Language Detection using Spectral Analysis

Aviral Singh

Vellore Institute of Technology, Chennai Tamil Nadu, India

Yug Agarwal

Vellore Institute of Technology, Chennai Tamil Nadu, India

**Abstract— The identification of languages in daily life is an extremely important field of work due to the large number of languages that are used in the world today. Accurately determining a language through speech is a very tricky endeavour since various differences can be observed in the same language which are caused by variations in speech patterns. This project aims to identify the language being spoken in a given audio excerpt using its phonetic properties rather than using audio – to – text transcription. The mechanism that will be used will extract spectrograms from a given audio file. These spectrograms will be used to train a popular image detection deep learning model which will facilitate the detection of a language with high accuracy.**

**Keywords— Convolutional Neural Network (CNN), Spectrograms, Short-Time Fourier Transform (STFT)**

1. **INTRODUCTION**

In today's digital age, with the proliferation of multimedia content, there is a growing demand for robust language detection systems capable of accurately identifying spoken languages from audio data. Traditional methods often struggle to cope with the complexities of diverse accents, environmental noise, and linguistic variations. Therefore, there's an urgent need to develop deep learning models tailored for audio-based language detection that can effectively handle such challenges. These models must exhibit high accuracy across a wide range of languages and dialects, while also being computationally efficient for real-time or near-real-time processing. Addressing this requires the design of sophisticated deep learning architectures, extensive labelled audio datasets covering diverse linguistic contexts, and considerations for practical deployment in real-world scenarios, such as multimedia content analysis, voice-controlled applications, and language interpretation services.

The aim of this paper is to implement a robust and versatile language detection system which does not focus on existing methods of speech-to-text transcription. The system should be able to detect the language being spoken in a given audio file with acceptable accuracy by analysing audio characteristics of an audio excerpt. The mechanism is trained on a dataset of containing audio files in 4 languages, namely, English, Arabic, French and Japanese. Each audio file in the dataset is converted into a spectrogram image, which is then fed into a pre-trained Convolutional Neural Network (CNN). This network predicts probabilities for the 4 languages and the highest predicted probability is considered as the prediction for the mechanism.

1. **LITERATURE REVIEW**

Language detection in audio is an exciting space of research due to recent advancements in Natural Language Processing technology. Existing research focuses on transcription and feature vectorization to accurately detect a spoken language in an audio file [1][4]. Unsupervised learning methods were used to extract audio metrics and the data was trained on classifiers such as SVM, among many others. Most research projects on audio detection use language detection as a subsystem for a larger mechanism and rely on pre-existing Python libraries to implement such a system for their needs [3]. Audio hotspotting has also been used in recent literature [2], which is also known as keyword spotting or audio indexing. It is a form of information retrieval employing speech recognition that is used for quickly identifying passages of interest within audio files. Among these methodologies, language detection in broadcast audio with ensemble learning has also been explored without the use of deep learning-based mechanisms [5].

1. **METHODOLOGY**

The dataset that was used for this task was sourced from Kaggle [6]. The dataset contains 1000 samples in total with 250 samples for each of the following languages:

* Arabic
* English
* French
* Japanese

The sampling rate for each audio file in the dataset is 16000 Hz. The first step in detecting the language being spoken in a given audio file is to convert the audio file into a usable numerical format.

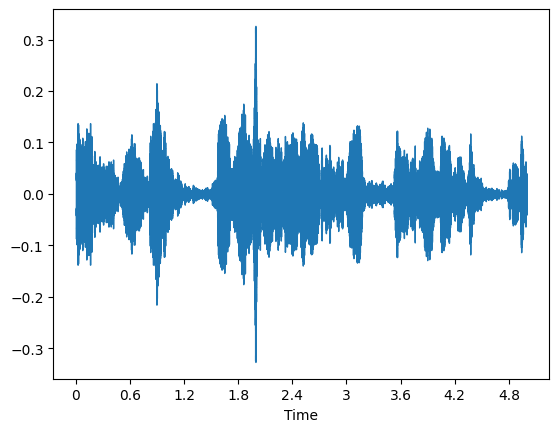


Fig 1: Sample waveform of an audio file in the dataset

After converting the audio waveform into a usable format of a numerical matrix, further preprocessing steps are applied.

