Vector of Locally Aggregated Embeddings for Text Representation

Hadi Amiri[a] **and Mitra Mohtarami**[b]
[a]Harvard, Department of Biomedical Informatics, Boston, MA, USA
[b]MIT, Computer Science and Artificial Intelligence Laboratory, Cambridge, MA, USA
hadi@hms.harvard.edu, mitra@csail.mit.edu

Abstract

We present **V**ector of **L**ocally **A**ggregated **E**mbeddings (VLAE) for effective and, ultimately, lossless representation of textual content. Our model encodes each input text by effectively identifying and integrating the representations of its semantically-relevant parts. The proposed model generates high quality representation of textual content and improves the classification performance of current state-of-the-art deep averaging networks across several text classification tasks.

1 Introduction

Representation learning algorithms can reveal intrinsic low-dimensional structure in data (Rumelhart et al., 1986; Bengio et al., 2013; LeCun et al., 2015). In particular, deep averaging networks (DANs) are effective for text classification (Shen et al., 2018; Arora et al., 2017; Wieting et al., 2016; Iyyer et al., 2015). They achieve their improvement through use of word embeddings, weighted averaging, and deepening networks. The above works show that DANs can outperform RNNs and CNNs in text classification while taking only a fraction of their training time.

In this work, with a special focus on DANs, we study the effect of information loss associated with average word embeddings and develop algorithms that are robust against information loss for text representation. We show that divergence of word embeddings from their average can be considered as a good proxy to quantify information loss; in particular, longer documents suffer from significant information loss when represented by average word embeddings. These results inspire our work to develop a novel representation learning approach based on Vector of Locally Aggregated Descriptors (VLAD) (Jégou et al., 2010; Arandjelovic and Zisserman, 2013)–an effective approach to integrate image descriptors for large scale image datasets. Our model identifies semantically-relevant parts of documents and locally integrates their representations through clustering and autoencoding. In contrast to averaging, our model prevents larger semantically-relevant parts of inputs to dominate final representations. It improves DANs by 5.30 macro-F1 points in classifying longer texts and show comparable performance to them on shorter text.

2 Preliminary Analysis

How can information loss be quantified when word embeddings are averaged? How important it is to address information loss when representing textual content? Are representation learning algorithms robust against information loss? We conduct experiments to answer these questions with respect to deep averaging network (DANs). Our study can inspire works in more complex averaging approaches such as those reported in (Torabi Asr et al., 2018; Kiela et al., 2015) as well as recent works on unsupervised semantic similarity (Pagliardini et al., 2018). We use the DAN developed in (Joulin et al., 2017) and several datasets containing short and long documents to answer these questions.

