Dynamic Word Embeddings via Skip-Gram Filtering

Robert Bamler ROBERT.BAMLER@DISNEYRESEARCH.COM
Disney Research, 4720 Forbes Avenue, Pittsburgh, PA 15213 USA

Stephan Mandt STEPHAN.MANDT@DISNEYRESEARCH.COM
Disney Research, 4720 Forbes Avenue, Pittsburgh, PA 15213 USA

Abstract

We present a probabilistic language model for time-stamped text data which tracks the semantic evolution of individual words over time. The model represents words and contexts by latent trajectories in an embedding space. At each moment in time, the embedding vectors are inferred from a probabilistic version of word2vec (Mikolov et al., 2013b). These embedding vectors are connected in time through a latent diffusion process. We describe two scalable variational inference algorithms—skip-gram smoothing and skip-gram filtering—that allow us to train the model jointly over all times; thus learning on all data while simultaneously allowing word and context vectors to drift. Experimental results on three different corpora demonstrate that our dynamic model infers word embedding trajectories that are more interpretable and lead to higher predictive likelihoods than competing methods that are based on static models trained separately on time slices.

1. Introduction

Language evolves over time and words change their meaning due to cultural shifts, technological inventions, or political events. We consider the problem of detecting shifts in the meaning and usage of words over a given time span based on text data. Capturing these semantic shifts requires a dynamic language model.

Word embeddings are a powerful tool for modeling semantic relations between individual words (Bengio et al., 2003; Mikolov et al., 2013a; Pennington et al., 2014; Mnih & Kavukcuoglu, 2013; Levy & Goldberg, 2014; Vilnis & McCallum, 2014; Rudolph et al., 2016). Word embeddings model the distribution of words based on their surrounding words, and summarize these statistics in terms of low-dimensional vector representations. Geometric distances between word vectors reflect semantic similarity (Mikolov et al., 2013a) and difference vectors encode semantic and syntactic relations (Mikolov et al., 2013c), which shows that they are sensible representations of language. Pretrained word embeddings are useful for various supervised tasks, including sentiment analysis (Socher et al., 2013b), semantic parsing (Socher et al., 2013a), and computer vision (Fu & Sigal, 2016). As unsupervised models, they have also been used for the exploration of word analogies and linguistics (Mikolov et al., 2013c).

Word embeddings are currently formulated as static models, which assumes that the meaning of any given word is the same across the entire text corpus. In this paper, we propose a generalization of word embeddings to sequential data, such as corpora of historic texts or streams of text in social media.

Current approaches to learning word embeddings in a dynamic context rely on grouping the data into time bins and training the embeddings separately on these bins (Kim et al., 2014; Kulkarni et al., 2015; Hamilton et al., 2016). This approach, however, raises three fundamental problems. First, since word embedding models are non-convex, training them twice on the same data will lead to different results. Thus, embedding vectors at successive times can only be approximately related to each other, and only if the embedding dimension is large (Hamilton et al., 2016). Second, dividing a corpus into separate time bins may lead to training sets that are too small to train a word embedding model. Hence, one runs the risk of overfitting to few data whenever the required temporal resolution is fine-grained, as we show in the experimental section. Third, due to the finite corpus size the learned word embedding vectors are subject to random noise. It is difficult to disambiguate this noise from systematic semantic drifts between subsequent times, in particular over short time spans, where we expect only minor semantic drift.

In this paper, we circumvent these problems by introducing a dynamic word embedding model. Our contributions are as follows:

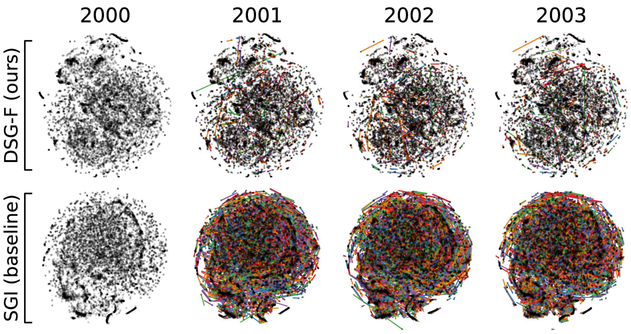

Figure 1. Word embeddings over a sequence of years trained on Google books, using dynamic skip-gram filtering (proposed, top row) and compared to the static method by Hamilton et al. (2016) (bottom). We used dynamic t-SNE (Rauber et al., 2016) for dimensionality reduction. Colored lines in the second to fourth column indicate the trajectories from the previous year. Our method infers smoother trajectories with only few words that move quickly. Figure 3 shows that these effects persist in the original embedding space.

- We derive a probabilistic state space model where word and context embeddings evolve in time according to a diffusion process. It generalizes the skip-gram model (Mikolov et al., 2013b; Barkan, 2016) to a dynamic setup, which allows end-to-end training. This leads to continuous embedding trajectories, smoothes out noise in the word-context statistics, and allows us to share information across all times.

- We propose two scalable black-box variational inference algorithms (Ranganath et al., 2014; Rezende et al., 2014) for filtering and smoothing. These algorithms find word embeddings that generalize better to held-out data. Our smoothing algorithm carries out efficient black-box variational inference for Gaussian structured variational distributions with tridiagonal precision matrices, and applies more broadly.

- We analyze three massive text corpora that span over long periods of time. Our approach allows us to automatically find the words that change their meaning the most rapidly. It results in smooth word embedding trajectories and therefore allows us to measure and visualize the continuous dynamics of the entire embedding cloud as it deforms over time.

The benefits of our approach are best visualized in Figure 1. Here, we show the word embeddings for four consecutive years according to our proposed dynamical model (top) and a static model (bottom). Colored lines indicate the trajectories from the previous year. Both models were trained on the large Google books corpus of historical text (Michel et al., 2011). Our approach (top row) leads to more interpretable trajectories, inferring semantic change over a single year for only few words in the vocabulary. This is not an artifact of the dimensionality reduction, as we show in Figure 3 in the experimental section.

Our paper is structured as follows. In section 2 we discuss related work, and we introduce our model in section 3. In section 4 we present two efficient variational inference algorithms for our dynamic model. We show experimental results in section 5. Section 6 summarizes our findings.

2. Related Work

Probabilistic models that have been extended to latent time series models are ubiquitous (Blei & Lafferty, 2006; Wang et al., 2008; Sahoo et al., 2012; Gultekin & Paisley, 2014; Charlin et al., 2015; Ranganath et al., 2015; Jerfel et al., 2017), but none of them relate to word embeddings. The closest of these models is the dynamic topic model (Blei & Lafferty, 2006; Wang et al., 2008), which learns the evolution of latent topics over time. Topic models are based on bag-of-word representations and thus treat words as symbols without modelling their semantic relations. They therefore serve a different purpose.

Mikolov et al. (2013a;b) proposed the skip-gram model with negative sampling (word2vec) as a scalable word embedding approach that relies on stochastic gradient descent. This approach has been formulated in a Bayesian setup (Barkan, 2016), which we discuss separately in section 3.1. These models, however, do not allow the word embedding vectors to change over time.

Several authors have analyzed different statistics of text data to analyze semantic changes of words over time (Mihalcea & Nastase, 2012; Sagi et al., 2011; Kim et al., 2014; Kulkarni et al., 2015; Hamilton et al., 2016). None of them explicitly model a dynamical process; instead, they slice the data into different time bins, fit the model separately on each bin, and further analyze the embedding vectors in post-processing. By construction, these static models can therefore not share statistical strength across time. This limits the applicability of static models to very large corpora.

Most related to our approach are methods based on word embeddings. Kim et al. (2014) fit word2vec separately on different time bins, where the word vectors obtained for the previous bin are used to initialize the algorithm for the next time bin. The bins have to be sufficiently large and the found trajectories are not as smooth as ours, as we demonstrate in this paper. Hamilton et al. (2016) also trained word2vec separately on several large corpora from different decades. If the embedding dimension is large enough (and hence the optimization problem less non-convex), the authors argue that word embeddings at nearby times ap-

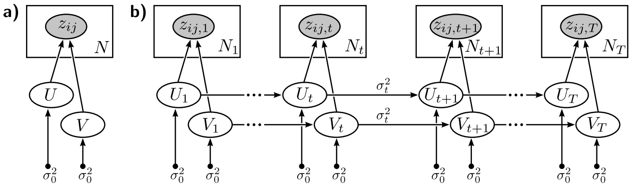

Figure 2. a) The Bayesian skip-gram model (Barkan, 2016); N denotes the total number of (positive and negative) examples and σ_0^2 is the variance of the Gaussian prior. b) our proposed dynamic skip-gram model; T copies of the Bayesian skip-gram model are connected by a Kalman filter (Kalman et al., 1960), constraining the drift of embedding vectors between nearby times.

proximately differ by a global rotation in addition to a small semantic drift, and they approximately compute this rotation. As the latter does not exist in a strict sense, it is difficult to distinguish artifacts of the approximate rotation from a true semantic drift. As discussed in this paper, both variants result in trajectories which are noisier.

