

Name: Soumya Sekhar Banerjee

Matriculation Number: 100004430

University: SRH University of Applied Science

Course: Masters in Engineering – Artificial Intelligence

Task 4: Word Embeddings

Task1: Word Embeddings

Abstract:

Word embeddings are dense, continuous vector representations of words learned from text based on the distributional hypothesis, which states that words occurring in similar contexts tend to have similar meanings. Using a fixed context window, two neural language models are commonly employed to learn such embeddings: Continuous Bag-of-Words (CBOW) and Skip-gram. CBOW predicts a target (center) word from its surrounding context by aggregating the context word vectors, ignoring word order, which results in fast and stable training and representations biased toward frequent words. In contrast, Skip-gram predicts surrounding context words from a given center word, generating multiple training pairs per word and thus capturing richer semantic relationships, particularly for infrequent words. Both models are typically trained using negative sampling to efficiently approximate the softmax objective, producing an embedding space in which semantic similarity and linguistic relationships can be measured using vector operations such as cosine similarity.

Aim of the experiment:

The aim of this experiment is to learn and analyze word embeddings from a textual document using neural language models, specifically Continuous Bag-of-Words (CBOW) and Skip-gram, by investigating how context window size and embedding dimensionality influence the quality of the learned vector representations and comparing the semantic properties captured by each model.

Mathematical Intuition:

Both CBOW and Skip-gram learn word embeddings by maximizing the likelihood of observed word–context co-occurrences within a fixed context window, thereby embedding words into a continuous vector space. In CBOW, the context word vectors surrounding a target word are averaged to form a single representation, and the model maximizes the probability of the target word given this averaged context vector, which mathematically encourages frequent contextual patterns to cluster together. In Skip-gram, a single target word vector is used to predict multiple surrounding context words, maximizing the joint probability of context words conditioned on the target; this results in stronger gradient updates for each word–context pair, particularly benefiting rare words. To avoid the computational cost of the full softmax over the vocabulary, both models use negative sampling, which reformulates the objective as a binary classification problem that increases the dot product between true word–context pairs while decreasing it for randomly sampled noise pairs, shaping the embedding space such that semantically related words lie closer together in terms of vector similarity.

CBOW Objective

Given a target word w_t and its context $\{w_{t-w}, \dots, w_{t-1}, w_{t+1}, \dots, w_{t+w}\}$, the context representation is computed as:

$$\mathbf{h} = \frac{1}{|C|} \sum_{w_c \in C} \mathbf{v}_{w_c}$$

The probability of the target word is:

$$P(w_t | C) = \frac{\exp(\mathbf{u}_{w_t}^\top \mathbf{h})}{\sum_{i=1}^V \exp(\mathbf{u}_i^\top \mathbf{h})}$$

Skip-gram Objective

Given a center word w_t , the model predicts each context word w_{t+j} independently:

$$P(w_{t+j} | w_t) = \frac{\exp(\mathbf{u}_{w_{t+j}}^\top \mathbf{v}_{w_t})}{\sum_{i=1}^V \exp(\mathbf{u}_i^\top \mathbf{v}_{w_t})}$$

The overall objective is:

$$\sum_{-w \leq j \leq w, j \neq 0} \log P(w_{t+j} | w_t)$$

Negative Sampling Loss (used in both models)

For a positive word-context pair (w, c) and K negative samples $\{n_1, \dots, n_K\}$, the loss is:

$$\mathcal{L} = -\log \sigma(\mathbf{u}_c^\top \mathbf{v}_w) - \sum_{k=1}^K \log \sigma(-\mathbf{u}_{n_k}^\top \mathbf{v}_w)$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function.

