seventh conference on International Language Resources and Evaluation*.

Cristian Danescu-Niculescu-Mizil, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. No Country for Old Members: User Lifecycle and Linguistic Change in Online Communities. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 307–318. Association for Computing Machinery.

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)*, pages 2069–2075.

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)*, pages 4171–4186.

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*, pages 457–470.

Lea Frermann and Mirella Lapata. 2016. A Bayesian Model of Diachronic Meaning Change. *Transactions of the Association for Computational Linguistics*, 4:31–45.

William A Gale, Kenneth W Church, and David Yarowsky. 1992. Work on statistical methods for word sense disambiguation. In *Working Notes of the AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, volume 54, page 60.

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*, Online. Association for Computational Linguistics.

Kristina Gulordava and Marco Baroni. 2011. A distributional similarity approach to the detection of semantic change in the Google Books Ngram corpus. In *Proceedings of the GEMS 2011 Workshop on Geometrical Models of Natural Language Semantics*, pages 67–71.

William L Hamilton, Jure Leskovec, and Dan Jurafsky. 2016. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1489–1501.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

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*, pages 3899–3908.

Yoon Kim, Yi-I Chiu, Kentaro Hanaki, Darshan Hegde, and Slav Petrov. 2014. Temporal Analysis of Language through Neural Language Models. In *Proceedings of the ACL 2014 Workshop on Language Technologies and Computational Social Science*, pages 61–65.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings*.

Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Conference Track Proceedings*.

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

Andrey Kutuzov, Lilja Øvrelid, Terrence Szymanski, and Erik Velldal. 2018. Diachronic Word Embeddings and Semantic Shifts: A Survey. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1384–1397.

Julian John McAuley and Jure Leskovec. 2013. From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise through Online Reviews. In *Proceedings of the 22nd International Conference on World Wide Web*, pages 897–908. Association for Computing Machinery.

Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K Gray, Joseph P Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, et al. 2011. Quantitative Analysis of Culture Using Millions of Digitized Books. *Science*, 331(6014):176–182.

Rada Mihalcea and Vivi Nastase. 2012. Word Epoch Disambiguation: Finding how Words Change over Time. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 259–263.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Sunny Mitra, Ritwik Mitra, Martin Riedl, Chris Biemann, Animesh Mukherjee, and Pawan Goyal. 2014. That's sick dude!: Automatic identification of word sense change across different timescales. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1020–1029.

Valerio Perrone, Marco Palma, Simon Hengchen, Alessandro Vatri, Jim Q Smith, and Barbara McGillivray. 2019. GASC: Genre-Aware Semantic Change for Ancient Greek. In *Proceedings of the 1st International Workshop on Computational Approaches to Historical Language Change*, pages 56–66.

Julia Rodina, Daria Bakshandaeva, Vadim Fomin, Andrey Kutuzov, Samia Touileb, and Erik Velldal. 2019. Measuring Diachronic Evolution of Evaluative Adjectives with Word Embeddings: the Case for English, Norwegian, and Russian. In *Proceedings of the 1st International Workshop on Computational Approaches to Historical Language Change*, pages 202–209.

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)*, pages 474–484.

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

Eyal Sagi, Stefan Kaufmann, and Brady Clark. 2009. Semantic Density Analysis: Comparing Word Meaning across Time and Phonetic Space. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 104–111. Association for Computational Linguistics.

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. *arXiv preprint arXiv:1906.02979*.

Dominik Schlechtweg, Barbara McGillivray, Simon Hengchen, Haim Dubossarsky, and Nina Tahmasebi. 2020. SemEval-2020 Task 1: Unsupervised Lexical Semantic Change Detection. In *To appear in Proceedings of the 14th International Workshop on Semantic Evaluation*, Barcelona, Spain. Association for Computational Linguistics.

Dominik Schlechtweg, Sabine Schulte im Walde, and Stefanie Eckmann. 2018. Diachronic Usage Relatedness (DURel): A Framework for the Annotation of Lexical Semantic Change. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 169–174.

Peter H Schönemann. 1966. A Generalized Solution of the Orthogonal Procrustes Problem. *Psychometrika*, 31(1):1–10.

Hinrich Schütze. 1998. Automatic word sense discrimination. *Computational linguistics*, 24(1):97–123.

Philippa Shoemark, Farhana Ferdousi Liza, Dong Nguyen, Scott A Hale, and Barbara McGillivray. 2019. Room to Glo: A Systematic Comparison of Semantic Change Detection Approaches with Word Embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language*

Processing and the 9th International Joint Conference on Natural Language Processing, pages 66–76.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

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

Xuri Tang. 2018. A State-of-the-Art of Semantic Change Computation. *Natural Language Engineering*, 24(5):649–676.

Adam Tsakalidis, Marya Bazzi, Mihai Cucuringu, Pierpaolo Basile, and Barbara McGillivray. 2019. Mining the UK Web Archive for Semantic Change Detection. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1212–1221.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic Word Embeddings for Evolving Semantic Discovery. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 673–681. ACM.

A List of Hyperparameters

Our models We test the following hyper-parameters for our $\texttt{seq2seq}_{r/f/rf}$ models:

- encoder_LSTM$_0$, number of units: [128, 256, 512] ([32,64,128,256,512] for seq2seq$_f$).
- encoder_LSTM$_1$, number of units: [32, 64]
- decoder_LSTM$_0$, number of units: [32, 64] (x2, for the case of $\texttt{seq2seq}_{rf}$ – for (a) the autoencoding and (b) future prediction component)
- decoder_LSTM$_1$, number of units: [128, 256, 512] (x2, for the case of $\texttt{seq2seq}_{rf}$; [32,64,128,256,512] for seq2seq$_f$).
- dropout rate in dropout layers: [.1, .25, .5]
- batch size: [32, 64, 128, 256, 512, 1024]
- number of epochs: [10, 20, 30, 40, 50]

We optimise our parameters using the Adam optimiser in `keras`, using the default learning rate (.001).