3. Model

We propose the dynamic skip-gram model, a generalization of the skip-gram model (word2vec) (Mikolov et al., 2013b) to sequential text data. The model finds word embedding vectors that continuously drift over time, allowing to track changes in language and word usage over short and long periods of time. Dynamic skip-gram is a probabilistic model which combines a Bayesian version of the skip-gram model (Barkan, 2016) with a latent time series. It is jointly trained end-to-end and scales to massive data by means of approximate Bayesian inference.

The observed data consist of sequences of words from a finite vocabulary of size L. In section 3.1, all sequences (sentences from books, articles, or tweets) are considered time-independent; in section 3.2 they will be associated with different time stamps. The goal is to maximize the probability of every word that occurs in the data given its surrounding words within a so-called context window. As detailed below, the model learns two vectors $u_i, v_i \in \mathbb{R}^d$ for each word i in the vocabulary, where d is the embedding dimension. We refer to u_i as the word embedding vector and to v_i as the context embedding vector.

3.1. Bayesian Skip-Gram Model

The Bayesian skip-gram model (Barkan, 2016) is a probabilistic version of word2vec (Mikolov et al., 2013b) and forms the basis of our approach. The graphical model is shown in Figure 2a). For each pair of words i, j in the vocabulary, the model assigns probabilities that word i appears in the context of word j. This probability is $\sigma(u_i^\top v_j)$ with the sigmoid function $\sigma(x) = 1/(1+e^{-x})$. Let $z_{ij} \in \{0,1\}$ be an indicator variable that denotes a draw from that probability distribution, hence $p(z_{ij}=1) = \sigma(u_i^\top v_j)$. The generative model assumes that many word-word pairs (i,j) are uniformly drawn from the vocabulary and tested for being a word-context pair; hence a separate random indicator z_{ij} is associated with each drawn pair.

Focusing on words and their neighbors in a context window, we collect evidence of word-word pairs for which $z_{ij} = 1$. These are called the positive examples. Denote n_{ij}^+ the number of times that a word-context pair (i,j) is observed in the corpus. This is a sufficient statistic of the model, and its contribution to the likelihood is $p(n_{ij}^+|u_i, v_j) = \sigma(u_i^\top v_j)^{n_{ij}^+}$. However, the generative process also assumes the possibility to reject word-word pairs if $z_{ij} = 0$. Thus, one needs to construct a fictitious second training set of rejected word-word pairs, called negative examples. Let their counts be n_{ij}^-. The total likelihood of both positive and negative examples is then

$$p(n^+, n^-|U,V) = \prod_{i,j=1}^{L} \sigma(u_i^\top v_j)^{n_{ij}^+} \sigma(-u_i^\top v_j)^{n_{ij}^-}. \quad (1)$$

Above we used the antisymmetry $\sigma(-x) = 1 - \sigma(x)$. In our notation, dropping the subscript indices for n^+ and n^- denotes the entire $L \times L$ matrices, $U = (u_1, \cdots, u_L) \in \mathbb{R}^{d\times L}$ is the matrix of all word embedding vectors, and V is defined analogously for the context vectors. To construct negative examples, one typically chooses $n_{ij}^- \propto P(i)P(j)^{3/4}$ (Mikolov et al., 2013b), where $P(i)$ is the frequency of word i in the training corpus. Thus, n^- is well-defined up to a constant factor which has to be tuned.

Defining $n^\pm = (n^+, n^-)$ the combination of both positive and negative examples, the resulting log likelihood is

$$\log p(n^\pm|U,V) = \sum_{i,j=1}^{L} \left(n_{ij}^+ \log\sigma(u_i^\top v_j) + n_{ij}^- \log\sigma(-u_i^\top v_j)\right). \quad (2)$$

This is exactly the objective of the (non-Bayesian) skip-gram model, see (Mikolov et al., 2013b). The count matrices n^+ and n^- are either pre-computed for the entire corpus, or estimated based on stochastic subsamples from the data in a sequential way, as done by word2vec. Barkan (2016) give an approximate Bayesian treatment of the model with Gaussian priors on the embeddings.

3.2. Dynamic Skip-Gram Model

The key extension of our approach is to use a Kalman filter as a prior for the time-evolution of the latent embeddings (Welch & Bishop, 1995). This allows us to share information across all times while still allowing the embeddings to drift.

Notation. We consider a corpus of T documents which were written at time stamps $\tau_1 < \ldots < \tau_T$. For each time step $t \in \{1, \ldots, T\}$ the sufficient statistics of word-context pairs are encoded in the $L \times L$ matrices n_t^+, n_t^- of positive and negative counts with matrix elements $n_{ij,t}^+$ and $n_{ij,t}^-$, respectively. Denote $U_t = (u_{1,t}, \cdots, u_{L,t}) \in \mathbb{R}^{d \times L}$ the matrix of word embeddings at time t, and define V_t correspondingly for the context vectors. Let $U, V \in \mathbb{R}^{T \times d \times L}$ denote the tensors of word and context embeddings across all times, respectively.

Model. The graphical model is shown in Figure 2b). We consider a diffusion process of the embedding vectors over time. The variance σ_t^2 of the transition kernel is

$$\sigma_t^2 = D(\tau_{t+1} - \tau_t), \tag{3}$$

where D is a global diffusion constant and $(\tau_{t+1} - \tau_t)$ is the time between subsequent observations (Welch & Bishop, 1995). At every time step t, we add an additional Gaussian prior with zero mean and variance σ_0^2 which prevents the embedding vectors from growing very large, thus

$$p(U_{t+1}|U_t) \propto \mathcal{N}(U_t, \sigma_t^2)\, \mathcal{N}(0, \sigma_0^2). \tag{4}$$

Computing the normalization, this results in

$$U_{t+1}|U_t \sim \mathcal{N}\left(\frac{U_t}{1 + \sigma_t^2/\sigma_0^2}, \frac{1}{\sigma_t^{-2} + \sigma_0^{-2}} I\right), \tag{5}$$

$$V_{t+1}|V_t \sim \mathcal{N}\left(\frac{V_t}{1 + \sigma_t^2/\sigma_0^2}, \frac{1}{\sigma_t^{-2} + \sigma_0^{-2}} I\right). \tag{6}$$

In practice, $\sigma_0 \gg \sigma_t$, so the damping to the origin is very weak. This is also called Ornstein-Uhlenbeck process (Uhlenbeck & Ornstein, 1930). At time index $t = 1$, we define $p(U_1|U_0) \equiv \mathcal{N}(0, \sigma_0^2 I)$ and do the same for V_1. Our joint distribution factorizes as follows:

$$p(n^\pm, U, V) = \prod_{t=0}^{T-1} p(U_{t+1}|U_t)\, p(V_{t+1}|V_t) \times \prod_{t=1}^{T} \prod_{i,j=1}^{L} p(n_{ij,t}^\pm | u_{it}, v_{jt}) \tag{7}$$

The prior model enforces that the model learns embedding vectors which vary smoothly across time. This allows to associate words unambiguously with each other and to detect semantic changes. The model efficiently shares information across the time domain, which allows to fit the model in setups where the data at every given point in time are small, but the data in total are large.

4. Inference

We discuss two scalable approximate inference algorithms for our model, taking into account different versions of inference: filtering, which only uses information from the past as required in streaming applications, and smoothing, which learns better embeddings but requires full knowledge of the sequence of documents. In Bayesian inference, we start by formulating a joint distribution (Eq. 7) over observations $n^\pm$ and parameters U and V, and we are interested in the posterior distribution over parameters conditioned on observations,

$$p(U, V|n^\pm) = \frac{p(n^\pm, U, V)}{\int p(n^\pm, U, V)\, dU dV} \tag{8}$$

The problem is that the normalization is intractable. In variational inference (VI) (Jordan et al., 1999; Blei et al., 2016) one sidesteps this problem and approximates the posterior with a simpler variational distribution $q_\lambda(U, V)$ by minimizing the Kullback-Leibler divergence to the posterior. This is equivalent to optimizing the evidence lower bound (ELBO) (Blei et al., 2016),

$$\mathcal{L}(\lambda) = \mathbb{E}_{q_\lambda}[\log p(n^\pm, U, V)] - \mathbb{E}_{q_\lambda}[\log q_\lambda(U, V)]. \tag{9}$$

For a restricted class of models, the ELBO can be computed in closed-form (Hoffman et al., 2013). Our model is non-conjugate and requires instead black-box VI using the reparameterization trick, where one maximizes $\mathcal{L}(\lambda)$ with stochastic gradient ascent and estimates the gradient $\nabla_\lambda \mathcal{L}$ by sampling from the variational distribution (Rezende et al., 2014; Kingma & Welling, 2014).

4.1. Skip-Gram Filtering

In many applications such as streaming, the data arrive sequentially. Thus, we can only condition our model on past and not on future observations. We will first describe inference in such a (Kalman) filtering setup (Kalman et al., 1960; Welch & Bishop, 1995).

