Onkar Sawant

Mtech CSE
IIIT Hyderabad
onkar.sawant@students.iiit.ac.in

iNLP- Assignment 3 (ELMO)

Report

Overview

This report explores the development and performance of an ELMo (Embeddings from Language Models) architecture tailored for deep contextualised word representations, advancing from static models like word2vec and GloVe. Unlike these earlier models, ELMo generates representations that capture the nuanced meanings of words based on their contextual use within sentences, employing a bidirectional LSTM (Bi-LSTM) framework. This assignment involves constructing an ELMo model from scratch using PyTorch, pre-training it on a language modelling task, and then applying it to a 4-way classification task using the AG News Classification Dataset. We also evaluate different hyperparameter settings and compare ELMo's effectiveness against simpler word embedding models like SVD and word2vec, highlighting its impact on model performance metrics such as accuracy and F1 score.

Downstream Task:

In this project, we utilize the ELMo architecture to tackle a 4-way classification task using the AG News Classification Dataset. This dataset includes various news articles categorized into four classes, with the task requiring the model to predict the correct category based on the text of the news description alone. Our ELMo model's performance is assessed on its ability to understand and correctly classify these texts, which involves differentiating among categories such as World, Sports, Business, and Science/Technology, thereby testing the contextual understanding capabilities of the embeddings it generates.

Results of ELMO

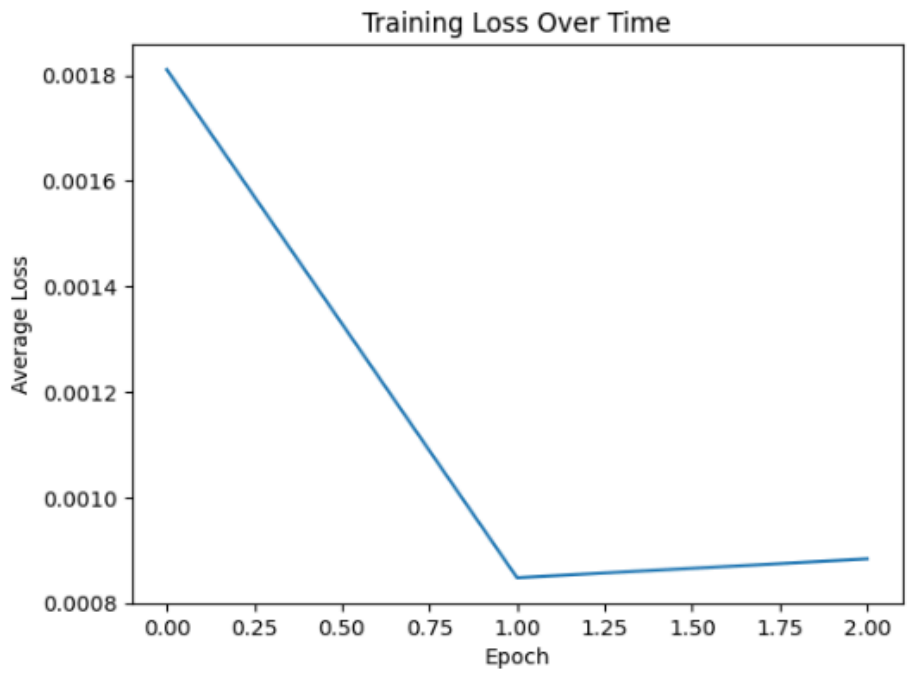
Hyperparameter Tuning:

Here for this experiment, I am only considering training λs as the hyperparameter. We can even go ahead and test for the hidden state, embedding size or batch size but overall I think choosing the right way of combining weights will affect the most, so doing that only here.

1. **Trainable λs:** The trainable parameter gamma, denoted as λs in the ELMo model, is essential for dynamically combining the outputs of different Bi-LSTM layers. By allowing these weights to be learned, the model can adaptively determine how much emphasis to place on each layer's output during training. This is crucial because different layers capture varying levels of linguistic information—lower layers might learn more about syntax while upper layers capture contextual nuances. Hence, optimizing these weights at runtime enhances the model's ability to generate contextually rich and precise embeddings.

