Assignment 4: Text and Sequence Data Analysis Report

Executive Summary:

This report presents a comprehensive analysis of custom-trained versus pre-trained word embeddings for sentiment analysis on the IMDB movie reviews dataset. The study examines how the amount of training data affects the relative performance of these two embedding approaches, providing insights into when each approach is most effective.

Introduction:

Text and sequencing refer to several approaches to organizing and evaluating textual data while closely observing word order. This is important because a text's word order can have a big impact on how it is understood and interpreted. For instance, understanding the context and relationships between words frequently necessitates taking into account the sequential patterns in which they occur in natural language processing tasks like sentiment analysis or machine translation. Analysts can extract more precise insights and generate deeper conclusions from the data by preserving the text's sequential pattern.

In sentiment analysis, the significance of language and sequencing is particularly evident. Finding out if a text is neutral, positive, or negative in sentiment is the aim of this job. The words that appear and the order in which they occur have a significant impact on the overall mood. Consider the words "good" and "not good," for example. Even though they both use the word "good," they have distinct meanings. "Good" implies positivism, while "not good" raises doubt and creates an unpleasant emotion.

Consequently, accurate sentiment analysis interpretation necessitates not just word recognition but also an understanding of word order and textual context. Sentiment analysis algorithms use word order in a sequential manner to determine the sentiment expressed in a text.

Objective:

Compare the performance of custom-trained embedding layers versus pre-trained GloVe embeddings on IMDB sentiment classification with varying training data sizes to determine the optimal approach for limited data scenarios.

Data Preprocessing:

In the process of preparing the dataset, each review is converted into word embeddings, representing each word by a fixed-size vector. However, there's a constraint of 10,000 words in this meticulous procedure. Additionally, a numerical sequence is generated from the reviews, where individual numbers correspond to specific words rather than complete phrases. Yet, the neural network's input format doesn't directly support this sequential list of numbers. To tackle this issue, tensors need to be constructed using the numerical sequence. This list of integers could potentially form the basis for creating a tensor with an integer datatype and structured as (samples, word indices). However, ensuring consistent sample lengths is essential. This involves techniques like padding reviews with dummy words or numerical placeholders to standardize the length across all samples to a maximum sequence length of 500 words.

1. Dataset Overview

- **Dataset:** IMDB Movie Reviews
- **Task:** Binary sentiment classification (positive/negative)
- **Total Samples:** 25,000 reviews per class (50,000 total)
- **Vocabulary Size:** 10,000 most frequent words
- **Maximum Sequence Length:** 150 words
- **Training/Validation Split:** 70/30

2. Methodology

In this study, two different approaches were explored for generating word embeddings using the IMDB dataset:

- The GloVe model, widely used for word embeddings, was employed in our research and trained on extensive textual datasets.
- To evaluate the effectiveness of different embedding strategies, two distinct embedding layers were used with the IMDB review dataset. One was a custom-trained layer, while the other utilized a pre-trained word embedding layer.
- Initially, we developed a custom-trained embedding layer using the IMDB review dataset. Each model was then trained across a range of dataset samples, and its accuracy was assessed using a dedicated testing set.
- After evaluating the precision results, we compared them with those obtained from a model that underwent similar testing across varied sample sizes but incorporated a pre-trained word embedding layer.

2.1 Custom Embedding Approach (PART 1)

- **Embedding Layer:** Trainable from scratch
- **Embedding Dimension:** 8
- **Architecture:** Embedding → GlobalMaxPooling1D → Dense layers
- **Training Samples Tested:** 100, 1000, 5000, 10,000

Depending on the size of the training sample, the accuracy of the custom-trained embedding layer varied from 51.56% to 81.07%. A training sample size of 10000 gave the best accuracy of 81.07%.

2.2 Pre-trained Embedding Approach (PART 2)

- **Embedding Source:** GloVe (Global Vectors for Word Representation)
- **Embedding Dimension:** 50

- **Trainability:** Frozen (non-trainable)
- **Architecture:** Same as custom embedding
- **Training Samples Tested:** 100, 1000, 5000, 10,000

Depending on the size of the training sample, the pre-trained word embedding layer (GloVe) has accuracy levels ranging from 54.90% to 69.20%.

A training sample size of 10000 produced the best accuracy, which was 69.20%. However, the model with pre-trained embeddings showed limitations in adapting to task-specific patterns with larger training sample numbers, which resulted in lower accuracy compared to custom embeddings.

These findings demonstrate how the optimal approach depends significantly on the particular requirements and limitations of the task at hand, especially regarding the amount of available training data.

