ANLP – Assignment 2

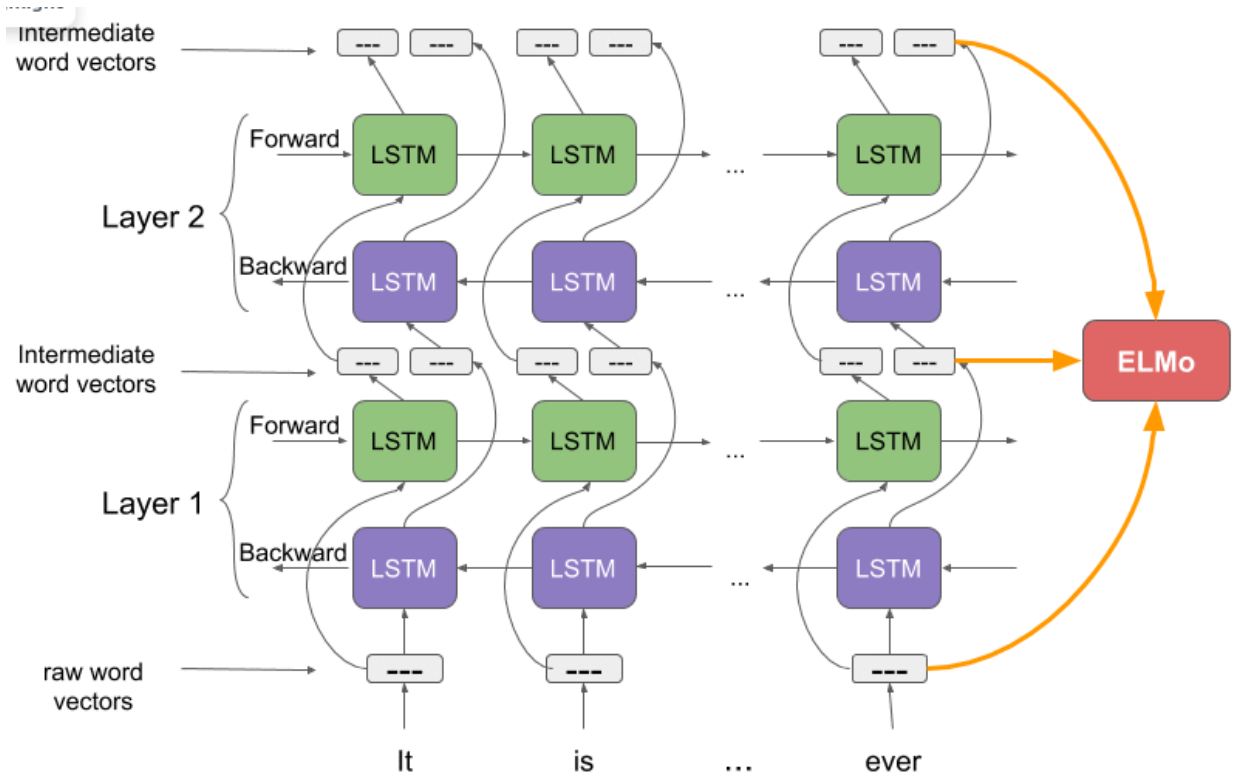
Course Coordinator: Manish Srivatsava
Name : Lakshmipathi Balaji
Roll No: 2021114007
Mail : lakshmipathi.balaji@gmail.com

Question:

How does ELMo differ from CoVe? Discuss and differentiate the strategies used to obtain the contextualised representations with equations and illustrations as necessary.

ELMO:

- ELMo derives word representations using a neural language model consisting of a character-based encoding layer followed by two BiLSTM layers.
- Instead of generating a dictionary of words and their respective vectors, ELMo evaluates words in their actual usage context.
- ELMo can craft vector representations for unknown or out-of-vocabulary words due to its character-based nature.
- It bears similarity to word2vec in its ability to anticipate the next word in a sequence.