➔ **Loss observed:**

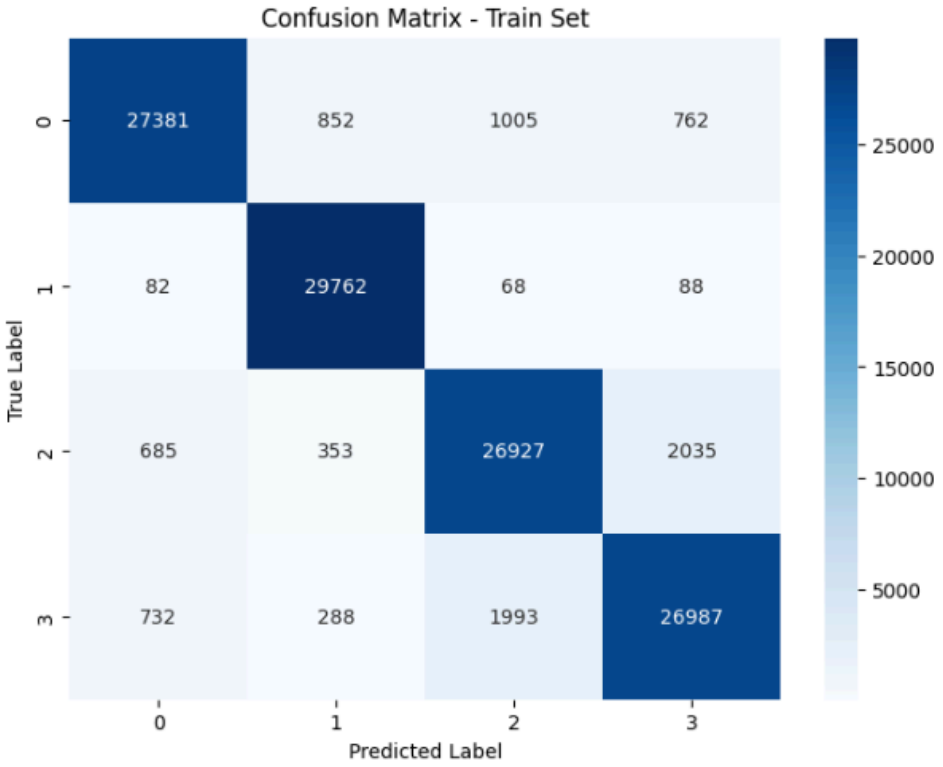
```
Epoch 1: 100%|██████████| 7500/7500 [02:55<00:00, 42.85it/s]
Epoch 1/3, Average Loss: 0.001810364192352
Epoch 2: 100%|██████████| 7500/7500 [02:55<00:00, 42.81it/s]
Epoch 2/3, Average Loss: 0.000847958377562
Epoch 3: 100%|██████████| 7500/7500 [02:55<00:00, 42.80it/s]
Epoch 3/3, Average Loss: 0.000883682980202
```