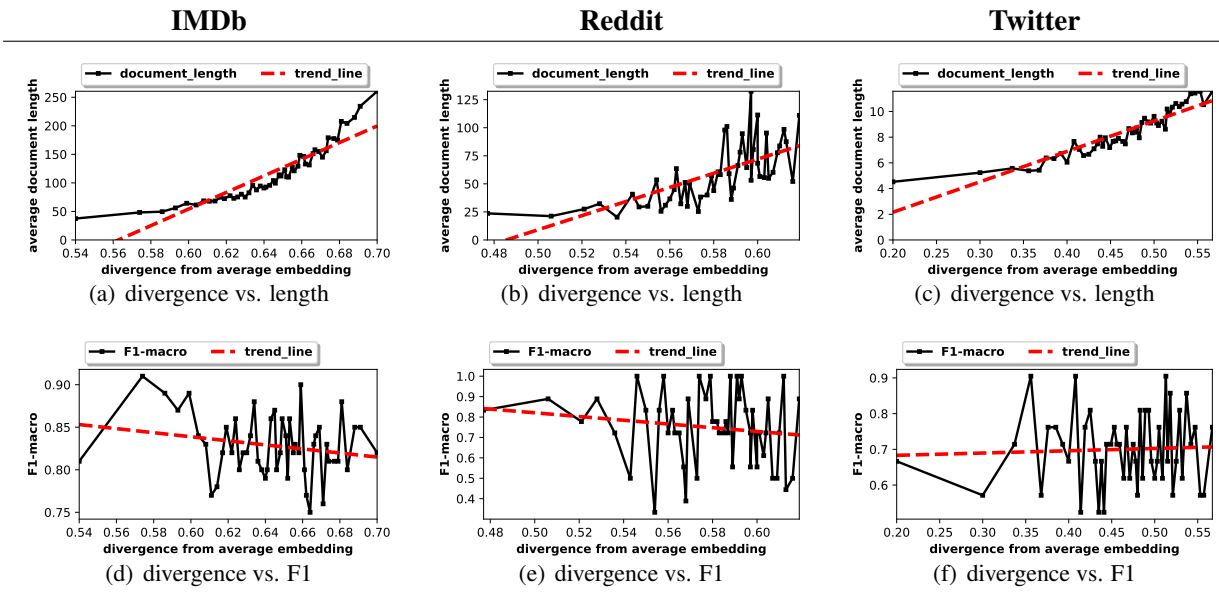
2.1 Quantifying Information Loss

Let's assume a d-dimensional word embedding space. We quantify the amount of information loss in the average word embedding vector of a given document $\mathbf{S} \in \mathbb{R}^{n \times d}$ by computing the average divergence (or distance) between its word embeddings, $\mathbf{w}_i \in \mathbb{R}^d \; \forall i \in \{1 \dots n\}$, and their average vector, $\bar{\mathbf{s}} = 1/n \sum_i \mathbf{w}_i$, $\bar{\mathbf{s}} \in \mathbb{R}^d$, as follows:

$$divergence = \frac{1}{n} \sum_i (1 - cosine(\bar{\mathbf{s}}, \mathbf{w}_i)). \quad (1)$$

Figures 1(a)–1(c) show strong positive correlation between divergence and document length

Proceedings of NAACL-HLT 2019, pages 1408–1414
Minneapolis, Minnesota, June 2 - June 7, 2019.



(a) divergence vs. length (b) divergence vs. length (c) divergence vs. length

(d) divergence vs. F1 (e) divergence vs. F1 (f) divergence vs. F1

Figure 1: Quantification of information loss associated with average word embeddings. Divergence indicates the average distance between individual word embeddings (of size $d = 300$) and their average embedding, see Equation (1). (a-c): show strong positive correlation between divergence and average document length (number of words in a document) across datasets. (d-f): show macro F1 performance of the deep averaging network developed in (Joulin et al., 2017) across datasets: performance considerably drops for higher values of information loss/divergence, e.g. divergence values above 0.55. Note that, we sort and bin instances based on their divergence values and report average length (a-c) and macro-F1 (d-f) for each bin.

across long and short text datasets. Given these results and if we assume longer documents should suffer from greater information loss if represented by average word embeddings, divergence from mean can be a good proxy to quantify information loss associated with average embeddings.

2.2 Effect of Information Loss

As Figures 1(d) and 1(e) show, DAN's macro-F1 classification performance considerably decreases as divergence (or text length) increases for IMDb and Reddit datasets; note that we sort and bin instances based on their divergence values and report average macro-F1 for each bin. In particular, as the trend lines in Figures 1(d) and 1(e) show, the average macro-F1 performance drops from 0.86 and 0.82 on shorter IMDb and Reddit posts to 0.82 and 0.71 on their longer posts respectively. In addition, the result on Twitter dataset, Figure 1(f), shows that DANs are robust against small information loss, i.e. small divergence values below 0.55 do not inversely affect macro-F1 performance. This result is also observed on the other two datasets, see macro-F1 performance for small divergence values (≤ 0.55) in Figures 1(d) and 1(e).

The above experiments show that (a): significant information loss can occur when word embeddings are averaged, in particular, when representing longer documents, and (b) such information loss can inversely affect the performance of downstream classifiers like DANs on longer texts. In this paper, we develop an effective representation learning model to tackle this problem.

