Language & Accent Classification.

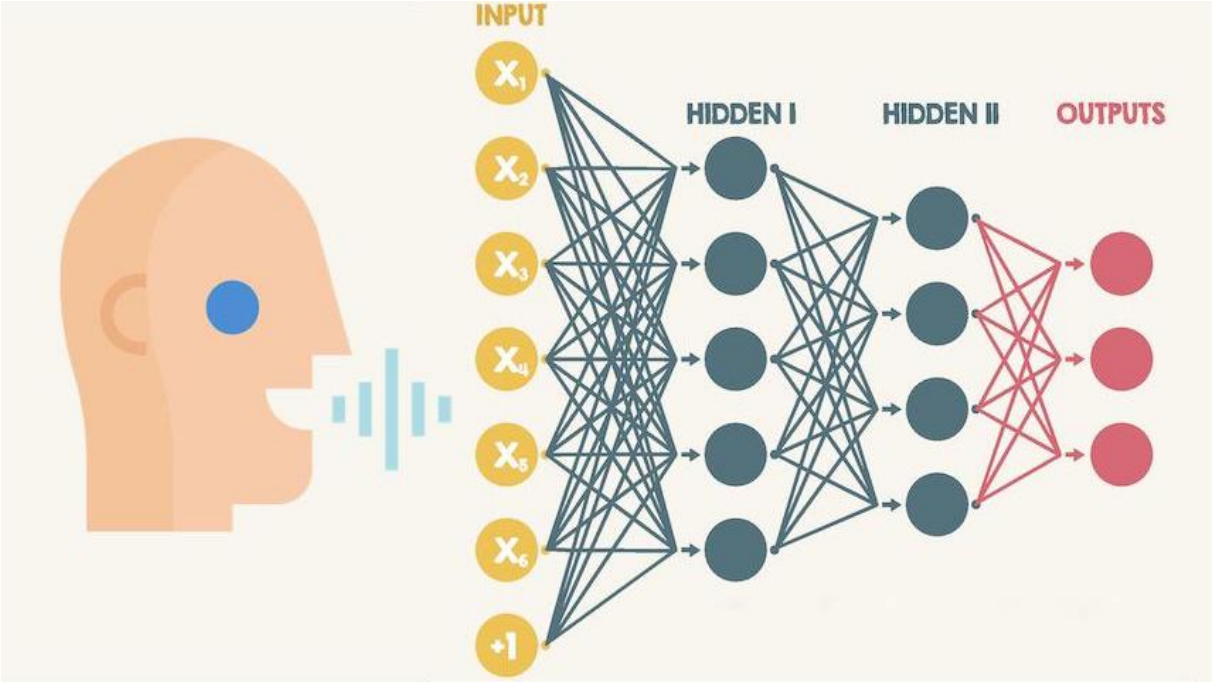
Nadav Epstein 205918980, Or Trabelsi 206066326

May 27, 2022

This article is written following a project we worked on during our first degree in computer science
The course focuses on Deep learning techniques for voice classification.
our lecturer for this course is Dr. Or Anidjar the CEO & Chief Data Scientist @ libonea.ai.
Link to our code :Our Project Github

