# Factors Influencing the Surprising Instability of Word Embeddings

**Laura Wendlandt, Jonathan K. Kummerfeld** and **Rada Mihalcea**
Computer Science & Engineering
University of Michigan, Ann Arbor
{wenlaura,jkummerf,mihalcea}@umich.edu

## Abstract

Despite the recent popularity of word embedding methods, there is only a small body of work exploring the limitations of these representations. In this paper, we consider one aspect of embedding spaces, namely their stability. We show that even relatively high frequency words (100-200 occurrences) are often unstable. We provide empirical evidence for how various factors contribute to the stability of word embeddings, and we analyze the effects of stability on downstream tasks.

## 1 Introduction

Word embeddings are low-dimensional, dense vector representations that capture semantic properties of words. Recently, they have gained tremendous popularity in Natural Language Processing (NLP) and have been used in tasks as diverse as text similarity (Kenter and De Rijke, 2015), part-of-speech tagging (Tsvetkov et al., 2016), sentiment analysis (Faruqui et al., 2015), and machine translation (Mikolov et al., 2013a). Although word embeddings are widely used across NLP, their stability has not yet been fully evaluated and understood. In this paper, we explore the factors that play a role in the stability of word embeddings, including properties of the data, properties of the algorithm, and properties of the words. We find that word embeddings exhibit substantial instabilities, which can have implications for downstream tasks.

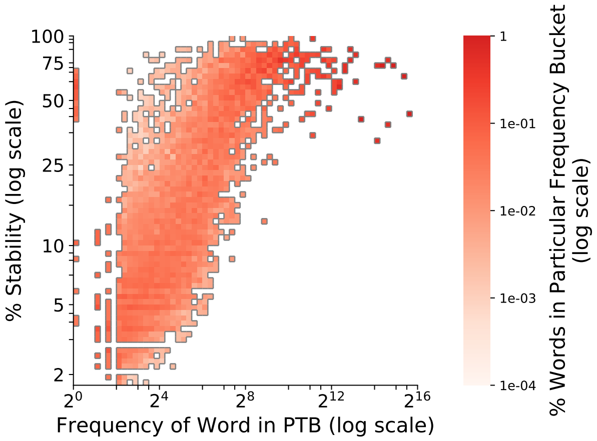


Figure 1: Stability of *word2vec* as a property of frequency in the PTB. Stability is measured across ten randomized embedding spaces trained on the training portion of the PTB (determined using language modeling splits (Mikolov et al., 2010)). Each word is placed in a frequency bucket (x-axis), and each column (frequency bucket) is normalized.

Using the overlap between nearest neighbors in an embedding space as a measure of stability (see section 3 below for more information), we observe that many common embedding spaces have large amounts of instability. For example, Figure 1 shows the instability of the embeddings obtained by training *word2vec* on the Penn Treebank (PTB) (Marcus et al., 1993). As expected, lower frequency words have lower stability and higher frequency words have higher stability. What is surprising however about this graph is the medium-frequency words, which show huge variance in stability. This cannot be explained by frequency, so there must be other factors contributing to their instability.

In the following experiments, we explore which factors affect stability, as well as how this stability affects downstream tasks that word embeddings are commonly used for. To our knowledge, this is the first study comprehensively examining the factors behind instability.

## 2 Related Work

There has been much recent interest in the applications of word embeddings, as well as a small, but growing, amount of work analyzing the properties of word embeddings.

Here, we explore three different embedding methods: PPMI (Bullinaria and Levy, 2007),

*Proceedings of NAACL-HLT 2018*, pages 2092–2102
New Orleans, Louisiana, June 1 - 6, 2018. ©2018 Association for Computational Linguistics

*word2vec* (Mikolov et al., 2013b), and GloVe (Pennington et al., 2014). Various aspects of the embedding spaces produced by these algorithms have been previously studied. Particularly, the effect of parameter choices has a large impact on how all three of these algorithms behave (Levy et al., 2015). Further work shows that the parameters of the embedding algorithm *word2vec* influence the geometry of word vectors and their context vectors (Mimno and Thompson, 2017). These parameters can be optimized; Hellrich and Hahn (2016) posit optimal parameters for negative sampling and the number of epochs to train for. They also demonstrate that in addition to parameter settings, word properties, such as word ambiguity, affect embedding quality.

In addition to exploring word and algorithmic parameters, concurrent work by Antoniak and Mimno (2018) evaluates how document properties affect the stability of word embeddings. We also explore the stability of embeddings, but focus on a broader range of factors, and consider the effect of stability on downstream tasks. In contrast, Antoniak and Mimno focus on using word embeddings to analyze language (e.g., Garg et al., 2018), rather than to perform tasks.

At a higher level of granularity, Tan et al. (2015) analyze word embedding spaces by comparing two spaces. They do this by linearly transforming one space into another space, and they show that words have different usage properties in different domains (in their case, Twitter and Wikipedia).

Finally, embeddings can be analyzed using second-order properties of embeddings (e.g., how a word relates to the words around it). Newman-Griffis and Fosler-Lussier (2017) validate the usefulness of second-order properties, by demonstrating that embeddings based on second-order properties perform as well as the typical first-order embeddings. Here, we use second-order properties of embeddings to quantify stability.