Since a major part of characteristics analysis of audio is frequency analysis, the obtained audio waveform is transformed from the time domain into the frequency domain. This is done because differences in human languages can be observed by measuring variations in frequency because different languages have varying phonetic patterns, causing variations in frequency signatures in continued speech.

The conversion is performed by employing the Short-Time Fourier Transform (STFT). The Short-Time Fourier Transform (STFT) of a signal is a way to analyze its frequency content as it evolves over time. It can be defined mathematically as follows:

Given a signal , the STFT is computed by dividing the signal into short segments and then taking the Fourier Transform of each segment. Let be the input signal, be the window function, and be the window's time duration.

Segmentation and Windowing

A windowed segment at time may be defined as:

Here represents a window function centered at time , which helps focus on a specific segment of the signal.

Fourier Transform

The Fourier transform of each windowed segment to obtain its frequency representation:

Here, is the frequency representation of the signal at time and frequency .

Time-Frequency Representation

The result is a 2D representation of the signal in the time-frequency domain, typically denoted as , called the Short-Time Fourier Transform:

This representation shows how the frequency content of the signal changes over time. The STFT essentially captures how the frequency content of a signal varies over time by examining short segments of the signal and analysing their frequency components. It is an extremely powerful tool for analysing time-varying signals in various fields, such as audio signal processing, speech recognition, and vibration analysis.

The next pre-preprocessing step is to convert the present waveform’s amplitude scale into a decibel scale. This is done to differentiate between the sound intensities of various sections of speech in a given language. This can be done using the following equation:

dB = 20 \* log10(amplitude)

Decibels are different from other familiar scales of measurement. While many standard measuring devices, such as rulers, are linear, the decibel scale is logarithmic. This kind of scale better represents how changes in sound intensity actually feel to the human ear.

After these pre-processing steps are applied to the input data, the resultant data is converted into a spectrogram to obtain the visual characteristics of an audio file. A spectrogram is a visual representation of the frequencies of a signal over time, and can also show the signal's strength at different frequencies. Spectrograms are sometimes called voiceprints, sonographs, or voicegrams when applied to audio signals. When the data are represented in a 3D plot, they may be called waterfall displays.

A spectrogram shows the frequencies that make up the sound, from low to high, and how they change over time, from left to right. It can reveal broadband, electrical, or intermittent noise in audio. For example, a spectrogram can show whether there is more or less energy at 2 Hz vs 10 Hz, and how energy levels vary over time.

The spectrogram will be obtained by calculating the magnitude of the previously obtained frequency representation from applying the Short-Time Fourier Transform (STFT) to the input audio signal. The magnitude spectrum from the complex FFT results is calculated by taking the absolute value:

The magnitude spectra of each windowed segment is arranged over time to form the spectrogram . This can be represented as a matrix or a 2D plot, where the horizontal axis represents time , the vertical axis represents frequency , and the color or intensity represents the magnitude of each frequency component. Some sample example spectrograms are showcased in Fig. 2.

