Name-Based Gender Classification

Ch Muhammad Awais

Collection of gender information while filling the forms can hurt the sentiments of users, so most online applications do not ask for this information. However, it's a recommendation system that should be capable enough for inferring the genders for recommending the related stuff. Therefore it is necessary to infer the gender of those users who did not to provide this information during registration. We are predicting the gender of registered users based on their declared names. By analyzing the first names of 100k+ users, we found that genders can be very effectively classified using the composition of the name strings. We propose a number of hyperparameter tuning on two different machine learning models, LSTM and Neural Networks, and demonstrate that our models are able to infer the gender of users with much higher accuracy than baseline models if the hyperparameters are tuned better.

Brief task definition and data description

We have a dataset of names with their specified genders, and we built a classifier to predict the gender of the name, our dataset consists of two columns, name, and gender. We converted the gender into 0-female, 1-male form, and name into normalized format e.g the name Elizabeth will be represented as [5 38 35 52 27 28 31 46 34 0 0 0 0 0 0], the ending 0's are used for padding, as we assumed that the max length of a name will be 15, so the names with lesser characters are padded with 0's.

We have discussed two machine learning models, LSTM and Neural Nets, with different hyperparameters, the comparison metric is accuracy. Our baseline accuracies are as follows:

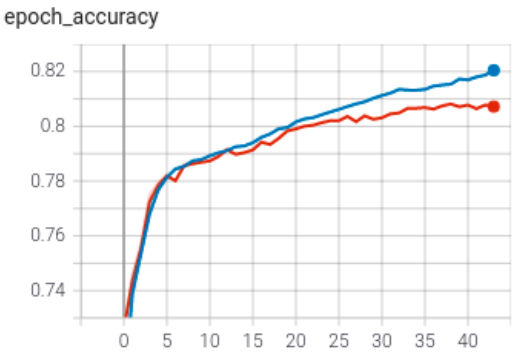
Model	Accuracy
LSTM	• 80.21%
Neural Network	• 75.72%

LSTM performance

Different hyper-parameters tested are as follows:

- Learning rate: 0.001, 0.01, 0.1, 0.2, 0.3
- Epochs: 10, 50, 100
- Batch Size: 10, 20, 40, 60, 80, 100
- hidden_state_size: 5, 10, 15, 20, 25

📌 **SUMMARY:** After testing different parameters, we found out that learning rate, epochs, and batch size were not affecting the performance of our LSTM model, but the change in hidden state size made an effect on the accuracy. The graphs below are showing the change in accuracy and loss on different epochs.

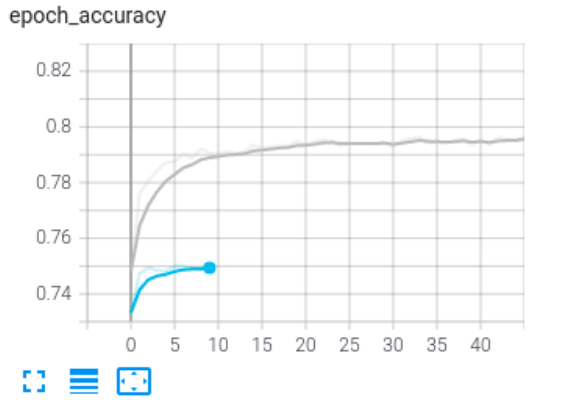


Classical Neural Network Performance

We created a classical neural network with 5 layers, i.e. embedding, flatten, dense, dropout, dense, the hyper-parameters used are as follows:

- Learning rates: 0.001
- Activation Functions: sigmoid
- Epochs: 100
- Batch Size: 64

📌 **SUMMARY:** Classical neural network gave us an accuracy of 75% when we set up the learning rate to 0.001, epochs and batch size was also set the same. 75% is the best accuracy achieved after checking different learning, rates, hidden_state_sizes, and epochs. The difference between accuracy and loss can be seen in LSTM and Neural Network, so based on this information, and the different in accuracy we can say that LSTM performs better than Neural Network for this task, given that the hyper-parameters are same for both.



epoch_loss