Asadullah Hill Galib
CSE 891-001: Deep Learning
Programming Assignment 1
September, 29

# 1 Linear Embedding - GLoVe

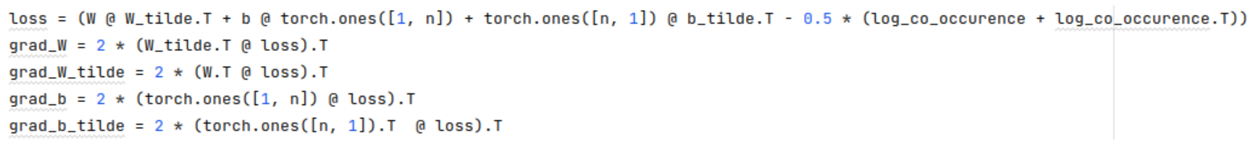
## 1.1 GLoVe Parameter Count

Number of trainable parameters: $2(V \times d + V)$

## 1.2 Gradient Expression

$$\frac{\partial L}{\partial w_i} = 2 \times \sum_{i,j=1}^{V} (w_i^T \tilde{w}_j + b_i + \tilde{b_j} - \log X_i j)\, \tilde{w}_j$$

## 1.3 Implementation



```
loss = (W @ W_tilde.T + b @ torch.ones([1, n]) + torch.ones([n, 1]) @ b_tilde.T - 0.5 * (log_co_occurence + log_co_occurence.T))
grad_W = 2 * (W_tilde.T @ loss).T
grad_W_tilde = 2 * (W.T @ loss).T
grad_b = 2 * (torch.ones([1, n]) @ loss).T
grad_b_tilde = 2 * (torch.ones([n, 1]).T  @ loss).T
```

Figure 1: Implementation of the gradient update

*Corresponding python file (`glove.py`) is also attached.

## 1.4 Embedding Dimension

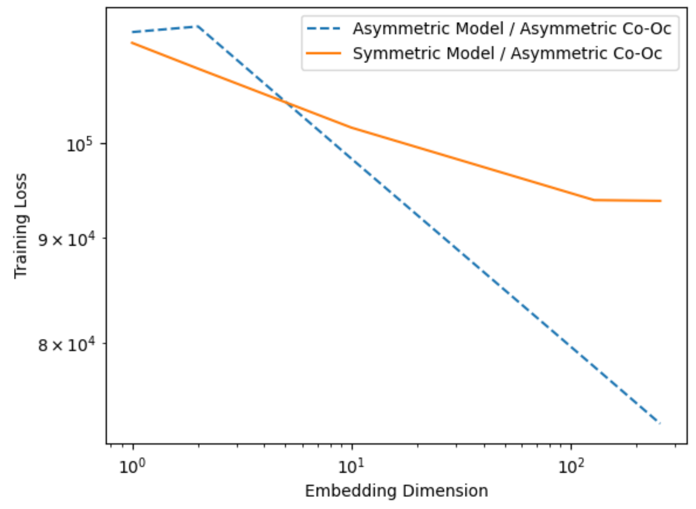


Figure 2: Training Performance

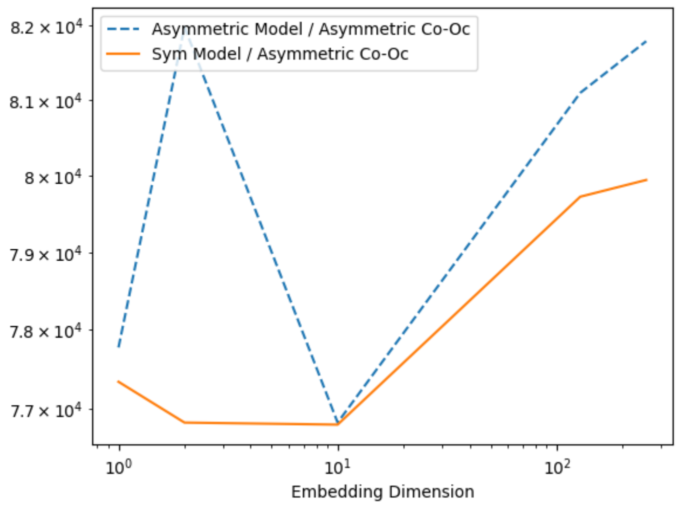


Figure 3: Validation Performance

The results are shown in Fig. 2 and Fig. 3. Comment on the results:

- $d = 10$ leads to optimal validation performance for both of the models.
- Larger d doesn't always lead to better validation error, as it makes the embedding space not consistent with the vocabulary size. If the vocabulary size is small, a large embedding space may make over speculation (confounding the embedding representation). So, it should be tuned with respect to vocabulary size.
- Symmetric model is performing better, because the asymmetric model tries to learn the relation from both directions, which confound the training process. Like, from deep learning perspective, learning the relation from a single direction (symmetric) is more convenient than learning from both direction (asymmetric) - which may induce more non-linearity.

## 2 Network Architecture

**Number of trainable parameters** $= (V \times D) + (N \times D \times H) + H + (V \times H) + V$

$V \times D$, from `word_embbeding_weights`;

$N \times D \times H$, `from embed_to_hid_weights`;

$V \times H$, `from hid_to_output_weights`;

$H$, `from hid_bias`;

$V$, `from output_bias`.

As $V \gg H > D > N$, the `hid_to_output_weights` part has the largest number of trainable parameters $(V \times H)$. **Reasoning**: $V \times H$ is greater than $V \times D$, as $H > D$. Also, $V \gg H > D > N$. So, $V > N \times D$ which indicates $V \times H > N \times D \times H$.

## 3 Training Neural Networks

The `layer.py` file is attached.

