Machine Learning Challenge

Introduction

In this challenge, the goal was to build a machine learning algorithm which takes audio recording data as input and predicts which of several english words are pronounced in that audio recording.

Preprocessing and extracting features

We concluded that it would be best to use the given features as our input data, but had to make some changes to the data structure. The features came in one dimension, consisting of 10535 examples, each example came with 99 lists and each list contained 13 features in total. After preliminary EDA of the data, we realized that some examples contained less than 99 lists. In order to make up for the missing data, we added n-lists of 13 zero values to each incomplete example, so that at the end, each example consisted of 99 lists in total. Next, we split the feature data into a training and test set. In order to enable efficient feature engineering for our first algorithm (random forest), we changed the data into a three dimensional array in the following format: (Example, lists, features).

Learning algorithms

In total we used two different algorithms, a random forest and lastly a convolutional neural network. Because it is a multiclass classification problem, we first tried to implement a random forest algorithm. As our prediction accuracy was very low (23%) and did not change significantly even after adjusting the forest depth, we disregarded it and implemented a convolutional neural network since it has been demonstrated to be a powerful algorithm for audio classification (Hershey et al., 2017). We used a CNN from a tutorial on Datacamp as our baseline model and adjusted several parameters to our task demands ("Convolutional Neural Networks in Python", 2019).

Hyperparameter tuning

For the random forest we adjusted the number of nodes to see how it affected the prediction accuracy. The accuracy did not change significantly after adjusting the number of nodes but 300 nodes led to the best accuracy (Table1). For the convolutional neural network we had to adjust the amount of hidden layers, batch size, epochs and kernel size. Through trial and error we finally found the perfect balance of parameters, which led to the best overall prediction accuracy (Table 1)

Results

Table 1 depicts the best performing algorithms we used for the task and the subsequent prediction accuracy on the validation data. After submitting our neural network on codalab, we obtained an overall accuracy of 94.55%.

Group Information

Codalab account: Avengers
Group: 38

Name	Snr	Anr
Francesco Lauria	2041074	819639
Athanasios Toulias	2038923	682034
Mory Kaba	2047205	234755

Task Division

Mory Kaba did the preprocessing of the data and wrote the report. Francesco Lauria build the random forest algorithm and created the convolutional neural network together with Athanasios. Athanasios ran the convolutional neural network and we collectively tuned the hyperparameters accordingly, to get the best results.

References

Convolutional Neural Networks in Python. (2019). Retrieved 7 December 2019, from https://www.datacamp.com/community/tutorials/convolutional-neural-networks-python

Hershey, S., Chaudhuri, S., Ellis, D., Gemmeke, J., Jansen, A., & Moore, R. et al. (2017). CNN architectures for large-scale audio classification. *2017 IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP)*. doi: 10.1109/icassp.2017.7952132

Appendix

Table 1: Algorithms

Type of Algorithm	**Hyperparameters**	**Accuracy Validation Data**
Random Forest	Number of Nodes: 300	23%
Convolutional Neural Network	Batch size: 256 Epochs: 100 Number of hidden layers: 10 Kernel size = 6	95.06%