In the filtering scenario, the inference algorithm iteratively updates the variational distribution q as evidence from each time step t becomes available. We thereby use a variational distribution that factorizes across all times, $q(U, V) = \prod_{t=1}^T q(U_t, V_t)$ and we update the variational factor at a given time t based on the evidence at time t and the approximate posterior of the previous time step. Furthermore, at every time t we use a fully-factorized distribution:

$$q(U_t, V_t) = \prod_{i=1}^{L} \mathcal{N}(u_{i,t}; \mu_{ui,t}, \Sigma_{ui,t})\, \mathcal{N}(v_{i,t}; \mu_{vi,t}.\Sigma_{vi,t}),$$

The variational parameters are the means $\mu_{ui,t}, \mu_{vi,t} \in \mathbb{R}^d$ and the covariance matrices $\Sigma_{ui,t}$ and $\Sigma_{vi,t}$, which we restrict to be diagonal (mean-field approximation).

We now describe how we sequentially compute $q(U_t, V_t)$ and use the result to proceed to the next time step. As other

Markovian dynamical systems, our model assumes the following recursion,

$$p(U_t, V_t|n^{\pm}_{1:t}) \propto p(n^{\pm}_t|U_t, V_t)\, p(U_t, V_t|n^{\pm}_{1:t-1}). \qquad (10)$$

Within our variational approximation, the ELBO (Eq. 9) therefore separates into a sum of T terms, $\mathcal{L} = \sum_t \mathcal{L}_t$ with

$$\mathcal{L}_t = \mathbb{E}[\log p(n^{\pm}_t|U_t, V_t)] + \mathbb{E}[\log p(U_t, V_t|n^{\pm}_{1:t-1})] \\ - \mathbb{E}[\log q(U_t, V_t)], \qquad (11)$$

where all expectations are taken under $q(U_t, V_t)$. We compute the entropy term in Eq. 11 analytically and estimate the gradient of the log likelihood by sampling from the variational distribution and using the reparameterization trick (Kingma & Welling, 2014; Salimans & Kingma, 2016). However, the second term of Eq. 11, containing the prior at time t, is still intractable. We approximate the prior as

$$\begin{aligned} p(U_t, V_t|n^{\pm}_{1:t-1}) &\equiv \\ &\mathbb{E}_{p(U_{t-1},V_{t-1}|n^{\pm}_{1:t-1})}\big[p(U_t, V_t|U_{t-1}, V_{t-1})\big] \\ &\approx \mathbb{E}_{q(U_{t-1},V_{t-1})}\big[p(U_t, V_t|U_{t-1}, V_{t-1})\big]. \qquad (12) \end{aligned}$$

The remaining expectation involves only Gaussians and can be carried-out analytically. The resulting approximate prior is a fully factorized distribution $p(U_t, V_t|n^{\pm}_{1:t-1}) \approx \prod_{i=1}^{L} \mathcal{N}(u_{i,t}; \tilde{\mu}_{ui,t}, \tilde{\Sigma}_{ui,t})\, \mathcal{N}(v_{i,t}; \tilde{\mu}_{vi,t}, \tilde{\Sigma}_{vit})$ with

$$\begin{aligned} \tilde{\mu}_{ui,t} &= \tilde{\Sigma}_{ui,t} \left(\Sigma_{ui,t-1} + \sigma_t^2 I\right)^{-1} \mu_{ui,t-1}; \\ \tilde{\Sigma}_{ui,t} &= \left[\left(\Sigma_{ui,t-1} + \sigma_t^2 I\right)^{-1} + (1/\sigma_0^2) I\right]^{-1} \end{aligned} \qquad (13)$$

Analogous update equations hold for $\tilde{\mu}_{vi,t}$ and $\tilde{\Sigma}_{vi,t}$. The expected log prior therefore also yields an analytic contribution to $\mathcal{L}_t$.

4.2. Skip-Gram Smoothing

In contrast to filtering, where inference is conditioned on past observations until a given time t, (Kalman) smoothing performs inference based on the entire sequence of observations $n^{\pm}_{1:T}$. This approach results in smoother trajectories and typically higher likelihoods than with filtering, because evidence is used from both future and past observations.

Besides the new inference scheme, we also use a different variational distribution. As the model is fitted jointly to all time steps, we are no longer restricted to a variational distribution that factorizes in time. For simplicity we focus here on the variational distribution for the word embeddings U; the context embeddings V are treated identically. We use a factorized distribution over both embedding space and vocabulary space,

$$q(U_{1:T}) = \prod_{i=1}^{L} \prod_{k=1}^{d} q(u_{ik,1:T}). \qquad (14)$$

In the time domain, our variational approximation is structured. To simplify the notation we now drop the indices for words i and embedding dimension k, hence we write $q(u_{1:T})$ for $q(u_{ik,1:T})$ where we focus on a single factor. This factor is a multivariate Gaussian distribution in the time domain with tridiagonal precision matrix Λ,

$$q(u_{1:T}) = \mathcal{N}(\mu, \Lambda^{-1}) \qquad (15)$$

Both the means $\mu = \mu_{1:T}$ and the entries of the tridiagonal precision matrix $\Lambda \in \mathbb{R}^{T \times T}$ are variational parameters. This gives our variational distribution the interpretation of a posterior of a Kalman filter (Blei & Lafferty, 2006) which captures correlations in time.

We fit the variational parameters by training the model jointly on all time steps, using black-box VI and the reparameterization trick. As the computational complexity of an update step scales as $\Theta(L^2)$, we first pretrain the model by drawing minibatches of $L' < L$ random words and L' random contexts from the vocabulary (Hoffman et al., 2013). We then switch to the full batch to reduce the sampling noise. Since the variational distribution does not factorize in the time domain we always include all time steps $\{1, \ldots, T\}$ in the minibatch.

We also derive an efficient algorithm that allows us to estimate the reparametrization gradient using $\Theta(T)$ time and memory, while a naive implementation of black-box variational inference with our structured variational distribution would require $\Theta(T^2)$ of both resources. The main idea is to parameterize $\Lambda = B^{\top} B$ in terms of its Cholesky decomposition B, which is bidiagonal (Kılıç & Stanica, 2013), and to express gradients of the dense (upper triangular) matrix B^{-1} in terms of gradients of B, which are sparse. We use mirror ascent (Ben-Tal et al., 2001; Beck & Teboulle, 2003) to enforce positive definiteness of B. The algorithm is detailed in our supplementary material. It does not depend on any specific aspects of the dynamic skip-gram model and applies to other latent time-series models.

5. Experiments

We evaluate our method on three time-stamped text corpora. We demonstrate that our algorithms find smoother embedding trajectories than methods based on a static model. This allows us to track semantic changes of individual words by following nearest-neighbor relations over time. In our quantitative analysis, we find higher predictive likelihoods on held-out data compared to our baselines.

Algorithms and Baselines. We report results from our proposed algorithms from section 4 and compare against baselines from section 2:

- SGI denotes the non-Bayesian skip-gram model with independent random initializations of word vectors (Mikolov et al., 2013b). We used our own implementation of the model by dropping the Kalman filtering prior and point-estimating embedding vectors. Word vectors at nearby times are made comparable by approximate orthogonal transformations, which corresponds to Hamilton et al. (2016).
- SGP denotes the same approach as above, but with word and context vectors being pre-initialized with the values from the previous year, as in Kim et al. (2014).
- DSG-F: dynamic skip-gram filtering (proposed).
- DSG-S: dynamic skip-gram smoothing (proposed).

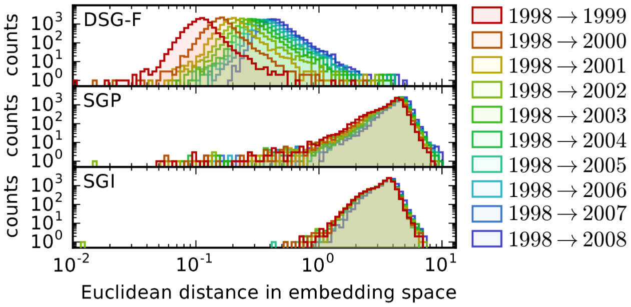

Figure 3. Histogram of distances between word vectors in the year 1998 and their positions in subsequent years (colors). DSG-F (top panel, proposed) displays a continuous growth of these distances over time, indicating a directed motion. In contrast, in SGP (middle) (Kim et al., 2014) and SGI (bottom) (Hamilton et al., 2016), the distribution of distances jumps from the first to the second year but then remains largely stationary. This indicates that the random motion due to noise dominates over the directed drift.

Data and preprocessing. Our three corpora exemplify opposite limits both in the covered time span and in the amount of text per time step.

