ARI3210 – Speaker Classification with Deep Learning

Matthew Kenely[1]

[1]University of Malta

matthew.kenely.21@um.edu.mt

Abstract

In this project, a neural network architecture for the use training a Speaker Identification classifier is proposed. When trained on the ABI-1 Corpus, a deep learning classifier using the proposed architecture achieves an F-score of 0.899

Index Terms: speech recognition, deep learning, CNN, LSTM, speaker identification

1. Introduction

The aim of this project is to design a neural network architecture using a combination of convolution and LSTM-type RNN blocks and train a deep-learning classifier using this architecture. The classifier will be used to carry out Speaker Identification (SID) (the process of identifying speakers through the analysis of samples of their speech) on a given pool of speakers.

2. Implementation

2.1. Architecture

Shown in Table 1 is the proposed architecture:

No.	Layer Type	Detail
1	conv2D	3x3, 32 filters
2	batchNormalization	-
3	ReLU	-
4	maxPooling	2x2, stride (1,1)
5	conv2D	3x3, 64 filters
6	batchNormalization	-
7	ReLU	-
8	maxPooling	2x2, stride (2,2)
9	conv2D	5x5, 32 filters
10	batchNormalization	-
11	ReLU	-
12	dropout	10%
13	maxPooling	2x2, stride (2,2)
14	LSTM	32 units
15	batchNormalization	-
16	dropout	50%
17	fullyConnected	285 neurons
18	softMax	285 outputs

Table 1: *Proposed SID Architecture*

Layers 1 – 7 and 16 – 17 are derived from the architectures proposed in [1] and [2] and were used as a basis for the classifier. Using solely Layers 1 – 7 combined with Layers 16 – 17 yielded good accuracy on the training set (> 0.98), but the classifier failed to generalize well, reaching a maximum accuracy of around 0.62 on the validation set.

An LSTM layer was integrated into the architecture with 32 units and a dropout rate of 0.5 (determined through trial and error) into the architecture (Layers 14 – 16), improving the classifier's accuracy on the validation set by around 0.1.

To further increase the classifier's ability to generalize on the speaker data, an additional convolution layer (Layers 8 – 13) was introduced, followed by dropout with a rate of 0.1.

2.2. Hyperparameters

Optimizer	Adam
Learning Rate	Scheduled
Momentum	0.9
Loss Function	Categorical cross-entropy
Metrics	Accuracy
Mini-batch size	128
Gradient Clipping	0.5

Table 2: *Model Compilation Configuration*

Adam [3] is a popular optimization algorithm that combines the benefits of both AdaGrad and RMSProp and is known for its adaptive learning rate and momentum features. It has been shown to perform well on LSTM-based models [4].

The use of a learning rate scheduler was chosen after a rigorous experimentation process. Lower learning rates would cause the classifier to learn excessively slowly and fall into local minima during initial epochs, and higher learning rates would prevent it from converging on the validation set, thus a learning scheduler was used as follows:

Epoch	**Learning** Rate
0	0.001
5	0.0005
10	0.0001
15	0.00005
20	0.00001

Table 3: *Learning rate scheduler*

Categorical cross-entropy is commonly used loss function for multi-class classification problems and has seen use in LSTM-based models [5]. It measures the discrepancy between the true class distribution and the predicted class distribution.

Accuracy is a commonly used metric for classification problems. It represents the ratio of correctly predicted instances to the total instances, providing an overall assessment of model performance.

An ideal mini-batch size of 128 was determined through trial and error. A batch size of 64 resulted in excessive training time for similar results. A batch size of 256 proved detrimental to the classifier's ability to generalize, likely due to it being exposed to less noise [6].

Gradient clipping was used to minimize instability in validation loss during training.

3. Evaluation

3.1. Loss curves

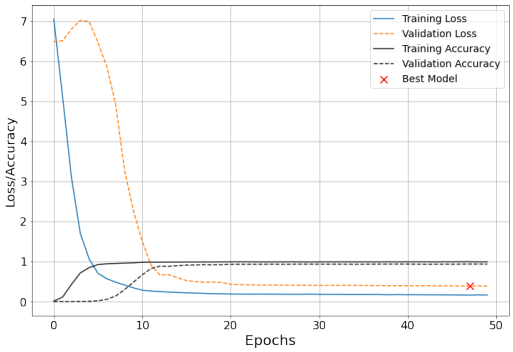

Figure 1: *Training/validation loss/accuracy curve*

The training progress of the classifier over 100 epochs is shown in Figure 1. The classifier converged after around 20 epochs, and achieved a maximum accuracy of 0.931 on the validation set.

The classifier does not display any overfitting as the number of epochs increases, validation loss decreases consistently.

3.2. F-score

The classifier was able to achieve an average F-score of **0.977** across 5 test set splits, with a minimum of 0.974 and a maximum of 0.982 being achieved in these test sets. This low variance indicates a high degree of generalization.

The observed F-score increased throughout the project as the ability for the classifier to generalize, i.e. performance on the validation and test sets, increased through the steps taken in 2.1.