## 3 Defining Stability

We define *stability* as the percent overlap between nearest neighbors in an embedding space.[1] Given a word $W$ and two embedding spaces $A$ and $B$, take the ten nearest neighbors of $W$ in both $A$ and $B$. Let the stability of $W$ be the percent overlap between these two lists of nearest neighbors. 100% stability indicates perfect agreement between the two embedding spaces, while 0% stability indicates complete disagreement. In order to find the ten nearest neighbors of a word $W$ in an embedding space $A$, we measure distance between words using cosine similarity.[2] This definition of stability can be generalized to more than two embedding spaces by considering the average overlap between two sets of embedding spaces. Let $X$ and $Y$ be two sets of embedding spaces. Then, for every pair of embedding spaces $(x, y)$, where $x \in X$ and $y \in Y$, take the ten nearest neighbors of $W$ in both $x$ and $y$ and calculate percent overlap. Let the stability be the average percent overlap over every pair of embedding spaces $(x, y)$.

| Model 1 | Model 2 | Model 3 |
|---|---|---|
| **metropolitan** | *ballet* | **national** |
| **national** | **metropolitan** | *ballet* |
| *egyptian* | bard | **metropolitan** |
| *rhode* | chicago | institute |
| *society* | **national** | glimmerglass |
| debut | *state* | *egyptian* |
| folk | *exhibitions* | intensive |
| reinstallation | *society* | jazz |
| chairwoman | whitney | *state* |
| philadelphia | *rhode* | *exhibitions* |

Table 1: Top ten most similar words for the word *international* in three randomly intialized *word2vec* models trained on the NYT Arts Domain. Words in all three lists are in bold; words in only two of the lists are italicized.

Consider an example using this metric. Table 1 shows the top ten nearest neighbors for the word *international* in three randomly initialized *word2vec* embedding spaces trained on the NYT Arts domain (see Section 4.3 for a description of this corpus). These models share some similar words, such as *metropolitan* and *national*, but there are also many differences. On average, each pair of models has four out of ten words in common, so the stability of *international* across these three models is 40%.

The idea of evaluating ten best options is also found in other tasks, like lexical substitution (e.g., McCarthy and Navigli, 2007) and word associa-

[1] This metric is concurrently explored in work by Antoniak and Mimno (2018).

[2] We found comparable results for other distance metrics, such as $l^1$ norm, $l^2$ norm, and $l^\infty$ norm, but we report results from cosine similarity to be consistent with other word embedding research.

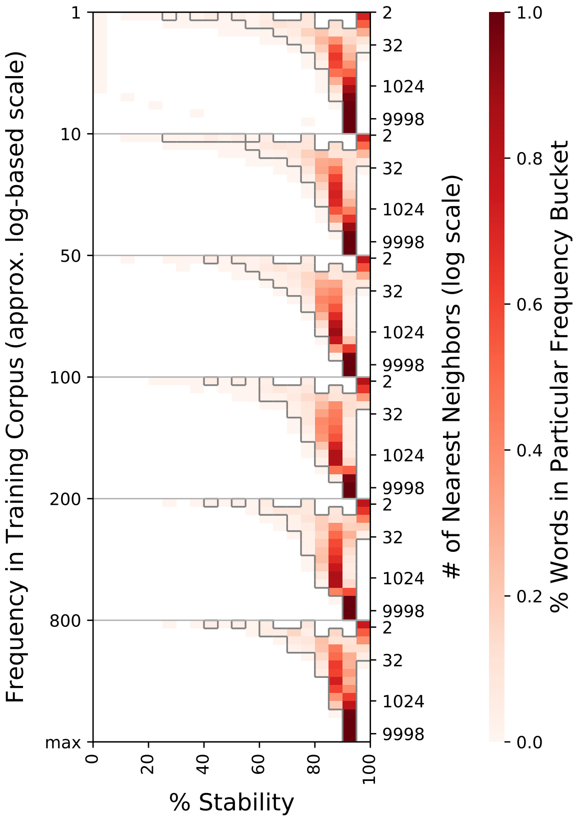


Figure 2: Stability of GloVe on the PTB. Stability is measured across ten randomized embedding spaces trained on the training data of the PTB (determined using language modeling splits (Mikolov et al., 2010)). Each word is placed in a frequency bucket (left y-axis) and stability is determined using a varying number of nearest neighbors for each frequency bucket (right y-axis). Each row is normalized, and boxes with more than 0.01 of the row's mass are outlined.

tion (e.g., Garimella et al., 2017), where the top ten results are considered in the final evaluation metric. To give some intuition for how changing the number of nearest neighbors affects our stability metric, consider Figure 2. This graph shows how the stability of GloVe changes with the frequency of the word and the number of neighbors used to calculate stability; please see the figure caption for a more detailed explanation of how this graph is structured. Within each frequency bucket, the stability is consistent across varying numbers of neighbors. Ten nearest neighbors performs approximately as well as a higher number of nearest neighbors (e.g., 100). We see this pattern for low frequency words as well as for high frequency words. Because the performance does not change substantially by increasing the number of nearest neighbors, it is computationally less intensive to use a small number of nearest neighbors. We choose ten nearest neighbors as our metric throughout the rest of the paper.

## 4 Factors Influencing Stability

As we saw in Figure 1, embeddings are sometimes surprisingly unstable. To understand the factors behind the (in)stability of word embeddings, we build a regression model that aims to predict the stability of a word given: (1) properties related to the word itself; (2) properties of the data used to train the embeddings; and (3) properties of the algorithm used to construct these embeddings. Using this regression model, we draw observations about factors that play a role in the stability of word embeddings.