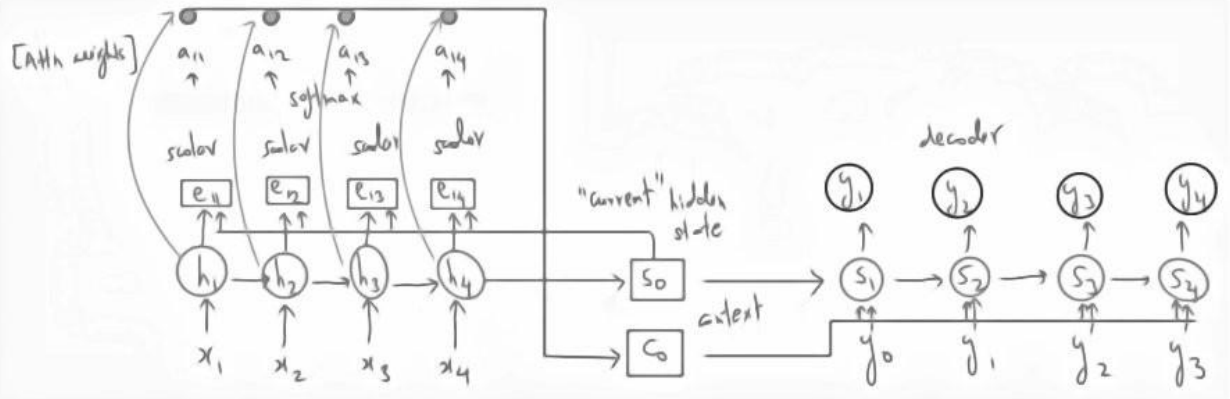
$$h_{k,j}^{LM} = RNN_k(h_{k,j-1}^{LM}, x_j^{LM})$$

Visualising the ELMo model:
Consider LHS as the hidden state at position *j* from the *k*-th layer of the biLM (bi-directional Language Model). RNNk is the RNN operating in either forward or backward direction for layer k, while x is the input to layer *k* at position *j*.

Advantages:
- ELMo's character-focused nature helps in representing out-of-vocabulary words.
- Words are gauged within their actual context, which might enhance representation accuracy.
- Like word2vec, it can predict the upcoming word in a sequence.

CoVe:
- CoVe employs a deep LSTM encoder from a sequence-to-sequence model with attention pre-trained for machine translation to bestow context to word vectors.
- The entirety of the input sentence influences different from conventional word embeddings, CoVe word representations.

Illustrating the CoVe model:



$$h_t^{MT} = LSTM(h_{t-1}^{MT}, [x_t; h_t^{s2s}])$$

Let LHS be the hidden state from the machine translation model at the t-th time step.LSTM is the specific unit, xt is the input at *t* time step, and h is the hidden state at t-1 time step of a sequence-to-sequence model.

Advantages:
- CoVe takes advantage of a sequence-to-sequence model developed for machine translation, ensuring its representations encapsulate the entire input.
- It's proficient in determining a word's contextual meaning as well as the overall sentiment of the sentence.

Differentiation between both the models:
ELMo (Embeddings from Language Models):

Model Structure: ELMo utilizes a bi-directional language model built on top of a character-based convolutional neural network (CNN) and two BiLSTM layers.
Representation: Instead of maintaining a fixed embedding for each word, ELMo offers embeddings that are functions of the entire sentence. This allows for words to have different representations based on their context.
Character-based: ELMo can form embeddings for out-of-vocabulary or unknown words due to its character-based CNN layer, making it versatile.
Training Objective: ELMo is trained as a language model to predict the next word in a sequence, allowing it to understand and generate word sequences.
Application: ELMo embeddings are usually added to existing models to improve their performance, especially in tasks that benefit from understanding the context, like named entity recognition and sentiment analysis.
CoVe (Contextualized Word Vectors):

