Artificial Intelligence 2

HW 3

Στέφανος Μπακλαβάς 1115201700093

For the purposes of this hw only the attention layer was not completed.

1.Model

1.1 Data Pre-Proccessing

The data pre-proccessing is the same as of hw2.First we remove every whitespace bigger than two spaces .Then we tun capital letters into small letters. An extra step was done by removing all numbers and emojis of the tweets but then it was found that our model performs better if we leave them and replace the with the mean vector of the tweet that they come from.

1.2 Glove Vectors

Instead of using an Embedding layer glove embeddings were shaped manually because it was found in articles that putting the embeddings manually has a little better performance than using an empedding layer.

https://towardsdatascience.com/pre-trained-word-embeddings-or-embedding-layer-a-dilemma-8406959fd76c

The glove vectors set that is used is the 'glove.twitter.27B.50d.txt'.This set was chosen because it is trained on twitter data which is also the theme of our model. Each glove vector was put into a dictionary with the word as a key and the vector as an item. Then for each tweet of our dataset we found the words of the tweet that were dictionary keys and represented each tweet as vectors of its word vectors. If a word does not appear in glove then we put into the tweet the mean vector of its words . Finally if a tweet has no words in glove we represent the tweet as a zero vector of length 50.

1.3 Model examples

In this HW I did not magane to run many tests because of the time it took for each test to run wich was approximately 3 hours (I was banned of the use of GPU because of Colab usage in other courses also) . The optimizer that was used in all of them was the torch.optim.Adam(lstm_model.parameters(), lr=0.001).

In our dataloader we used batch_size = 32 for training and 8 for validation because validation set is a lot smaller. The scores are the **max** scores of each model performance. Recall score was not used as it had the same result as precision.

Examples without using skip connections:

Model #	Type of cells	Hidden size	num_layers	Dropout between rnn cells	f1_score	Precision score
1	LSTM	16	1	---	0.6824	0.7184
2	GRU	16	1	---	0.6522	0.7025
3	LSTM	16	2	---	0.6469	0.7074
4	GRU	16	2	---	0.7013	0.7462
5	LSTM	16	2	0.3	0.6586	0.7119
6	GRU	16	2	0.3	0.6906	0.7209
7	LSTM	32	2	---	0.6443	0.6999
8	LSTM	16	4	0.3	0.5701	0.6523
9	GRU	16	4	0.3	0.6607	0.7400

Model with skip connections where the output of i-th layer is added to the output of layer i+1.

10	LSTM	16	4	0.3	0.6075	0.6672
11	GRU	16	4	0.3	0.6441	0.7063

Model #4 which performed better with clipping:

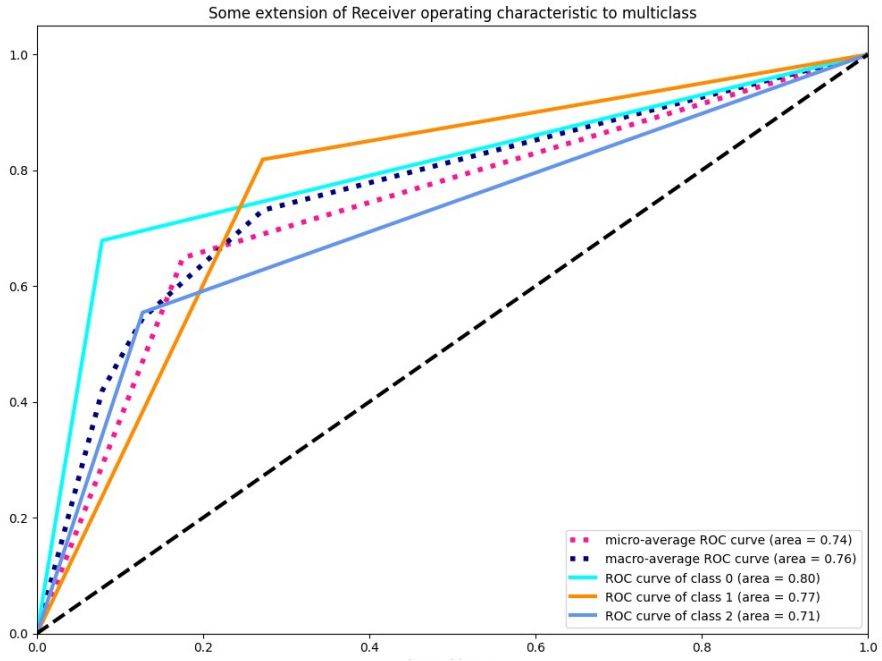
```
torch.nn.utils.clip_grad_norm(lstm_model.parameters(),max_norm=2.0, norm_type=2)
```