3 Method

We propose to utilize semantically-relevant parts of inputs to tackle information loss associated with average word embeddings. Assuming that semantically-relevant words are closer to each other in semantic space (constructed over a global vocabulary), we expect divergence between words in semantically-relevant parts of inputs (i.e. information loss associated to their average word embedding) to be very small. Thus, as Figure 2 illustrates, we propose to cluster the semantic space to first identify semantically-relevant parts of inputs over a global vocabulary; we then effectively integrate these parts to represent documents.

Let's assume a global vocabulary $\mathcal{V}$ in which words are represented in a d-dimensional space, $\mathbf{w} \in \mathbb{R}^d$. As Figure 2 shows, we first cluster this

semantic space into k clusters through the following objective function over $\mathcal{V}$:

$$min \sum_{\mathbf{w} \in \mathcal{V}} ||f(\mathcal{C}, \mathbf{w}) - \mathbf{w}||^2, \quad (2)$$

where $\mathcal{C}$ is the set of k cluster centers, $|\mathcal{C}| = k$, and $f(\mathcal{C}, \mathbf{w})$ returns the nearest cluster center $\mathbf{c} \in \mathcal{C}$ to the embedding vector $\mathbf{w}$ based on `cosine` similarity among embeddings or Euclidean distance in case of K-Means.[1]

Given a document $\mathbf{S} \in \mathbb{R}^{n \times d}$ with an arbitrary number of $n \geq 1$ words, and the above k cluster centers, we compute the representation of the document in each cluster $\mathbf{c}_i$, $i = 1 \ldots k$ as follows:

$$\mathbf{a}_i = \frac{1}{z_i} \sum_{j: f(\mathcal{C}, \mathbf{w}_j) = \mathbf{c}_i} \mathbf{w}_j \quad (3)$$

$$z_i = |j : f(\mathcal{C}, \mathbf{w}_j) = \mathbf{c}_i|,$$

where $\mathbf{a}_i \in \mathbb{R}^d$ indicates the representation of the document at cluster $\mathbf{c}_i$ and is obtained by taking the average embedding of words of the document that have been assigned to cluster $\mathbf{c}_i$ according to Equation (2), and z_i is the number of such words in cluster $\mathbf{c}_i$. To this end, each document can be represented by $\mathbf{A} \in \mathbb{R}^{d \times k}$ which is obtained by concatenating its cluster-level representations. Note that we didn't observe any performance difference between the above averaging process versus computing *residuals* (differences between word embeddings and corresponding cluster centroids) which is commonly used to represent cluster-level image descriptors (Jégou et al., 2010; Arandjelovic and Zisserman, 2013) in image processing.

Since $\mathbf{A}$s are of fixed length, they can be readily used as features in traditional classification and clustering algorithms. However, they can cause efficiency issues because of their large size ($d \times k$); note that the typical value for embedding dimension d is 300 (Pennington et al., 2014; Mikolov et al., 2013). To tackle this issue, we further integrate cluster-level representations, at the cost of some further information loss, to create representations of lower dimension for inputs.

In particular, given all input documents with k cluster-level representations $\mathbf{A} \in \mathbb{R}^{d \times k}$ for each document, we develop an autoencoder with one hidden layer that integrates these cluster-level representations to create a final representation for each document, vector $\mathbf{a} \in \mathbb{R}^{d \times m}$ where m is the dimensionality reduction parameter and $m \times d$ is length of the representation (final layer of the encoder) and is smaller than $d \times k$ for $m < k$. Training a single-layer autoencoder corresponds to optimizing the learning parameters to minimize the overall loss between inputs and their reconstructions. For real-valued $\mathbf{A}$, squared loss is often used (Vincent et al., 2010), i.e. $l(\mathbf{A}) = ||\mathbf{A} - \hat{\mathbf{A}}||^2$ where $\hat{\mathbf{A}} \in \mathbb{R}^{d \times k}$ is reconstruction of $\mathbf{A}$ and generated by the decoder from $\mathbf{a}$. Our intuition is that if $\mathbf{a}$ leads to a good reconstruction of $\mathbf{A}$, it has retained all information available in the input.