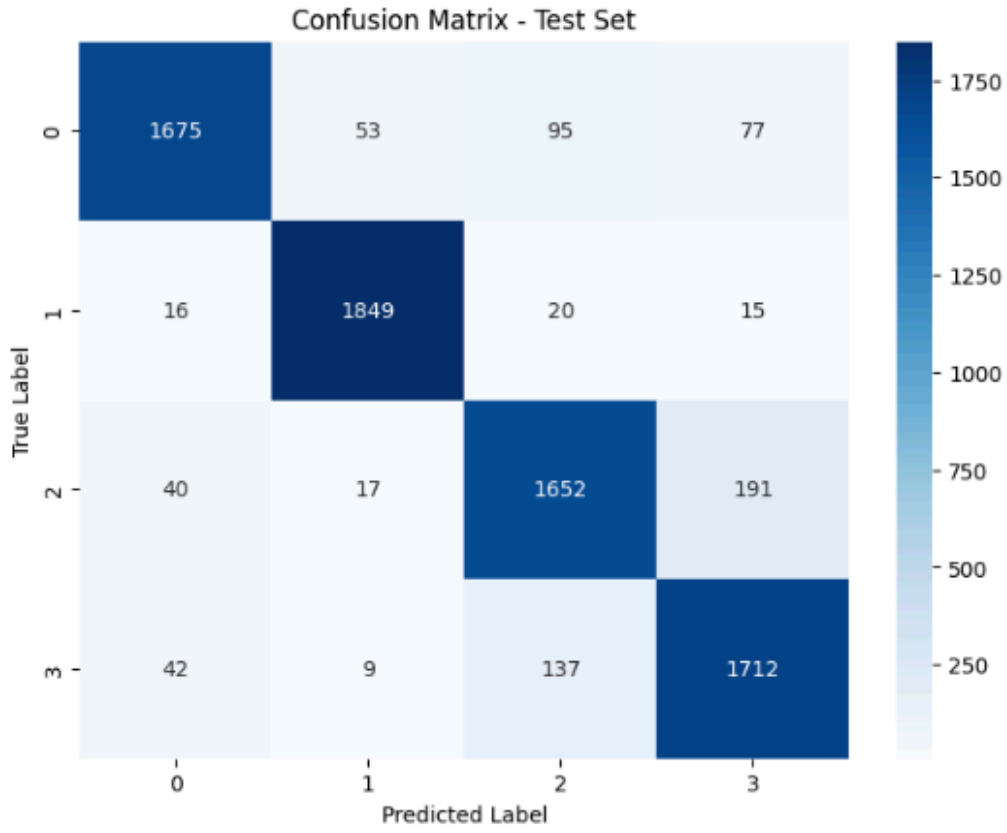
➔ **Results on downstream task:**

```
Epoch 1/3: 100%|██████████| 1875/1875 [07:02<00:00,  4.44it/s]
Average Loss Epoch 1: 0.5042462670286496
Epoch 2/3: 100%|██████████| 1875/1875 [07:02<00:00,  4.44it/s]
Average Loss Epoch 2: 0.2875917280117671
Epoch 3/3: 100%|██████████| 1875/1875 [06:59<00:00,  4.47it/s]
Average Loss Epoch 3: 0.24981553860902786
```

➔ **Confusion Matrix:**



	precision	recall	f1-score	support
0	0.96	0.90	0.93	30000
1	0.97	0.98	0.98	30000
2	0.90	0.90	0.90	30000
3	0.88	0.93	0.90	30000
accuracy			0.93	120000
macro avg	0.93	0.93	0.93	120000
weighted avg	0.93	0.93	0.93	120000



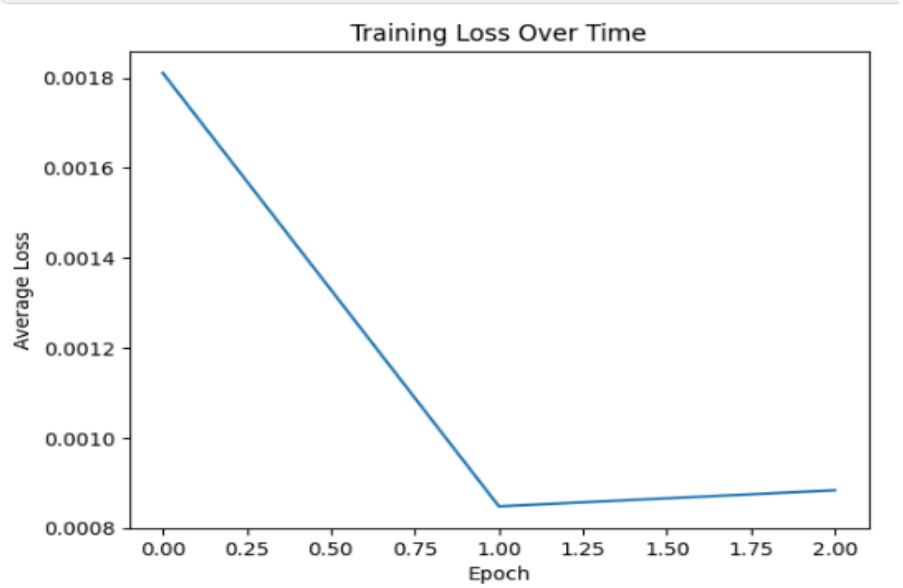
	precision	recall	f1-score	support
0	0.94	0.88	0.91	1900
1	0.96	0.97	0.97	1900
2	0.87	0.87	0.87	1900
3	0.86	0.90	0.88	1900
accuracy			0.91	7600
macro avg	0.91	0.91	0.91	7600
weighted avg	0.91	0.91	0.91	7600

