Skip-Gram Word2Vec with Negative Sampling on Wikipedia

1. Introduction

Natural Language Processing systems require numerical representations of text to perform learning and inference. Traditional representations such as one-hot encoding fail to capture semantic similarity between words. Word embeddings address this limitation by representing words as dense vectors in a continuous space. In this assignment, I implemented the Skip-Gram Word2Vec model with Negative Sampling and trained it on a Wikipedia-based corpus (enwik8).

2. Dataset and Preprocessing

The dataset used is the enwik8 corpus, a cleaned subset of English Wikipedia text. The preprocessing steps included converting text to lowercase, removing punctuation and special characters, tokenizing the text into words, and removing rare words using a minimum frequency threshold. These steps ensured a clean and consistent input for training the model.

3. Skip-Gram Model

The Skip-Gram model learns word representations by predicting surrounding context words given a center word. For example, in the sentence 'the king ruled the kingdom', the word 'king' is used as the center word and surrounding words such as 'the', 'ruled', and 'kingdom' are treated as context words. Training pairs are created from these center–context relationships.

4. Negative Sampling

Computing probabilities over the entire vocabulary using softmax is computationally expensive. Negative Sampling addresses this by converting the problem into a binary classification task. For each positive (center, context) pair, a fixed number of negative samples are drawn from a noise distribution. The model learns to assign high similarity to true word pairs and low similarity to randomly sampled pairs.

5. Model Architecture and Training

The model consists of two embedding matrices: an input embedding matrix for center words and an output embedding matrix for context words. Both matrices are learned during training. The model was trained using PyTorch with an embedding dimension of 100, a context window size of 5, and 5 negative samples per positive pair. Training was performed for 3 epochs using the Adam optimizer.

6. Evaluation Using Cosine Similarity

Cosine similarity was used to evaluate semantic similarity between words. For example, the cosine similarity between the learned embeddings for 'king' and 'queen' was higher than that between 'king' and 'apple'. These similarities were compared with pretrained Word2Vec embeddings provided by Gensim, and similar semantic trends were observed.

7. Word Analogy Examples

Word analogy tasks were evaluated using vector arithmetic. Example cases tested include: king – man + woman ≈ queen, and paris – france + italy ≈ rome. These examples demonstrate that the learned embeddings capture linear semantic relationships between words.

8. Bias Detection Examples

To analyze bias in the learned embeddings, a gender direction was computed using word pairs such as man – woman and he – she. Profession-related words were projected onto this direction. For example, words such as 'engineer' and 'programmer' showed stronger association with male terms, while 'nurse' and 'assistant' showed stronger association with female terms.

9. Conclusion

This assignment demonstrates the successful implementation of Skip-Gram Word2Vec with Negative Sampling trained on Wikipedia text. The learned embeddings capture meaningful semantic relationships, support analogy reasoning, and reveal biases present in the training data, highlighting both the strengths and limitations of distributional word representations.