### 4.1 Methodology

We use ridge regression to model these various factors (Hoerl and Kennard, 1970). Ridge regression regularizes the magnitude of the model weights, producing a more interpretable model than non-regularized linear regression. This regularization mitigates the effects of multicollinearity (when two features are highly correlated). Specifically, given $N$ ground-truth data points with $M$ extracted features per data point, let $\mathbf{x}_n \in \mathbb{R}^{1\times M}$ be the features for sample $n$ and let $\mathbf{y} \in \mathbb{R}^{1\times N}$ be the set of labels. Then, ridge regression learns a set of weights $\mathbf{w} \in \mathbb{R}^{1\times M}$ by minimizing the least squares function with $l^2$ regularization, where $\lambda$ is a regularization constant:

$$L(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{y}_n - \mathbf{w}^\top \mathbf{x}_n)^2 + \frac{\lambda}{2}||\mathbf{w}||^2$$

We set $\lambda = 1$. In addition to ridge regression, we tried non-regularized linear regression. We obtained comparable results, but many of the weights were very large or very small, making them hard to interpret.

The goodness of fit of a regression model is measured using the coefficient of determination $R^2$. This measures how much variance in the dependent variable $\mathbf{y}$ is captured by the independent variables $\mathbf{x}$. A model that always predicts the expected value of $\mathbf{y}$, regardless of the input features, will receive an $R^2$ score of 0. The highest possible $R^2$ score is 1, and the $R^2$ score can be negative.

Given this model, we create training instances by observing the stability of a large number of words across various combinations of two embedding spaces. Specifically, given a word $W$ and two embedding spaces $A$ and $B$, we encode properties of the *word* $W$, as well as properties of the *datasets* and the *algorithms* used to train the embedding spaces $A$ and $B$. The target value associated with this features is the stability of the word $W$ across embedding spaces $A$ and $B$. We repeat this process for more than 2,500 words, several datasets, and three embedding algorithms.

Specifically, we consider all the words present in all seven of the data domains we are using (see Section 4.3), 2,521 words in total. Using the feature categories described below, we generate a feature vector for each unique word, dataset, algorithm, and dimension size, resulting in a total of 27,794,025 training instances. To get good average estimates for each embedding algorithm, we train each embedding space five times, randomized differently each time (this does not apply to PPMI, which has no random component). We then train a ridge regression model on these instances. The model is trained to predict the stability of word $W$ across embedding spaces $A$ and $B$ (where $A$ and $B$ are not necessarily trained using the same algorithm, parameters, or training data). Because we are using this model to learn associations between certain features and stability, no test data is necessary. The emphasis is on the model itself, not on the model's performance on a specific task.

We describe next each of the three main categories of factors examined in the model. An example of these features is given in Table 2.

## 4.2 Word Properties

We encode several features that capture attributes of the word $W$. First, we use the primary and secondary part-of-speech (POS) of the word. Both of these are represented as bags-of-words of all possible POS, and are determined by looking at the primary (most frequent) and secondary (second most frequent) POS of the word in the Brown corpus[3] (Francis and Kucera, 1979). If the word is not present in the Brown corpus, then all of these POS features are set to zero.

| **Word Properties** | |
|---|---|
| Primary part-of-speech | `Adjective` |
| Secondary part-of-speech | `Noun` |
| # Parts-of-speech | `2` |
| # WordNet senses | `3` |
| Syllables | `5` |
| **Data Properties** | |
| Raw frequency in corpus $A$ | `106` |
| Raw frequency in corpus $B$ | `669` |
| Diff. in raw frequency | `563` |
| Vocab. size of corpus $A$ | `10,508` |
| Vocab. size of corpus $B$ | `43,888` |
| Diff. in vocab. size | `33,380` |
| Overlap in corpora vocab. | `17%` |
| Domains present | `Arts, Europarl` |
| Do the domains match? | `False` |
| Training position in $A$ | `1.02%` |
| Training position in $B$ | `0.15%` |
| Diff. in training position | `0.86%` |
| **Algorithm Properties** | |
| Algorithms present | *word2vec*, `PPMI` |
| Do the algorithms match? | `False` |
| Embedding dimension of $A$ | `100` |
| Embedding dimension of $B$ | `100` |
| Diff. in dimension | `0` |
| Do the dimensions match? | `True` |

Table 2: Consider the word *international* in two embedding spaces. Suppose embedding space $A$ is trained using *word2vec* (embedding dimension 100) on the NYT Arts domain, and embedding space $B$ is trained using PPMI (embedding dimension 100) on Europarl. This table summarizes the resulting features for this word across the two embedding spaces.

To get a coarse-grained representation of the polysemy of the word, we consider the number of different POS present. For a finer-grained representation, we use the number of different WordNet senses associated with the word (Miller, 1995; Fellbaum, 1998).

We also consider the number of syllables in a word, determined using the CMU Pronuncing Dictionary (Weide, 1998). If the word is not present in the dictionary, then this is set to zero.

