Understanding Text Embedding Spaces

Angikar Ghosal, Tingnan Hu, Xingyu Zhu
Supervised by Prof. Rong Ge Ph.D. and Muthu Chidambaram

Duke Computer Science

Introduction

Text embeddings are learned vector representation of text in which similar words/sentences have a similar encoding. It is the proxy for a wide collection of machine learning models to understand text information. Text embeddings can be trained using various models, corpora, initializations, etc. In this project we ask:

Do different embeddings capture the same information?
What are the similarities and differences between embeddings learnt with different models, datasets, and initializations?

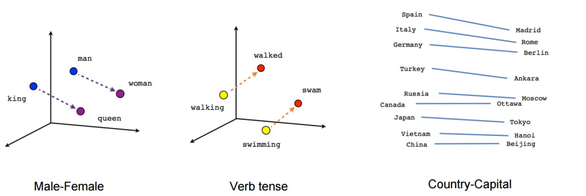

Figure 1. Text embeddings encode various linguistic information.[1]

Objectives

Get Different Embeddings	Train/Inference different word/sentence embedding models using different corpus and from different initializations.
Embedding Alignment	Find a reasonable alignment mechanism that can deal with embeddings of distinctive dimensions, vocabulary size, etc.
Difference Correlation	Determine whether there exists correlation between embedding similarities and their downstream performances.
Embedding Ensemble	Ensemble word/sentence embeddings or outputs of models that take these embeddings as inputs for better performance.

Preliminaries and Methods

Alignment and Matching of Text Embeddings

Embeddings that capture the same information may appear to be completely different; to correctly find similar embeddings, we can align embeddings before comparing them, as text embeddings trained by most used algorithms are **rotationally invariant.**

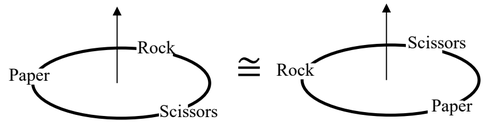

Figure 2. Equivalent but "Different" Embeddings

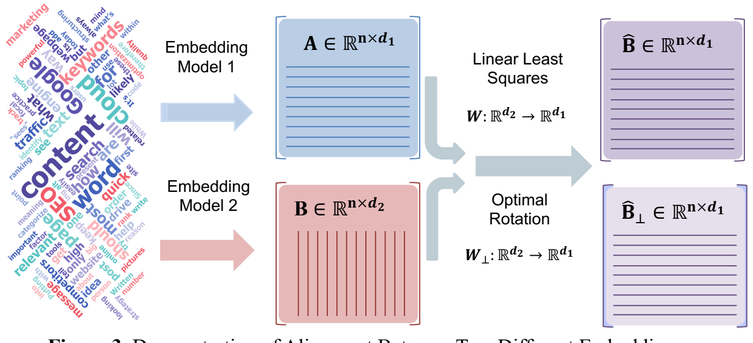

Figure 3. Demonstration of Alignment Between Two Different Embeddings

Optimal Transport Matching

To evaluate the similarities of embeddings A and $\hat{B}$ after alignment, we used the matching induced by the earth mover optimal transport allocation.

With cosine similarity as the distance metric, we consider the matching induced by the doubly stochastic matrix $G \in \mathbb{R}^{d_1 \times d_1}$ that solves

$$\sum_{i,j=1}^{d_1} G_{i,j} \cos\left(A_i, \hat{B}_j\right)$$

When no weighting is applied, this linear program returns a permutation matrix from which we can directly infer the matching.

Word Embedding Ensemble

Previous works have shown that in vision classification tasks, due to the "multi-view" property[2] of the input data one can improve model accuracy by averaging the logit outputs of different deep learning models.

Likewise, we propose three ways of ensembling-like methods.

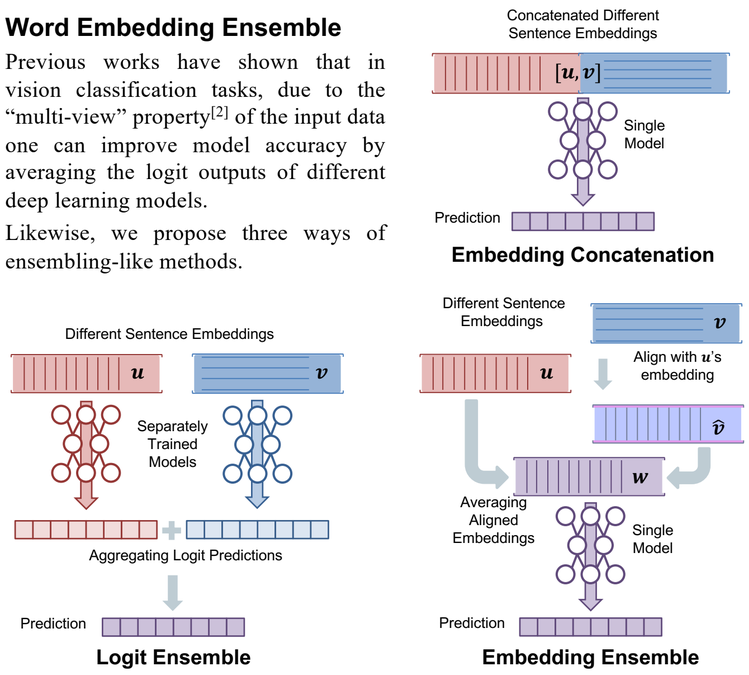

Figure 4. Illustration of Different Ensembling Methods
In this figure u, v corresponds to different embeddings of the same sentence

Results

Embedding Alignment and Optimal Transport Matching

Table 1 presents the comparisons between a collection of word embeddings pairs. For every embedding pair A and B, we compute the aligned embedding $\hat{B}$ via optimal rotation. We then use the following metrics to compare A and $\hat{B}$:

1. The average l_2 distance between embedding vectors.
2. Accuracy of the nearest neighbor search from A and $\hat{B}$.
3. Accuracy of the optimal-transport-induced matching.

