CSC2611 Lab: Word Embedding and Semantic Change*

Linfeng Du

1 Synchronic Word Embeddings

In this section, we build a vocabulary of 5031 words by combining the 5000 most common English words in the Brown Corpus and the words in Table 1 of RG65. Based on which, the word-context model is built by collecting bigram counts throughout the corpus. We then build the PPMI model based on the word-context matrix, and the LSA model by applying truncated SVD factorization to the PPMI matrix (dimension after truncation is shown as subscript). As for the word2vec model, we use pretrained embeddings and remove out-of-vocabulary words to ensure fair comparison in word analogy test. All vector models are processed into "gensim.models.KeyedVectors" which provides out-of-the-box API for evaluating model performance on word similarity and word analogy tests.

1.1 Word Similarity Test

We evaluate model performance on word similarity test by calculating Pearson correlation between cosine similarities of word vectors and human-judged similarities. Using Table 1 of RG65 as the test set, we report the results of different word vector models in Table 1.

It is observed that the correlation coefficient between word2vec-based similarities and human-judged similarities is significantly greater than all other methods, indicating that word2vec is more capable of capturing word similarity. However, it is worth noticing that word2vec is trained on a much larger corpus and with a much larger vocabulary, the richer semantic information in which could also benefit other methods[1]. It is also worth noticing that word-context model demonstrates better correlation than PPMI and LSA, this might be due to low frequency of the tested words in the Brown Corpus.

Model	Pearson R (p-value)
word-context	0.34 (0.01)
PPMI	0.26 (0.04)
LSA_{10}	0.20 (0.10)
LSA_{100}	<u>0.31</u> (0.01)
LSA_{300}	0.30 (0.01)
word2vec	**0.77** (0.00)

Table 1: Model performance on word similarity test.

1.2 Word Analogy Test

We evaluate the performance of word2vec and LSA on semantic and syntactic analogy tests. Results are reported in Table 2.

Model	Semantic Acc (%)	Syntactic Acc (%)
LSA_{300}	<u>3.47</u>	<u>9.50</u>
word2vec	**90.28**	**75.01**

Table 2: Model performance on word analogy tests.

*Code repository: https://github.com/CapFreddy/CSC2611-Computational-Models-of-Semantic-Change

[1] As Hamilton et al. shows, SVD outperforms word2vec on this task in a more controlled setting.

It is observed that word2vec outperforms LSA on both tests by a large margin. Besides the factor of training settings described in the previous section, the low performance of LSA could also be due to small context window (since we only consider bigrams for constructing word-context matrix), its training objective, and its linearity. As Mikolov et al. points out, the capacity of word2vec in doing analogy via vector arithmetic could come from the context prediction objective and their log-linear modeling, while SVD aims at finding the optimum low-rank approximation of the word-context matrix via a linear setting. This discrepancy in objective and modeling is likely to be the main reason for SVD to have a low performance in this task.

1.3 Improving Vector-Based Models in Capturing Word Similarities

One possible way of improving vector-based models in capturing word similarities is to leverage known examples of word similarity and/or word analogy as prior knowledge, and incorporate them into the training objective as weighted regularization terms.

Formally, we denote known similar words as $(w_u, w_v) \sim P_s$ and known word analogy tuple as $(w_u, w_v, w_x, w_y) \sim P_a$, then we can rewrite the loss function L as:

$$L = L' - \alpha \sum_{(w_u, w_v) \sim P_s} \text{cos-sim}(v_{w_u}, v_{w_v}) - \beta \sum_{(w_u, w_v, w_x, w_y) \sim P_a} \text{cos-sim}(v_{w_v} - v_{w_u} + v_{w_x}, v_{w_y}).$$

Adding such constraints on the learned embeddings space can help generalize to unseen examples which might not be captured by the original training objective. For example, constraining on (Athens, Greece, Baghdad, Iraq) and (Athens, Greece, Beijing, China) could help reasoning with (Baghdad, Iraq, Beijing, China).

2 Diachronic Word Embedding

In this section, we study the semantic shift of words in a diachronic setting, where word embeddings of each time snapshot are pretrained on the corresponding temporal corpus. As pointed out by Hamilton et al., the stochastic nature of word2vec may result in arbitrary orthogonal transformations of the embedding space, which hinders comparison of the same word across time. Therefore, we follow their work to align the embedding spaces using orthogonal Procrustes, which can be solved based on the result of SVD. Following Kulkarni et al., we align all embedding spaces to that in the last snapshot.