1. We used data from the **Google books** corpus[1] (Michel et al., 2011) from the last two centuries ($T = 209$). This amounts to 5 million digitized books and approximately 10^{10} observed words. The corpus consists of n-gram tables with $n \in \{1, \ldots, 5\}$, annotated by year of publication. We considered the years from 1800 to 2008. In 1800, the size of the data is approximately $\sim 7 \cdot 10^7$ words. We used the 5-gram counts, resulting in a context window size of 4.
2. We used the "State of the Union" (**SoU**) addresses of U.S. presidents which spans more than two centuries, resulting in $T = 230$ different time steps and approximately 10^6 observed words.[2] Some presidents gave both a written and an oral address; if these were less than a week apart we concatenated them and used the average date. We converted all words to lower case and constructed the positive sample counts n^+_{ij} using a context window size of 4.
3. We used a **Twitter** corpus of news tweets for 21 randomly drawn dates from 2010 to 2016. The median number of tweets per day is 722. We converted all tweets to lower case and used a context window size of 4, which we restricted to stay within single tweets.

Hyperparameters. The vocabulary for each corpus was constructed from the 10,000 most frequent words throughout the given time period. In the Google books corpus, the number of words per year grows by a factor of 200 from the year 1800 to 2008. To avoid that the vocabulary is dominated by modern words we normalized the word frequencies separately for each year before adding them up.

For the Google books corpus, we chose the embedding dimension $d = 200$, which was also used in Kim et al. (2014). We set $d = 100$ for SoU and Twitter, as $d = 200$ resulted in overfitting on these much smaller corpora. The ratio $\eta = \sum_{ij} n^-_{ij,t} / \sum_{ij} n^+_{ij,t}$ of negative to positive word-context pairs was $\eta = 1$. The precise construction of the matrices $n^\pm_t$ is explained in the supplementary material. We used the global prior variance $\sigma_0^2 = 1$ for all corpora and all algorithms, including the baselines. The diffusion constant D controls the time scale on which information is shared between time steps. The optimal value for D depends on the application. A single corpus may exhibit semantic shifts of words on different time scales, and the optimal choice for D depends on the time scale in which one is interested. We measured time in years and used $D = 10^{-3}$ for Google books, $D = 10^{-4}$ for SoU, and $D = 1$ for the Twitter corpus, which spans a much shorter time range. In the supplementary material, we provide details of the optimization procedure.

Qualitative results. We show that our approach results in smooth word embedding trajectories on all corpora. We can automatically detect words that undergo significant semantic changes over time.

Figure 1 in the introduction visualizes word embedding clouds over four subsequent years of Google books, where we compare DSG-F against SGI. We mapped the normalized embedding vectors to two dimensions using dynamic t-SNE (Rauber et al., 2016) (see supplement for details).

[1] http://storage.googleapis.com/books/ngrams/books/datasetsv2.html

[2] http://www.presidency.ucsb.edu/sou.php

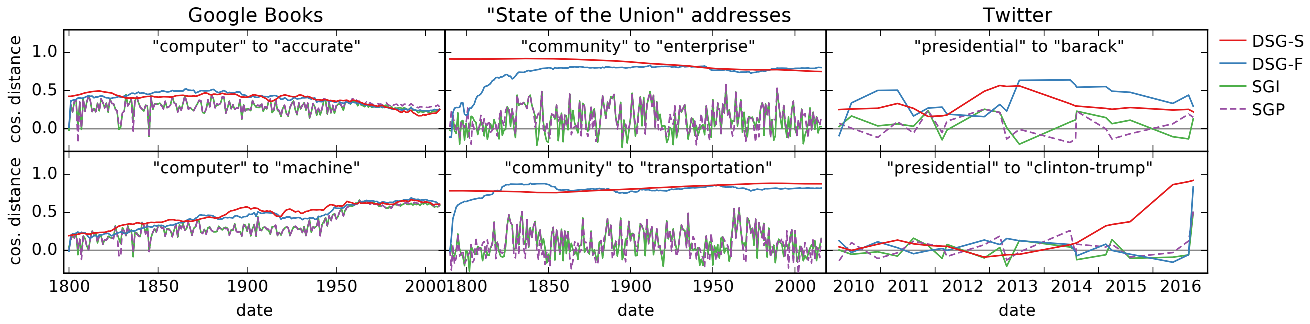

Figure 4. Smoothness of word embedding trajectories, compared across different methods. We plot the cosine distance between two words (see captions) over time. High values indicate similarity. Our methods (DSG-S and DSG-F) find more interpretable trajectories than the baselines (SGI and SGP). The different performance is most pronounced when the corpus is small (SoU and Twitter).

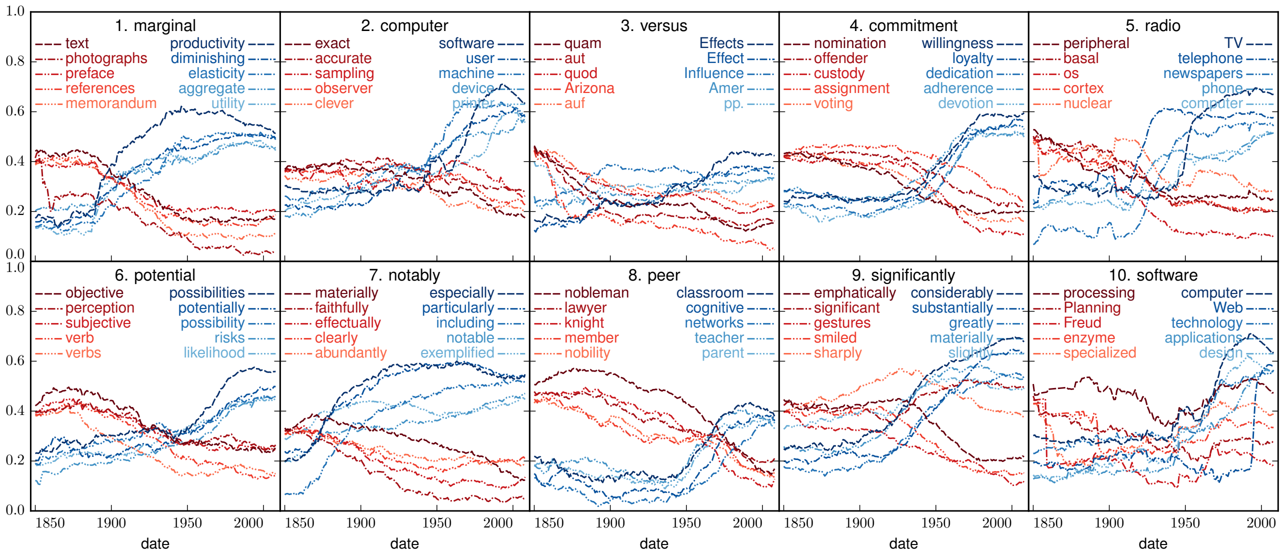

Figure 5. Evolution of the 10 words whose embedding vector changed the most (in cosine distance) from 1850 to 2008 (DSG-F on Google books). Red (blue) curves correspond to the five closest words at the beginning (end) of the time span, respectively.

Lines indicate shifts of word vectors relative to the preceding year. In our model only few words change their position in the embedding space rapidly, while embeddings using SGI show strong fluctuations, making the cloud's motion hard to track.

Figure 3 visualizes the smoothness of the trajectories directly in the embedding space (without the projection to two dimensions). We consider differences between word vectors in the year 1998 and the subsequent 10 years. In more detail, we compute histograms of the Euclidean distances $||u_{it} - u_{i,t+\delta}||$ over the word indexes i, where $\delta = 1, \ldots, 10$ (as discussed previously, SGI uses a global rotation to optimally align embeddings first). In our model, embedding vectors gradually move away from their original position as time progresses, indicating a directed motion. In contrast, both baseline models show no directed motion after the first time step, suggesting that most temporal changes are due to finite-size fluctuations of $n^{\pm}_{ij,t}$.

Our approach allows us to detect semantic shifts in the usage of specific words. Figures 4 and 5 both show the cosine angle between a given word and its neighboring words (colored lines) as a function of time. Figure 4 shows results on all three corpora and focuses on the comparisons across methods. We see that for DSG-S and DSG-F (proposed) result in trajectories which display less noise than the baselines SGP and SGI. The fact that the baselines predict zero cosine angle (no correlation) between the chosen word pairs on the SoU and Twitter corpora suggests that these corpora are too small to successfully fit these models, in contrast to our approach which shares information in the time domain.

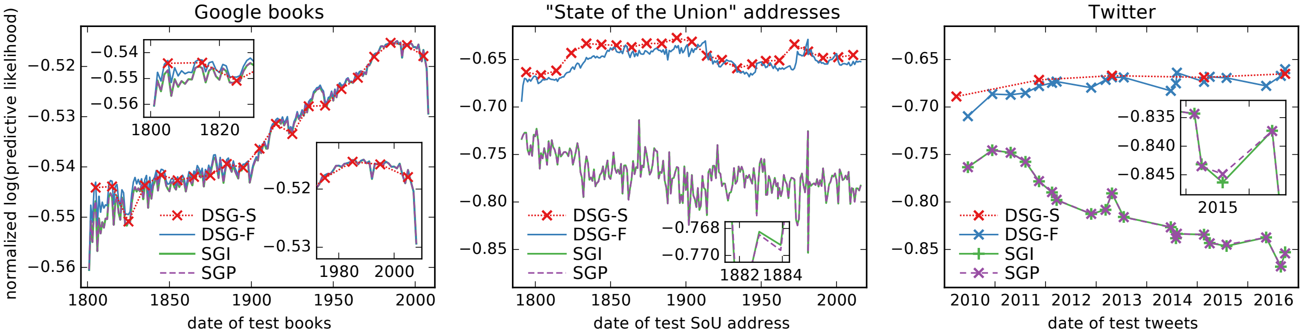

Figure 6. Predictive log-likelihoods (Eq. 16) for two proposed versions of the dynamic skip-gram model (DSG-F & DSG-S) and two competing methods SGI (Hamilton et al., 2016) and SGP (Kim et al., 2014) on three different corpora (high values are better).