Model Structure: CoVe employs the encoder from a sequence-to-sequence model with attention that was pre-trained on a machine translation task.
Representation: CoVe, similar to ELMo, offers word representations that are influenced by the entire sentence. This means a word's embedding can vary depending on its context within a sentence.
Machine Translation Roots: CoVe's embeddings are derived from a model trained on machine translation. This gives it a unique perspective on word meanings, as it has been trained to map sentences between languages.
Training Objective: CoVe's underlying model is trained to translate from one language to another. This means it's particularly adept at capturing semantic meanings of words and their relationships to other words in a sentence.
Application: CoVe embeddings can be incorporated into various NLP tasks to provide a richer understanding of word meanings and relationships. Their foundation in translation models might offer unique advantages in tasks involving multiple languages or semantic understanding.

Question:
The architecture described in the ELMo paper includes a character convolutional layer at its base. Find out more on this, and describe this layer. Why is it used? Is there any alternative to this? [Hint: Developments in word tokenisation]

ELMo's Special Layer: A Peek Inside

So, let's dive into ELMo's structure. At its heart, ELMo has this nifty layer called the "character convolutional layer." Why does it matter? This layer is like ELMo's secret weapon for handling those pesky unfamiliar words. Here's the lowdown:

- Imagine the incoming data as a bunch of boxes (or a tensor, for the tech-savvy). Each box has words, and each word has a max of 50 characters.
- ELMo then breaks these words into individual characters and gives each character a unique number tag.
- Each tag gets its own mini-representation or vector.

- Then, the model does some cool math stuff (1D convolutions) on these vectors.
- Finally, the results are all bundled up and sent off to the next stage, known as "highway networks."

Shoutout to Kim and team in 2015 for cooking up this design! Thanks to this layer, ELMo can understand words by looking at their characters, kind of like reading a word letter by letter.

But, Are There Other Ways?

Sure, ELMo loves its character convolutions, but there are other ways to look at words too. You've got techniques like GloVe or Word2Vec that deal with whole words. But they might find themselves in a pickle when it comes to unknown words.

There's also this fancy technique called Tropical Convolutional Neural Networks (or TCNNs for short). But when it comes to understanding the tiny details of language, ELMo's method might be more sharp.

Tokenisation

If ELMo was a camera, its character-based approach would be like zooming in super close, looking at each character in a word. This gives ELMo a unique perspective, letting it pick up on tiny language details.

But wait, there's more! Techniques like Byte-Pair Encoding (BPE) are like detectives for language. They start with characters and then start pairing them up based on how often they appear together. It's a neat trick for understanding unfamiliar words.

Another cool technique? Morphological analysis. Think of it as breaking words into their tiniest meaningful parts, giving an even deeper insight into the language.

REPORT

The AG News Classification Dataset dataset is used for the task and Elmo pretraining and used the first 7600 entries of the train as the development set. A sequence length of 45 is used because the 95% quartile for the given dataset is observed to be the same value. Other configs are mentioned in the below table,

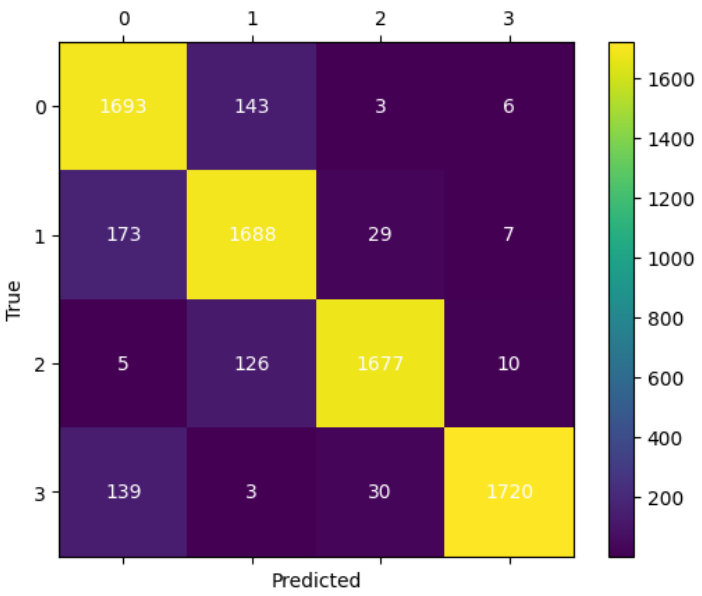
Parameter	Value
embedding_dim	300
hidden_dim	150
num_layers	2
optimizer	adam
batch_size	64
random_seed	42

