Audio Signal Processing for Language Classification Using Machine Learning

Márcio Lopes Júnior
Technology Center
Federal University of Rio Grande do Norte
Natal, Brazil
marciolopesjr@ufrn.edu.br

***Abstract*—There are many uses for audio transcription nowadays, and audio language classification can be very useful to help improve transcription accuracy in scenarios where the speaker's language is unknown. In this work we explore two different ways of targeting language classification: using extracted features, when we created a numerical matrix of data from the original signals and processed it with a random forest classifier; and using convolutional neural networks, when we used spectrograms from the original signals to train a neural network for language classification. Results show that both models are capable to differentiate the languages nicely even with few samples and features.**

***Index Terms*—machine learning, audio signal processing, classification, neural networks**

I. Introduction

Probably the most well-known kind of signal, audio has a plentiful of applications, and there's a rich history of audio scientific analysis. In the world of smartphones and smart homes, when voice has gained an even bigger importance as it is now a standard way of communicating with machines, these studies are extremely important to communication.

One major problem of modern tech industry, then, is to make sure their devices can properly understand what their user says. Natural Language Processing (NLP) is not a trivial task, and popular used applications for speech recognition, such as the ones attached to messaging apps, usually ask the users to pre-define the language they're speaking before "guessing" what was said. This is perfectly fine for that situation, and definitely will help the algorithm's chances of a right answer, but what about a situation when you don't know who's talking? It can happen, for instance, when dealing with tourists (public service) or with unknown messages (public safety).

In order to improve the chance of guessing what a user says, we could first try to guess in what language they are speaking. And to do so, we would have to guess it without identifying the words. All we can use is the audio signal and its properties.

This paper will explore the potential of using audio signals to predict spoken language by testing two model types: the first is based on extracting numerical features from the audio signal, and then processing those features to train a machine learning model with them; the second uses a popular neural network algorithm – Convolutional Neural Networks – and trains it with images of each audio signal's mel spectrogram.

II. Methodology

A. Dataset

It was necessary to have a multi-language dataset available in order to develop this project. Here, the *Mozilla Common Voice* dataset was chosen based on the dataset size, variety, and also the fact that it is publicly available.

The Mozilla Common Voice is a project by the Mozilla Foundation that intents to serve the community with an open-source audio signal database in order to make voice recognition open to everyone. The Common Voice dataset currently has audio data available for use in 29 languages.

Every signal is sampled at a similar rate (48kHz) and only signals lasting at least 5 seconds were used in the processing. All the used signals were also limited to 5 seconds, with part of the audio being removed beforehand.

B. Data Extration

Working with signals (or time-series) is a common machine learning problem and there are several ways of dealing with that. In this case, the goal is to classify the signal into n categories (the languages). To find these categories, we need meaningful data that can help our model define what it should label each signal.

Having the whole signal information – or a resampled copy of it – could be important if there is a need to predict future data points or when the timing of a future event is more important than an older event – and vice-versa. But this is not the case in this problem, here, we're just trying to use anything a user said to predict their language, regardless of when. Thus, we can forget about the signal itself, once we have defined the features that can help us classify it.

There are several features that can be extracted from digital signals, both in time domain and frequency domain. Here, the frequency domain signal is calculated by the Fast Fourier Transform.

Six different features were extracted from the signals: (1) spectral centroid, (2) zero crossing rate, (3) spectral contrast, (4) spectral flatness, (5) spectral rollof, and (6) root-mean-square.

Each of these features were extracted for different time range splits, and after all features were calculated by range, four properties were calculated per feature: *minimum*, *maximum*, *mean*, and *standard deviation*.

1) Spectral centroid: The spectral centroid of a signal is the mean magnitude value of the spectrum after normalization.

$$centroid[t] = \frac{\sum_k S[k,t] * f[k]}{\sum_j S[j,t]}$$

Where S is the computed spectrogram, and f is the range of computed frequencies. [1]

2) Zero Crossing Rate (ZCR): ZCR is defined as the number of times (in time domain) where zero-crossing happens. Zero-crossing is when the magnitude of the signal goes from positive to negative, or from negative to positive. [2]

$$zcr = \frac{1}{T-1} \sum_{t=1}^{T-1} 1_{\mathbb{R}_{<0}}(s_t s_{t-1})$$

That is, if s_t multiplied by s_{t-1} is less than zero (negative), then s_t and s_{t-1} have different signs and zero is crossed between the two. When that happens, 1 is added to the summation, and at the end of the sample of size $T-1$, the result is divided by that size.