Figure 5 then focuses on the Google books corpus. Here, we show the ten words that change their meaning most rapidly in terms of cosine distance. We thereby automatically discover words such as ”computer”, ”radio”, and ”software” that changed their meaning due to technological advances, but also words as ”peer” and ”notably” whose semantic shift is arguably less obvious.

Quantitative results. We show that our approach generalizes better to unseen data. We thereby analyze held-out predictive likelihoods on word-context pairs at a given time t, where t is excluded from the training set,

$$\frac{1}{|n_t^\pm|} \log p(n_t^\pm | \tilde{U}_t, \tilde{V}_t). \tag{16}$$

Above, $|n_t^\pm| = \sum_{i,j} \left(n_{ij,t}^+ + n_{ij,t}^-\right)$ denotes the total number of word-context pairs at time τ_t. Since inference is different in all approaches, the definitions of word and context embedding matrices $\tilde{U}_t$ and $\tilde{V}_t$ in Eq. 16 have to be adjusted:

- For SGI and SGP, we did a chronological pass through the time sequence and used the embeddings $\tilde{U}_t = U_{t-1}$ and $\tilde{V}_t = V_{t-1}$ from the previous time step to predict the statistics $n_{ij,t}^\pm$ at time step t.

- For DSG-F, we did the same pass to test $n_{ij,t}^\pm$. We thereby set $\tilde{U}_t$ and $\tilde{V}_t$ to be the modes U_{t-1}, V_{t-1} of the approximate posterior at the previous time step.

- For DSG-S, we held out 10%, 10% and 20% of the documents from the Google books, SoU, and Twitter corpora for testing, respectively. After training, we estimated the word (context) embeddings $\tilde{U}_t$ ($\tilde{V}_t$) in Eq. 16 by linear interpolation between the values of U_{t-1} (V_{t-1}) and U_{t+1} (V_{t+1}) in the mode of the variational distribution, taking into account that the time stamps τ_t are in general not equally spaced.

The predictive likelihoods as a function of time τ_t are shown in Figure 6. Differences between the two implementations of the static model (SGI and SGP) are small. This suggests that pre-initializing the embeddings with the previous result may improve their continuity but seems to have a minor impact on predictive power.

We see that both proposed versions of the dynamic model (DSG-F and DSG-S) outperform the baselines SGP and SGI. The improvements are particularly pronounced in the SoU and Twitter corpora (center and right panels in Figure 6), for which sharing information between time steps is crucial because there is little data at each time slice. In the Google Books corpus (left panel), the number of words per year grows by a factor of 230 from 1800 to 2008. This explains why the quantitative improvements by the dynamic model are only noticeable at the beginning of the considered time span, and why the performance of all methods increases over time. Further, smoothing (DSG-S) outperforms filtering (DSG-F), as this algorithm can use information from both past and future observations.

6. Conclusions

We presented the dynamic skip-gram model: a Bayesian probabilistic model that combines word2vec with a latent continuous time series. We showed experimentally that both skip-gram filtering (which conditions only on past observations) and skip-gram smoothing (which uses all data) lead to smoothly changing embedding vectors that are better at predicting word-context statistics at held-out time bins. The benefits are most drastic when the data at individual time steps is small; making it hard to fit a static word embedding model. Our approach may be used as a data mining and anomaly detection tool when streaming text on social media, as well as a tool for historians and social scientists interested in the evolution of language.

References

Barkan, Oren. Bayesian Neural Word Embedding. *arXiv preprint arXiv:1603.06571*, 2016.

Beck, Amir and Teboulle, Marc. Mirror Descent and Nonlinear Projected Subgradient Methods for Convex Optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Ben-Tal, Aharon, Margalit, Tamar, and Nemirovski, Arkadi. The Ordered Subsets Mirror Descent Optimization Method with Applications to Tomography. *SIAM Journal on Optimization*, 12(1):79–108, 2001.

Bengio, Yoshua, Ducharme, Réjean, Vincent, Pascal, and Jauvin, Christian. A Neural Probabilistic Language Model. *journal of machine learning research*, 3(Feb): 1137–1155, 2003.

Blei, David M and Lafferty, John D. Dynamic Topic Models. In *Proceedings of the 23rd international conference on Machine learning*, pp. 113–120. ACM, 2006.

Blei, David M., Kucukelbir, Alp, and McAuliffe, Jon D. Variational Inference: A Review for Statisticians. *arXiv preprint arXiv:1601.00670*, 2016.

Charlin, Laurent, Ranganath, Rajesh, McInerney, James, and Blei, David M. Dynamic Poisson Factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pp. 155–162, 2015.

Fu, Yanwei and Sigal, Leonid. Semi-Supervised Vocabulary-Informed Learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5337–5346, 2016.

Gultekin, San and Paisley, John. A Collaborative Kalman Filter for Time-Evolving Dyadic Processes. In *International Conference on Data Mining*, pp. 140–149, 2014.

Hamilton, William L, Leskovec, Jure, and Jurafsky, Dan. Diachronic word embeddings reveal statistical laws of semantic change. *arXiv preprint arXiv:1605.09096*, 2016.

Hoffman, Matthew D, Blei, David M, Wang, Chong, and Paisley, John William. Stochastic Variational Inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Jerfel, Ghassen, Basbug, Mehmet E, and Engelhardt, Barbara E. Dynamic Compound Poisson Factorization. In *Artificial Intelligence and Statistics*, 2017.

Jordan, Michael I, Ghahramani, Zoubin, Jaakkola, Tommi S, and Saul, Lawrence K. An Introduction to Variational Methods for Graphical Models. *Machine learning*, 37(2):183–233, 1999.

Kalman, Rudolph Emil et al. A New Approach to Linear Filtering and Prediction Problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

Kılıç, Emrah and Stanica, Pantelimon. The Inverse of Banded Matrices. *Journal of Computational and Applied Mathematics*, 237(1):126–135, 2013.

Kim, Yoon, Chiu, Yi-I, Hanaki, Kentaro, Hegde, Darshan, and Petrov, Slav. Temporal Analysis of Language Through Neural Language Models. *arXiv preprint arXiv:1405.3515*, 2014.

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

Kulkarni, Vivek, Al-Rfou, Rami, Perozzi, Bryan, and Skiena, Steven. Statistically Significant Detection of Linguistic Change. In *Proceedings of the 24th International Conference on World Wide Web*, pp. 625–635, 2015.

Levy, Omer and Goldberg, Yoav. Neural Word Embedding as Implicit Matrix Factorization. In *Advances in neural information processing systems*, pp. 2177–2185, 2014.

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

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

Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Efficient Estimation of Word Representations in Vector Space. In *International Conference on Learning Representations*, 2013a.

Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pp. 3111–3119, 2013b.

Mikolov, Tomas, Yih, Wen-tau, and Zweig, Geoffrey. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT-2013)*, pp. 746–751, 2013c.

Mnih, Andriy and Kavukcuoglu, Koray. Learning Word Embeddings Efficiently with Noise-Contrastive Estimation. In *Advances in Neural Information Processing Systems*, pp. 2265–2273, 2013.

Pennington, Jeffrey, Socher, Richard, and Manning, Christopher D. Glove: Global Vectors for Word Representation. In *EMNLP*, volume 14, pp. 1532–43, 2014.

Ranganath, Rajesh, Gerrish, Sean, and Blei, David M. Black Box Variational Inference. In *AISTATS*, pp. 814–822, 2014.

Ranganath, Rajesh, Perotte, Adler J, Elhadad, Noémie, and Blei, David M. The Survival Filter: Joint Survival Analysis with a Latent Time Series. In *UAI*, pp. 742–751, 2015.

Rauber, Paulo E., Falcão, Alexandre X., and Telea, Alexandru C. Visualizing Time-Dependent Data Using Dynamic t-SNE. In *EuroVis 2016 - Short Papers*, 2016.

Rezende, Danilo Jimenez, Mohamed, Shakir, and Wierstra, Daan. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 23rd international conference on Machine learning*, 2014.

Rudolph, Maja, Ruiz, Francisco, Mandt, Stephan, and Blei, David. Exponential Family Embeddings. In *Advances in Neural Information Processing Systems*, pp. 478–486, 2016.