Note that the same parameters are consistently used for downstream tasks as well.

By the requirements outlined in the assignment, I have diligently completed the analysis of the ELMo model, both in its pretraining phase and in its application to the downstream task.

To ensure thoroughness and clarity in our understanding, the report provides a detailed exposition of the hyperparameters employed during training. This encompasses the selection rationale for loss functions, the number of epochs, the adopted learning rate, and the chosen optimiser.

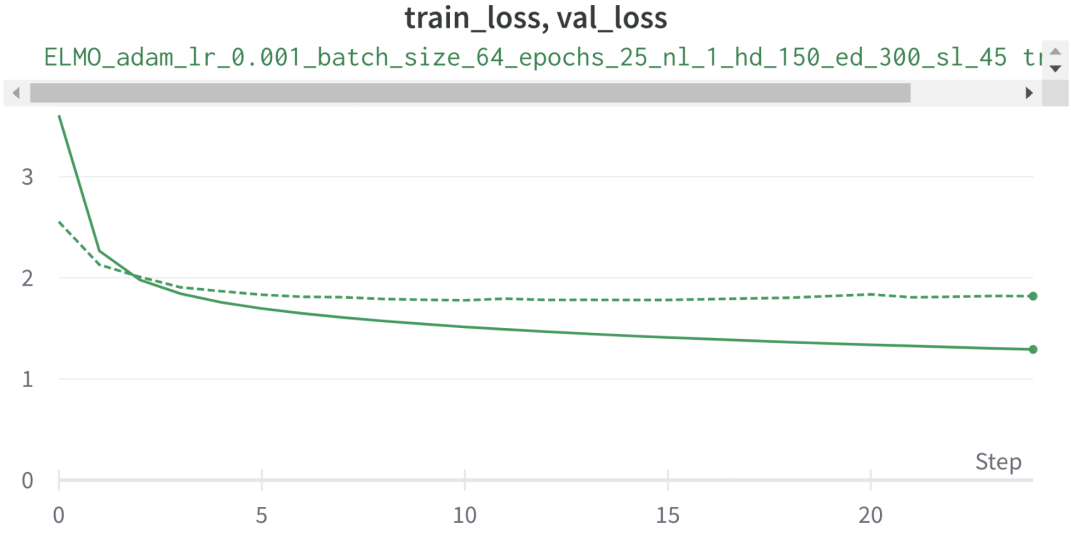
The performance metrics have been comprehensively analysed, focusing on accuracy and the micro F1 score. A confusion matrix has been incorporated into the report further to elucidate the model's performance across various classes.



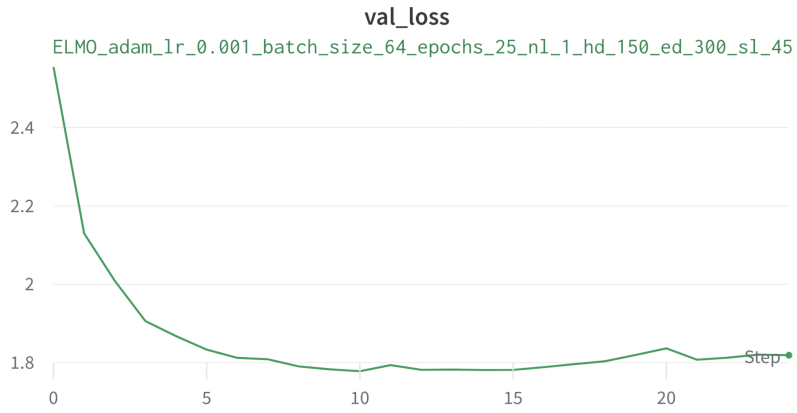
Note that this for learnable parameters for the test set.