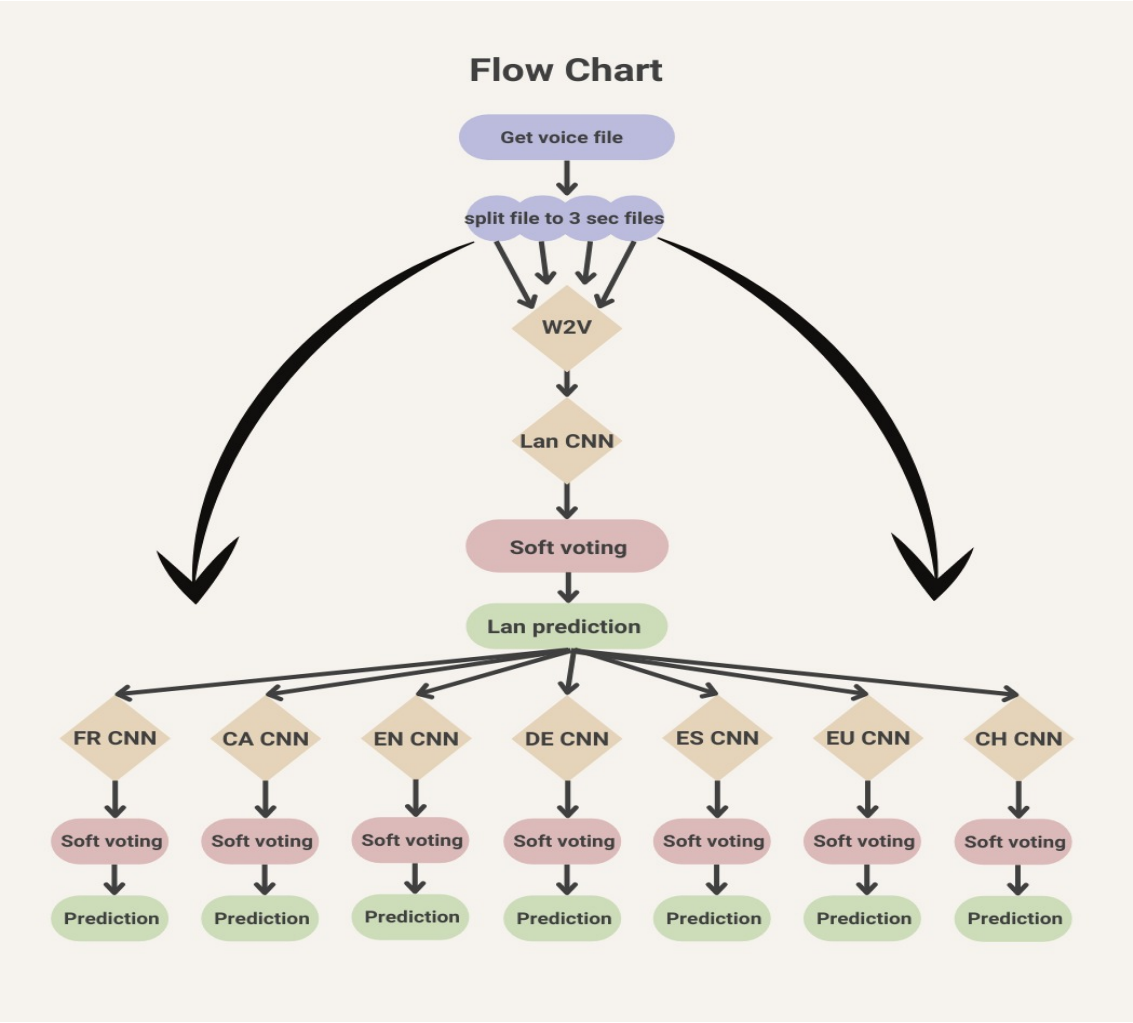
Abstract

In today's world we can learn a lot about a person just by hearing his voice. Our goal is to
identify just by voice a person language and accent which can help in a variety of sections such
as Marketing, Scamming, Security and etc.
In order to do that we used a model based on deep learning that classifies the language and after that the accent of that language.



1 Introduction

Most of the companies that deals with customer service via telephones records all the conversation's their employees had every day which means that every day they get another thousands of record hours, You're probably ask yourself "what are they doing with it?" , apparently not much the absolute majority of those recording just sitting there taking a lot of space without any use. That's where we get in by using our model (and other voice classification models) we can use that data to help to improve the customer service. How? using our model we can predict in less then a second what language the person on the other side of the phone is talking and not just that , also his accent. when we have this information we can use it to our own advantage by selecting a representative of the company that can speak the other person's language and even in the same accent, By doing that we already improved our customer service because we wont have problems of communication between the customer and the company representative and it only took one second. this is just one out of many problems our project can solve and improve. How does it work? We built out model using tagged data (Records that we already know what language and accent is heard) and we built a model using deep learning methods and trained him with the tagged data so he can learn how to tell just by voice what is the spoken language and accent. If deep learning is a new subject to you please click here before continuing.



2 Methodology

2.1 Gather Data

In order to start a project from this kind first of all we need to find tagged data in order to do that we visited Hugging Face website which is a data-set library for easily accessing and sharing data-sets and evaluation of audio tasks (also they have evaluation metrics for Natural Language Processing and computer vision).

2.2 Deal with Data

Once we have the data we want to split it to test and train we did it in a way that test was 2

0% of the data and train was 80% from the data. By doing it we made sure that after we trained our model it wont know the answers to the test data. It is important to check that we have all the languages in both test and train.