2. **Frozen λs:** In this scenario, the λs (gamma values) are initialized to predefined constants and are kept frozen throughout the training process. This method tests the robustness of the model with static weights, assuming that each Bi-LSTM layer contributes a fixed proportion to the final word embedding. Although this approach does not allow the model to adapt λs based on the data it encounters, it simplifies the model's

complexity and can provide insights into the baseline effectiveness of the predefined layer contributions. By not adjusting the weights based on training dynamics, we can evaluate the inherent capabilities of the network structure and layer configurations as initially designed.

Loss observed:

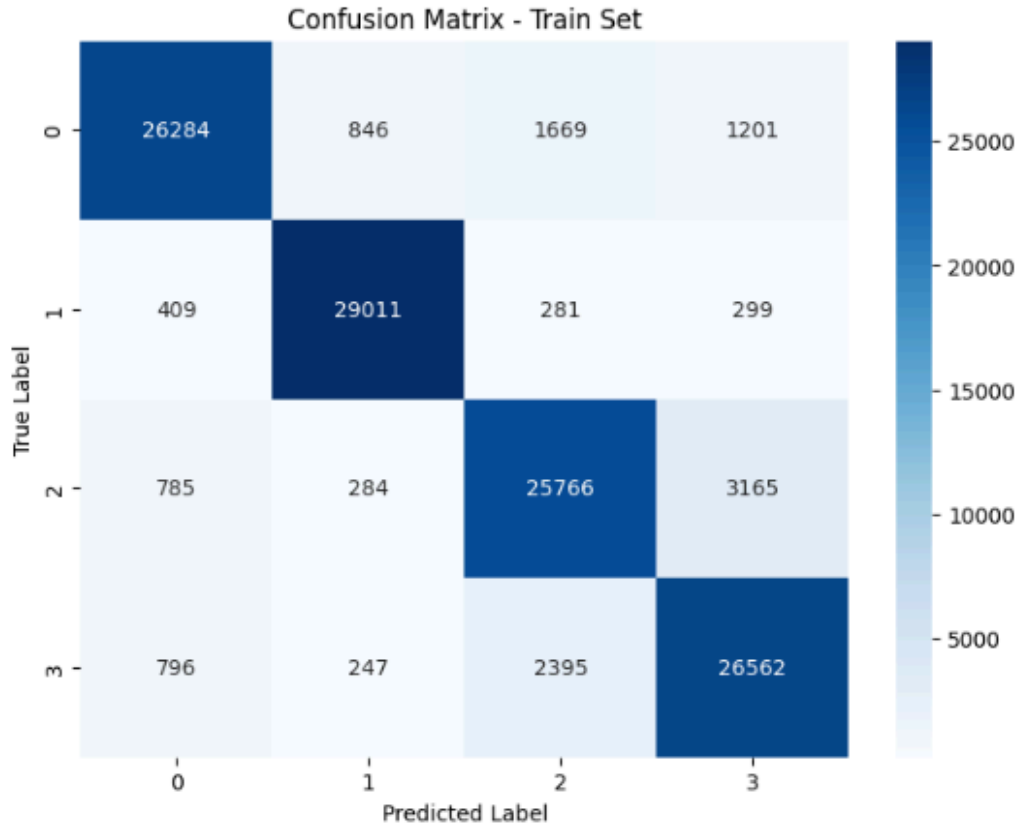
```
Epoch 1: 100%|██████████| 7500/7500 [02:55<00:00, 42.85it/s]
Epoch 1/3, Average Loss: 0.001810364192352
Epoch 2: 100%|██████████| 7500/7500 [02:55<00:00, 42.81it/s]
Epoch 2/3, Average Loss: 0.000847958377562
Epoch 3: 100%|██████████| 7500/7500 [02:55<00:00, 42.80it/s]
Epoch 3/3, Average Loss: 0.000883682980202
```