Furthermore, as part of the advanced analysis, I investigated the effect of hardcoded configurations for layer-wise representation weights in the ELMo model. Specifically, I undertook experiments with four distinct configurations, one exclusively utilising the embeddings from the final Bi-LSTM layer. The outcomes of these experiments, along with their implications, have been methodically documented for a comparative study. Below is a brief description of the runs for ELMO and Downstream tasks. A deeper understanding of all metrics and plots in wandb can be expected during the evals.

Plot comparing Train and Val loss in ELMO pretraining.

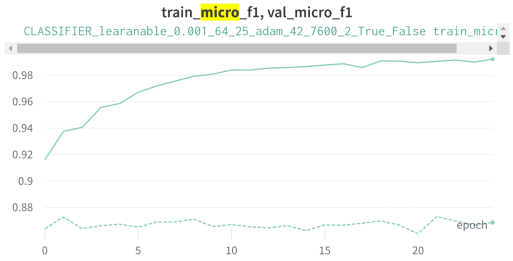


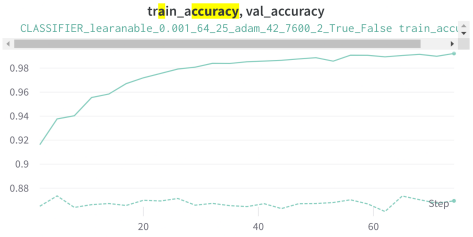
As we can observe after few epochs the model started to over fit to the training set so the best model is taken as consideration for getting embeddings.



It can be noticed that model started overfitting thus val_loss is increased.
Analysis of downstream classification task using Elmo embeddings:

Let's first analyse results with learnable weights for getting embedding from the Elmo model and then go to the hardcoded weights section.





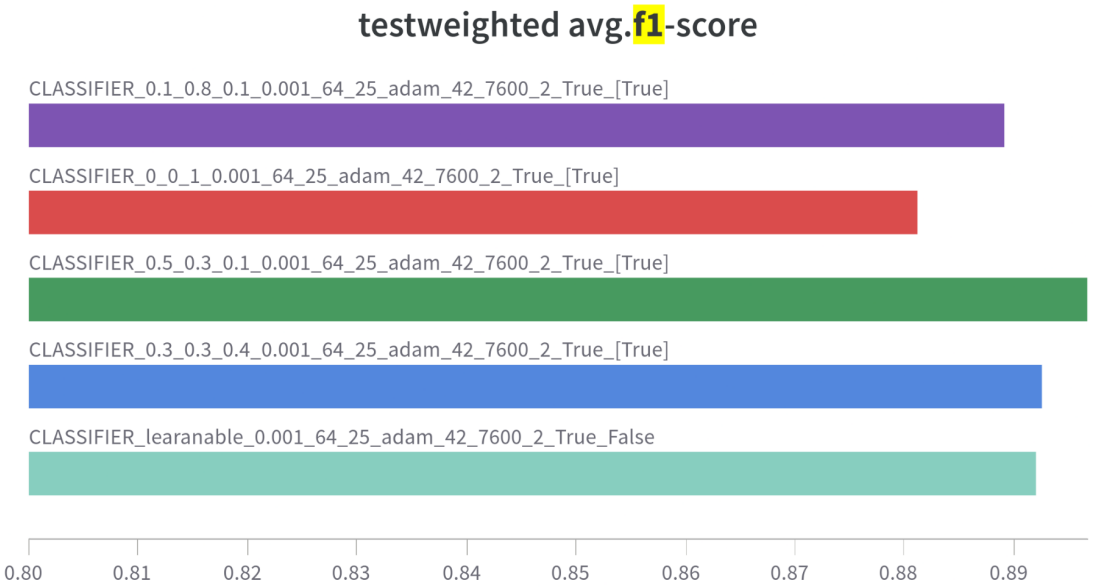
By above plots we can observe that mod tions in the first few epochs and in consistent for next epochs too.
The results for this run is

