TÉCNICO LISBOA

Speech Processing

Instituto Superior Técnico

2º Semster - 2020/2021

MEEC

LAB3

Grupo Nº:
nº 100660 Gustavo Bakker
nº 90817 Vasco Araújo

May 2021

Part 0: VoxCelebPT

For this part we will analyze the VoxCelebPT dataset. It has 26 663 transcribed utterances from 51 speakers (28M/23F) and a total duration of 17:55:14 hours.

Because it doesn't have many speakers, the dataset will be divided into three groups: 70% for train, 20% for dev and 10% for test. We will answer two questions regarding this dataset.

Q: Is such a partition of data a valid approach to train and/or evaluate paralinguistic tasks?

Yes, this is a valid partition of data for paralinguistic tasks. In order to get rid of overfitting, Machine Learning engineers typically divide the dataset into two or three sets. This is so that the model doesn't overfit to the data available and therefore is able to correctly predict the result when a different data is introduced. The difference between using two and three partitions is the addition of the development set. This partition is used to tune the hyperparameters of the model.

Q: Is such a partition of data a valid approach to (only) evaluate Speaker Recognition tasks?

A problem that may arise from such partition is that the data from a single person could be only on one of the partitions. For example, say we only have a single sound file from Person A. if that data is on the test set, the model would not be able to learn from it and as such the probability of a failure to recognize Person A would be very high. To counteract this we could make sure that the data from each person is sliced proportionally across all partitions to increase distribution and probability of success.

Part 1: Age Regression

Q: Do you consider age detection an easy task? Identify and explain possible age groups you think make sense considering what happens when you age.

Depending on how the question is framed, aged detection can range from very easy to very difficult. For example, it is not that hard to divide a person into a child, adult or elder based only on speech. On these groups there are very clear differences in speech such as F0, jitter, vocal tract length, pitch, etc.

After downloading the datasets aGender (used for the train and development partition) and VoxCelebPT (used for test partition) we had to do the feature extraction, which provided no difficulty since the code was given.

After this we were ready to do Age Regression in two ways: SVR (Support Vector Regressor) and with Neural Networks.

SVR

For the SVR, we started by normalizing all data along the train dataset. If we have chosen a linear kernel, normalizing wouldn't much effect. However, if we chose the RBF kernel whose accuracy heavily depends on the data being normalized it would be very important.

Regarding what Kernel to choose, we mainly played with two options, Linear and RBF (Radial Basis Function) which is a non-linear filter.

The RBF kernel has basically 3 parameters to adjust: C, the regularization, g, gamma and d, the order of the polynomial. In our tests, d proved to make no real change. In a test where C and g where fixed, we tried to compute predictions with d equals to 1 and with d equals to 20. Both on the test set and the devel set, the predictions were exactly the same. Because of this, we chose to have d equal to 1.

C and gamma, however, have a great effect on the overall result. C, or the regularization, sets the toleration of the model to allow misclassification of data points. This means that, the higher the C, the more we would penalize a misclassification. Because of this, with a large C we have a higher the accuracy for the training data, but at a cost of possibly overfitting and failing to generalize to unseen data. Gamma is the lenght of points the model considers in order to compute the separation line. A high gamma will mean that the radius of influence is very small and the model will overfit, while a low gamma is not very accurate as it resembles the behaviour of a linear kernel.

Trying several combinations, we concluded that the best parameters are a C of 1, a gamma 0f 0.01 and a d of 1. Of course there may be a better combination, these values provided a good enough result. While we didn't test the model on the test set, we believe these values are robust enough to fit correctly.

The Linear and the Polynomial kernels proved to be worse than the RBF.

Neural Networks

As for the SVR, we start by normalizing all data with regards to the train dataset. Using a Neural Network means we can choose from a wide range of hyperparameters such as number of epochs, the learning rate, the learning decay, batch size or dropout probability. On top of that, we can also choose how to con strut our neural network by tuning the number of neurons or if we want to use Batch Normalization or Dropout.

In our testing, we concluded that 100 epochs, 0.1 for learning rate, 0.001 for learning decay, 128 for batch size, 0.1 for dropout probability and had the best results.