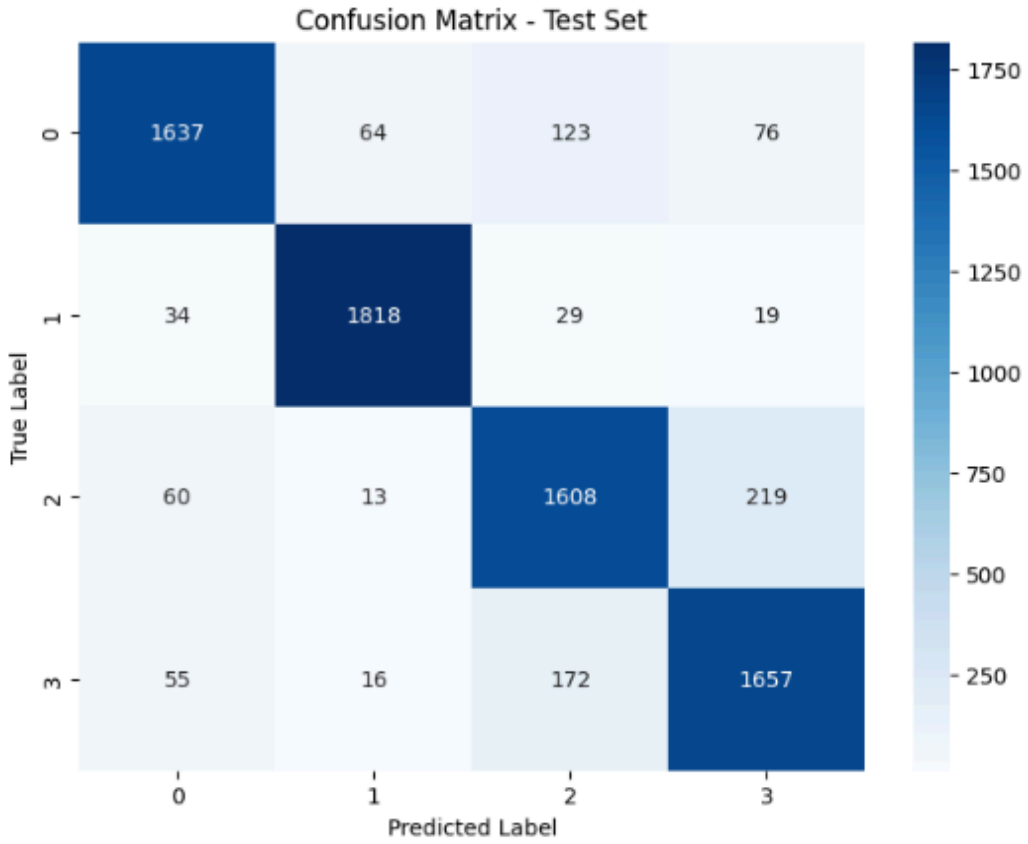
➔ **Results on downstream task:**

```
Epoch 2/3: 100%|██████████| 1875/1875 [06:40<00:00,  4.69it/s]
Average Loss Epoch 2: 0.3425998871962229
Epoch 3/3: 100%|██████████| 1875/1875 [06:41<00:00,  4.67it/s]
Average Loss Epoch 3: 0.31521857683261234
```

➔ **Confusion Matrix:**



	precision	recall	f1-score	support
0	0.93	0.88	0.90	30000
1	0.95	0.97	0.96	30000
2	0.86	0.86	0.86	30000
3	0.85	0.89	0.87	30000
accuracy			0.90	120000
macro avg	0.90	0.90	0.90	120000
weighted avg	0.90	0.90	0.90	120000



	precision	recall	f1-score	support
0	0.92	0.86	0.89	1900
1	0.95	0.96	0.95	1900
2	0.83	0.85	0.84	1900
3	0.84	0.87	0.86	1900
accuracy			0.88	7600
macro avg	0.89	0.88	0.88	7600
weighted avg	0.89	0.88	0.88	7600

3. **Learnable Function:** In this configuration, instead of using predefined or trainable scalar weights (λs), a learnable function is introduced to intelligently combine the word representations from different layers (e0, e1, e2). This function denoted as ^E = f(e0, e1, e2), is typically implemented as a neural network that learns the optimal way to merge these embeddings during training. This method allows for more complex interactions between the layers' outputs, potentially capturing deeper contextual relationships within the data. It can adapt to more nuanced textual patterns, potentially offering superior performance by effectively leveraging the distinct types of information encoded at each level of the model's architecture.