2.3 Build Model

After we have the data we need to build the model for the train (We will expand on our model later on in this article) basically the model will train him-self by the tagged data that we have and after we trained him long enough we can try and test the data that we kept in the test data and it will show us how good is the model.

2.4 Train Test and score

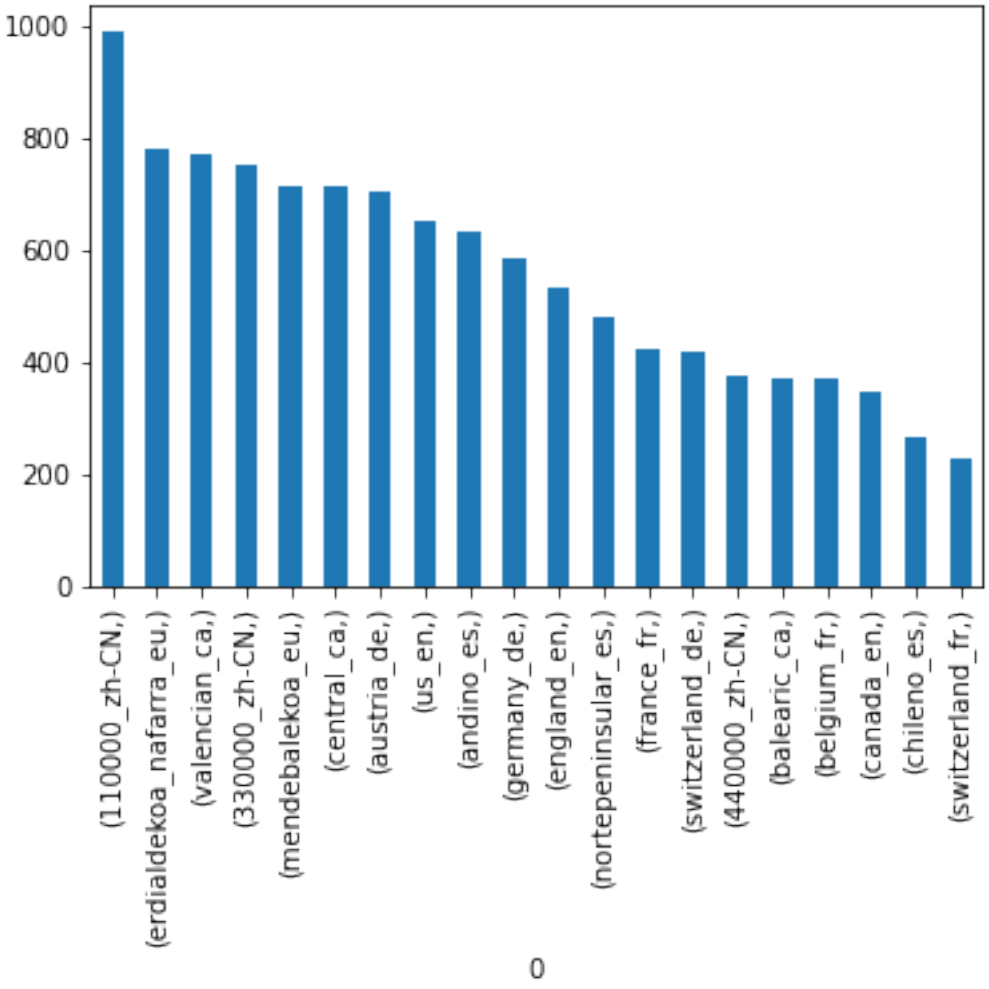
How do we know if the model had trained long enough? When we see that our percentage of the accuracy using F1 score(We'll expand about this method later on in the article) isn't going up any more or its going down we try to test the model. When we test the model we get the prediction and then we check the F1 score. Now we comapre the Train F1 score with the Test F1 score and we check if there's over-fitting if there isn't we can stop train the model.

2.5 Using the model

After the model is finally tuned and ready and we satisfied with our score we now can use it to predict new audio files. We'll get a new audio file as a live record from microphone or as a previous recorded file. Now the file will go through a series of preparation step and after that will enter the model and we'll get the desire prediction.