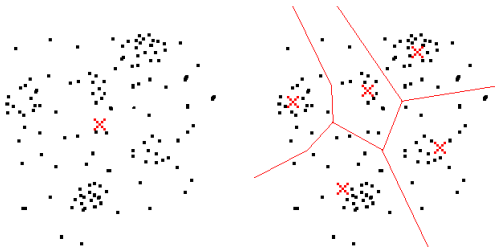


Figure 2: Illustration of VLAEs: squares show word embeddings of a hypothetical document and crosses show their corresponding average. (a): shows average word embedding in center - the case of high information loss or divergence from mean, (b): shows average word embeddings in corresponding word clusters; these embeddings are used to produce VLAEs.

We refer to $\mathbf{a} \in \mathbb{R}^{d \times m}$ as the Vector of Locally Aggregated Embeddings (VLAE). We expect this final representation to be robust against information loss due to its cluster-level local aggregation which prevents larger portions of semantically-similar words to dominate the representation.

4 Experiments

Data: We investigate VLAEs in three binary classification tasks: sentiment classification on IMDb (Maas et al., 2011), disease-text classification on Reddit, where the task is to classify reddit posts as relevant or irrelevant to specific diseases, and churn prediction on Twitter (Amiri and Daumé III, 2015), where the task is to classify/predict if given tweets indicate user intention about leaving brands, e.g. the tweet "my days with BRAND are numbered" is a *churny* tweet. See details in Table 1. For pre-processing, we change all texts to lowercase, and remove stop words, user names, and URLs from texts.

[1] This problem can be solved through gradient descent seeded with an initial set of k examples drawn uniformly at random from $\mathcal{V}$ (Bottou and Bengio, 1995; Sculley, 2010).

	Train	Val	Test	Unlabeled
IMDb	40K	5K	5K	50K
Reddit	2K	1K	1K	100K
Twitter	3K	1K	1K	100K

Table 1: Statistics of dataset used in experiments.

Settings: We use validation data for hyperparameter tuning and model selection. We use 300-dimensional word embeddings ($d = 300$) provided by Google (Mikolov et al., 2013), and for greater number of ds, we train `word2vec` on unlabeled data, see Table 1. In addition, we set the dimensionality reduction parameter m from $\{1 \dots 4\}$ using validation data. The best value of m is the same across tasks/datasets, $m = 2$. Furthermore, we determine the number of clusters k for VLAEs by choosing the optimal k from $\{2^i, i = \{1 \dots 7\}\}$ using validation data of each dataset. We learn optimal k with respect to task, but not embedding space, due to significant density of the semantic space of word embeddings, see Note on Clustering Word Embeddings.

Baselines: We consider two versions of DANs as baselines: `Avg_small` and `Avg_large` which represent documents by average word embedding of size $d = 300$ and $d = m \times 300$ respectively. Note that, for fair comparison, `Avg_large` has the exact same size as our model (`VLAE`); however, depending on m, their network size is 1.3-1.6 times greater than that of `Avg_small` due to difference in input dimensions. We use 3 hidden layers of size 300 for above networks. Also, to directly evaluate the effect of averaging, we do not adjust initial word embeddings during training.

Experimental Results: Table 2 shows the performance of different models across datasets. The results show that `VLAE` significantly outperforms `Avg_small` and `Avg_large` by 2.6 and 7.2 points in Macro-F1 on IMDb. The corresponding values on Reddit dataset are 6.7 and 3.4 points respectively. We believe these improvements are due to more effective and lossless representation of inputs. We note that `Avg_large` performs worse than `Avg_small` on IMDb. This could be attributed to the size of training data which may not be enough to train `Avg_large`, or to lower quality of input representations in `Avg_large` compared to `Avg_small` in case of IMDb. Note that although `VLAE` has the same number of parameters as `Avg_large`, it uses autoencoding to effectively filter redundant information. Verifying these hypotheses will be the subject of future work. In addition, `VLAE` show lower performance than `Avg_large` on Twitter dataset, F1 of 72.62 versus 73.08. We attribute this result to the shorter length of tweets for which, as we experimentally showed before, averaging does not cause major divergence in representations. On average, `VLAE` improves `Avg_small` and `Avg_large` by 4.7 and 5.3 F1 points on IMDb and Reddit (longer texts) respectively. It also shows comparable performance to best performing model on Twitter (shorter texts).