Sagi, Eyal, Kaufmann, Stefan, and Clark, Brady. Tracing Semantic Change with Latent Semantic Analysis. *Current methods in historical semantics*, pp. 161–183, 2011.

Sahoo, Nachiketa, Singh, Param Vir, and Mukhopadhyay, Tridas. A Hidden Markov Model for Collaborative Filtering. *MIS Quarterly*, 36(4):1329–1356, 2012.

Salimans, Tim and Kingma, Diederik P. Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks. In *Advances in Neural Information Processing Systems*, pp. 901–901, 2016.

Socher, Richard, Bauer, John, Manning, Christopher D, and Ng, Andrew Y. Parsing with Compositional Vector Grammars. In *ACL (1)*, pp. 455–465, 2013a.

Socher, Richard, Perelygin, Alex, Wu, Jean Y, Chuang, Jason, Manning, Christopher D, Ng, Andrew Y, and Potts, Christopher. Recursive Deep Models for Semantic Compositionality over a Sentiment Treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, pp. 1642, 2013b.

Uhlenbeck, George E and Ornstein, Leonard S. On the Theory of the Brownian Motion. *Physical review*, 36(5): 823, 1930.

Vilnis, Luke and McCallum, Andrew. Word Representations via Gaussian Embedding. In *International Conference on Learning Representations*, 2014.

Wang, Chong, Blei, David, and Heckerman, David. Continuous Time Dynamic Topic Models. In *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, 2008.

Welch, Greg and Bishop, Gary. An introduction to the kalman filter. Technical report, University of North Carolina at Chapel Hill, 1995.

Supplementary Material to “Dynamic Word Rmbeddings via Skip-Gram Filtering”

Robert Bamler ROBERT.BAMLER@DISNEYRESEARCH.COM
Disney Research, 4720 Forbes Avenue, Pittsburgh, PA 15213 USA

Stephan Mandt STEPHAN.MANDT@DISNEYRESEARCH.COM
Disney Research, 4720 Forbes Avenue, Pittsburgh, PA 15213 USA

Table S1. Hyperparameters for skip-gram filtering and skip-gram smoothing.

PARAMETER	COMMENT
$L = 10^4$	vocabulary size
$L' = 10^3$	batch size for smoothing
$d = 100$	embedding dimension for SoU and Twitter
$d = 200$	embedding dimension for Google books
$N_{\text{tr}} = 5000$	number of training steps for each t (filtering)
$N'_{\text{tr}} = 5000$	number of pretraining steps with minibatch sampling (smoothing; see Algorithm 2)
$N_{\text{tr}} = 1000$	number of training steps without minibatch sampling (smoothing; see Algorithm 2)
$c_{\max} = 4$	context window size for positive examples
$\eta = 1$	ratio of negative to positive examples
$\gamma = 0.75$	context exponent for negative examples
$D = 0.01$	diffusion const. per year (Google books & SoU)
$D = 0.1$	diffusion const. per year (Twitter)
$\sigma_0^2 = 1$	variance of overall prior
$\alpha = 10^{-2}$	learning rate (filtering)
$\alpha' = 10^{-2}$	learning rate during minibatch phase (smoothing)
$\alpha = 10^{-3}$	learning rate after minibatch phase (smoothing)
$\beta_1 = 0.9$	decay rate of 1st moment estimate
$\beta_2 = 0.99$	decay rate of 2nd moment estimate (filtering)
$\beta_2 = 0.999$	decay rate of 2nd moment estimate (smoothing)
$\delta = 10^{-8}$	regularizer of Adam optimizer

1. Dimensionality Reduction in Figure 1

To create the word-clouds in Figure 1 of the main text we mapped the fitted word embeddings from $\mathbb{R}^d$ to the two-dimensional plane using dynamic t-SNE (Rauber et al., 2016). Dynamic t-SNE is a non-parametric dimensionality reduction algorithm for sequential data. The algorithm finds a projection to a lower dimension by solving a non-convex optimization problem that aims at preserving nearest-neighbor relations at each individual time step. In addition, projections at neighboring time steps are aligned with each other by a quadratic penalty with prefactor $\lambda \geq 0$ for sudden movements.

There is a trade-off between finding good local projections for each individual time step ($\lambda \to 0$), and finding smooth projections (large λ). Since we want to analyze the smoothness of word embedding trajectories, we want to avoid bias towards smooth projections. Unfortunately, setting $\lambda = 0$ is not an option since, in this limit, the optimization problem is invariant under independent rotations at each time, rendering trajectories in the two-dimensional projection plane meaningless. To still avoid bias towards smooth projections, we anneal λ exponentially towards zero over the course of the optimization. We start the optimizer with $\lambda = 0.01$, and we reduce λ by 5% with each training step. We run 100 optimization steps in total, so that $\lambda \approx 6 \times 10^{-6}$ at the end of the training procedure. We used the open-source implementation,[1] set the target perplexities to 200, and used default values for all other parameters.

2. Hyperparemeters and Construction of $n^{\pm}_{1:T}$

Table S1 lists the hyperparameters used in our experiments. For the Google books corpus, we used the same context window size $c_{\max}$ and embedding dimension d as in (Kim et al., 2014). We reduced d for the SoU and Twitter corpora to avoid overfitting to these much smaller data sets.

In constrast to word2vec, we construct our positive and negative count matrices $n^{\pm}_{ij,t}$ deterministically in a preprocessing step. As detailed below, this is done such that it resembles as closely as possible the stochastic approach in word2vec (Mikolov et al., 2013). In every update step, word2vec stochastically samples a context window size uniformly in an interval $[1, \cdots, c_{max}]$, thus the context size fluctuates and nearby words appear more often in the same context than words that are far apart from each other in the sentence. We follow a deterministic scheme that results in similar statistics. For each pair of words (w_1, w_2)

[1] https://github.com/paulorauber/thesne

Algorithm 1 Skip-gram filtering; see section 4 of the main text.

```
Remark: All updates are analogous for word and context vectors; we drop their indices for simplicity.
Input: number of time steps $T$, time stamps $\tau_{1:T}$, positive and negative examples $n^{\pm}_{1:T}$, hyperparameters.

Init. prior means $\tilde{\mu}_{ik,1} \leftarrow 0$ and variances $\tilde{\Sigma}_{i,1} = I_{d\times d}$.
Init. variational means $\mu_{ik,1} \leftarrow 0$ and var. $\Sigma_{i,1} = I_{d\times d}$.
for $t = 1$ to $T$ do
  if $t \neq 1$ then
    Update approximate Gaussian prior with params. $\tilde{\mu}_{ik,t}$ and $\tilde{\Sigma}_{i,t}$ using $\mu_{ik,t-1}$ and $\Sigma_{i,t-1}$, see Eq. 13.
  end if
  Compute entropy $\mathbb{E}_q[\log q(\cdot)]$ analytically.
  Compute expected log Gaussian prior with parameters $\tilde{\mu}_{ik,t}$ and $\tilde{\Sigma}_{k,t}$ analytically.
  Maximize $\mathcal{L}_t$ in Eq. 11, using black-box VI with the reparametrization trick.
  Obtain $\mu_{ik,t}$ and $\Sigma_{i,t}$ as outcome of the optimization.
end for
```

in a given sentence, we increase the counts $n^{+}_{i_{w_1} j_{w_2}}$ by $\max(0, 1 - k/c_{\text{max}})$, where $0 \leq k \leq c_{max}$ is the number of words that appear between w_1 and w_2, and i_{w_1} and j_{w_2} are the words' unique indices in the vocabulary.

We also used a deterministic variant of word2vec to construct the negative count matrices n^{-}_{t}. In word2vec, η negative samples (i, j) are drawn for each positive sample (i, j') by drawing η independent values for j from a distribution $P'_t(j)$ defined below. We define $n^{-}_{ij,t}$ such that it matches the expectation value of the number of times that word2vec would sample the negative word-context pair (i, j). Specifically, we define

$$P_t(i) = \frac{\sum_{j=1}^{L} n^{+}_{ij,t}}{\sum_{i',j=1}^{L} n^{+}_{i'j,t}}, \tag{S1}$$

$$P'_t(j) = \frac{\big(P_t(j)\big)^{\gamma}}{\sum_{j'=1}^{L} \big(P_t(j')\big)^{\gamma}}, \tag{S2}$$

$$n^{-}_{ij,t} = \Bigg(\sum_{i',j'=1}^{L} n^{+}_{i'j',t}\Bigg)\eta P_t(i) P'_t(j). \tag{S3}$$

We chose $\gamma = 0.75$ as proposed in (Mikolov et al., 2013), and we set $\eta = 1$. In practice, it is not necessary to explicitly construct the full matrices n^{-}_{t}, and it is more efficient to keep only the distributions $P_t(i)$ and $P'_t(j)$ in memory.

3. Skip-gram Filtering Algorithm

The skip-gram filtering algorithm is described in section 4 of the main text. We provide a formulation in pseudocode in Algorithm 1.