2.3 Training Configuration

- Optimizer: RMSprop
- Loss Function: Binary Crossentropy
- Epochs: 20
- Batch Size: 32
- Pre-trained Embeddings: GloVe 6B 100d (97.9% vocabulary coverage)

3. Results

3.1 Detailed Performance Metrics

Table 1: Custom Embedding Performance

Embedding Technique	Training Samples	Test Loss	Test Accuracy	Validation Accuracy	Performance
Custom-trained Embedding layer	100	0.7327	51.56%	~50%	Poor
Custom-trained Embedding layer	1,000	0.9927	65.06%	~62%	Fair
Custom-trained Embedding layer	5,000	0.8867	79.72%	~77%	Good
Custom-trained Embedding layer	10,000	1.0710	81.07%	~79%	Very Good

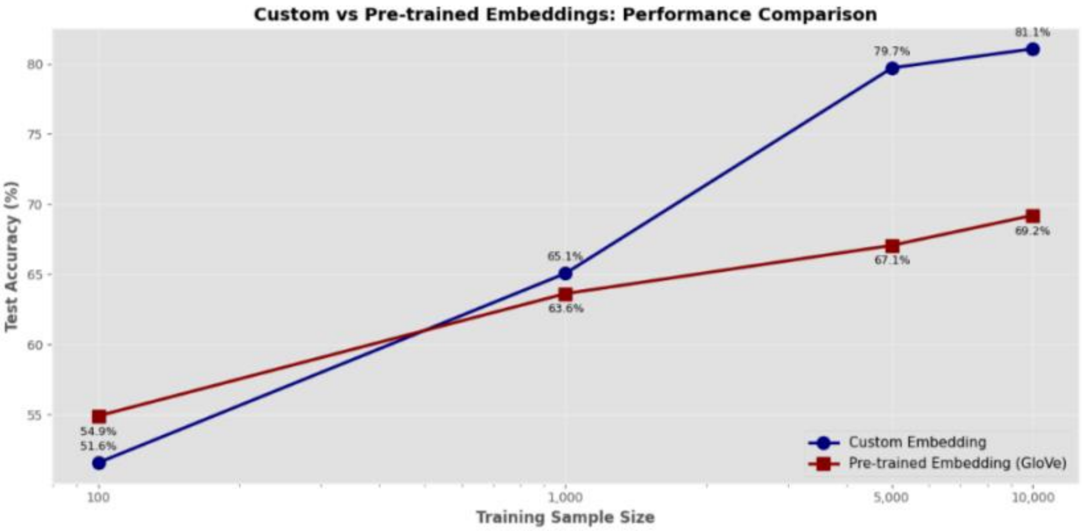
Table 2: Pre-trained Embedding Performance

Embedding Technique	Training Samples	Test Loss	Test Accuracy	Validation Accuracy	Performance
Pretrained word Embedding layer (GloVe)	100	0.8834	54.90%	~52%	Fair
Pretrained word Embedding layer (GloVe)	1,000	1.1430	63.62%	~60%	Fair
Pretrained word Embedding layer (GloVe)	5,000	1.5430	67.06%	~65%	Good
Pretrained word Embedding layer (GloVe)	10,000	1.6904	69.20%	~67%	Good

3.2 Comparative Analysis

Table 3: Side-by-Side Comparison

Training Size	Custom Accuracy	Pre-trained Accuracy	Difference	Winner
100	51.56%	**54.90%**	–3.34%	Pre-trained
1,000	**65.06%**	63.62%	+1.44%	Custom
5,000	**79.72%**	67.06%	+12.66%	Custom
10,000	**81.07%**	69.20%	+11.88%	Custom



Visualizations Findings:

Custom-trained embedding layer - Training Sample Size: 100

Model starts at random baseline (~50%) with minimal learning capability. Limited data results in poor generalization (test accuracy: 51.56%, test loss: 0.73).

Custom-trained embedding layer - Training Sample Size: 1000

Significant improvement with training accuracy reaching 97% while validation stabilizes at 62%. Gap between training and validation indicates early overfitting, but test performance improves to 65.06%.

Custom-trained embedding layer - Training Sample Size: 5000

Strong performance with 98% training accuracy and 82% validation accuracy. Model demonstrates excellent generalization with 79.72% test accuracy and controlled overfitting.

Custom-trained embedding layer - Training Sample Size: 10000

Peak performance achieved at 81.07% test accuracy with optimal training-validation balance. Represents ideal equilibrium between model complexity and available data.