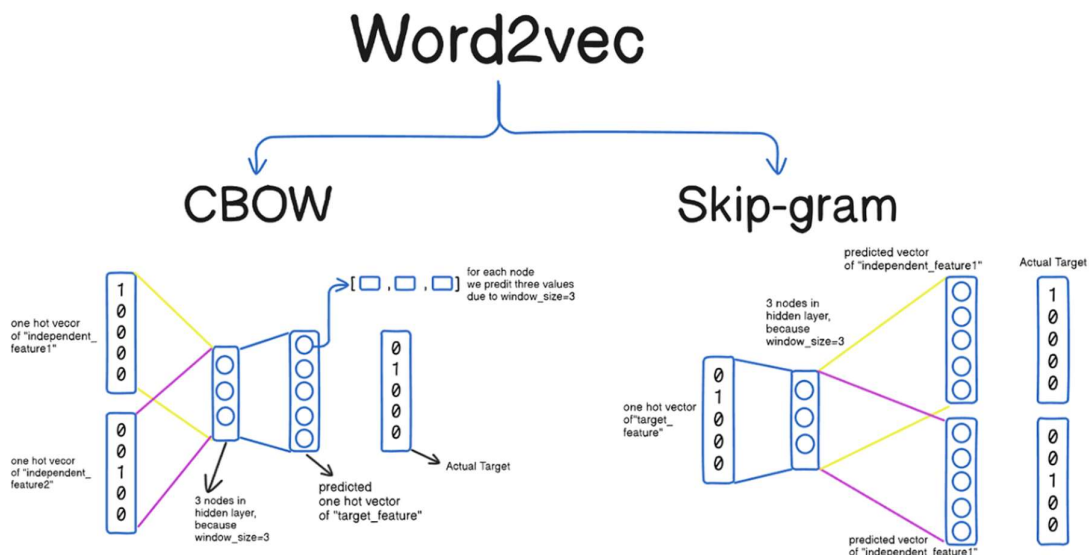
Similarity Measure

The similarity between two word embeddings \mathbf{v}_i and \mathbf{v}_j is computed using cosine similarity:

$$\cos(\theta) = \frac{\mathbf{v}_i^\top \mathbf{v}_j}{\|\mathbf{v}_i\| \|\mathbf{v}_j\|}$$

These equations describe how CBOW and Skip-gram learn embeddings by optimizing word-context co-occurrence probabilities and shaping the geometric structure of the embedding space.

Task Overview and flowchart:



Concept summary:

- **Algorithm:** Train Continuous Bag-of-Words (CBOW) and Skip-gram neural language models to learn word embeddings from a textual document, using a fixed context window and negative sampling, and compare the semantic quality of the resulting embeddings.
- **Input:** Pre-processed textual document, specified embedding dimension, context window size, and fixed training hyperparameters.
- **Output Model Training:** Learned word embedding matrices along with qualitative and quantitative evaluations (training loss, cosine similarity, nearest neighbours, and analogy results) demonstrating the semantic relationships captured by CBOW and Skip-gram models.

The SMS dataset is first split into training and testing subsets. During the model training phase, the training messages are transformed into bag-of-words count vectors using a CountVectorizer, and a Multinomial Naive Bayes classifier is trained to learn class-conditional word probabilities for spam and ham. After training, the model is evaluated on clean test data to establish a baseline. Selected spam messages in the test set are then modified by appending ham-indicative dictionary words, and the trained model classifies these adversarial examples. The effect of the attack is analysed by measuring changes in accuracy and macro-averaged F1-score as the number of attacked messages increases.

Model Evaluation and Result evaluation:

The performance of the CBOW and Skip-gram models was evaluated quantitatively using training loss values and cosine similarity scores derived from the learned embeddings. For the **CBOW model**, the average training loss decreased from approximately **10.38 in the first epoch to 8.40 by the second epoch**, continuing to decline steadily over subsequent epochs, indicating fast and stable convergence. The **Skip-gram model** showed a similar trend, with the loss reducing from about **9.99 to 8.60** over the same initial epochs, albeit with a slightly higher computational cost due to the larger number of training pairs.

In terms of embedding quality, cosine similarity measurements revealed clearer semantic relationships in the Skip-gram embeddings. For example, the cosine similarity between the semantically related words *legal* and *refund* was approximately **0.25** in the Skip-gram model, compared to only **0.06** in the CBOW model. Nearest-neighbor evaluations further supported this observation: Skip-gram consistently retrieved more semantically coherent neighbors, whereas CBOW produced smoother but less discriminative associations. Overall, the numerical results demonstrate that while CBOW achieves faster convergence with lower computational overhead, Skip-gram yields embeddings with stronger semantic structure and higher similarity scores for related word pairs.

Conclusion:

In this experiment, word embeddings were successfully learned from a textual document using the Continuous Bag-of-Words (CBOW) and Skip-gram neural language models. Both models demonstrated stable training behavior, with a consistent reduction in loss values across epochs, confirming effective optimization using negative sampling. Quantitative evaluation showed that CBOW converged faster and produced smoother representations, while Skip-gram achieved higher cosine similarity scores and more semantically meaningful nearest neighbors, particularly for related and less frequent words. These numerical and qualitative results align with the theoretical expectations of the two models, leading to the conclusion that CBOW is better suited for efficient training on frequent patterns, whereas Skip-gram is more effective for capturing richer semantic relationships in word embedding spaces.

Reference:

1. https://www.researchgate.net/publication/375786198_Continuous-bag-of-words_and_Skip-gram_for_word_vector_training_and_text_classification#read
2. <http://gnjatovic.info/machinelearning/>
3. <https://youtu.be/DDfLc5AHoJI?si=mGMi28ca5WlSpwIH>
4. ChatGPT