INLP Assignment - 4 Report

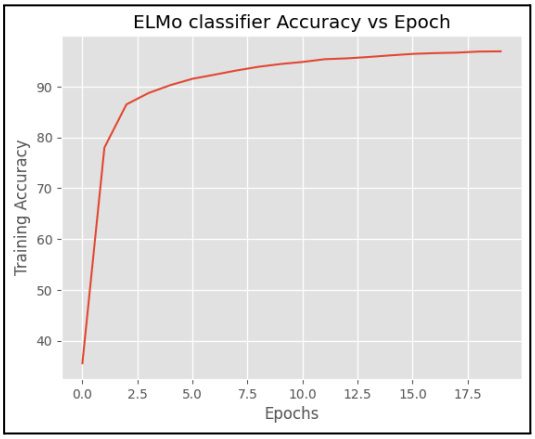
Ishan Kavathekar

2022121003

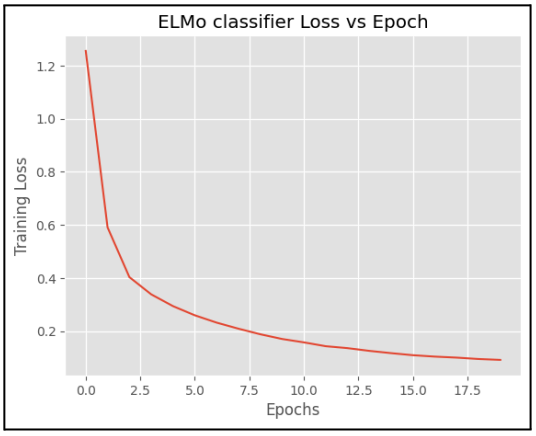
Downstream classification task:

The embedding size and the hidden size is 300. This can be changed by changing the input in the scripts.

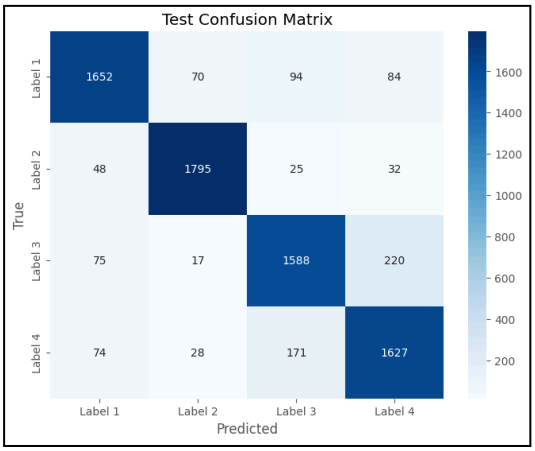
1. Relu



Training Accuracy



Training Loss

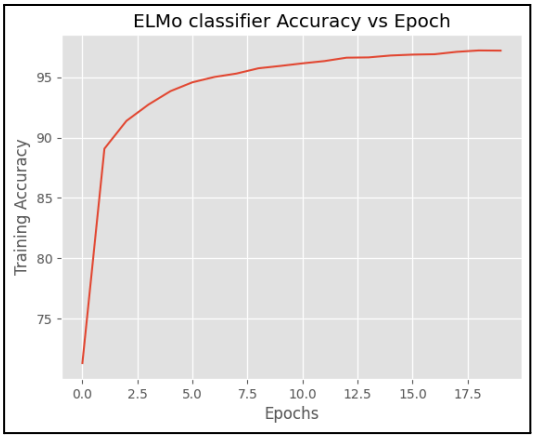


Test Confusion Matrix

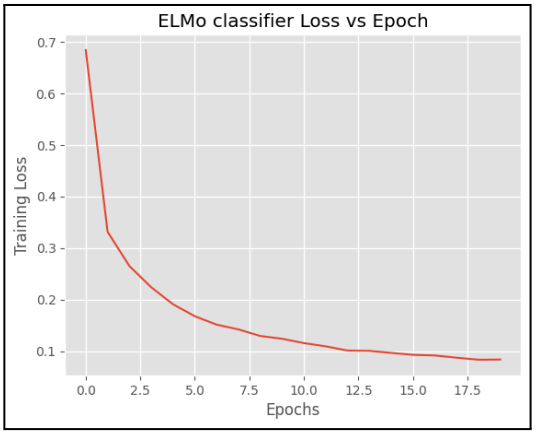
	precision	recall	f1-score	support
0	0.89	0.87	0.88	1900
1	0.94	0.94	0.94	1900
2	0.85	0.84	0.84	1900
3	0.83	0.86	0.84	1900
accuracy			0.88	7600
macro avg	0.88	0.88	0.88	7600
weighted avg	0.88	0.88	0.88	7600