## 4 Analysis

### 4.1 1

(a) Fig. 4 denotes the 2-dimensional visualization for the trained model. There are quite a lot of clusters can be found. Like, (most, some, many), (man, woman), (between, against, another, by), etc. The words in each cluster share related context.

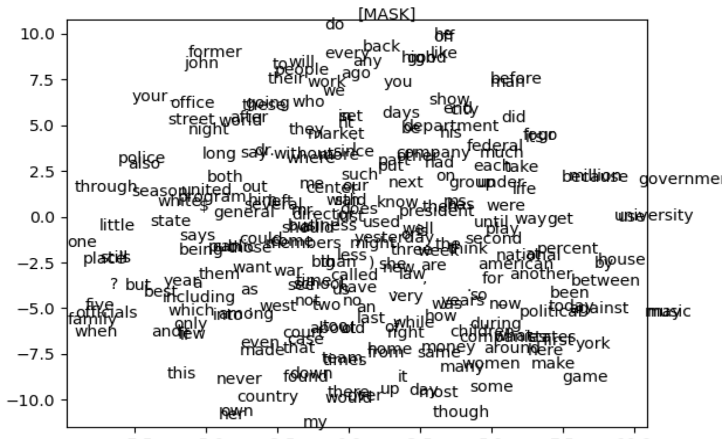


Figure 4: Trained Model

(b) Fig. 5 denotes the 2-dimensional visualization for the GloVe model. There are a lot of clusters. Like, (new, york, city), (can, could, would, should, will), (so, much, too), (several, very, few), (office, department, officials, program) etc. Here, also the words in each cluster share strongly related context, like similar prepositions are in one cluster, similar adverb in one cluster, etc. In comparison with the trained model, it looks like better in terms of number of clusters and relevance of words in clusters.

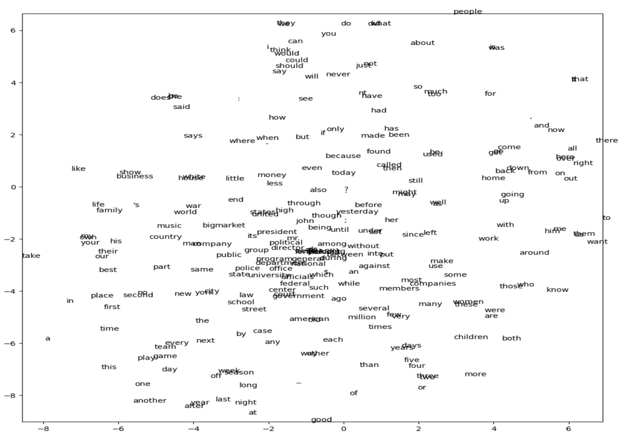


Figure 5: GLoVe (TSNE Plot)

(c) Fig. 6 shows the 2-dimensional visualization of the learned representation. It shows few but large clusters. Also, it looks like quite different from the t-SNE embeddings. Like, the very common words, like pronoun, preposition, article, auxiliary verbs, are not making clusters here. Rather, the uncommon and specific words are making clusters here.

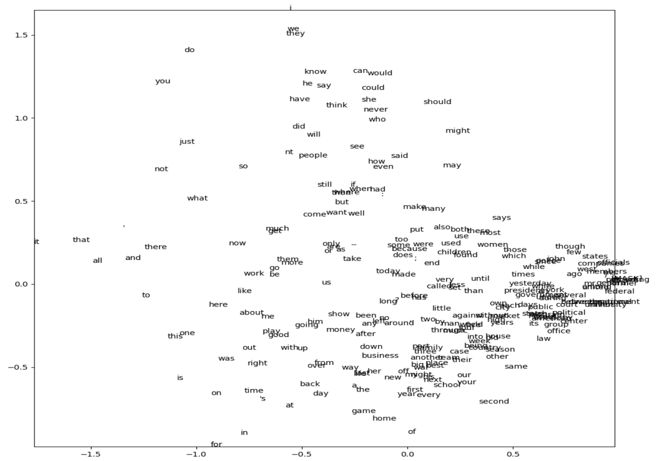


Figure 6: GLoVe (Learned Representation)

## 4.2 Analysis Continued: Word Analogy

### 4.2.1 1. he, him, her

1. **The closest word:**
   - Symmetric Embedding: *she*
   - Averaging Asymmetrical GLoVe Embedding: *she*
   - Concatenating Asymmetrical GLoVe Embedding: *she*
   - Neural Network Word Embedding: *here*
2. **Distance to the closest word:**
   - Symmetric Embedding: *1.41*
   - Averaging Asymmetrical GLoVe Embedding: *1.82*
   - Concatenating Asymmetrical GLoVe Embedding: *3.52*
   - Neural Network Word Embedding: *6.78*

In the tSNE plots (Fig. 4 and Fig. 5), these words form kind of a parallelogram structure for the gLoVe model, but no straight pattern is found for the neural network.

### 4.2.2 2. (Extra) four, two, three

1. **The closest word:**
   - Symmetric Embedding: *five*

  - Averaging Asymmetrical GLoVe Embedding: *five*
  - Concatenating Asymmetrical GLoVe Embedding: *five*
  - Neural Network Word Embedding: *five*

2. **Distance to the closest word:**
  - Symmetric Embedding: *0.79*
  - Averaging Asymmetrical GLoVe Embedding: *1.43*
  - Concatenating Asymmetrical GLoVe Embedding: *2.86*
  - Neural Network Word Embedding: *1.43*

In the tSNE plots (Fig. 4 and Fig. 5), these words form kind of a parallelogram shape for the gLoVe model.

## 5 Most Illuminating Part:

The word analogy things.

## 6 Most Difficult Part:

Backward and derivative things.