As recommended, we also implemented a second and a third layer for our neural network, which proved to be very successful. With only the first layer, the Neural Network has worse results than the SVM, but with all three layers it performed a bit better.

The final results from the SVM and NN can be seen on Table 1.

Tabela 1: Baseline Results

	Hyperparameters	**Train MAE**	**Devel MAE**
	RBF(C=1,g=0.01,d=1)	13.780	16.187
	NN(Linear(128-¿64-¿32)+Batch+ReLU+Dropout), sgd	11.236	15.540

The plots for the trining loss and the validation mean absolute error are shown in Figure 1 and 2. As expected, the training loss diminishes severely as the epochs pass. However, as we can see, the validation mean absolute error, although trending downwards, it is very noisy. Decreasing

the learning rate we could reduce the noise of the plot, but the train and development MAE were slightly worse, so we didn't change it.

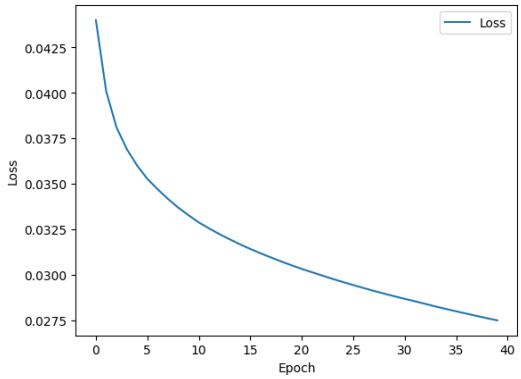

Figura 1: Training loss

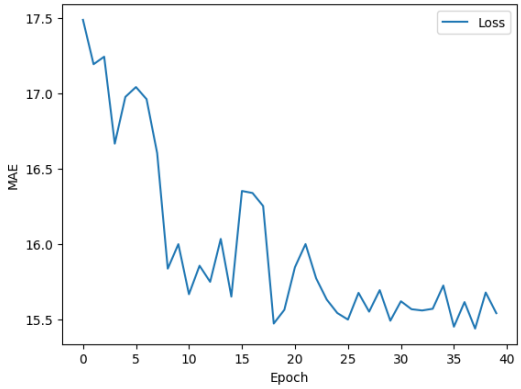

Figura 2: Validation mean absolute error

Q: What performance do you expect from your models on the test set (when compared with dev)? Same, better, worse? Explain your intuition.

We expect that the model performs worse on the test set compared with the dev set. The dev set is partitioned from the same dataset as the train set, so it is normal that the model, because the data used to train it was from the same source, performs better on the dev set. The test is a completely different dataset, and because of that we will be able to see if the model is correctly trained and not overfitted.

Part 2: Speaker Identification

Q: What is a threshold in this context? Can you use this threshold for other datasets?

We can define threshold as the decision of whether an embedding truly corresponds to a certain speaker or not. The threshold in this context is given by the condition of whether or not the embedding of a certain utterance corresponds to the average of the embeddings for that speaker. Normally the threshold could be something in the middle, like 0.5, but as we can see in the

metrics given by our calculated scores on similarity, we can see different thresholds for EER and minDCF.

Tabela 2: EER (ECAPA-TDNN)

EER	**EER Threshold**
0.020	0.333

Tabela 3: minDCF (ECAPA-TDNN)

minDCF	**minDCF Threshold**
0.0012	0.494

As we can see in the tables below, the threshold varies not only by the metric used, but also by the classification model used, in which we see a threshold of about 50% for the ECAPA-TDNN and a threshold of more than 90% for the XVector model.

Tabela 4: EER (XVector)

EER	**EER Threshold**
0.085	0.939

Tabela 5: minDCF (XVector)

minDCF	**minDCF Threshold**
0.008	0.959

So the threshold for a different dataset could vary and we conclude that this threshold could not be the best decision value in a different dataset.

Q: What if instead of selecting the best score, you based your decision on whether the score was higher or lower than the threshold you got on the previous task. Would the solution (speaker) for a given test utterance be unique?

If we used the threshold, we could have more than one result, since more than one speaker could have a score higher then that threshold, so the solution would not be unique for this case. Using a threshold means that our system can predict that all the scores that satisfy a minimum threshold are considered a possible speaker.