## 4.3 Data Properties

Data features capture properties of the training data (and the word in relation to the training data). For this model, we gather data from two sources: New York Times (NYT) (Sandhaus, 2008) and Europarl (Koehn, 2005). Overall, we consider seven domains of data: (1) NYT - U.S., (2) NYT - New York and Region, (3) NYT - Business, (4) NYT - Arts, (5) NYT - Sports, (6) All of the data from

[3] Here, we use the universal tagset, which consists of twelve possible POS: adjective, adposition, adverb, conjunction, determiner / article, noun, numeral, particle, pronoun, verb, punctuation mark, and other (Petrov et al., 2012).

| Dataset | Sentences | Vocab.<br>Size | Num. Tokens /<br>Vocab. Size |
|---|---|---|---|
| NYT US | 13,923 | 5,787 | 64.37 |
| NYT NY | 36,792 | 11,182 | 80.41 |
| NYT Business | 21,048 | 7,212 | 75.96 |
| NYT Arts | 28,161 | 10,508 | 65.29 |
| NYT Sports | 21,610 | 5,967 | 77.85 |
| All NYT | 121,534 | 24,144 | 117.98 |
| Europarl | 2,297,621 | 43,888 | 1,394.28 |

Table 3: Dataset statistics.

domains 1-5 (denoted "All NYT"), and (7) All of English Europarl. Table 3 shows statistics about these datasets. The first five domains are chosen because they are the top five most common categories of news articles present in the NYT corpus. They are smaller than "All NYT" and Europarl, and they have a narrow topical focus. The "All NYT" domain is more diverse topically and larger than the first five domains. Finally, the Europarl domain is the largest domain, and it is focused on a single topic (European Parliamentary politics). These varying datasets allow us to consider how data-dependent properties affect stability.

We use several features related to domain. First, we consider the raw frequency of word $W$ in both the domain of data used for embedding space $A$ and the domain of data for space $B$. To make our regression model symmetric, we effectively encode three features: the higher raw frequency (between the two), the lower raw frequency, and the absolute difference in raw frequency.

We also consider the vocabulary size of each corpus (again, symmetrically) and the percent overlap between corpora vocabulary, as well as the domain of each of the two corpora, represented as a bag-of-words of domains. Finally, we consider whether the two corpora are from the same domain.

Our final data-level features explore the role of curriculum learning in stability. It has been posited that the order of the training data affects the performance of certain algorithms, and previous work has shown that for some neural network-based tasks, a good training data order (curriculum learning strategy) can improve performance (Bengio et al., 2009). Curriculum learning has been previously explored for *word2vec*, where it has been found that optimizing training data order can lead to small improvements on common NLP tasks (Tsvetkov et al., 2016). Of the embedding algorithms we consider, curriculum learning only affects *word2vec*. Because GloVe and PPMI use the data to learn a complete matrix before building embeddings, the order of the training data will not affect their performance. To measure the effects of training data order, we include as features the first appearance of word $W$ in the dataset for embedding space $A$ and the first appearance of $W$ in the dataset for embedding space $B$ (represented as percentages of the total number of training sentences)[4]. We further include the absolute difference between these percentages.

### 4.4 Algorithm Properties

In addition to word and data properties, we encode features about the embedding algorithms. These include the different algorithms being used, as well as the different parameter settings of these algorithms. Here, we consider three embedding algorithms, *word2vec*, GloVe, and PPMI. The choice of algorithm is represented in our feature vector as a bag-of-words.

PPMI creates embeddings by first building a positive pointwise mutual information word-context matrix, and then reducing the dimensionality of this matrix using SVD (Bullinaria and Levy, 2007). A more recent word embedding algorithm, *word2vec* (skip-gram model) (Mikolov et al., 2013b) uses a shallow neural network to learn word embeddings by predicting context words. Another recent method for creating word embeddings, GloVe, is based on factoring a matrix of ratios of co-occurrence probabilities (Pennington et al., 2014).

For each algorithm, we choose common parameter settings. For *word2vec*, two of the parameters that need to be chosen are window size and minimum count. Window size refers to the maximum distance between the current word and the predicted word (e.g., how many neighboring words to consider for each target word). Any word appearing less than the minimum count number of times in the corpus is discarded and not considered in the *word2vec* algorithm. For both of these features, we choose standard parameter settings, namely, a window size of 5 and a minimum count of 5. For GloVe, we also choose standard parameters. We

[4] All *word2vec* experiments reported here are run in a multi-core setting, which means that these percentages are approximate. However, comparable results were achieved using a single-core version of *word2vec*.

| Feature | Weight |
|---|---|
| Lower training data position of word $W$ | -1.52 |
| Higher training data position of $W$ | -1.49 |
| Primary POS = Numeral | 1.12 |
| Primary POS = Other | -1.08 |
| Primary POS = Punctuation mark | -1.02 |
| Overlap between corpora vocab. | 1.01 |
| Primary POS = Adjective | -0.92 |
| Primary POS = Adposition | -0.92 |
| Do the two domains match? | 0.91 |
| Primary POS = Verb | -0.88 |
| Primary POS = Conjunction | -0.84 |
| Primary POS = Noun | -0.81 |
| Primary POS = Adverb | -0.79 |
| Do the two algorithms match? | 0.78 |
| Secondary POS = Pronoun | 0.62 |
| Primary POS = Determiner | -0.48 |
| Primary POS = Particle | -0.44 |
| Secondary POS = Particle | 0.36 |
| Secondary POS = Other | 0.28 |
| Primary POS = Pronoun | -0.26 |
| Secondary POS = Verb | -0.25 |
| Number of *word2vec* embeddings | -0.21 |
| Secondary POS = Adverb | 0.15 |
| Secondary POS = Determiner | 0.14 |
| Number of NYT Arts Domain | -0.14 |
| Number of NYT All Domain | 0.14 |
| Number of GloVe embeddings | 0.13 |
| Number of syllables | -0.11 |