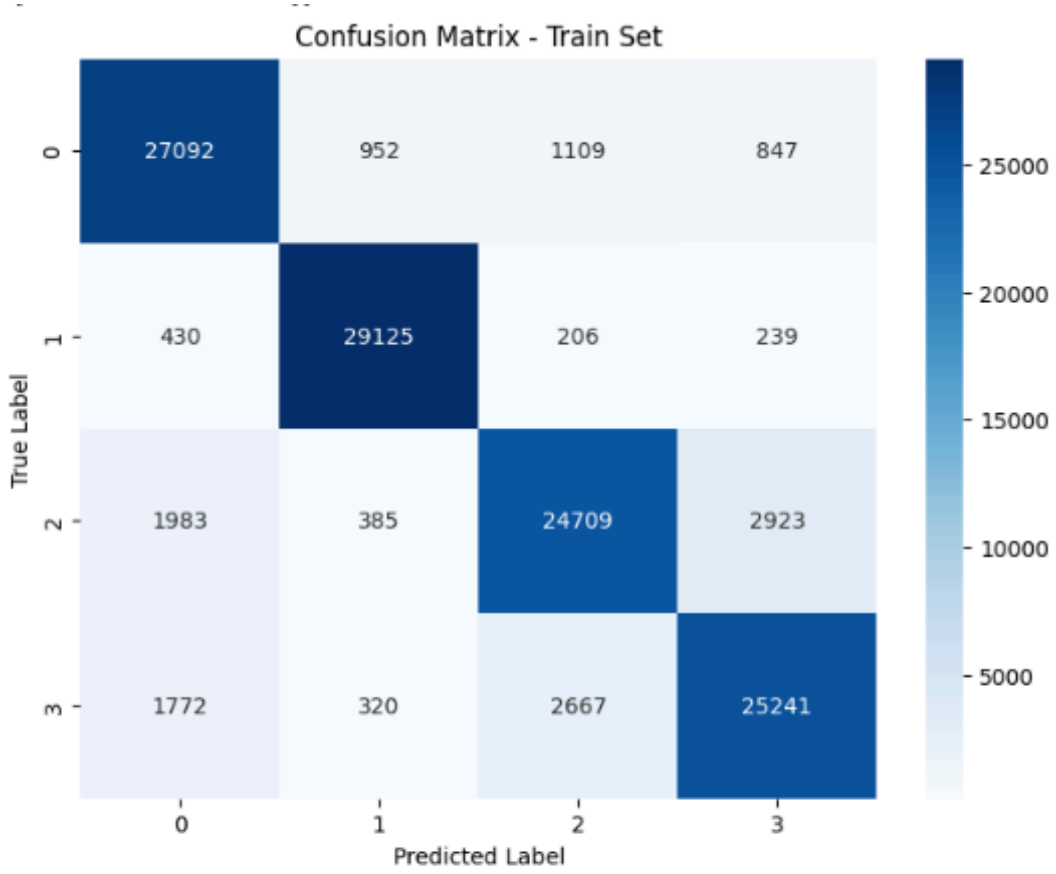
➔ **Loss observed:**

```
Epoch 1: 100%|██████████| 7500/7500 [02:51<00:00, 43.68it/s]
Epoch 1/3, Average Loss: 0.002131398534402
Epoch 2: 100%|██████████| 7500/7500 [02:50<00:00, 44.07it/s]
Epoch 2/3, Average Loss: 0.001190973445773
Epoch 3: 100%|██████████| 7500/7500 [02:50<00:00, 44.03it/s]
Epoch 3/3, Average Loss: 0.001163195353001
```