3) Spectral Contrast: ratio between the magnitude of *peaks* and *valleys* in the frequency spectrum. *Peaks* and*valleys* are names given to the top and bottom quantiles in the spectrum. [3]

4) Spectral Flatness: a property created to measure how tone-lie or noise-like a signal is. [4] If the spectral flatness is close to 1, than it means the signal is close to white-noise.

$$SF = \frac{e^{(\frac{1}{T} \sum_{t=0}^{T-1} \log S_{max})}}{\frac{1}{T} \sum_{t=0}^{T-1} S_{max}}$$

Where S_t is given by the maximum value in :

$$S_{max} = \max S^2$$

5) Spectral Rolloff: The Spectral Rolloff is a frequency (f_{roff}) in which the sum of its magnitude and the magnitude of all lower frequencies combine to 85% of all the spectrum magnitude. Its calculation is given by:

$$\sum_{n=1}^{f_{roff}} M_t[n] = 0.85 * \sum_{n=1}^{N} M_t[n]$$

Where N is the total number of frequencies in the discrete frequency spectrum. [5]

Although having a different – probably more complicated – calculation process, it's close to the statistical idea of a *percentile*.

6) Root-Mean-Square (RMS): The RMS is a feature that is calculated by taking the square of every value in a specific range, summing these values and then getting the squared root of the sum. It is commonly used to measure error in machine learning projects, bu here it is used to help us represent the power of the signal.

$$rms = \sqrt{\frac{1}{T} * \sum_{t=0}^{T-1} {s_t}^2}$$

C. Mel Spectrogram

In the neural networks experiment, the model will be trained with images that can represent the data – taken from their spectrograms. And each of these will be rescaled into the Mel Scale.

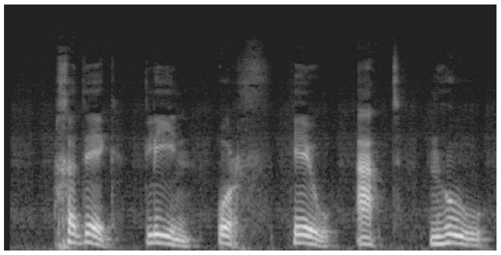

Fig. 1. Mel scale spectrogram of Welsh language audio.

1) Spectrogram: A spectrogram is a visual representation of a signal, or more specifically, of its intensity over time and frequency. The y-axis of a spectrogram shows the frequencies, the x-axis shows the time, and a third dimension shows the intensity (or amplitude) of the signal at each <time, frequency> point.

2) Mel Scale: The mel scale is a frequency scale born the 1940s through experiments on the human auditory system. The experiments showed that human perception of audio is not linear to frequencies higher than 1000Hz. Then, in order to have a scale that approximates better to how humans *feel* the difference, the mel scale was developed. It is a quasi-logarithmic scale and it is given by the formula:

$$mel = 2595.0 * \log_{10}(1.0 + f/700.0)$$

Where f is the original frequency in hertz.

D. Convolutional Neural Networks

A special set of neural networks, convolutional neural networks, or CNNs, differentiate themselves from other networks because of their *convolutional layers*. These layers apply a kernel (or filter) over the input data, extracting characteristics through a 2-dimensional convolution process. The CNN job, then, is focused on optimizing the kernel's values so that it can find the right characteristics to improve the model's performance.

CNNs are extremely popular and efficient in *image processing*, [6] and when we extract the Mel Spectrograms from the signals, we are in fact making our signal processing problem also an image processing problem. In that sense, CNNs are expected to perform well on finding out what defines each language.

The model developed for this project is shown in Fig. 2. It consists of (1) four convolutional layers with different kernel sizes, (2) a max-pooling layer for each convolutional layer, and (3) three dense layers, with the last of these being the output layer. The final score is calculated by a *Softmax* activation function, as it is one of the standards on multi-classification problems.

In Fig. 2, the depths of the layers represent the number of neurons in each of them. And the larges vector, of size 65536 is a flattening layer, responsible for turning the 32x16x128 tensor from the last max-pooling layer into a one-dimensional vector.

III. Results

The two models were tested on audio samples of four different languages: Spanish, Dutch, Welsh, and Basque (Euskara). Basque is a language spoken in the Basque Country, a region of Spain, which makes it heavily influenced by Spanish speakers, and the choice of having both languages is also to test if they are understood as more similar by the trained model, even though they're very distinct grammatically.

The first model was tested on a set of 4,000 audio samples, 1,000 from each selected language. The second model was tested on 5,000 audio spectrograms, 1,250 from each language, where 250 were for testing, and 1,000 for training.

A. First model (Random Forests)

In the first model, the data went through a few preprocessing steps before being fed to a classifier: (1) The data was centered and scaled to unit variance, (2) new polynomial features were created from the product of original features, (3) and each column was rescaled to values between 0 and 1.

In Fig. 3, the results from one of the outputs of the model can be seen. As it shows, the model was able to correctly classify the samples most of the time, with Dutch having the lowest accuracy. Spanish and Basque were the most commonly mistaken language for each other, which may indicate some closeness.