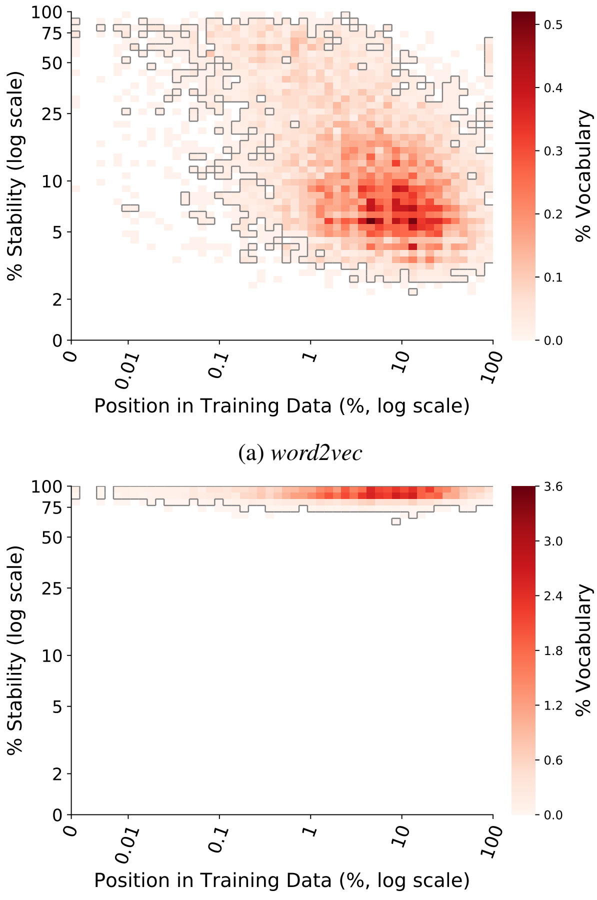
Table 4: Regression weights with a magnitude greater than 0.1, sorted by magnitude.

use 50 iterations of the algorithm for embedding dimensions less than 300, and 100 iterations for higher dimensions.

We also add a feature reflecting the embedding dimension, namely one of five embedding dimensions: 50, 100, 200, 400, or 800.

## 5 Lessons Learned: What Contributes to the Stability of an Embedding

Overall, the regression model achieves a coefficient of determination ($R^2$) score of 0.301 on the training data, which indicates that the regression has learned a linear model that reasonably fits the training data given. Using the regression model, we can analyze the weights corresponding to each of the features being considered, shown in Table 4. These weights are difficult to interpret, because features have different distributions and ranges. However, we make several general observations about the stability of word embeddings.



(a) *word2vec*

(b) GloVe

Figure 3: Stability of both *word2vec* and GloVe as properties of the starting word position in the training data of the PTB. Stability is measured across ten randomized embedding spaces trained on the training data of the PTB (determined using language modeling splits (Mikolov et al., 2010)). Boxes with more than 0.02% of the total vocabulary mass are outlined.

**Observation 1. Curriculum learning is important.** This is evident because the top two features (by magnitude) of the regression model capture where the word first appears in the training data. Figure 3 shows trends between training data position and stability in the PTB. This figure contrasts *word2vec* with GloVe (which is order invariant).

To further understand the effect of curriculum learning on the model, we train a regression model with all of the features except the curriculum learning features. This model achieves an $R^2$ score of 0.291 (compared to the full model's score of 0.301). This indicates that curriculum learning is a factor in stability.

**Observation 2. POS is one of the biggest factors in stability.** Table 4 shows that many of the top weights belong to POS-related features (both primary and secondary POS). Table 5 compares aver-

| Primary POS | Avg. Stability |
|---|---|
| Numeral | 47% |
| Verb | 31% |
| Determiner | 31% |
| Adjective | 31% |
| Noun | 30% |
| Adverb | 29% |
| Pronoun | 29% |
| Conjunction | 28% |
| Particle | 26% |
| Adposition | 25% |
| Punctuation mark | 22% |

Table 5: Percent stability broken down by part-of-speech, ordered by decreasing stability.

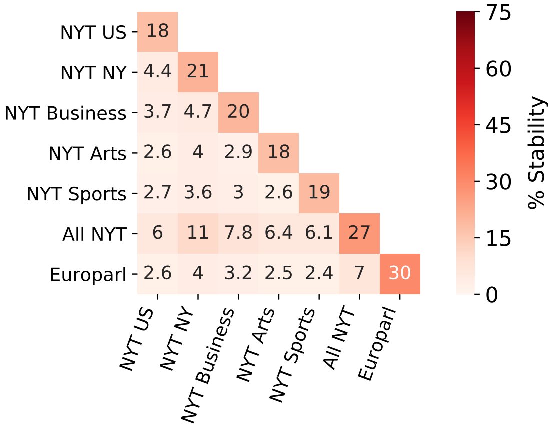


Figure 4: Percent stability broken down by domain.

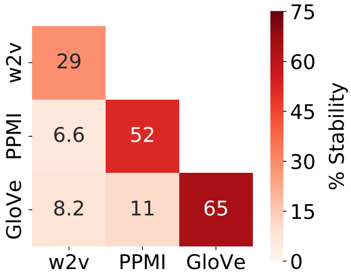


Figure 5: Percent stability broken down between algorithm (in-domain data only).

age stabilities for each primary POS. Here we see that the most stable POS are numerals, verbs, and determiners, while the least stable POS are punctuation marks, adpositions, and particles.