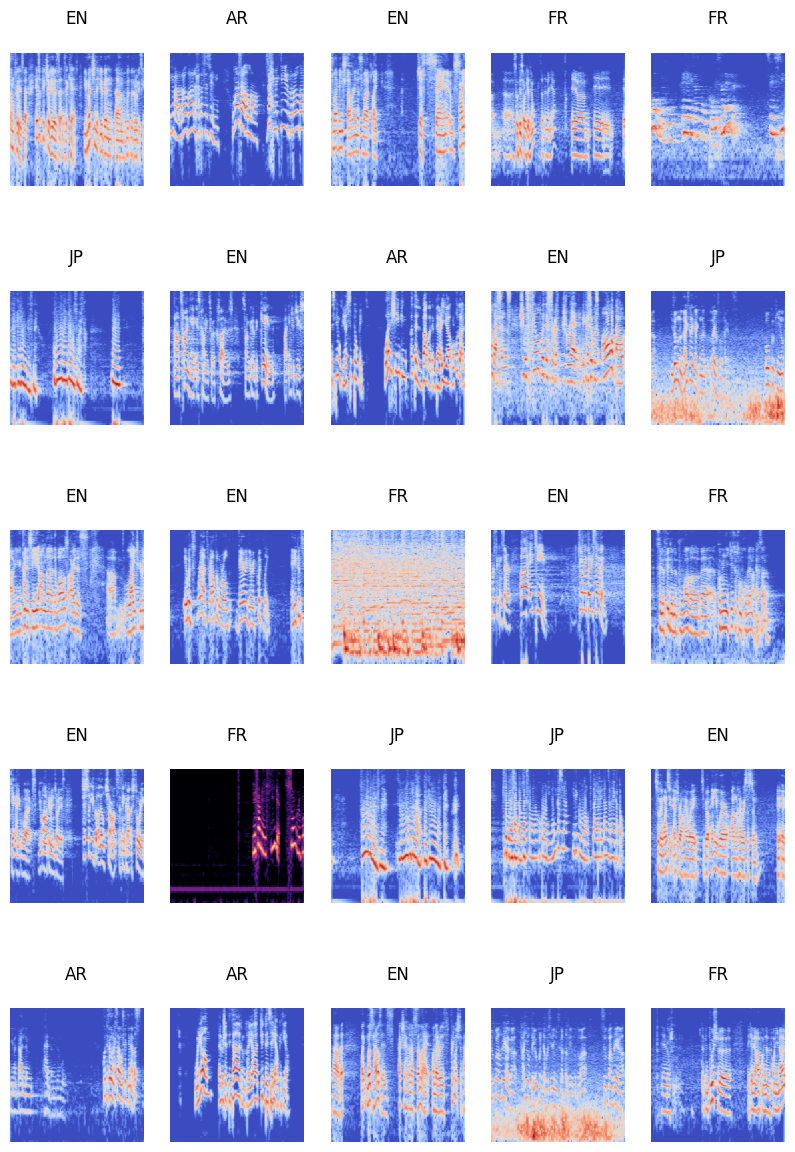


Fig 2: Sample spectrograms for different languages in the dataset

1. **MODELLING AND TRAINING**

After obtaining the spectrograms for each audio file in the dataset, the next objective is to find a suitable deep learning model which can differentiate between the subtle phonetic characteristics of spectrograms for different languages in the dataset. Many pre-trained deep learning models are available which are extremely proficient in image detection tasks. Most of these models have variations of the Convolutional Neural Network (CNN) architecture. A Convolutional Neural Network (CNN) is an extremely powerful tool for image detection tasks. It is a deep-learning based mechanism which focuses on generating abstract representations of images which can be used to separate various entities in an image. These abstract representations are generated through various operations which are given below:

* Convolutional Layer: Every Convolutional Neural Network (CNN) contains a certain number of convolution layers. Each layer consists of a set of learnable filters or kernels. These filters are convolved with the input data, performing a dot product between the filter and a small region (receptive field) of the input data. This operation results in feature maps, which represent the presence of specific features or patterns in the input data.
* Pooling Layer: Pooling layers are often used after convolutional layers to reduce the spatial dimensions of the feature maps while retaining important information. Common pooling operations include max pooling and average pooling, which down-sample the feature maps by taking the maximum or average value within a small region.
* Activation Functions: Activation functions introduce non-linearities into the network, allowing CNNs to learn complex mappings between the input and output. Common activation functions used in CNNs include ReLU (Rectified Linear Unit), sigmoid, and tanh.
* Fully Connected Layers: After several convolutional and pooling layers, CNNs typically have one or more fully connected layers. These layers connect every neuron in one layer to every neuron in the next layer, enabling the network to learn complex relationships between high-level features extracted by previous layers.
* Loss Function and Optimization: CNNs are trained using a loss function that measures the difference between the predicted output and the true labels. Common loss functions for classification tasks include cross-entropy loss. The weights of the network are updated using optimization algorithms such as stochastic gradient descent (SGD) or its variants (e.g., Adam, RMS Prop) to minimize the loss function.
* Training: CNNs are trained using large datasets through a process called backpropagation, where gradients of the loss function with respect to the network parameters are computed and used to update the weights iteratively.

The model which is employed for the task at hand is the MobileNet V2[7]. The MobileNet V2 model is extremely performant in object detection tasks in images and is built to be operable on lightweight devices such as mobile phones. This model has been chosen because of these aforementioned qualities which will help the deployment of this mechanism in a relatively low-power environment. The architecture for the MobileNet V2 is illustrated in Fig. 3.