3 Data Set

As explained before we took our data from Data Set Source. at first we wanted to predict accent from all the languages at the data-set (59 languages) but we saw that the majority of the languages didn't had tagged accent so we could only use 7 languages (French , Catalan, English, German, Espaniol, Basque and Chinese). Each of those languages has few accents which means our classification is between 20 accents. We had a problem downloading all the data-set at once because of it size so we used "Datasets" library this library provides accesses to stream many data-sets from Hugging Face which help us to get all the records one by one and we can stop at any time after we have enough data.



4 Pre-Processing

Before we enter the data to the model it has to go through few steps.

1. **Balancing the data manually**: Because the data set had few languages with a lot of tagged accent and the others with much less we put a higher limit so we wont have to much from one language and it will cause over-fitting.

2. **Division by Language**: Each sample was sent to her own pickle divided by language.

3. **Casting data to 16,000 sampling rate**: We needed to make the data uniform so every audio had been cast to 16,000 sampling rate.

4. **Save & load the data using pickle** : Pickle is a binary protocol for serialization we use this protocol to save and load efficiently our data/model. we need it so we can work on the data continuously and save checkpoints along the data processing and the model train.

5. **Division to Train & Test** : We decided to divide ton 20% test and 80% train

6. **Division to three seconds** : We planned that the minimum time for a record will be 3 seconds and for every additional 3 seconds our model will preform "soft voting" which will improve our prediction significantly .

7. **Wav2Vec**: Now days there is a very popular approach to train complicated models with small data-sets. Using pre-trained models that were trained on huge data-sets and find tuning them to your specific purpose. This approach called transfer learning as we transfer the knowledge from the pre-trained model. Wav2Vec is one of the current state of the art models for automatic speech recognition. Because this model works on unlabeled data it help us to create a smart matrix that represents the audio file such that audio files that are similar will represented with similar matrix's.

5 Our Model

Our flow composed from 8 models. the first model predicates the language and for each language we have there own model that predicates the accent. All of the models architecture are the same and were built as follow :

5.1 Architecture

We have six convolution layers for each of the five first layers we will preform batch normalization and relu and for must of them we will preform max-pooling after the six layer we'll preform drop-out that will help us to avoid over-fitting. the last layer is fully connected that connects to log soft max. A little bit about the method we used:

1. **Convolution layer**: Is a deep learning algorithm which Uses the convolution operation and proved to be very efficient while preforming on computer vision, NLP and voice.

2. **Batch normalization**: Normalize the value for each mini batch this technique stabilizes the learning process.

3. **Max-Pooling**: We use this method to down-sample the feature map by taking the max for every X*X sub matrix as we chose.

4. **Relu**: Activation layer which gives the value X to every X bigger then 0 and zero to every X smaller then 0.

5. **Drop out**: We use this method to prevent over fitting by dropping X percentage of the input units to zero.

6. **Log Soft-Max**: This is a common function that helps to classify for multi classes by normalize the result. This function takes the exponent of the result and divided by the sum of the exponent of all the classes. The Log-softmax help us to spread the values to larger range that improves numerical performance and gradient optimization.

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K$$
$$LogSoftMax = \log \sigma(z_i)$$

5.2 Loss function

Our goal is to minimize the loss to zero(with the help of the optimizer). We use Cross Entropy loss function in order to do that. This function measures the model error

5.3 Optimizer

The goal of an optimizer is to minimize the loss. We choose the Adam optimizer this optimizer is a fusion of two common optimizers SGD and RMS. This optimizer uses moving average to get to the minimum faster and increase very small steps and decrease very large steps.

5.4 Hyper Parameters

The main hyper parameters for the model are :

1. **Learning rate** : this parameter decides the step size the optimizer will make in each iteration in order to minimize the loss.

2. **Epochs** : number of iteration over all of the data.

3. **Batch size** : number of samples for each iteration.

4. **Class weights** : for imbalanced data multiple the weights for the low class size.

For every Class (language) we find the optimal parameters (from our attempts) and every one of them had different parameters in consideration towards the class data.

5.5 Final Prediction and Scoring

1. **Prediction**

   After the model is trained we can evaluate the model quality and predict new samples. As explained in the pre-processing section we take each audio file and split it every three seconds. Every sub sample (long 3 seconds) is sent to the model and we get the model prediction by a vector of probabilities in the length of the classes(this is the output of the softmax).The model prediction will be the argmax of this vector. How do we know which one to choose? In a perfect world all of the predictions should be the same but we know that the reality isn't perfect so we use soft-voting method, soft voting combining the probabilities of each prediction and picking the prediction with the highest total probability. We return the top 3 accent prediction because of leak of data the results of the model wasn't as we expected.