**Observation 3. Stability within domains is greater than stability across domains.** Table 4 shows that many of the top factors are domain-related. Figure 4 shows the results of the regression model broken down by domain. This figure shows the highest stabilities appearing on the diagonal of the matrix, where the two embedding spaces both belong to the same domain. The stabilities are substantially lower off the diagonal.

Figure 4 also shows that "All NYT" generalizes across the other NYT domains better than Europarl, but not as well as in-domain data ("All NYT" encompasses data from US, NY, Business, Arts, and Sports). This is true even though Europarl is much larger than "All NYT".

**Observation 4. Overall, GloVe is the most stable embedding algorithm.** This is particularly apparent when only in-domain data is considered, as in Figure 5. PPMI achieves similar stability, while *word2vec* lags considerably behind.

To further compare *word2vec* and GloVe, we look at how the stability of *word2vec* changes with the frequency of the word and the number of neighbors used to calculate stability. This is shown in Figure 6 and is directly comparable to Figure 2. Surprisingly, the stability of *word2vec* varies substantially with the frequency of the word. For lower-frequency words, as the number of nearest neighbors increases, the stability increases approximately exponentially. For high-frequency words, the lowest and highest number of nearest neighbors show the greatest stability. This is different than GloVe, where stability remains reasonably constant across word frequencies, as shown in Figure 2. The behavior we see here agrees with the conclusion of (Mimno and Thompson, 2017), who find that GloVe exhibits more well-behaved geometry than *word2vec*.

**Observation 5. Frequency is not a major factor in stability.** To better understand the role that frequency plays in stability, we run separate ablation experiments comparing regression models with frequency features to regression models without frequency features. Our current model (using raw frequency) achieves an $R^2$ score of 0.301. Comparably, a model using the same features, but with normalized instead of raw frequency, achieves a score of 0.303. Removing frequency from either regression model gives a score of 0.301. This indicates that frequency is not a major factor in stability, though normalized frequency is a larger factor than raw frequency.

Finally, we look at regression models using only frequency features. A model using only raw frequency features has an $R^2$ score of 0.008, while

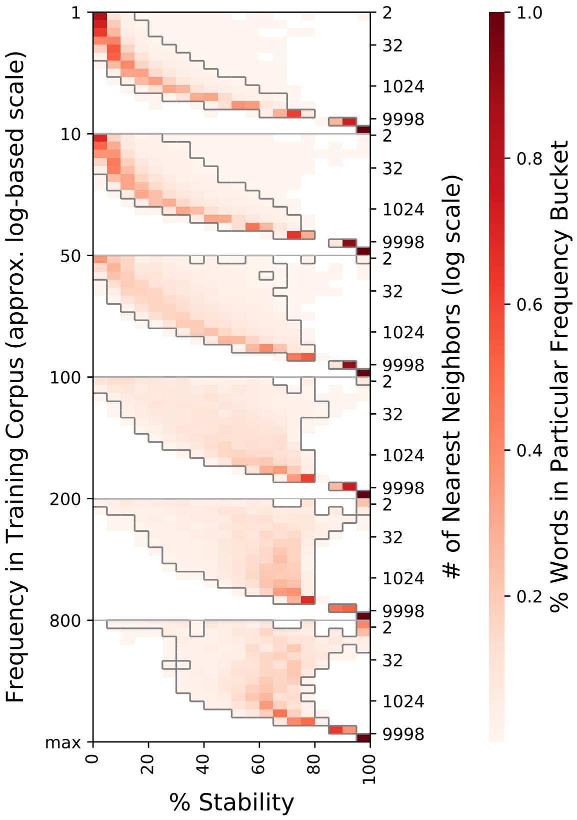


Figure 6: Stability of *word2vec* on the PTB. Stability is measured across ten randomized embedding spaces trained on the training data of the PTB (determined using language modeling splits (Mikolov et al., 2010)). Each word is placed in a frequency bucket (left y-axis) and stability is determined using a varying number of nearest neighbors for each frequency bucket (right y-axis). Each row is normalized, and boxes with more than 0.01 of the row's mass are outlined.

a model with only normalized frequency features has an $R^2$ score of 0.0059. This indicates that while frequency is not a major factor in stability, it is also not negligible. As we pointed out in the introduction, frequency does correlate with stability (Figure 1). However, in the presence of all of these other features, frequency becomes a minor factor.

# 6 Impact of Stability on Downstream Tasks

Word embeddings are used extensively as the first stage of neural networks throughout NLP. Typically, embeddings are initalized based on a vector trained with *word2vec* or GloVe and then further modified as part of training for the target task. We study two downstream tasks to see whether stability impacts performance.

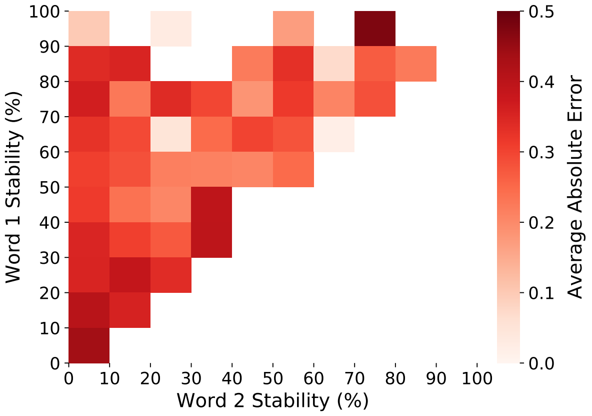


Figure 7: Absolute error for word similarity.

