Lab Report Exercise 03

Preprocessing

The preprocessing was very important in this exercise. We have **removed the URLs, @someone mentions and the HTML sequences**. We have also removed all languages that had less than 1000 instances. This left us with 11 languages to work with.
After a while, we realized that there was an ‘und’ label present in the train/validation sets, which stands for an ‘undefined’ language. This probably caused issues, as we’d also added an ‘<unk>’ class as a fallback case for *unknowns*.

With this issue, we decided to **exclude the ‘und’ instances** from the train/val set. Especially because some of those ‘und’ instances were actually in languages that were part of the dataset! This could potentially mess with the weights. However, we have also changed our ‘<unk>’ class to ‘und’. With this change, the ‘und’ class that could possibly be in the test set wouldn’t be new to the model. We suspect that this issue with the ‘und’ class (especially in the test set) is the reason for somewhat low accuracies.

We have also added class weights as a parameter for the loss function (CELoss) while training to account for the obvious class imbalance.

The model

We chose to work with PyTorch. Our CNN model has one convolutional layer, followed by a maxpool and two fully connected (dense) layers. One-hot encoded characters were used for the input representation. The input layer’s dimensions were 256x5613. These dimensions translate to max_tweet_length x number_of_unique_characters.

Hyperparameters

As you can see, we have tried a few different hyperparameters
The hidden neuron size and the number of filters were also changed quite a bit in the process, resulting in significant changes in the accuracy of our models.
We have only reported our 5 best models here, but a lot of different combinations were tried. One you can’t see here much is the dropout. We have tried to modify the dropout to 0.6 and 0.7, but with inconclusive results. You can see this with the last row: if you look closely, the model is the same as the third one, only the dropout changes (in pink). However, this difference in dropout resulted in a loss of accuracy of more than 10%.

When it came to the optimizer, we saw Adam was usually preferred with CNNs, so we decided to stick to it. We also didn’t change the pooling strategy, using maxpooling every time.

The stride of one was chosen to allow for overlaps in the n-grams: if we had a stride of 2 with bigrams, it would mean that the bigrams wouldn't overlap and we feared it would negatively affect our model. Coming to the n-grams : we decided to keep the kernel size rather low (up to 4), first because all our best models in ex01 were based on bigrams. The training time was also taken into consideration when making this choice.

Optimizer	Learning rate	Epochs	dropout	# of filters	stride	Kernel size	Pooling	Number of hidden neurons	Batch size	Accuracy on test set (validation set)
Adam	0.0001 first 5, 0.00001 last epoch	6	0.5	300	1	4	Max	128	32	78.46% (89.02%)
Adam	0.005, decreasing multistep ([5, 10], gamma=0.1)	15	0.5	500	1	2	Max	256	32	78.11% (86.57%)
Adam	0.0001	3	0.5	300	1	3	Max	128	32	78.02% (88.45%)
Adam	0.0001	3	0.5	300	1	3	Max	350	32	73.41% (83.01%)
Adam	0.0001	3	0.5	300	1	2	Max	128	32	73.39% (82.23%)
Adam	0.0001	3	0.7	300	1	3	Max	128	32	67.24% (75.69%)

(early stopping was not used)

A little remark on the accuracies: we can see that almost all models seem to have trouble generalizing, as the accuracies on the test set are always lower than the one on the validation set. This could be related to the 'und' labels we mentioned earlier. It might also be overfitting a little bit, although it is rather unlikely given the low numbers. In any case, the accuracies are not very high: we had expected (or hoped for) accuracies ranging above 80-85%%, which we couldn't obtain.

References

Delip Rao and Brian McMahan. *Natural language processing with PyTorch: build intelligent language applications using deep learning*. O'Reilly, First edition ed., 2019.

Sasikumar Ivek. Image classification - CNN with PyTorch. Nov 2018. https://medium.com/@vivekvscool/image-classification-cnn-with-pytorch-5b2cb9ef9476

Cornelisse Daphne. An intuitive guide to Convolutional Neural Networks. Apr 2018. https://www.freecodecamp.org/news/an-intuitive-guide-to-convolutional-neural-networks-260c2de0a050/