Experiment on the King corpus

Thomas Keller

Oct 2023

Abstract

This study explores different parameters of a model architecture for speaker recognition using the King Speaker Verification corpus, focusing on the Nutley, NJ, recordings from the International Telephone and Telegraph Corporation. For a given neural network architecture, the error rate is evaluated, and the influence is assessed for different parameters such as the number of hidden nodes, hidden layers, and regularization penalties on model performance. Through the experiment, the findings show substantial improvements in speaker recognition tasks with an error rate of 4.8%. The results demonstrate the efficiency of the selected architecture with selected parameters.

1 Introduction

Speech processing has evolved as a prominent field within the realm of artificial intelligence and machine learning. One notable application within this field is speaker recognition, which aims to pinpoint individuals only by their vocal attributes. While this task may seem deceptively simple to the human ear, it presents a significant challenge for automated systems. The sound of a voice is influenced by a variety of factors, including individual habits, emotional states, external influences, and physiological properties. Given this diversity, the crux of the challenge lies in the precision required to adeptly extract these unique vocal features and patterns that distinctly define each individual from the extensive variability of influences [1].

2 Methods

For this project, the dataset of the King Speaker Verification corpus [2] is used. The King corpus consists of 51 male speakers recorded by the International Telephone and Telegraph Corporation in San Diego, CA, and Nutley, NJ. As a part of this assignment [3], only the Nutley recordings are used. The Nutley database consists of 25 speakers, which are recorded with a sample frequency of 16kHz. Each speaker has 10 recordings and the first five are used to train the model and the last five to test. The files are split into frames of a length of 20ms and overlapping of 10ms. For the reason that it is really hard to identify the speaker on a 20ms frame, the recognition is made on the whole file.

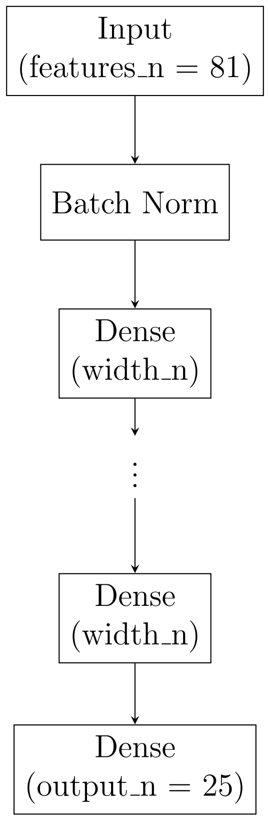

Figure 1: Architecture of the neural network model with an input layer with 81 features, a batch normalization layer with default values, 2-3 dense layers with a specific width_n, and an output layer with 25 categories.

Figure 1 shows the architecture that is implemented. The input layer consists of features with a dimension of features_n = 81. Followed by the batch normalization, so that the features have a consistent scale. After that, dense layers are implemented; the number depends on the experiment and the width_n as well. The final layer, is the output layer with dimension of output_n = 25, corresponding to the number of categories the model predicts.

Another aspect of the architecture is the regularization that is used in the dense layers. To prevent overfitting and to encourage weight sparsity, L2 norm regularization is applied.

3 Experiments

The study tries to find the best parameters for the given architecture by changing one parameter at a time. The data is split 1:1 into training and test data files. For the training process, the first 5 files of each speaker are used. For the validation during the training the train_test_split function is applied with ratio of 1:9. The last 5 files of each speaker are used for the testing.

The experiment primarily examines the influence of the number of nodes (width_n) on the error rate. While the number of hidden layers (hidden_n) is varied between 2 and 3, the study also inquires about the effect of the regularization (l2_penalty), testing values from 0 to 0.0001. Additionally, batch sizes from the default value of 32 up to 100 are evaluated.

4 Results

Table 1 shows all the experiments with different parameters for architecture 1 and their outcomes.

No.	**(features_n, hidden_n, width_n, l2_penalty, output_n)**	**Epochs**	**Error Rate**
1	81, 2, 25, 0.001, 25	10	23.20%
2	81, 2, 50, 0.001, 25	10	14.40%
3	81, 2, 100, 0.001, 25	10	13.60%
4	81, 2, 200, 0.001, 25	10	12.00%
5	81, 2, 200, 0.0001, 25	10	8.80%
6	81, 3, 200, 0.0001, 25	10	8.00%
7	81, 3, 200, 0.0001, 25, batchsize=100	10	9.60%
8	81, 3, 400, 0.0001, 25	10	6%
9	81, 3, 250, 0.0001, 25	10	7.20%
10	81, 3, 300, 0.0001, 25	10	6.40%
11	81, 3, 350, 0.0001, 25	10	4.80%
12	81, 3, 350, 0, 25	10	7.20%
13	81, 3, 350, 0.0001, 25	20	5.60%

Table 1: Summary table of neural network architecture parameters and their performance regarding error rate.