Classification Report

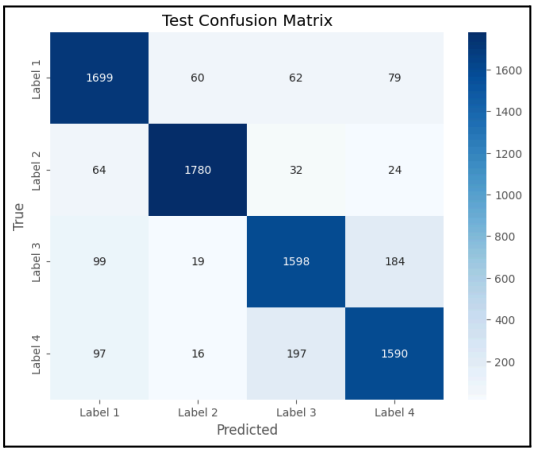
2. Tanh



Training Accuracy



Training Loss

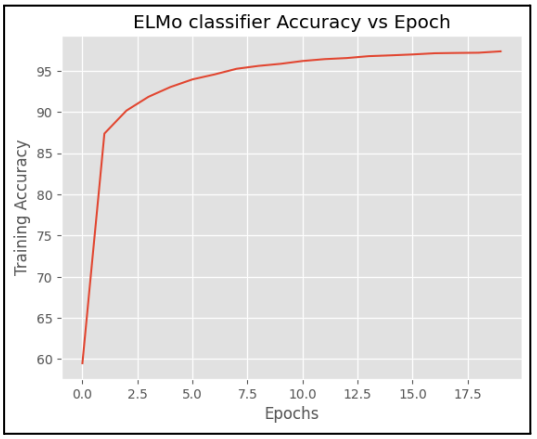


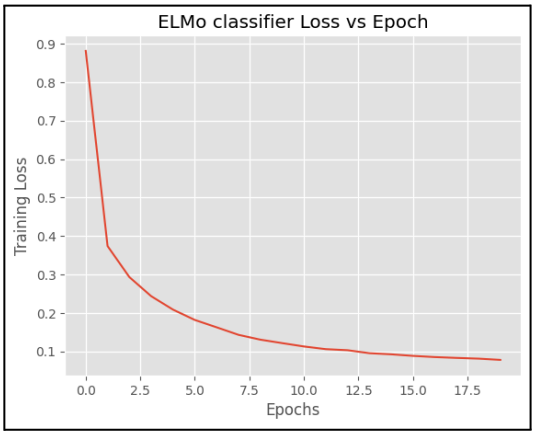
Test Confusion Matrix

	precision	recall	f1-score	support
0	0.87	0.89	0.88	1900
1	0.95	0.94	0.94	1900
2	0.85	0.84	0.84	1900
3	0.85	0.84	0.84	1900
accuracy			0.88	7600
macro avg	0.88	0.88	0.88	7600
weighted avg	0.88	0.88	0.88	7600