The model takes an image of size 224x224 pixels as input. The pre-trained version of this model is publicly available for use online and is used for various image detection tasks. A final output layer is attached to the model which produces the output probabilities for each class in the dataset. A train-test split of 90% and 10% has been used for this task due to the low number of samples available for training. The mechanism may be generalised and used for other languages and with more extensive datasets.

The output layer will consist of 4 neurons corresponding to each language class. The softmax activation function will be used to produce the probabilities of each class. The softmax activation function is used for multi-class prediction tasks and outputs a set of likelihood probabilities for each class in the dataset. It can be mathematically represented as follows:

where is the input vector and N is the total number of elements in the input vector. By dividing each exponential score by the sum of all exponential scores, the softmax function normalizes the input vector into a probability distribution.

The loss function which will be used to evaluate the chosen model while training is categorical cross-entropy. Categorical cross-entropy, also known as softmax cross-entropy or simply cross-entropy loss, is a commonly used loss function in machine learning, particularly in multi-class classification problems. It quantifies the difference between the predicted probability distribution and the actual probability distribution of categorical variables. The mathematical equation for the same is given below:

where is the true probability distribution for class , is predicted probability distribution (output of the softmax function) for class and N is the total number of classes.

The optimization algorithm chosen for training the deep learning model is the Adam [8] optimizer. The Adam optimizer is a popular optimization algorithm used in training deep learning models, particularly in neural networks. It combines the ideas of momentum-based optimization and adaptive learning rates to achieve faster convergence and better generalization.

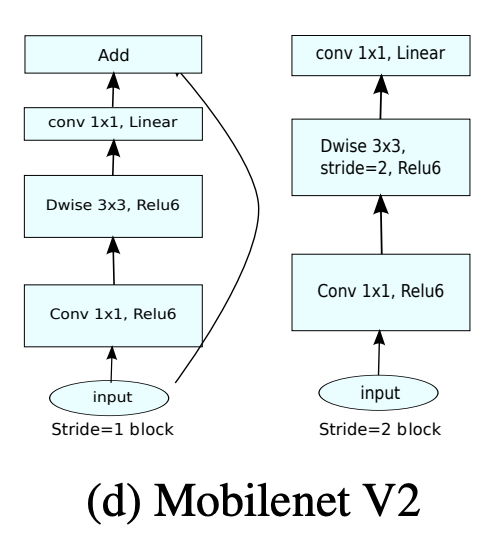


Fig 3: Architecture of the MobileNet V2 model [7]

1. **RESULTS**

The model used for this task will be evaluated using the following metrics:

* Confusion Matrix
* Accuracy Score
* Precision Score
* Recall Score
* F1 Score

The confusion matrix obtained for the model in use is given in Fig. 4.