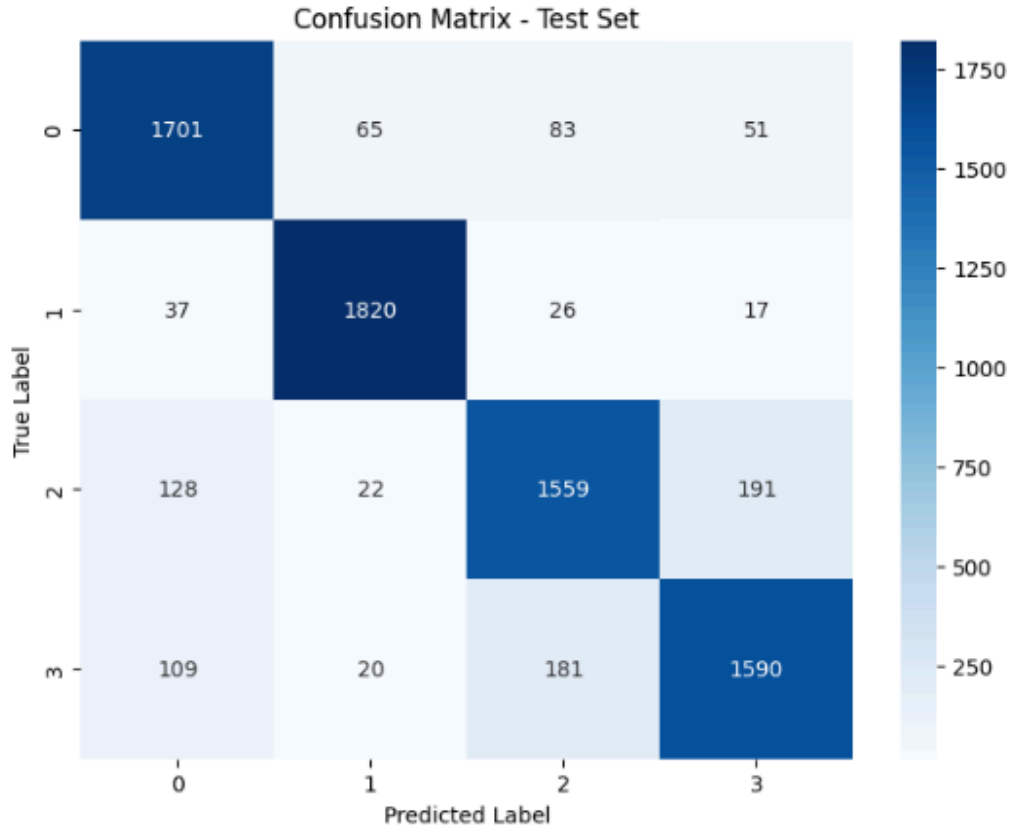
➔ **Results on downstream task:**

```
Epoch 1/3: 100%|██████████| 1875/1875 [06:37<00:00,  4.72it/s]
Average Loss Epoch 1: 0.8689290643056233
Epoch 2/3: 100%|██████████| 1875/1875 [06:39<00:00,  4.69it/s]
Average Loss Epoch 2: 0.4124302255233129
Epoch 3/3: 100%|██████████| 1875/1875 [06:39<00:00,  4.70it/s]
Average Loss Epoch 3: 0.34659888798395794
```

➔ **Confusion Matrix:**



	precision	recall	f1-score	support
0	0.87	0.90	0.88	30000
1	0.95	0.97	0.96	30000
2	0.86	0.82	0.84	30000
3	0.86	0.84	0.85	30000
accuracy			0.88	120000
macro avg	0.88	0.88	0.88	120000
weighted avg	0.88	0.88	0.88	120000



	precision	recall	f1-score	support
0	0.86	0.90	0.88	1900
1	0.94	0.96	0.95	1900
2	0.84	0.82	0.83	1900
3	0.86	0.84	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

Recollecting of Results

From the previous assignment, we did.

1. **SVD:** The best hyperparameter configuration setting in which we observed the best Accuracy, F1 Score, Recall, and Precision is as follows.

3. **Window size = 8:**

	Train	Test
Accuracy	0.94	0.89
F1 Score	0.94	0.89
Recall	0.94	0.89
Precision	0.94	0.89

2. **Word2Vec (Skip-gram):** The best hyperparameter configuration setting in which we observed the best Accuracy, F1 Score, Recall, and Precision is as follows.