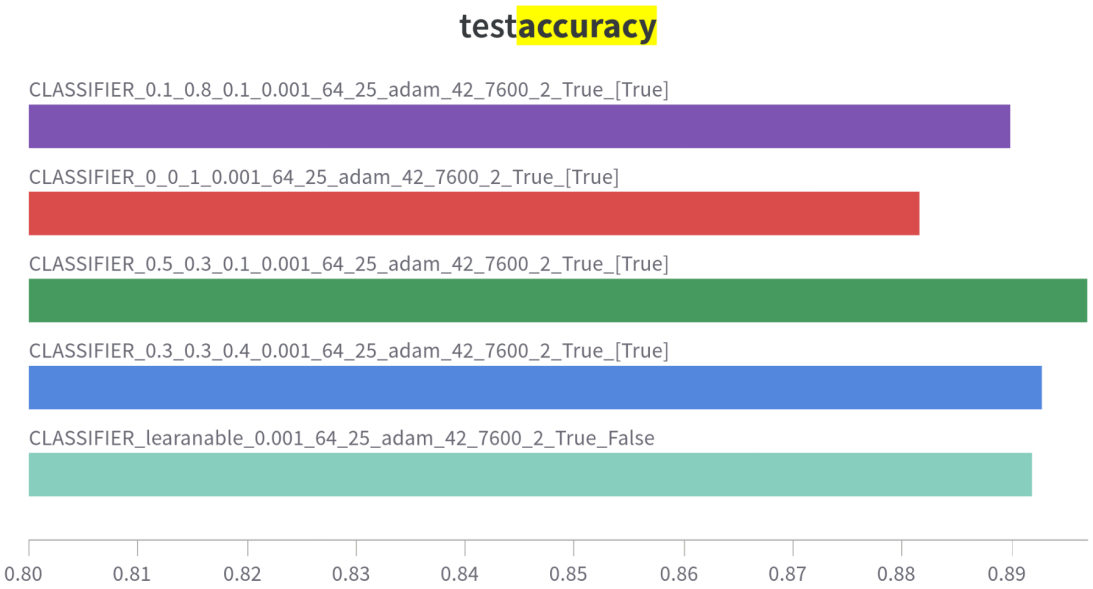
Test weighted avg.f1-score	0.8919681245464093
Test weighted avg.precision	0.8920332708031508
Test weighted avg.recall	0.8919594683510988

Note that any other metrics can be observed in wandb plots for all metrics.

Comparing this with the remaining hyperparameters tried with hardcoded weights.
Weights tried:

RUNS	Embeddings weight	1st layer state weights	2nd layer state weights
Run1	0.3	0.3	0.4
Run2	0	0	1 (only last layer state)
Run3	0.5	0.3	0.1
Run4	0.1	0.8	0.1





Comparison of test accuracies and microf1 scores for all the parameters tried. It is observed that the model with weights [0.5,0.3,0.1] performed

better than all other models, it should have performed better than the model of the learnable weight might be because of the fact that the gradients didn't affect the gradients much or it fitted well on the training set with learnable weights but performed worse on the test set with those weights.

Conclusion:

Upon the successful completion of this assignment, I have gained comprehensive insights into the intricacies of deep learning models, particularly the ELMo model and its architectural nuances. I delved deep into the role and significance of the character convolutional layer, understanding its relevance in addressing out-of-vocabulary words and improving the granularity of word representations.

Link to wandb project - https://wandb.ai/lakshmipathi-balaji/anlp_a2
Link to the models and embeddings - ass2