Since we are interested in seeing the impact of word vector stability, we choose tasks that have an intuitive evaluation at the word level: word similarity and POS tagging.

## 6.1 Word Similarity

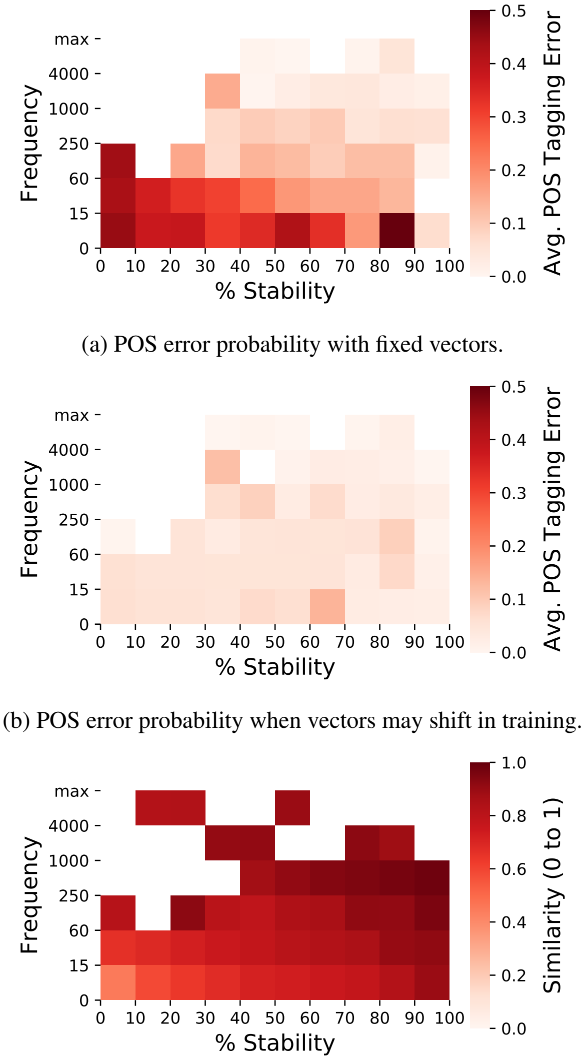
To model word similarity, we use 300-dimensional *word2vec* embedding spaces trained on the PTB. For each pair of words, we take the cosine similarity between those words averaged over ten randomly initialized embedding spaces.

We consider three datasets for evaluating word similarity: WS353 (353 pairs) (Finkelstein et al., 2001), MTurk287 (287 pairs) (Radinsky et al., 2011), and MTurk771 (771 pairs) (Halawi et al., 2012). For each dataset, we normalize the similarity to be in the range $[0, 1]$, and we take the absolute difference between our predicted value and the ground-truth value. Figure 7 shows the results broken down by stability of the two words (we always consider Word 1 to be the more stable word in the pair). Word similarity pairs where one of the words is not present in the PTB are omitted.

We find that these word similarity datasets do not contain a balanced distribution of words with respect to stability; there are substantially more unstable words than there are stable words. However, we still see a slight trend: As the combined stability of the two words increases, the average absolute error decreases, as reflected by the lighter color of the cells in Figure 7 while moving away from the (0,0) data point.

## 6.2 Part-of-Speech Tagging

Part-of-speech (POS) tagging is a substantially more complicated task than word similarity. We use a bidirectional LSTM implemented using DyNet (Neubig et al., 2017). We train nine sets of



(a) POS error probability with fixed vectors.

(b) POS error probability when vectors may shift in training.

(c) Cosine similarity between original word vectors and shifted word vectors.

Figure 8: Results for POS tagging. (a) and (b) show average POS tagging error divided by the number of tokens (darker is more errors) while either keeping word vectors fixed or not during training. (c) shows word vector shift, measured as cosine similarity between initial and final vectors. In all graphs, words are bucketed by frequency and stability.

128-dimensional word embeddings with *word2vec* using different random seeds. The LSTM has a single layer and 50-dimensional hidden vectors. Outputs are passed through a *tanh* layer before classification. To train, we use SGD with a learning rate of 0.1, an input noise rate of 0.1, and recurrent dropout of 0.4.

This simple model is not state-of-the-art, scoring 95.5% on the development set, but the word vectors are a central part of the model, providing a clear signal of their impact. For each word, we group tokens based on stability and frequency. Figure 8 shows the results.[5] Fixing the word vectors provides a clearer pattern in the results, but also leads to much worse performance: 85.0% on the development set. Based on these results, it seems that training appears to compensate for stability. This hypothesis is supported by Figure 8c, which shows the similarity between the original word vectors and the shifted word vectors produced by the training. In general, lower stability words are shifted more during training.

Understanding how the LSTM is changing the input embeddings is useful information for tasks with limited data, and it could allow us to improve embeddings and LSTM training for these low-resource tasks.

## 7 Conclusion and Recommendations

Word embeddings are surprisingly variable, even for relatively high frequency words. Using a regression model, we show that domain and part-of-speech are key factors of instability. Downstream experiments show that stability impacts tasks using embedding-based features, though allowing embeddings to shift during training can reduce this effect. In order to use the most stable embedding spaces for future tasks, we recommend either using GloVe or learning a good curriculum for *word2vec* training data. We also recommend using in-domain embeddings whenever possible.