TABLE I
Results from first model test

Language	Precision	Recall	F1-score
Welsh	0.59	0.62	0.61
Spanish	0.55	0.51	0.53
Euskara	0.59	0.65	0.62
Dutch	0.57	0.52	0.54
Overall	**0.58**	**0.58**	**0.58**

B. Feature Importance

The first model was run in a slightly different configuration – without the polynomial features – in order to get a importance score for each feature from the RF classifier. Importances were taken from 3 folds of the dataset and the averaged was calculated.

As it is shown in Fig. 4, the feature that most strongly affected the model's decision was the standard deviation of the Spectral Flatness, followed by the maximum and mean of the Zero Crossing Rate. The minimum values of the features had the lowest importance for all but the Spectral Contrast – where it had the second lowest.

C. Second model (CNN)

The second model, the CNN, had only one preprocessing step, which was the transformation of each signal into images. The process involved loading the audio file, getting its spectrogram, and creating an image from it.

In Fig. 5 we can see the results for the second model. It's clear that it could predict correctly more often than the first model. With an increase of about 15% in accuracy, as shown in Table II. The least well predicted language was Spanish, which was mistaken by Basque in 18.4% of its samples.

TABLE II
Results from second model test

Language	Precision	Recall	F1-score
Welsh	0.77	0.74	0.75
Spanish	0.76	0.63	0.69
Euskara	0.67	0.78	0.72
Dutch	0.75	0.78	0.77
Overall	**0.74**	**0.73**	**0.73**

IV. Discussion

A. Features

When the used features were selected, the first idea was to use the maximum and mean values, and that the maximum would be most important as it could detect the outlying nature of some languages. Actually, what the RF classifier shows (Fig. 4) is that the mean was a more meaningful property than the maximum in all but one feature. Also, of the two later added properties – minimum and standard deviation – the first had the smaller impact of all properties, while the second proved to be very useful to some features – *spectral flatness* specially.

The importance of the *spectral flatness* is an interesting discovery. It could be that the model has identified some ways of speaking which are closer to some white-noise properties than others, but it could also be a problem in our dataset, that some collections of audio might had more noise than others and that affected the model output.

It's also notable that Spanish and Basque had the closest similarity. I believe that a more robust model, with a high classification accuracy to a larger number of languages, could be used as a tool to calculate language closeness.

B. First model

Considering the limited number of features used, and the also limited number of samples – both of which were due to processing power and time limitations – the results were

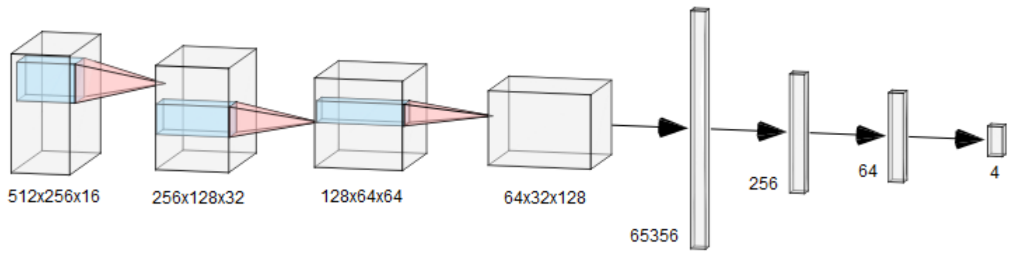

Fig. 2. Convolutional model architecture.

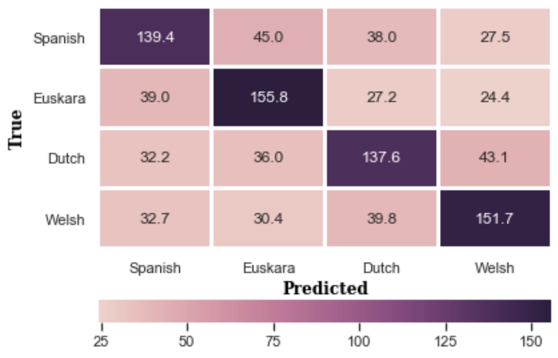

Fig. 3. Confusion matrix of the Random Forests classifier predictions (average over 20 tests).

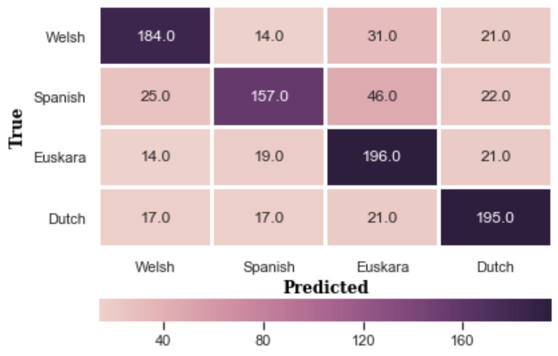

Fig. 5. Confusion matrix of the CNN model predictions on single test.