Algorithm 2 Skip-gram smoothing (batch version); see section 4.

```
Remark: As in Algorithm 1, we focus on a single word vector $u$, and we drop indices $i$ and $k$; we also just consider a single sample ($S = 1$), and we describe the basic algorithm without minibatch sampling.
Input: number of time steps $T$, time stamps $\tau_{1:T}$, positive and negative examples $n^{\pm}_{1:T}$, hyperparameters.

Find the upper bidiagonal matrix $B_0$ that is the Cholesky decomposition of the prior precision matrix $\Pi$, Eq. S11.
Initialize the variational parameters $\nu_{1:T}$ with the elements of the main diagonal of $B_0$.
Initialize the variation parameters $\omega_{1:T-1}$ with the elements of the secondary diagonal of $B_0$.
Initialize the posterior means $\mu_{1:T} \leftarrow 0$.
for $step = 1$ to $N_{\text{tr}}$ do
  Draw $T$ independent Gauss. noises $\epsilon_{1:T} \sim \mathcal{N}(0, I)$.
  Solve $Bx_{1:T} = \epsilon_{1:T}$ for $x_{1:T}$ in $\Theta(T)$ time using the bidiagonal structure of $B$ defined in Eq. S5.
  Compute $\Gamma_{1:T}$ from Eqs. S12 and S8.
  Obtain $\partial\mathcal{L}/\partial\mu_{1:T}$, i.e., the derivative of the stochastic ELBO $\mathcal{L}$ with respect to $\mu_{1:T}$, see Eq. S10.
  Solve $B^{\top} y_{1:T} = \partial\mathcal{L}/\partial\mu_{1:T}$ for $y_{1:T}$ in $\Theta(T)$ time using the bidiagonal structure of $B$, see Eq. S16.
  Obtain $\partial\mathcal{L}/\partial\nu_{1:T}$ from Eq. S14.
  Obtain $\partial\mathcal{L}/\partial\omega_{1:T-1}$ from Eq. S15.
  Do a stochastic gradient step in $\mu_{1:T}$, $\nu_{1:T}$, and $\omega_{1:T-1}$ using Adam optimizer and mirror ascent, see Eq. S18.
end for
```

4. Skip-gram Smoothing Algorithm

In this section, we give details for the skip-gram smoothing algorithm, see section 4 of the main text. A summary is provided in Algorithm 2.

Variational distribution. For now, we focus on the word embeddings, and we simplify the notation by dropping the indices for the vocabulary and embedding dimensions. The variational distribution for a single embedding dimension of a single word embedding trajectory is

$$q(u_{1:T}) = \mathcal{N}(\mu_{u,1:T}, (B_u^{\top} B_u)^{-1}). \tag{S4}$$

Here, $\mu_{u,1:T}$ is the vector of mean values, and B_u is the Cholesky decomposition of the precision matrix. We restrict the latter to be bidiagonal,

$$B_u = \begin{pmatrix} \nu_{u,1} & \omega_{u,1} & & & \\ & \nu_{u,2} & \omega_{u,2} & & \\ & & \ddots & \ddots & \\ & & & \nu_{u,T-1} & \omega_{u,T-1} \\ & & & & \nu_T \end{pmatrix} \tag{S5}$$

with $\nu_{u,t} > 0$ for all $t \in \{1, \ldots, T\}$. The variational parameters are $\mu_{u,1:T}$, $\nu_{u,1:T}$, and $\omega_{1:T-1}$. The variational distribution of the context embedding trajectories $v_{1:T}$ has the same structure.

With the above form of B_u, the variational distribution is a Gaussian with an arbitrary tridiagonal symmetric precision matrix $B_u^\top B_u$. We chose this variational distribution because it is the exact posterior of a hidden time-series model with a Kalman filtering prior and Gaussian noise (Blei & Lafferty, 2006). Note that our variational distribution is a generalization of a fully factorized (mean-field) distribution, which is obtained for $\omega_{u,t} = 0 \ \forall t$. In the general case, $\omega_{u,t} \neq 0$, the variational distribution can capture correlations between all time steps, with a dense covariance matrix $(B_u^\top B_u)^{-1}$.

Gradient estimation. The skip-gram smoothing algorithm uses stochastic gradient ascent to find the variational parameters that maximize the ELBO,

$$\mathcal{L} = \mathbb{E}_q\big[\log p(U_{1:T}, V_{1:T}, n^{\pm}_{1:T})\big] - \mathbb{E}_q\big[\log q(U_{1:T}, V_{1:T})\big]. \tag{S6}$$

Here, the second term is the entropy, which can be evaluated analytically. We obtain for each component in vocabulary and embedding space,

$$-\mathbb{E}_q[\log q(u_{1:T})] = -\sum_t \log(\nu_{u,t}) + \text{const.} \tag{S7}$$

and analogously for $-E_q[\log q(v_{1:T})]$.

The first term on the right-hand side of Eq. S6 cannot be evaluated analytically. We approximate its gradient by sampling from q using the reparameterization trick (Kingma & Welling, 2014; Rezende et al., 2014). A naive calculation would require $\Omega(T^2)$ computing time since the derivatives of $\mathcal{L}$ with respect to $\nu_{u,t}$ and $\omega_{u,t}$ for each t depend on the count matrices $n^{\pm}_{t'}$ of all t'. However, as we show next, there is a more efficient way to obtain all gradient estimates in $\Theta(T)$ time.

We focus again on a single dimension of a single word embedding trajectory $u_{1:T}$, and we drop the indices i and k. We draw S independent samples $u^{[s]}_{1:T}$ with $s \in \{1, \ldots, S\}$ from $q(u_{1:T})$ by parameterizing

$$u^{[s]}_{1:T} = \mu_{u,1:T} + x^{[s]}_{u,1:T} \tag{S8}$$

with

$$x^{[s]}_{u,1:T} = B_u^{-1} \epsilon^{[s]}_{u,1:T} \quad \text{where} \quad \epsilon^{[s]}_{u,1:T} \sim \mathcal{N}(0, I). \tag{S9}$$

We obtain $x^{[s]}_{u,1:T}$ in $\Theta(T)$ time by solving the bidiagonal linear system $B_u x^{[s]}_{u,1:T} = \epsilon^{[s]}_{u,1:T}$. Samples $v^{[s]}_{1:T}$ for the context embedding trajectories are obtained analogously. Our implementation uses $S = 1$, i.e., we draw only a single sample per training step. Averaging over several samples is done implicitly since we calculate the update steps using the Adam optimizer (Kingma & Ba, 2015), which effectively averages over several gradient estimates in its first moment estimate.

The derivatives of $\mathcal{L}$ with respect to $\mu_{u,1:T}$ can be obtained using Eq. S8 and the chain rule. We find

$$\frac{\partial \mathcal{L}}{\partial \mu_{u,1:T}} \approx \frac{1}{S} \sum_{s=1}^{S} \Big[\Gamma^{[s]}_{u,1:T} - \Pi u^{[s]}_{1:T}\Big]. \tag{S10}$$

Here, $\Pi \in \mathbb{R}^{T \times T}$ is the precision matrix of the prior $u_{1:T} \sim \mathcal{N}(0, \Pi^{-1})$. It is tridiagonal and therefore the matrix-multiplication $\Pi u^{[s]}_{1:T}$ can be carried out efficiently. The non-zero matrix elements of Π are

$$\begin{aligned}
\Pi_{11} &= \sigma_0^{-2} + \sigma_1^{-2} \\
\Pi_{TT} &= \sigma_0^{-2} + \sigma_{T-1}^{-2} \\
\Pi_{tt} &= \sigma_0^{-2} + \sigma_{t-1}^{-2} + \sigma_t^{-2} \quad \forall t \in \{2, \ldots, T-1\} \\
\Pi_{1,t+1} &= \Pi_{t+1,1} = -\sigma_t^{-2}.
\end{aligned} \tag{S11}$$

The term $\Gamma^{[s]}_{u,1:T}$ on the right-hand side of Eq. S10 comes from the expectation value of the log-likelihood under q. It is given by

$$\Gamma^{[s]}_{ui,t} = \sum_{j=1}^{L} \Big[\big(n^+_{ij,t} + n^-_{ij,t}\big)\, \sigma\Big(-u^{[s]\top}_{i,t} v^{[s]}_{j,t}\Big) - n^-_{ij,t}\Big] v^{[s]}_{j,t} \tag{S12}$$

where we temporarily restored the indices i and j for words and contexts, respectively. In deriving Eq. S12, we used the relations $\partial \log \sigma(x)/\partial x = \sigma(-x)$ and $\sigma(-x) = 1 - \sigma(x)$.