2. **Scoring** We choose to evaluate our model using F1 score and confusion matrix (as you can see in the last page)

$$F_1 = 2\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}$$

6 Previous attempts (failures and solution's)

6.1 Data without filter

At first we tried to use one model to predict accent (without predicting language first) we even took every language without checking whether it has enough tagged samples. After we tried to train the model we saw that we had a lot of languages which have only few tagged samples

How did we solve that problem? We deleted every accent that had less then 200 tagged samples.

6.2 Imbalanced data

We had few accents with huge tagged samples and few with only few hundreds of tagged samples which cause to high accuracy and low F1 score .

How did we solve that problem? We blocked the tagged samples that will only take 1000 samples maximum from each accent. but the data was still a little bit unbalanced so we decided to use class weights which helped us to give more weight to the classes with lower data size.

6.3 Too many classes

When we tried to predict the accent straight away without knowing the language first we saw that our model wasn't strong enough (because of the leak of data) and we didn't had good result.

How did we solve that problem? We decided to create a model for each language that will predict her accent and we created another model that predicates the language first. Then instead of 20+ classes we needed first to classify between 7 languages and then predict from each language her accent which is usually between 2-3 classes. that resulted a big improve on our model.

7 Development Environment

Because the process of training model is very rough and our computers aren't strong enough to train the model in time. We used google colab that give us access to free GPU and faster training. google colab also have verity of options if you are a paying member such as stronger GPU, you can train the model while the computer is shut down etc.

7.1 Main libraries that we used

pandas , pickle, Pytorch (torchaudio), numpy, seaborn, cv2.

8 Appendices

1. **classification report**:

   EN Test accuracy score: 65 %
   EN Test F1 score for 1: 73 %

   CA Test accuracy score: 58 %
   CA Test F1 score for 1: 56 %

   FR Test accuracy score: 51 %
   FR Test F1 score for 1: 64 %

   EU Test accuracy score: 75 %
   EU Test F1 score for 1: 78 %

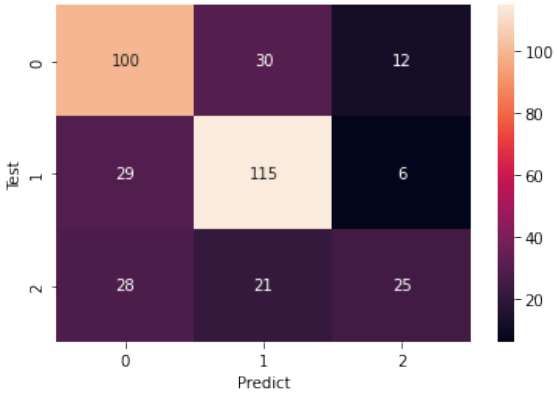
   ZH Test accuracy score: 66 %
   ZH Test F1 score for 1: 66 %

   ES Test accuracy score: 65 %
   ES Test F1 score for 1: 63 %

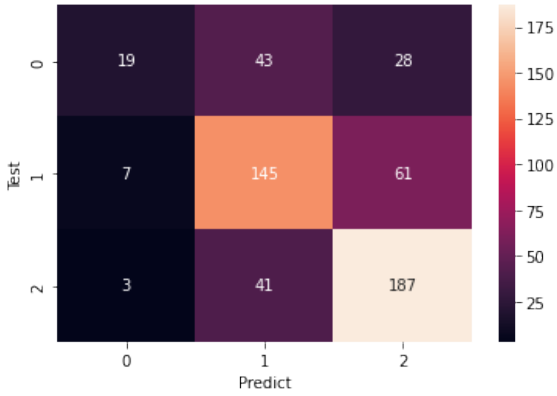
   DE Test accuracy score: 50%
   DE Test F1 score for 1: 53%

   Lan Test accuracy score: 82 %
   Lan Test F1 score for 1: 85 %

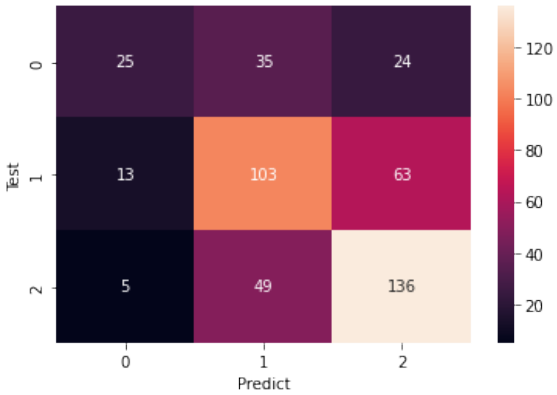
2. **Confusion matrix**:



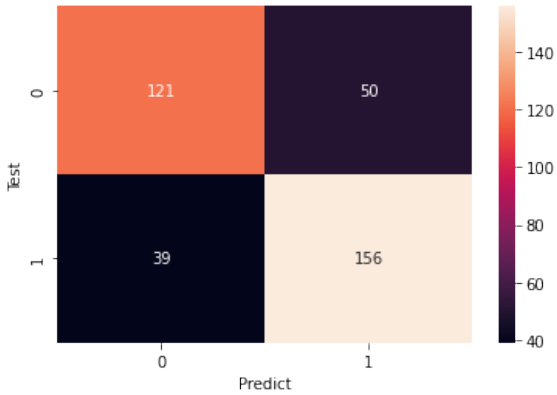
(a) EN



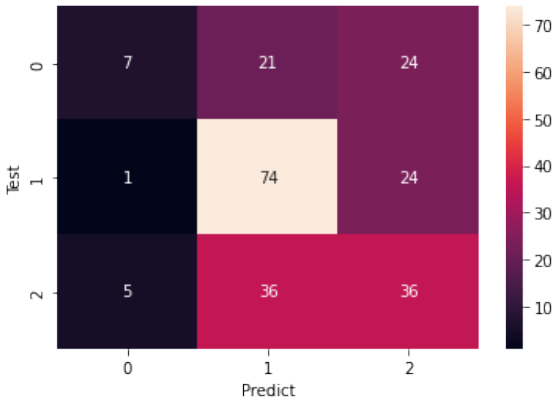
(b) CH



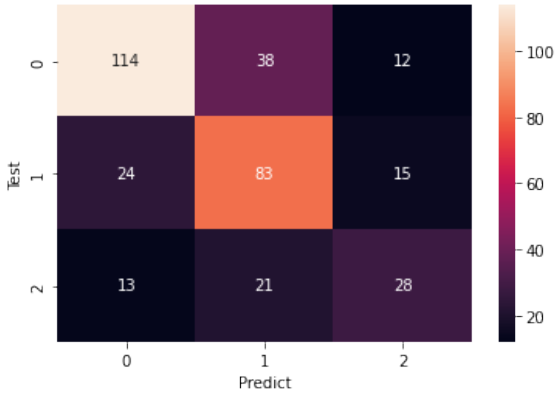
(c) CA



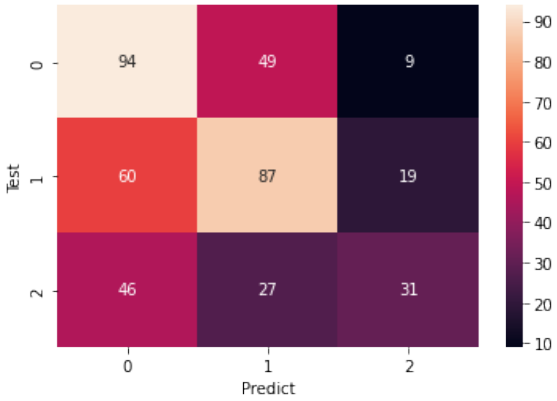
(d) EU



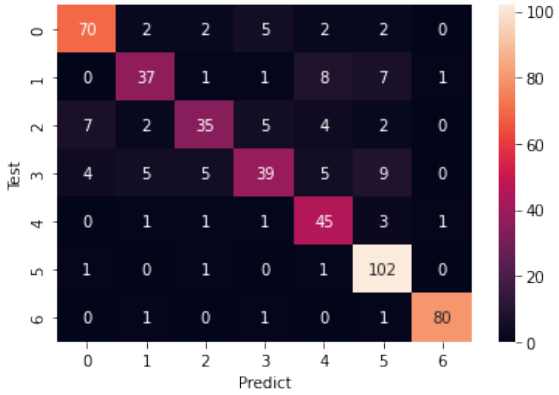
(a) FR



(b) ES



(c) DE



(d) LAN