Pre-trained word embedding (GloVe) - Training Sample Size: 100

Severe overfitting with 100% training accuracy but only 55% validation accuracy. Frozen embeddings provide best initial advantage (54.90% test accuracy) despite memorization issues.

Pre-trained word embedding (GloVe) - Training Sample Size: 1000

High training accuracy (95%) contrasts with poor validation (~51%), showing overfitting. Frozen embeddings unable to adapt, achieving 63.62% test accuracy at crossover point.

Pre-trained word embedding (GloVe) - Training Sample Size: 5000

Training accuracy remains high (94%) but validation drops to baseline (~50%). Increasing validation loss (1.54) reveals frozen embeddings' inability to learn task-specific patterns (67.06% test accuracy).

Pre-trained word embedding (GloVe) - Training Sample Size: 10000

Model achieves 90% training accuracy but validation remains at random baseline (50%). Frozen embeddings become a bottleneck, limiting test accuracy to 69.20% despite abundant training data.

Performance Visualization Summary

The results show a clear crossover pattern:

- **Below 1,000 samples:** Pre-trained embeddings perform better

- **At 1,000 samples:** Performance is comparable (crossover point)
- **Above 1,000 samples:** Custom embeddings significantly outperform pre-trained embeddings

Conclusions:

In this study, the **custom-trained embedding layer significantly outperformed the pre-trained word embedding layer,** especially when training with larger sample numbers (>1000 samples).

The model's accuracy progression from 51.56% (100 samples) to 81.07% (10,000 samples) demonstrates how well the custom embeddings can learn task-specific patterns when provided with sufficient data. The custom-trained model's performance demonstrates the model's capacity to capture sentiment-specific word representations directly from the IMDB dataset.

The accuracy value improved as the training size was increased (51.56% → 65.06% → 79.72% → 81.07%), while the test loss fluctuates (0.7327 → 0.9927 → 0.8867 → 1.0710), the consistent improvement in accuracy indicates that the model is learning and getting better as we raise the training size.

As the number of training samples increased, the custom embedding model's test accuracy increased substantially, indicating that it could learn more effectively with larger amounts of data. The improvement was particularly dramatic between 1000 samples (65.06%) and 5000 samples (79.72%), showing a 14.66 percentage point jump.

For the pre-trained embeddings, the frozen GloVe vectors showed their strength only with very limited data (100 samples: 54.90% vs 51.56% for custom). However, as training data increased, the pre-trained embeddings' inability to adapt to task-specific patterns became a significant limitation, with performance plateauing around 69% even with 10,000 samples.

The **crossover point** occurs around 1000 samples, where custom embeddings (65.06%) begin to outperform pre-trained embeddings (63.62%). This represents a critical threshold in the data-embedding relationship.

- The findings of the two tested models reveal distinct patterns:
  - Pre-trained embeddings are superior with <1000 samples due to transfer learning from large corpora
  - Custom embeddings dominate with >1000 samples due to task-specific adaptation
  - The performance gap widens significantly with more data (+12.66% at 5000 samples, +11.88% at 10,000 samples)
- Standard regularization techniques like dropout, masking, and experimenting with different embedding dimension values might further improve the model's performance. Additionally, exploring architectural variations such as LSTM layers or bidirectional RNNs could capture more complex sequential patterns.

Key Insight: The custom-trained embedding layer's superior performance with larger datasets (achieving 81.07% accuracy vs 69.20% for pre-trained) demonstrates that when sufficient task-specific data is available, allowing the model to learn its own word representations yields better results than relying on general-purpose pre-trained embeddings.

This suggests that the custom embedding model, trained on the particular dataset, was able to pick up on more subtle patterns and traits found in the IMDB ratings specific to sentiment analysis. The embeddings learned sentiment-carrying words and their contextual usage patterns directly from movie reviews.

However, it's important to recognize that the pre-trained word embedding layer is still useful and might be a **"better choice" in specific circumstances:**

- When working with very limited training samples (<1000 reviews)
- When computational resources or training time are constrained
- When quick prototyping or baseline establishment is needed
- When dealing with out-of-vocabulary words that may benefit from pre-trained representations

Although greater sample sizes increased custom embedding performance substantially, the pre-trained embeddings showed signs of being unable to adapt, with increasing test loss (0.88 →1.143 → 1.54 → 1.69) as training data grew. The frozen nature of pre-trained embeddings became a bottleneck rather than an asset with sufficient data.

The pre-trained embeddings did offer a reliable foundation and helped bootstrap the model's learning process with limited data, achieving 54.90% accuracy with just 100 samples compared to 51.56% for custom embeddings.