3.3. Confusion matrix

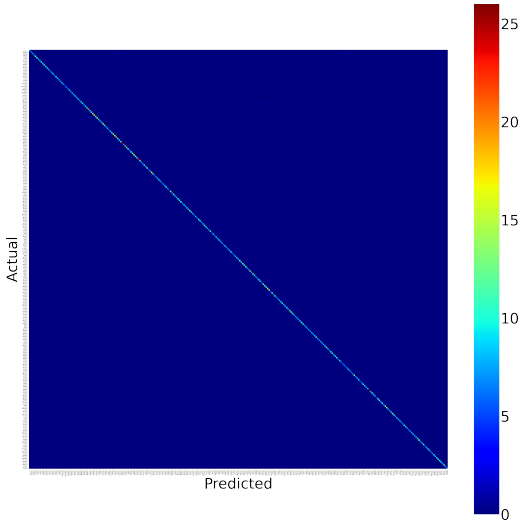

Figure 2: *Confusion matrix across all 285 classes.*

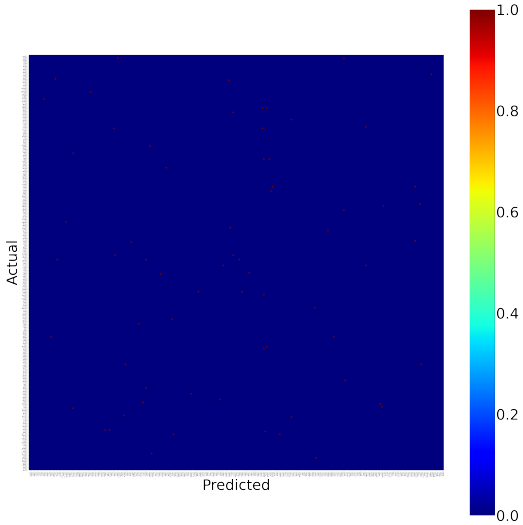

Figure 3: *Confusion matrix across all 285 classes, excluding the diagonal.*

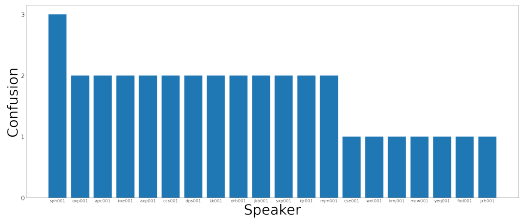

Figure 4: *Top 20 speakers with the largest quantity of incorrect predictions.*

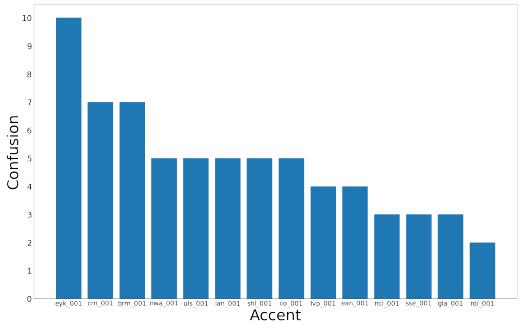

Figure 5: *Accents and cumulative confusion across all speakers.*

From Figure 4 it can be deduced that, given that the greatest amount of confusion was 3 (in only one case), there is no discernible pattern which made certain speakers harder to predict than others.

Figure 5 initially indicates a discrepancy in the predictability of certain accents, however when accounting for the distribution of accents in the dataset (as shown in Figure 6), a correlation between the quantity of speakers with each accent and the quantity of incorrect predictions is revealed, indicating that this confusion stems linearly from a greater quantity of speakers and not from the accents themselves.

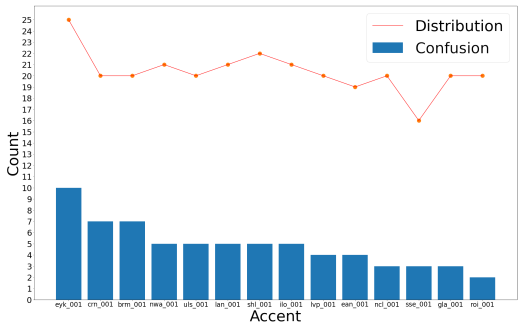

Figure 6: *Accents and cumulative confusion across all speakers, as well as corresponding quantity of speakers in the dataset.*

4. References

[1] S. Bunrit, T. Inkian, N. Kerdprasop, and K. Kerdprasop, "Text-independent speaker identification using deep learning model of convolution neural network," *International Journal of Machine Learning and Computing*, vol. 9, no. 2, pp. 143–148, 2019.

[2] Y. Lukic, C. Vogt, O. Dürr, and T. Stadelmann, "Speaker identification and clustering using convolutional neural networks," in *2016 IEEE 26th international workshop on machine learning for signal processing (MLSP)*. IEEE, 2016, pp. 1–6.

[3] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[4] Z. Chang, Y. Zhang, and W. Chen, "Effective adam-optimized lstm neural network for electricity price forecasting," in *2018 IEEE 9th international conference on software engineering and service science (ICSESS)*. IEEE, 2018, pp. 245–248.

[5] S. Kim and M. Kang, "Financial series prediction using attention lstm," *arXiv preprint arXiv:1902.10877*, 2019.

[6] H. Yong, J. Huang, D. Meng, X. Hua, and L. Zhang, "Momentum batch normalization for deep learning with small batch size," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XII 16*. Springer, 2020, pp. 224–240.