Figure 2 shows the confusion matrix for the evaluated model parameters (No. 11, 1) for the testing dataset, with an error rate of 4.8%. It results that 6 predictions of 125 (25 speaker and each speaker has 5 files) are wrong. One file of speakers 0,1, 9, 17, 19 and 20 is wrongly predicted.

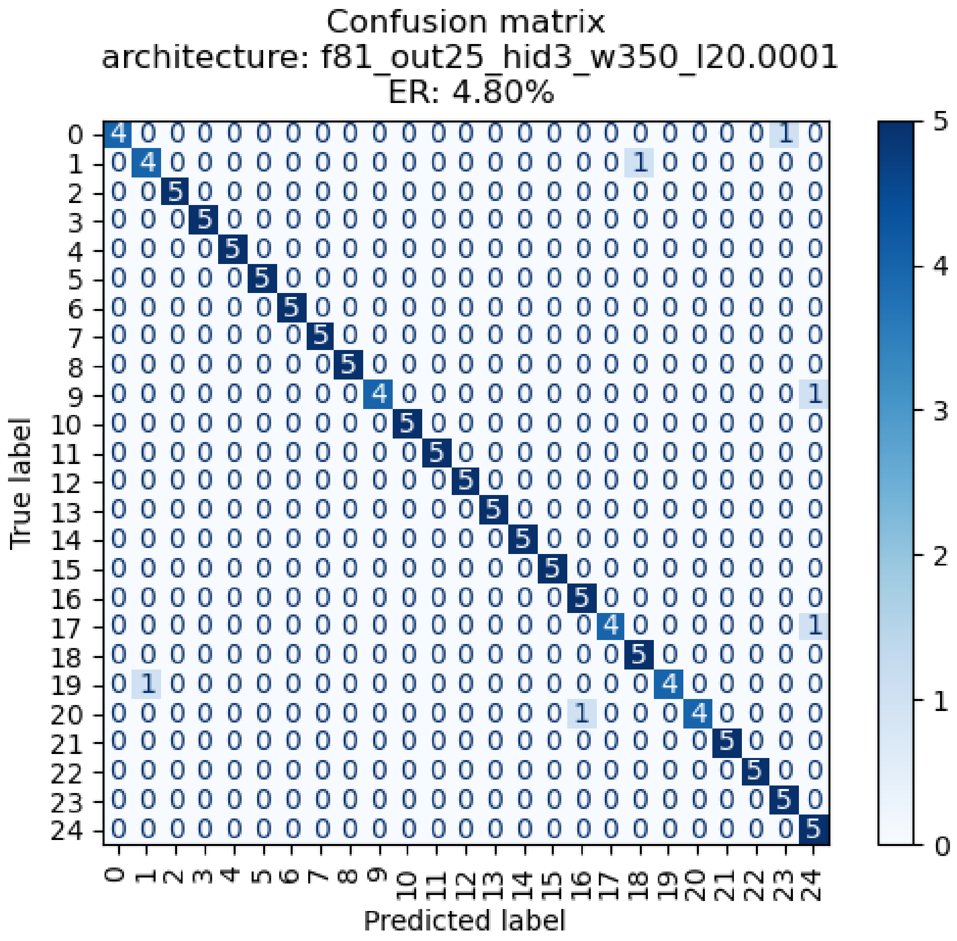

Figure 2: Confusion Matrix with an error rate of 4.8%; x-axis are the true labels and y-axis are the predicted labels; the parameters for architecture 1 are as follows: features_n = 81, hidden_n = 3, width_n = 350, l2_penalty = 0.0001, output_n = 25. Wrong predictions are speaker files 0,1, 9, 17, 19 and 20.

5 Discussion

The evaluated model (No. 11, 1) does reach an error rate of 4.8% on the test dataset, which consists of the last 5 files of each speaker. That is below the expected error rate of 20%. The width of the 3 hidden layers is 350 for the best-evaluated model. That results in more computation time compared to the training with a lower width_n of the hidden layers. That doesn't really fall into account for this experiment because the model training takes around 10-20 seconds, depending on the parameters. Compared to the width of 200 (No. 7, 1), the error rate is halved by increasing the width of the hidden layers by 150. Further, the default batch size of 32 gives a good result.

For a future experiment, the batch size could be decreased, which would decrease the computation time of the model. Further, it would be interesting to see how another architecture would perform. For example, by adding so-called dropout layers [4].

References

[1] Wikipedia. Speech recognition. Oct 2, 2023. https://en.wikipedia.org/wiki/Speech_recognition

[2] Higgins, A., and Vermilyea, D. (1995). "King Speaker Verification," edited by International Telephone and Telegraph Corporation (Linguistic Data Consortium, 3600 Market Street, Suite 810, Philadelphia, PA 19104-2653), doi:10.35111/j0af- qf40.

[3] Prof. Marie Roch (2023). Speech recognition CS682 Assignment 2. Oct 3, 2023. https://roch.sdsu.edu/cs682/assignments/A2.pdf

[4] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," edited by The Journal of Machine Learning Research 15, PA 1929-1958.