Table 1. Alignment results between different pairs of embeddings (vocab size = 5000)

Metric	C1-C2	G1-G2	C1-G1	LSTM-C1
Ave. l_2 distance (Least Squares)	0.309	0.623	0.965	0.937
Matching Acc. (nearest, top1)	1.000	0.634	0.4302	0.2284
Matching Acc. (nearest, top5)	1.000	0.8769	0.5712	0.4388
Matching Acc. (OT)	1.000	1.000	0.8042	0.6328

*C1 and C2 refer to Word2Vec (CBOW)[3] embeddings trained with different initializations; G1 and G2 refer to GloVe[1] embeddings trained with different initializations; all trained on IMDB[4] dataset. LSTM refers to pretrained Word2Vec embeddings fine-tuned by LSTM model[5] on sentiment analysis.

Embedding Similarity and Downstream Performance

Training with similar embeddings is **necessary but not sufficient** for downstream models to make similar logit predictions.

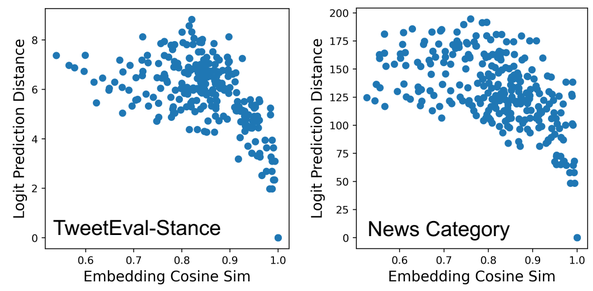

Figure 5. Correlation of Embedding Similarity and Downstream Performance on Classification Tasks. Each point corresponds to a pair of embeddings.

Embedding Ensemble

Accuracy on sentiment classification task are improved after ensembling pairs of embeddings using the three approaches illustrated in the previous section.

All ensembling methods result in an increase in accuracy. The first two methods preserved dimensions of all the original embeddings, resulting in more improvement. Embedding ensembling can improve accuracy with the same size of data.

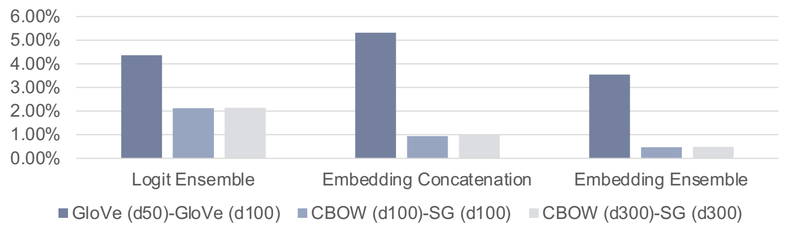

Figure 6. Accuracy Improvement on IMDB Sentiment Analysis After Ensembling
(The GloVe embeddings are pretrained on WikiText + GigaWord corpus[4] (6 billion tokens), and the other embeddings are pretrained on IMDB review corpus[5] (5 million tokens).)

Conclusions

- Embeddings trained on different corpora from different models and random initializations are different given the large geometric distances between these embeddings; nevertheless, they captured similar features of the input text data since optimal transport can find a reasonable matching between the aligned embeddings.
- For simple classification tasks such as sentiment analysis and news headline classification, dissimilar embeddings have different downstream performance, but it is inconclusive whether similar embeddings have similar downstream performance.
- For classification tasks, ensembling embeddings or their outputs can improve downstream performance; while the latter leads to a higher improvement in accuracy, the former provides insights for distilling better embeddings.

References

[1] Pennington, Jeffrey, Socher, Richard, and Manning, Christopher, "GloVe: Global Vectors for Word Representation", *aclanthology.org/D14-1162* (2014).
[2] Zeyuan Allen-Zhu and Yuanzhi Li. "Towards understanding ensemble, knowledge distillation and self-distillation in deep learning." *arXiv preprint arxiv.2012.09816* (2012).
[3] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, "Efficient Estimation of Word Representations in Vector Space", *arXiv preprint arXiv:1301.3781* (2013).
[4] Maas, Andrew L. and Daly, Raymond E. and Pham, Peter T. and Huang, Dan and Ng, Andrew Y. and Potts, Christopher, "Learning Word Vectors for Sentiment Analysis", *aclweb.org/anthology/P11-1015* (2011).
[5] Sachan, Devendra Singh, Manzil Zaheer, and Ruslan Salakhutdinov, "Revisiting LSTM Networks for Semi-Supervised Text Classification via Mixed Objective Function". *AAAI* 33 (01):6940-48. (2019)
[6] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. *2016.* "Pointer Sentinel Mixture Models", *arXiv preprint arXiv: 1609.07843* (2016).