Final Report

Introduction

Word embeddings are a sort of word representation that allows for the depiction of words with comparable meanings. They are a distributed representation for text that may be one of the major innovations behind deep learning techniques' excellent performance on difficult natural language processing challenges. With the use of unsupervised learning, the popular algorithm Word2Vec creates dense vector representations of words in huge corpora (Mikolov et al., 2013). Word2Vec creates vectors that represent the semantic connections between words. For instance, the vector that is closest to vec('queen') is produced by the operation vec('king') - vec('man') + vec('woman').

Implementation

Using Python, we created the Word2Vec model from scratch and trained it using the provided dataset. Preprocessing was done on the text data to get rid of punctuation, digits, and uncommon terms. The following settings were utilized for training: a window size of 5, a minimum word count of 2, and a word vector size of 100. The dimension of the word vectors is determined by the size parameter. The greatest separation between the target word and its nearby words is determined by the window size. We can ignore unusual terms by using the minimum word count.

Efficency

Execution time and memory utilization were used to gauge how effective the implementation was. In 1.32 seconds, our application produced word embeddings, using 245.74 MB of memory. These outcomes show how well our system scales with more datasets and is highly effective.

Visualization and Clustering

We used t-SNE, a dimensionality reduction approach that is especially well suited for the display of high-dimensional datasets, to visualize the word embeddings after acquiring them. We noticed that in the two-dimensional space, words with comparable semantic properties were clustered together.

By using k-Means clustering to the word embeddings, we were able to further investigate them. To investigate the impact of k on the generated clusters, we tested with different values of k. Semantically comparable terms were clustered together for smaller values of k. More precise clusters were seen as we raised k. With a larger k, for instance, terms associated with a particular subject, like "sports," formed a separate cluster.

Potential Problems and Suggestions for Improvement

Our implementation has various potential problems even if it delivered encouraging results. First, the selection of hyperparameters like size, window, and minimum word count can have a big impact on the embeddings' quality. The embeddings might be enhanced by adjusting these values while utilizing a validation set.

Second, although word order can be significant for many tasks, our approach does not take it into consideration. Future research could look towards models like recurrent neural networks (RNNs) that can collect this data.

Finally, terms that are not in the model's vocabulary are not handled. Character-level embeddings could be one solution to this problem.

Conclusions and Future Work

Word embeddings were successfully produced by our Word2Vec implementation, which we then visualized and clustered. Tuning hyperparameters, experimenting with other model architectures, and comparing the embeddings with those produced by other tools are all tasks that will need to be completed in the future.

References

1. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. In Proceedings of the International Conference on Learning Representations (ICLR).
2. Van der Maaten, L., & Hinton, G. (2008). Visualizing Data using t-SNE. Journal of Machine Learning Research, 9, 2579-2605.

3. Arthur, D., & Vassilvitskii, S. (2007). k-means++: The Advantages of Careful Seeding. In Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007.