	Avg_small	Avg_large	VLAE
IMDb	83.11	78.52	**85.72***
Reddit	59.42	62.72	**66.10***
Twitter	61.42	**73.08***	72.62
AVG	67.98	71.44	**74.81**

Table 2: Macro-F1 performance of different models across datasets. Asterisk mark (*) indicates significant difference between top two systems. `VLAE` outperforms other models on longer texts, and show comparable performance to `Avg_large` on shorter text.

We also compare models in terms of the quality of their representations. For this comparison, we ignore input preparation time and assume a model that generates better representations should converge faster than other models; note that the overall turnaround time of `VLAE` is greater than that of `Avg_small` or `Avg_large` because of its input preparation time which we ignore for the purpose of this experiment. The result show that `VLAE` leads to 7.5, 1.3, and 1.3 times faster convergence than `Avg_small` and 14.9, 2.6, and 1.8 times faster convergence than `Avg_large` on IMDb, Reddit, and Twitter datasets respectively. Considering the size of these networks, these results indicate that representations obtained from VLAE are much better than those of its counterparts.

Note on Clustering Word Embeddings: In experiments, we observe clusters obtained from word embeddings are often very dense. This is a challenge for our model because with small number of clusters (ks) potentially dissimilar words can appear in the same cluster, while with large ks semantically-similar words may appear in different clusters. Neither of these are desired.

To illustrate the above challenge, we report Silhouette Coefficient (SC) (Rousseeuw, 1987) of k-means with different number of clusters obtained from words embeddings across datasets. SC indicates how well cluster boundaries are detected

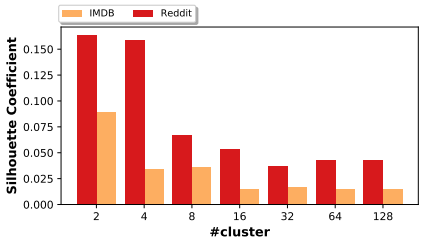


Figure 3: Mean Silhouette Coefficient computed for different number of clusters; a higher Silhouette Coefficient score indicates better defined clusters.

by a clustering model. It is calculated using the mean intra-cluster distance and the mean nearest-cluster distance for each sample. Specifically, the mean distance between each embedding and all other embeddings in the same cluster (mc), and the mean distance between the embedding and all other embeddings in the next nearest cluster (the nearest cluster that the embedding is not part of) (mn) are used to measure SC for the embedding:

$$\frac{mn - mc}{\max(mn, mc)}.$$

The best and worst SC scores are 1 and -1 which indicate ideal and worst clustering respectively. Also, values near 0 indicate overlapping clusters.

Figure 3 shows the mean SCs computed over all word embeddings for IMDb and Reddit datasets.[2] The results show that (a): the best number of clusters is $k = 2$ on both datasets, and (b): Silhouette Coefficient scores generally home in on values close to zero as the number of clusters increases. These results show significant density of embeddings in semantic space. Therefore, we optimize the number of clusters for creating VLAEs by resorting to validation data and measuring task-specific performance. From these results, we conclude that a hierarchical clustering approach that recursively combines pairs of semantically-similar clusters could help better defining these clusters and perhaps improve the performance of our model.

5 Related Work

Deep averaging networks (DANs) (Joulin et al., 2017; Iyyer et al., 2015; Arora et al., 2017; Shen et al., 2018) were developed based on the successes of vector operations in embedding space. In contrast to their simplicity, DANs showed high performance in text classification tasks.