12	LSTM	16	2	---	0.6226	0.6711

1.4 Best Model and comparisons

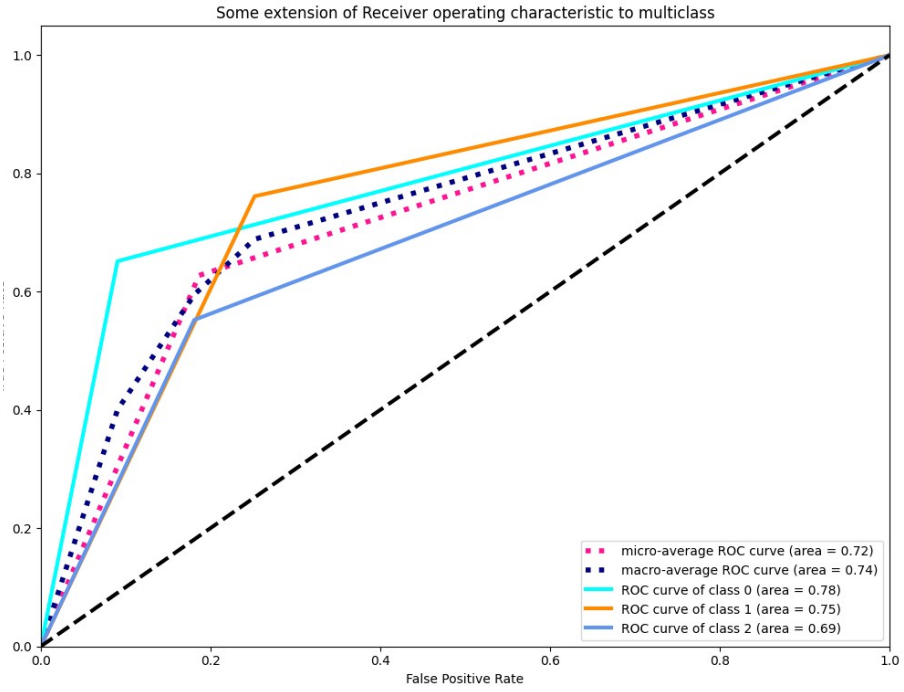
Of all the above models the model in examples #3 and #4 performed better and we saw that GRU cells in model #4 gave as the best results.

Model #4 (GRU)

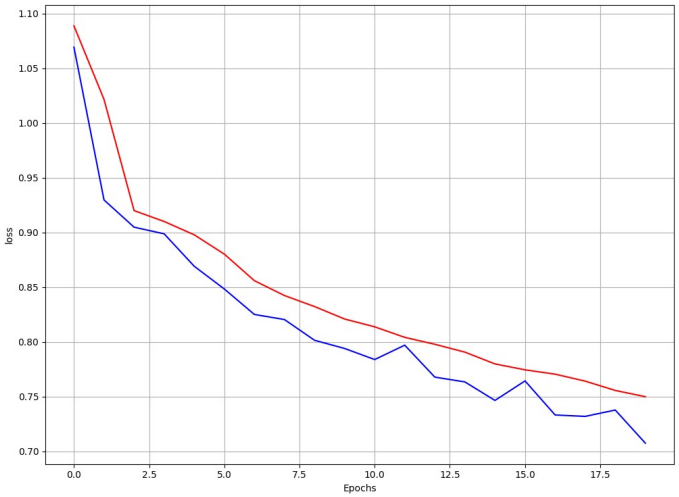


We can see in the ROC diagrams that LSTM and GRU cells have almost the same results in model true when comparing the false positives.In all three classes GRU cells predicted the right class a little better than the LSTM cells when using this model.

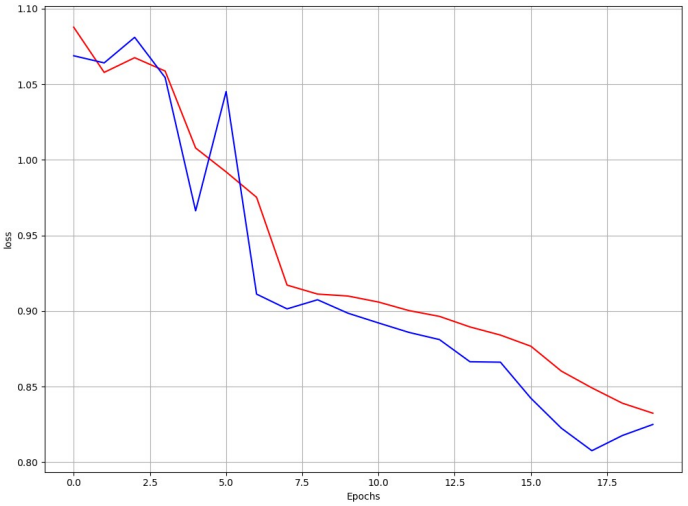
Model #3 (LSTM)



Model #4 (GRU)



Model #3 (LSTM)



We can also see in the train-test diagrams that GRU has a smoother progress during epochs.

Although we expected that clipping would help increase the performance of our model this did not happen (example #12 is example #4 with clipping). Maybe better clipping parameters would have achieved this goal.

On the other hand skip connections increased the performance of example 8 with LSTM cells but performed a little worse when used over example 9 with GRU cells (example 10 is example 8 with skip connections and example 11 is example 9 with skip connections) . The best model we found had only 2 RNN layers so we could not test skip-connections in that model.

There we can see the diagrams that were produced by example #11 because I could non run again this examples together with example 4 and put them into the .ipynb file.