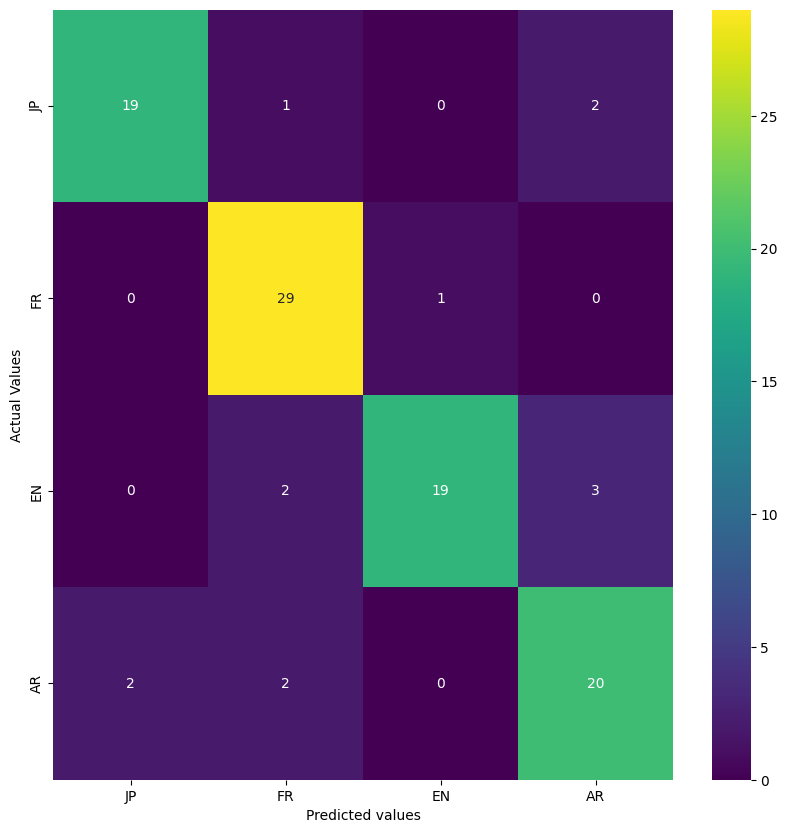


Fig. 4: Confusion Matrix obtained after testing the trained model

A confusion matrix is a table that is often used to evaluate the performance of a classification model. It presents a comprehensive summary of the predictions made by the model against the actual ground truth across all classes. The confusion matrix in Fig. 4 shows that the model correctly classified 19/22, 29/30, 19/24 and 20/24 samples for Japanese, French, English and Arabic respectively. Apart from the confusion matrix, the other scoring metrics are defined below:

* Accuracy: Accuracy measures the proportion of correct predictions among the total number of predictions. It is calculated as:
* Recall: Recall measures the proportion of actual positives that are correctly identified by the model. It can be defined as:
* Precision: Precision measures the proportion of true positives among the instances that the model predicted as positive. It is calculated as:
* F1 Score: The F1 score is the harmonic mean of precision and recall. It provides a balance between precision and recall, especially when the classes are imbalanced. It is calculated as:

Table 1 illustrates the scores that were obtained after testing.

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Metric** | **Score Obtained** |
| 1 | Accuracy Score | 0.87000 |
| 2 | Precision Score | 0.87493 |
| 3 | Recall Score | 0.87000 |
| 4 | F1 Score | 0.86948 |

TABLE 1: SCORING METRICS OBTAINED AFTER TESTING

1. **CONCLUSION**

The proposed mechanism that has been discussed is sufficiently performant for the chosen task. The system can correctly classify the language being spoken in a given audio file with up to 87% accuracy. This mechanism may be generalized for other languages and may offer differing results based on the language in consideration. More extensive datasets may also be used to improve the accuracy of the mechanism. The system can also be integrated as a component in a language translation system for correctly identifying the spoken language before translation. The major performance intensive preprocessing step which is required for detection is the generation of spectrograms which may be improved using alternative methods.

1. **REFERENCES**

1) Ashkan Moradi, Yasser Shekofteh (2023), “Spoken language identification using a genetic-based fusion approach to combine acoustic and universal phonetic results”, Computers and Electrical Engineering, Volume 105, 108549, ISSN 0045-7906.

2) Miller, Corey & Strong, Rachel & Jones, Evan & Vinson, Mark. (2014), “Employing Phonetic Speech Recognition for Language and Dialect Specific Search”, DOI:10.3115/v1/W14-5308

3) Riya Menon (2022), “Detectsy: A System for Detecting Language from the Text, Images, and Audio Files”, <https://www.ijraset.com/best-journal/detectsy-system-for-detecting-language-from-the-text-images-and-audio-files>. (Accessed on 22nd April, 2024).

4) Y. Rajanak, R. Patil and Y. P. Singh (2023), “Language Detection Using Natural Language Processing”, 9th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 2023, pp. 673-678, doi: 10.1109/ICACCS57279.2023.10112773.

5) Kotsakis, Rigas & Matsiola, Maria & Kalliris, George & Dimoulas, Charalampos. (2020), "Investigation of Spoken-Language Detection and Classification in Broadcasted Audio Content”, Information 2020, 11(4), 211; <https://doi.org/10.3390/info11040211>

6) Mehdi Nait Hammou (2023), “Audio sample language classification”, Kaggle; <https://www.kaggle.com/datasets/mehdinait/audio-sample-language-classification> (Accessed on 2nd May, 2024).

7) Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen (2019), “MobileNetV2: Inverted Residuals and Linear Bottlenecks”, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 4510-4520; <https://doi.org/10.48550/arXiv.1801.04381>

8) Diederik P. Kingma, Jimmy Ba (2017), “Adam: A Method for Stochastic Optimization”, Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015; <https://doi.org/10.48550/arXiv.1412.6980>