Arora et al. (2017) showed that sentences can be effectively represented by the weighted average of their word embeddings modified by PCA/SVD. In addition, the DANs developed in (Iyyer et al., 2015), (Joulin et al., 2017), and (Shen et al., 2018) were feed-forward networks that used average word embeddings to represent inputs; they were effective for several NLP tasks such as document categorization, text pair similarity, and short sentence classification. Furthermore, feed-forward architectures like DANs have been used for language modeling (Bengio et al., 2003) and greedy transition-based dependency parsing (Chen and Manning, 2014) with fast turnaround time.

In addition, previous research investigated a variety of vector operations that could replace the averaging operation used in the DANs. Many of these operations have been studied in (Mitchell and Lapata, 2008) for modeling the compositionality of short phrases, or showing the utility of simple vector computations(Banea et al., 2014). The operations in (Mitchell and Lapata, 2008) were also extended to use syntactic relation between words and grammar (Erk and Padó, 2008; Collobert and Weston, 2008). Also, clustering semantic space was studied in (Mekala et al., 2017) to learn context information for words and for tasks like topic coherence and information retrieval.

In this work, we built on previous work on DANs and investigated and tackled information loss associated with average word embeddings.

6 Conclusion and Future Work

We investigate information loss associated with average word embeddings. We show that averaging lead to significant information loss and propose to tackle the issue by identify semantically-similar parts of documents through clustering of semantic space at word-level and integrating cluster-level representations through autoencoding. A promising future direction is to use hierarchical clustering to create better cluster-level representations.

Acknowledgments

We thank anonymous reviewers for their insightful comments and constructive feedback.

[2]The corresponding values for Twitter data are [0.52, 0.48, 0.42, 0.31, 0.21, 0.13, 0.09] respectively. We have not reported these values in Figure 3 to better illustrate the scores for other datasets.

References

Hadi Amiri and Hal Daumé III. 2015. Target-dependent churn classification in microblogs. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2361–2367. AAAI Press.

Relja Arandjelovic and Andrew Zisserman. 2013. All about vlad. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1578–1585.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *International conference on learning representations (ICLR)*.

Carmen Banea, Di Chen, Rada Mihalcea, Claire Cardie, and Janyce Wiebe. 2014. Simcompass: Using deep learning word embeddings to assess cross-level similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 560–565.

Yoshua Bengio, Aaron Courville, and Pierre Vincent. 2013. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8).

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Leon Bottou and Yoshua Bengio. 1995. Convergence properties of the k-means algorithms. In *Advances in neural information processing systems*, pages 585–592.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Katrin Erk and Sebastian Padó. 2008. A structured vector space model for word meaning in context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 897–906. Association for Computational Linguistics.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of ACL-IJCNLP*.

Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. 2010. Aggregating local descriptors into a compact image representation. In *CVPR 2010-23rd IEEE Conference on Computer Vision & Pattern Recognition*, pages 3304–3311. IEEE Computer Society.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431.

Douwe Kiela, Felix Hill, and Stephen Clark. 2015. Specializing word embeddings for similarity or relatedness. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2044–2048. Association for Computational Linguistics.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, pages 142–150. Association for Computational Linguistics.

Dheeraj Mekala, Vivek Gupta, Bhargavi Paranjape, and Harish Karnick. 2017. Scdv : Sparse composite document vectors using soft clustering over distributional representations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 659–669. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. *Proceedings of ACL-08: HLT*, pages 236–244.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 528–540.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

DE Rumelhart, G Eoo Hinton, and RJ Williams. 1986. Learning internal representations by error propagation. *Parallel distributed processing: explorations in the microstructure of cognition*, 1.

David Sculley. 2010. Web-scale k-means clustering. In *Proceedings of the 19th international conference on World wide web*, pages 1177–1178. ACM.

Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. 2018. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, volume 1.

Fatemeh Torabi Asr, Robert Zinkov, and Michael Jones. 2018. Querying word embeddings for similarity and relatedness. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 675–684.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research*, 11:3371–3408.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1504–1515. Association for Computational Linguistics.