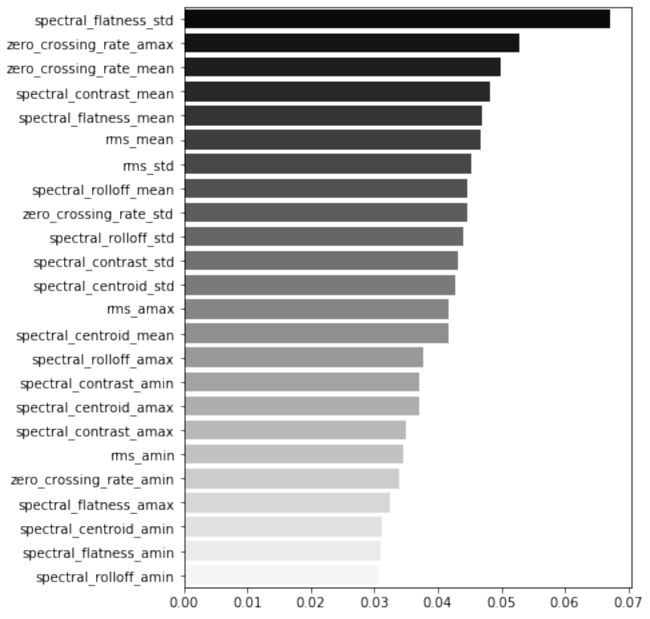

Fig. 4. Feature importance as calculated by Random Forests.

satisfactory, with accuracy being close to 60%. In a four-classes scenario, random guessing is expected to have a 25% accuracy, so having a much better value, shows that some – or many – of the features are proving to be meaningful to the problem of language classification.

Our model is very simple and it uses only a few of the many possible features to be extracted from signals. In a future research, with a better selection of extracting features, it should be possible to strongly improve this accuracy results. Still, it's important to point out that the model was tested with only 4 languages, and if the model is developed to be used in a real world application, it needs to take into account at least all the widely spoken languages. That being said, the features used here should have a worse result once the size of the problem increases.

For a more complete study on the subject, some deeper study on audio signal analysis and human languages should be done in order to find richer (more meaningful) features.

C. *Second model*

The CNN had some very good results when compared to the first model. It could achieve an accuracy slightly higher than 70%. It was also possible to observe from Table II how Spanish and Basque are the most similar of the four, as it causes the most confusion in the prediction.

The model training with the examples here only used 1,000 audio samples per language – total of 4,000. The ImageNet dataset, a famous image processing problem has more than 10,000,000 images, showing there's still a huge room for growth in data collection.

There was purposely no use of a state-of-the-art architecture here, as we're only exploring the possibilities of using these types of algorithms, but the CNN results did well enough with the limited 4,000 training images provided. The results of the CNN with little without an optimal architecture and with limited resources support the highly accurate results obtained in [7] when dealing with the same problem, but using a different dataset.

V. Conclusion

We have developed two models in order to explore the possibilities of language classification in audio signals. The first model showed it was possible to classify about 60% of the signals correctly by using only a few extracted features.

The second model was had an improvement of about 15% accuracy when compared to the first, making a good impression that CNN classifiers are a good and efficient option to classify signals.

Although the results were not good enough for real world applications, they were able to prove that both models are capable of dealing with our classification problem. If trained with larger amounts of data, and if more meaningful information are extracted from the signals, machine learning algorithms will likely be able to efficiently predict languages without any grammar, morphology or syntax.

References

[1] G. Peeters. "A large set of audio features for sound description (similarity and classification) in the CUIDADO project". 2004.

[2] D.S. Shete and S.B. Patil. "Zero crossing rate and Energy of the Speech Signal of Devanagari Script". IOSR journal of VLSI and Signal Processing. 4. 01-05. 2014.

[3] D. Jang, M. Jin and C. Yoo. "Music genre classification using novel features and a weighted voting method". 2008 IEEE International Conference on Multimedia and Expo, ICME 2008 - Proceedings. 1377 - 1380. 2008.

[4] S. Dubnov. "Generalization of Spectral Flatness Measure for Non-Gaussian Linear Processes. Signal Processing Letters," IEEE. 11. 698 - 701. 2004.

[5] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," in IEEE Transactions on Speech and Audio Processing, vol. 10, no. 5, pp. 293-302, July 2002.

[6] F. Sultana, A. Sufian and P. Dutta. "Advancements in Image Classification using Convolutional Neural Network". 122-129. 2018.

[7] S. Revay and M. Teschke. "Multiclass Language Identification using Deep Learning on Spectral Images of Audio Signals". 2019.