The code used in the experiments described in this paper is publicly available from http://lit.eecs.umich.edu/downloads.html.

## Acknowledgments

We would like to thank Ben King and David Jurgens for helpful discussions about this paper, as well as our anonymous reviewers for useful feedback. This material is based in part upon work supported by the National Science Foundation (NSF #1344257) and the Michigan Institute for Data Science (MIDAS). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or MIDAS.

[5] The unusual dark spot that occurs at medium-high stability and low frequency is caused primarily by words that have a substantially different POS distribution in the test data than in the training data.

## References

Maria Antoniak and David Mimno. 2018. Evaluating the stability of embedding-based word similarities. *Transactions of the Association for Computational Linguistics (TACL)* 6:107–119. https://transacl.org/ojs/index.php/tacl/article/view/1202.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the International Conference on Machine Learning (ICML)*. pages 41–48. https://dl.acm.org/citation.cfm?id=1553380.

John A Bullinaria and Joseph P Levy. 2007. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior Research Methods* 39(3):510–526. https://link.springer.com/article/10.3758/BF03193020.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL: HLT)*. pages 1606–1615. http://www.aclweb.org/anthology/N15-1184.

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library. https://onlinelibrary.wiley.com/doi/full/10.1002/9781405198431.wbeal1285.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2001. Placing search in context: The concept revisited. In *Proceedings of the International Conference on World Wide Web (WWW)*. pages 406–414. https://dl.acm.org/citation.cfm?id=372094.

W Nelson Francis and Henry Kucera. 1979. Brown corpus manual. *Brown University* 2.

Nikhil Garg, Londa Schiebinger, Dan Jurafsky, and James Zou. 2018. Word embeddings quantify 100 years of gender and ethnic stereotypes. *Proceedings of the National Academy of Sciences (PNAS)* https://doi.org/10.1073/pnas.1720347115.

Aparna Garimella, Carmen Banea, and Rada Mihalcea. 2017. Demographic-aware word associations. In *Proceedings of the International Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pages 2285–2295. http://www.aclweb.org/anthology/D17-1242.

Guy Halawi, Gideon Dror, Evgeniy Gabrilovich, and Yehuda Koren. 2012. Large-scale learning of word relatedness with constraints. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (SIGKDD)*. pages 1406–1414. https://dl.acm.org/citation.cfm?id=2339751.

Johannes Hellrich and Udo Hahn. 2016. Bad company-Neighborhoods in neural embedding spaces considered harmful. In *Proceedings of the International Conference on Computational Linguistics (COLING)*. pages 2785–2796. http://www.aclweb.org/anthology/C16-1262.

Arthur E Hoerl and Robert W Kennard. 1970. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 12(1):55–67. https://amstat.tandfonline.com/doi/abs/10.1080/00401706.1970.10488634.

Tom Kenter and Maarten De Rijke. 2015. Short text similarity with word embeddings. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management (CIKM)*. pages 1411–1420. https://dl.acm.org/citation.cfm?id=2806475.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Machine Translation Summit X*. volume 5, pages 79–86.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics (TACL)* 3:211–225. https://www.transacl.org/ojs/index.php/tacl/article/view/570.

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn treebank. *Computational Linguistics* 19(2):313–330. https://dl.acm.org/citation.cfm?id=972475.

Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the International Workshop on Semantic Evaluations (SemEval)*. pages 48–53. https://dl.acm.org/citation.cfm?id=1621483.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*. volume 2, page 3. https://www.isca-speech.org/archive/interspeech_2010/i10_1045.html.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168* https://arxiv.org/abs/1309.4168.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*. pages 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-andphrases.

George A Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM* 38(11):39–41. https://dl.acm.org/citation.cfm?id=219748.

David Mimno and Laure Thompson. 2017. The strange geometry of skip-gram with negative sampling. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pages 2863–2868. http://www.aclweb.org/anthology/D17-1308.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. DyNet: The dynamic neural network toolkit. *arXiv preprint arXiv:1701.03980* https://arxiv.org/abs/1701.03980.

Denis Newman-Griffis and Eric Fosler-Lussier. 2017. Second-order word embeddings from nearest neighbor topological features. *arXiv preprint arXiv:1705.08488* https://arxiv.org/abs/1705.08488.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. *Proceedings of The International Conference on Language Resources and Evaluation (LREC)* pages 2089–2096. http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf.

Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. 2011. A word at a time: Computing word relatedness using temporal semantic analysis. In *Proceedings of the International Conference on World Wide Web (WWW)*. pages 337–346. https://dl.acm.org/citation.cfm?id=1963455.

Evan Sandhaus. 2008. The New York Times annotated corpus. *Linguistic Data Consortium, Philadelphia* 6(12):e26752. https://catalog.ldc.upenn.edu/ldc2008t19.

Luchen Tan, Haotian Zhang, Charles LA Clarke, and Mark D Smucker. 2015. Lexical comparison between Wikipedia and Twitter corpora by using word embeddings. In *Assocation for Computational Linguistics (ACL)*. pages 657–661. http://www.aclweb.org/anthology/P15-2108.

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Brian MacWhinney, and Chris Dyer. 2016. Learning the curriculum with Bayesian optimization for task-specific word representation learning. *Association for Computational Linguistics (ACL)* pages 130–139. http://anthology.aclweb.org/P/P16/P16-1013.pdf.

Robert L Weide. 1998. The CMU pronouncing dictionary http://www.speech.cs.cmu.edu/cgibin/cmudict.