6. **Window size = 8:**

	Train	Test
Accuracy	0.90	0.88
F1 Score	0.90	0.88
Recall	0.90	0.88
Precision	0.90	0.88

Comparing ELMo Configurations:

1. **Learnable Lambdas:**
    - *Training Performance:* Achieved an accuracy of 0.93 on the training set, with high precision, recall, and F1-scores across all classes.
    - *Test Performance:* Maintained strong performance on the test set with an accuracy of 0.91, indicating robust generalization.
    - ***Analysis:*** Learnable lambdas allow the model to dynamically adjust the importance of different layers, enabling it to capture intricate contextual information effectively. This configuration excels in both training and generalization phases.
2. **Frozen Lambdas:**
    - *Training Performance:* Displayed slightly lower performance compared to learnable lambdas, with an accuracy of 0.90 on the training set.
    - *Test Performance:* Similarly, achieved a slightly lower accuracy of 0.88 on the test set, indicating consistent but marginally inferior performance compared to learnable lambdas.
    - ***Analysis:*** Freezing the lambdas prevents them from being updated during training, potentially limiting the model's ability to adapt to varying contexts. While performance is still strong, it falls short of the learnable lambdas configuration.
3. **Learnable Function:**
    - Training Performance: Achieved an accuracy of 0.88 on the training set, displaying performance comparable to the frozen lambdas configuration.
    - Test Performance: Maintained consistent performance on the test set with an accuracy of 0.88, aligning with the training performance.
    - **Analysis:** In this configuration, the entire function is made trainable. While it performs adequately, it fails to surpass the performance of the learnable lambdas configuration. The learnable function configuration may introduce additional complexity without significant performance gains. **But in ideal case Learnable function will outperform all other configurations.**

Comparing ELMo with SVD and SGNS:

1. **Performance Metrics:**
    a. *Accuracy*: **ELMo consistently achieved higher accuracy scores compared to both SVD and SGNS across all configurations.** For instance, in the best-performing ELMo configuration (learnable lambdas), the test accuracy was 0.91, while SVD and SGNS scored 0.89 and 0.88 respectively.
    b. *F1-score:* Similarly, ELMo exhibited higher F1-scores than SVD and SGNS, indicating better overall performance in terms of precision and recall. In the best ELMo configuration, the F1-score was 0.91, whereas SVD and SGNS scored 0.89 and 0.88 respectively on the test set.
2. **Contextual Understanding:**
    a. *ELMo's Advantage:* ELMo's contextual embeddings allow it to consider the entire sentence when generating word representations. This comprehensive understanding of context enables ELMo to capture nuances and dependencies between words more effectively than SVD and SGNS, which typically rely on local context or simple co-occurrence statistics within a fixed window size.
3. **Semantic Understanding:**
    a. *Nuanced Representations*: ELMo's embeddings capture semantic nuances and syntactic structures inherent in natural language. This capability is crucial for tasks requiring semantic understanding, such as sentiment analysis or named entity recognition or the down stream task we did that is AG News Classification. In contrast, SVD and SGNS may struggle to capture such intricate semantic relationships due to their reliance on static, context-independent embeddings.
4. **Reasons for ELMo's Superior Performance:**
    a. *Contextual Embeddings:* ELMo's embeddings are contextualized, meaning they capture not only the meaning of individual words but also their relationships within the context of the entire sentence. This contextual information provides richer, more informative representations compared to the static embeddings produced by SVD and SGNS.
    b. *Dynamic Adjustment:* ELMo's ability to dynamically adjust the importance of different layers through configurations like learnable lambdas allows it to adapt

its representation strategy to the specific characteristics of each word and context. This flexibility enhances its ability to generate context-aware embeddings tailored to the task at hand.

c. *Deep Learning Architecture:* ELMo leverages a deep neural network architecture, which enables it to learn complex patterns and representations in data. This deep learning approach allows ELMo to capture intricate linguistic features and nuances that may be overlooked by shallower models like those used in SVD and SGNS.

Conclusions:

In summary, ELMo's superior performance over SVD and SGNS can be attributed to its ability to generate contextual embeddings, dynamically adjust model parameters, and leverage the power of deep learning to capture complex linguistic patterns and relationships. These factors collectively contribute to ELMo's effectiveness in tasks requiring accurate and nuanced semantic understanding of text data.