Classification Report

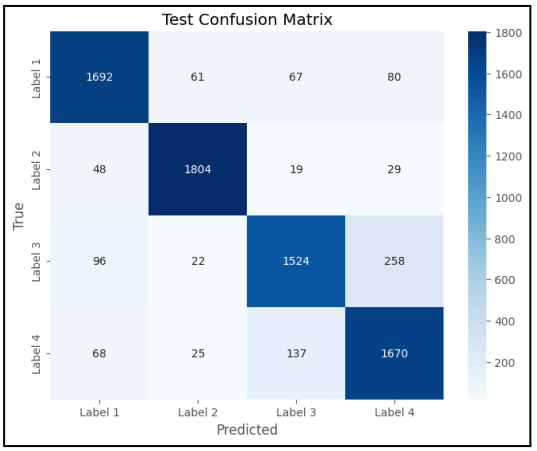
3. Train weights





Training Accuracy

Training Loss

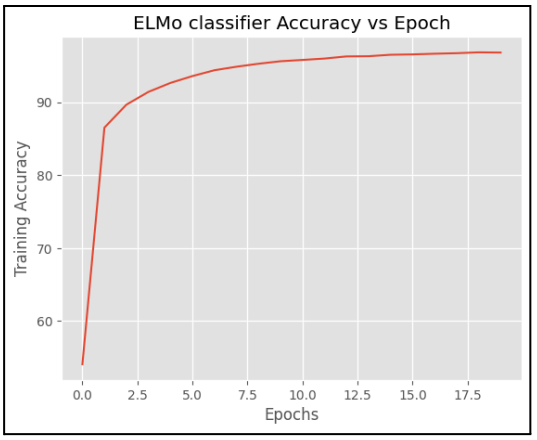


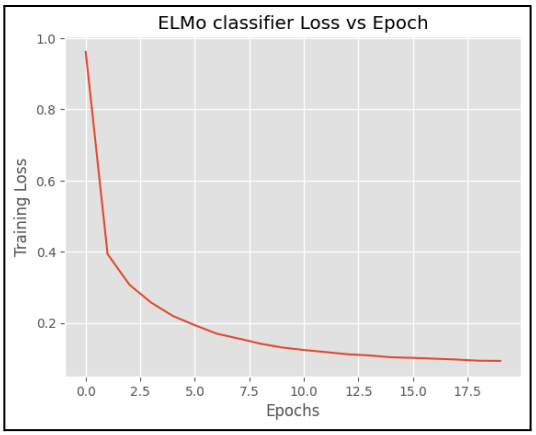
	precision	recall	f1-score	support
0	0.89	0.89	0.89	1900
1	0.94	0.95	0.95	1900
2	0.87	0.80	0.84	1900
3	0.82	0.88	0.85	1900
accuracy			0.88	7600
macro avg	0.88	0.88	0.88	7600
weighted avg	0.88	0.88	0.88	7600

Test Confusion Matrix

Classification Report

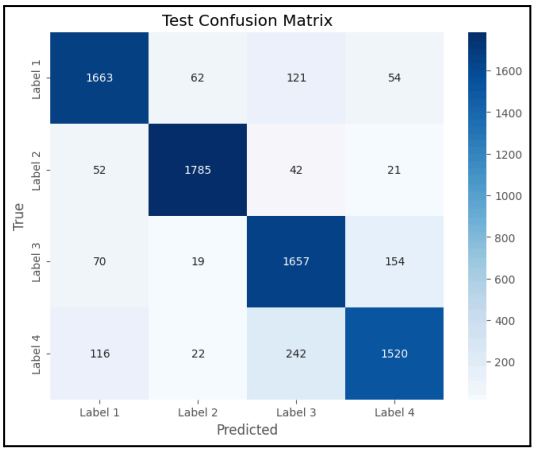
4. Frozen weights





Training Accuracy

Training Loss



Test Confusion Matrix

	precision	recall	f1-score	support
0	0.87	0.88	0.88	1900
1	0.95	0.94	0.94	1900
2	0.80	0.87	0.84	1900
3	0.87	0.80	0.83	1900
accuracy			0.87	7600
macro avg	0.87	0.87	0.87	7600
weighted avg	0.87	0.87	0.87	7600

Classification Report

Hyperparameters:

Learning rate: 0.001
Weight decay: 1e-5
Number of epochs: 20
The model with ReLU function with trainable weights performs the best on the training set. It reaches 97.4% of accuracy. The others models perform quite similarly for the hyperparameters. They attain a accuracy level of 88% with slight modifications.

Comparison between Word2Vec, SVD (Singular Value Decomposition), and ELMo (Embeddings from Language Models):

Word2Vec with skip gram with negative sampling employs a supervised learning methodology to learn distributed representations of words in a continuous vector space. It captures semantic relationships between words by representing each word as a dense vector. Word2Vec excels in capturing syntactic and semantic similarities among words, making it useful for tasks like word analogy and similarity detection. However, it struggles with capturing context-dependent meanings and rare words due to its reliance on local context.

On the other hand, SVD is a classical matrix factorization technique that decomposes a term-document matrix into three matrices: one capturing word semantics, one capturing document semantics, and one capturing the relationship between them. SVD-based vectorization tends to perform well in capturing global statistical properties of the data and is particularly effective with large, sparse datasets. However, it may not capture complex semantic relationships as effectively as neural network-based methods like Word2Vec and ELMo.

ELMo, a more recent addition to word vectorization methods, utilizes deep contextualized word representations. It leverages a deep bidirectional language model to generate word embeddings that capture both syntax and semantics in context. ELMo excels in capturing contextual information and handling polysemy, making it suitable for tasks where word meaning depends heavily on context, such as sentiment analysis and named entity recognition. However, its computational complexity and resource requirements may limit its applicability in certain contexts.

Comparing the performance of these methods depends on various factors such as the specific task, dataset size, and available computational resources. In tasks where context is crucial, such as text classification and sentiment analysis, ELMo often outperforms Word2Vec and SVD due to its ability to capture contextual information. Here, we observe that the Word2vec and SVD embeddings perform better than the ELMo embeddings but it can be attributed to the dataset quality or size. However, for simpler tasks or when computational resources are limited, Word2Vec may be preferred for its efficiency and effectiveness in capturing word similarities.