The derivatives of $\mathcal{L}$ with respect to $\nu_{u,t}$ and $\omega_{u,t}$ are more intricate. Using the parameterization in Eqs. S8–S9, the derivatives are functions of $\partial B_u^{-1}/\partial \nu_t$ and $\partial B_u^{-1}/\partial \omega_t$, respectively, where B_u^{-1} is a dense (upper triangular) $T \times T$ matrix. An efficient way to obtain these derivatives is to use the relation

$$\frac{\partial B_u^{-1}}{\partial \nu_t} = -B_u^{-1} \frac{\partial B_u}{\partial \nu_t} B_u^{-1} \tag{S13}$$

and similarly for $\partial B_u^{-1}/\partial \omega_t$. Using this relation and Eqs. S8–S9, we obtain the gradient estimates

$$\frac{\partial \mathcal{L}}{\partial \nu_{u,t}} \approx -\frac{1}{S} \sum_{s=1}^{S} y^{[s]}_{u,t} x^{[s]}_{u,t} - \frac{1}{\nu_{u,t}}, \tag{S14}$$

$$\frac{\partial \mathcal{L}}{\partial \omega_{u,t}} \approx -\frac{1}{S} \sum_{s=1}^{S} y^{[s]}_{u,t} x^{[s]}_{u,t+1}. \tag{S15}$$

The second term on the right-hand side of Eq. S14 is the derivative of the entropy, Eq. S7, and

$$y_{u,1:T}^{[s]} = (B_u^\top)^{-1} \left[\Gamma_{u,1:T}^{[s]} - \Pi u_{1:T}^{[s]} \right]. \tag{S16}$$

The values $y_{u,1:T}^{[s]}$ can again be obtained in $\Theta(T)$ time by bringing $B_u^\top$ to the left-hand side and solving the corresponding bidiagonal linear system of equations.

Sampling in vocabulary space. In the above paragraph, we described an efficient strategy to obtain gradient estimates in only $\Theta(T)$ time. However, the gradient estimation scales quadratic in the vocabulary size L because all L^2 elements of the positive count matrices n_t^+ contribute to the gradients. In order speed up the optimization, we pretrain the model using a minibatch of size $L' < L$ in vocabulary space as explained below. The computational complexity of a single training step in this setup scales as $(L')^2$ rather than L^2. After $N'_{\text{tr}} = 5000$ training steps with minibatch size L', we switch to the full batch size of L and train the model for another $N_{\text{tr}} = 1000$ steps.

The subsampling in vocabulary space works as follows. In each training step, we independently draw a set $\mathcal{I}$ of L' random distinct words and a set $\mathcal{J}$ of L' random distinct contexts from a uniform probability over the vocabulary. We then estimate the gradients of $\mathcal{L}$ with respect to only the variational parameters that correspond to words $i \in \mathcal{I}$ and contexts $j \in \mathcal{J}$. This is possible because both the prior of our dynamic skip-gram model and the variational distribution factorize in the vocabulary space. The likelihood of the model, however, does not factorize. This affects only the definition of $\Gamma_{uik,t}^{[s]}$ in Eq. S12. We replace $\Gamma_{uik,t}^{[s]}$ by an estimate $\Gamma_{uik,t}^{[s]\prime}$ based on only the contexts $j \in \mathcal{J}$ in the current minibatch,

$$\Gamma_{ui,t}^{[s]} = \frac{L}{L'} \sum_{j \in \mathcal{J}} \Big[\left(n_{ij,t}^+ + n_{ij,t}^- \right) \sigma\Big(-u_{i,t}^{[s]\top} v_{j,t}^{[s]} \Big) - n_{ij,t}^- \Big] v_{j,t}^{[s]}. \tag{S17}$$

Here, the prefactor L/L' restores the correct ratio between evidence and prior knowledge (Hoffman et al., 2013).

Enforcing positive definiteness. We update the variational parameters using stochastic gradient ascent with the Adam optimizer (Kingma & Ba, 2015). The parameters $\nu_{u,1:T}$ are the eigenvalues of the matrix B_u, which is the Cholesky decomposition of the precision matrix of q. Therefore, $\nu_{u,t}$ has to be positive for all $t \in \{1, \ldots, T\}$. We use mirror ascent (Ben-Tal et al., 2001; Beck & Teboulle, 2003) to enforce $\nu_{u,t} > 0$. Specifically, we update ν_t to a new value ν'_t defined by

$$\nu'_{u,t} = \frac{1}{2}\nu_{u,t} d[\nu_{u,t}] + \sqrt{\left(\frac{1}{2}\nu_{u,t} d[\nu_{u,t}] \right)^2 + \nu_{u,t}^2} \tag{S18}$$

where $d[\nu_{u,t}]$ is the step size obtained from the Adam optimizer. Eq. S18 can be derived from the general mirror ascent update rule $\Phi'(\nu'_{u,t}) = \Phi'(\nu_{u,t}) + d[\nu_{u,t}]$ with the mirror map $\Phi : x \mapsto -c_1 \log(x) + c_2 x^2/2$, where we set the parameters to $c_1 = \nu_{u,t}$ and $c_2 = 1/\nu_{u,t}$ for dimensional reasons. The update step in Eq. S18 increases (decreases) $\nu_{u,t}$ for positive (negative) $d[\nu_{u,t}]$, while always keeping its value positive.

Natural basis. As a final remark, let us discuss an optional extension to the skip-gram smoothing algorithm that converges in less training steps. This extension only increases the efficiency of the algorithm. It does not change the underlying model or the choice of variational distribution. Observe that the prior of the dynamic skip-gram model connects only neighboring time-steps with each other. Therefore, the gradient of $\mathcal{L}$ with respect to $\mu_{u,t}$ depends only on the values of $\mu_{u,t-1}$ and $\mu_{u,t+1}$. A naive implementation of gradient ascent would thus require $T-1$ update steps until a change of $\mu_{u,1}$ affects updates of $\mu_{u,T}$.

This problem can be avoided with a change of basis from $\mu_{u,1:T}$ to new parameters $\rho_{u,1:T}$,

$$\mu_{u,1:T} = A\rho_{u,1:T} \tag{S19}$$

with an appropriately chosen invertible matrix $A \in \mathbb{R}^{T \times T}$. Derivatives of $\mathcal{L}$ with respect to $\rho_{u,1:T}$ are given by the chain rule, $\partial\mathcal{L}/\partial\rho_{u,1:T} = (\partial\mathcal{L}/\partial\mu_{u,1:T})A$. A natural (but inefficient) choice for A is to stack the eigenvectors of the prior precision matrix Π, see Eq. S11, into a matrix. The eigenvectors of Π are the Fourier modes of the Kalman filtering prior (with a regularization due to σ_0). Therefore, there is a component $\rho_{u,t}$ that corresponds to the zero-mode of Π, and this component learns an average word embedding over all times. Higher modes correspond to changes of the embedding vector over time. A single update to the zero immediately affects all elements of $\mu_{u,1:T}$, and therefore changes the word embeddings at all time steps. Thus, information propagates quickly along the time dimension. The downside of this choice for A is that the transformation in Eq. S19 has complexity $\Omega(T^2)$, which makes update steps slow.

As a compromise between efficiency and a natural basis, we propose to set A in Eq. S19 to the Cholesky decomposition of the prior covariance matrix $\Pi^{-1} \equiv AA^\top$. Thus, A is still a dense (upper triangular) matrix, and, in our experiments, updates to the last component $\rho_{u,T}$ affect all components of $\mu_{u,1:T}$ in an approximately equal amount. Since Π is tridiagonal, the inverse of A is bidiagonal, and Eq. S19 can be evaluated in $\Theta(T)$ time by solving $A\mu_{u,1:T} = \rho_{u,1:T}$ for $\mu_{u,1:T}$. This is the parameterization we used in our implementation of the skip-gram smoothing algorithm.

References

Beck, Amir and Teboulle, Marc. Mirror Descent and Nonlinear Projected Subgradient Methods for Convex Optimization. *Operations Research Letters*, 31(3):167–175, 2003.

Ben-Tal, Aharon, Margalit, Tamar, and Nemirovski, Arkadi. The Ordered Subsets Mirror Descent Optimization Method with Applications to Tomography. *SIAM Journal on Optimization*, 12(1):79–108, 2001.

Blei, David M and Lafferty, John D. Dynamic Topic Models. In *Proceedings of the 23rd international conference on Machine learning*, pp. 113–120. ACM, 2006.

Hoffman, Matthew D, Blei, David M, Wang, Chong, and Paisley, John William. Stochastic Variational Inference. *Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

Kim, Yoon, Chiu, Yi-I, Hanaki, Kentaro, Hegde, Darshan, and Petrov, Slav. Temporal Analysis of Language Through Neural Language Models. *arXiv preprint arXiv:1405.3515*, 2014.

Kingma, Diederik and Ba, Jimmy. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations*, 2015.

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

Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. Distributed Representations of Words and Phrases and their Compositionality. In *Advances in Neural Information Processing Systems 26*, pp. 3111–3119, 2013.

Rauber, Paulo E., Falcão, Alexandre X., and Telea, Alexandru C. Visualizing Time-Dependent Data Using Dynamic t-SNE. In *EuroVis 2016 - Short Papers*, 2016.

Rezende, Danilo Jimenez, Mohamed, Shakir, and Wierstra, Daan. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 23rd international conference on Machine learning*, 2014.