Baselines We experiment with the following hyper-parameters per model:

- $\texttt{LSTM}_{r/f}$: we follow the exact same settings as in our seq2seq$_r$ and seq2seq$_f$ models, respectively.
- `RF`: we experiment with the number of trees ([50, 100, 150, 200]) and select the best model based on the maximum average cosine similarity across all predictions, as in our models.
- $\texttt{PROCR}_{k/kt}$: we experiment with different rate [.001, .01, .05, .1, .2, ...9] of anchor (or diachronic anchor) words on the basis of the size of the test set. We select to display in our results the best model based on the average performance in the test set (k=.9 for $\texttt{PROCR}_k$, k=.5 for $\texttt{PROCR}_{kt}$).
- $\texttt{GT}_c$: we explore different correlation metrics (Spearman Rank, Pearson Correlation, Kendall Tau) and select to display the best one (Pearson Correlation) on the basis of its average performance on the test set across all experiments. Due to the very poor performance of all metrics when operating on a small number of time-steps (≤ 2), we only provide the results in Table 1 (avg$\pm$std columns) when these models operate on longer sequences.
- `PROCR`, $\texttt{PROCR}_*$, $\texttt{GT}_\beta$, `RAND`: there are no hyper-parameter to tune in these models. In terms of preprocessing, for all `PROCR`-based baselines, we first subtract the mean and then we divide each matrix by its Frobenius norm, so that the resulting (transformed) matrices are in the same space.

B Complete Results on Real Data

The complete list of results (μ_r) of all models in all of the experiments with real data (section 5, Table 1 and Figure 5) are provided in Table 2. The interpretation of the "year" for each model is provided in Table 3.

year	PROCR	PROCR$_k$	PROCR$_{kt}$	RF	LSTM$_r$	LSTM$_f$	GT$_\beta$	GT$_c$	PROCR$_*$	seq2seq$_r$	seq2seq$_f$	seq2seq$_{rf}$
2000	-	-	-	-	-	-	-	-	-	-	23.86	-
2001	34.26	34.54	34.43	37.35	33.67	36.43	-	-	34.26	33.66	23.52	23.67
2002	32.70	32.64	32.41	34.94	31.20	32.98	-	-	32.98	34.06	23.39	23.42
2003	29.24	29.56	29.41	36.94	30.32	32.57	37.59	43.34	31.02	32.44	23.84	23.47
2004	25.46	25.35	25.03	27.25	24.66	26.08	35.43	42.98	28.68	30.01	24.21	23.50
2005	29.04	29.40	28.65	31.43	28.98	29.17	38.47	44.47	28.23	29.05	24.77	23.93
2006	27.73	27.89	27.38	28.86	26.61	26.55	38.74	44.45	27.71	28.58	25.62	24.28
2007	26.70	26.75	26.64	30.16	25.45	26.39	34.16	41.93	26.98	28.09	26.53	25.17
2008	28.30	28.34	27.87	32.77	26.25	27.86	35.02	42.86	26.72	27.38	27.30	24.44
2009	26.10	26.04	25.81	23.27	24.97	23.73	34.23	43.24	26.15	25.71	29.50	24.72
2010	27.95	27.96	27.38	28.25	28.18	28.19	36.04	44.77	25.81	25.84	30.91	24.83
2011	25.71	25.85	25.74	28.15	26.07	26.24	34.78	43.99	25.31	24.65	33.65	25.14
2012	26.77	27.34	27.44	26.51	27.52	27.12	35.18	44.53	24.94	24.42	36.09	24.93
2013	30.63	31.01	31.91	30.01	27.87	28.62	38.09	47.87	25.01	24.75	-	-
AVERAGE	28.51	28.67	28.47	30.45	27.83	28.61	36.16	44.04	27.99	28.36	27.17	24.29

Table 2: Complete μ_r scores across all runs.

Model	**Explanation**	**Example (year=2006)**
PROCR PROCR$_k$ PROCR$_{kt}$	Date to use for aligning the word vectors with their corresponding ones in the year 2000.	The model aligns the word vectors in the year 2006 with the word vectors in the year 2000.
LSTM$_r$	The date indicating the word vectors to reconstruct, along with those in the first time-step.	`LSTM`$_r$ receives as input the word vectors in the years 2000 and 2006 and reconstructs them.
LSTM$_f$, RF	The date indicating the word vectors to predict.	`LSTM`$_f$/`RF` receives the word vectors in the year 2000 & predicts the word vectors in the year 2006.
PROCR$_*$, GT$_c$, GT$_\beta$	Cut-off date to use for constructing the time series of the cosine distances.	The time series of cosine distances of every word are constructed based on the years [2000-2006].
seq2seq$_r$	Cut-off date in the input, indicating the range of years to reconstruct.	`seq2seq`$_r$ is fed with the word representations in the years [2000-2006] and reconstructs them.
seq2seq$_f$	Cut-off date in the input, affecting the range of years to predict.	`seq2seq`$_f$ predicts the word vectors during the years [2007-2013], given the vectors during the years [2000-2006] as input.
seq2seq$_{rf}$	Cut-off date in the input, indicating the range of years to reconstruct & affecting the range of dates to predict.	`seq2seq`$_{rf}$ receives the word vectors during the years [2000-2006] and (a) reconstructs them & (b) predicts their representations in [2007-2013].

Table 3: Explanation of the variable *"year"* in Table 2.