Design and Implementation of Word2Vec Model in PyTorch for Autonomous Flying Cab Voice Assistant

Introduction

The objective of this project is to create a Word2Vec word embedding model for a voice assistance for an autonomous flying cab. The model will be used to perform word clustering and visualization after being trained on the supplied datasets. The voice assistant's effectiveness and comprehension of natural language are improved by the word embeddings, which are input for various NLP subcomponents (Goldberg and Levy, 20141).

Word Embedding and Word2Vec

Word embeddings are a sort of word representation that enables the depiction of words with related meanings. A prominent technique for producing these word embeddings is Word2Vec, which makes use of a shallow neural network model (Mikolov et al., 20132). Word2Vec employs the Continuous Bag of Words (CBOW) and the Skip-gram techniques. As the CBOW model predicts a target word based on the context, it may be a better fit for our use case in this project.

Model Structure

Bag-of-Words Encoding, Feed-Forward Network, and Softmax will all be included in the CBOW model's PyTorch implementation.

Bag-of-Words Encoding: The context words are transformed into numerical vectors in this stage. We are implementing this encoding independently, without utilizing PyTorch's Embedding Layer, in order to comply with the criteria.

Feed-forward network: a straightforward neural network that learns to predict the target word based on context, will be fed the Bag-of-Words encoded vectors.

Softmax: In order to forecast the target word, the feed-forward network's output is sent via a softmax function, which converts the outputs into probabilities (Goodfellow et al., 20163).

Data Preprocessing

The preprocessed datasets will comprise lowercase conversion, tokenization, and perhaps stop words removal. The model is then trained using this tidy and organized data.

Implementation Plan

The model will develop a basic PyTorch code framework to compute word embeddings. This will entail establishing the forward pass, the training loop, and the model structure. We'll make sure that our code is modular, clear, and flexible.

Future Work

The next steps after the Word2Vec model have been successfully implemented are to train the model using the supplied datasets, conduct word clustering, and visualize the clusters. The results of the clustering will then be used to assess the model's efficacy.

References

1. Goldberg, Y., & Levy, O. (2014). word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method. arXiv preprint arXiv:1402.3722.
2. Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
3. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press. http://www.deeplearningbook.org