2.1 Quantifying Semantic Change

We measure the semantic displacement of a word between two snapshots via cosine distance and propose three methods to quantify the semantic change of a word over time: **Displacement between the first and last snapshot** (end2start), **Maximum displacement to the first snapshot over time** (max2start), and **Mean displacement to the first snapshot over time** (mean2start). We show the 20 most and least changed words detected by each method in Table 3.

We examine the agreement of the three methods by calculating Pearson correlations. Results are reported in Table 4.

2.2 Method Evaluation

We build a dataset for evaluating our methods by combining the 20 words from the reference dataset used by Kulkarni et al. and the words detected by their three methods. Specifically, Kulkarni et al. asked three human evaluators to give binary labels to the top 20 words detected by each of their methods. We take the mean of which to form our label. As for the words from the reference dataset, we simply assign 1 as their label since they are used in multiple previous works. 15 out of the 60 words appeared in our vocabulary. We evaluate our three methods by calculating Pearson correlation between method-based semantic change and the ground truth value. Results are reported in Table 5

Method	Most Changed	Least Changed
end2start	objectives programs radio patterns film sector assessment approach perspective media goals impact framework signal berkeley wilson economy pattern challenge jobs	april february november legislature september miles majority duties evening june christ officers morning december officer january church months afternoon court
max2start	objectives programs sector radio patterns approach goals wilson film perspective impact assessment input models evaluation media technology jobs berkeley princeton	april miles november february september months january july december brother legislature court vessels trees june university payment duties officers god
mean2start	objectives programs sector radio goals patterns wilson evaluation input jobs jones technology therapy wiley berkeley film princeton van perspective procedures	april september miles november january february months october july december legislature june duties years vessels christ majority court payment temperature

Table 3: The 20 most and least changed words detected by each method.

	end2start	max2start	mean2start
end2start	1.00	0.94	0.91
max2start	0.94	1.00	0.96
mean2start	0.91	0.96	1.00

Table 4: Correlation of the three methods.

2.3 Change Point Detection

Among the three methods, mean2start yields the highest correlation coefficient. We show the time course (displacement between each snapshot and the first one) of the top three changed words detected by mean2start, i.e. "objectives", "programs", and "sector", in Figure 1.

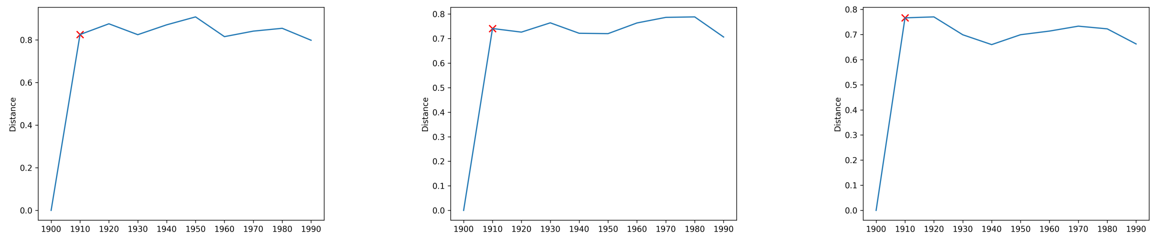
We calculate the change point as the time with the maximum absolute difference in the mean between past and future displacements. Formally:

$$\text{Change\_Point} = \underset{t\in[2,T-1]}{\arg\max} \left| \frac{1}{t-1}\sum_{k=1}^{t-1} d_k - \frac{1}{T-t+1}\sum_{k=t}^{T} d_k \right|$$

where d_k is the displacement from snapshot k to snapshot 1 of the current word. Under this criterion, the estimated change point (ECP) of all three words is 1910.

Method	Pearson R (p-value)
end2start	<u>0.32</u> (0.24)
max2start	0.29 (0.29)
mean2start	**0.45** (0.09)

Table 5: Method performance on detecting semantic change.



(a) Time course of "objectives" (b) Time course of "programs" (c) Time course of "sector"

Figure 1: Time courses of the top 3 changed words. ECP is marked with a red cross.