Automatic Language Identification in Audio & Video Clips

Akashdeep Balu, Kavya Nagaraju, Priyanka Patil
(Hc6943) (Hc3344) (Hc6653)

Abstract

Spoken language identification is the process by which the language in a spoken utterance is recognized automatically. Spoken language identification is commonly used in speech translation systems, in multi-lingual speech recognition. This project shows how to train a language recognizer from scratch that can distinguish between various Languages. Among the many available methods that can be applied to this classification task, modern machine learning and deep learning approaches have been reported as effective. The data from Common Voice has been used to build strong models. In this project, spoken language identification based on deep learning is presented.

Keywords: CNN, Spoken Language Recognition

1. Introduction

The application of voice when interacting with modern technology is rapidly increasing. Many intelligent products such as Amazon Alexa, Google Translate, and Apple's Siri are already applying speech recognition technologies to understand the context from speech and can subsequently be controlled by voice.

Spoken language Recognition is a programmed procedure that decides the character of the language verbally expressed in a speech utterance. Recent advancement in Machine Learning has hugely expanded the horizon for human-computer interaction using only their voice. The present-day technologies often require processed language to be explicitly declared. The capability to dynamically process input speech samples of different languages would expand the usability of existing speech processing technology and open up a wide range of additional functionality. Language Identification frameworks are utilized in a large number of applications: multilingual language interpretation, crisis or then again shopper call directing, reconnaissance, and security applications.

Several machine learning approaches have previously been used to construct language identification systems (LID) that attempt to solve the task of correctly classifying speech samples from different languages. Some common approaches are modern machine learning and deep learning approaches have been reported as effective.

2. Dataset Description

The dataset used in this project is based on Mozilla's Common Voice. Five Language datasets have been used for our Analysis. We are extracting the training and an evaluation dataset containing speech signals with a duration between 7.5 and 10 seconds.

2.1 Data Augmentation

The training dataset can be augmented by adding noise. This will later help to improve the robustness of the final model against noise-affected recordings. We use the NumPy function randomly. Normal for a normal (Gaussian) distribution, which gives us white noise.

2.2 Data Preprocessing

All audio files are preprocessed to extract a Mel-scaled spectrogram. This is done as below.

The audio is loaded and downsampled to 8 kHz to limit the bandwidth to 4 kHz. This helps to make the algorithm robust against noise in the higher frequencies. As most of the phonemes in various languages do not exceed 3 kHz. We duplicate the signal that is between 7.5 and 10 seconds long and cut it to 10 seconds.

The Mel-scaled spectrogram is computed from the audio. It helps to analyze the speech frequencies with the neural network. The Mel-scaling represents lower frequencies with a higher resolution than higher frequencies and considers the way humans perceive frequencies. Normalization takes place to adjust the values in the spectrogram between 0 and 1.

This step equalizes quiet and loud recordings to a common level. All files are finally stored as PNG files of size 500 x 128, but first, the normalized spectrograms need to be converted from decimal values between 0 and 1 to integer values between 0 and 255.

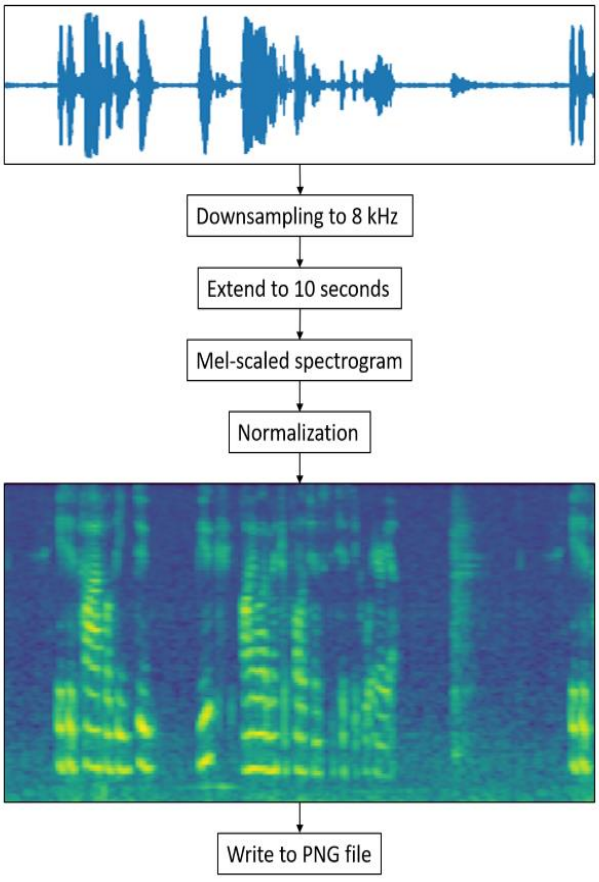
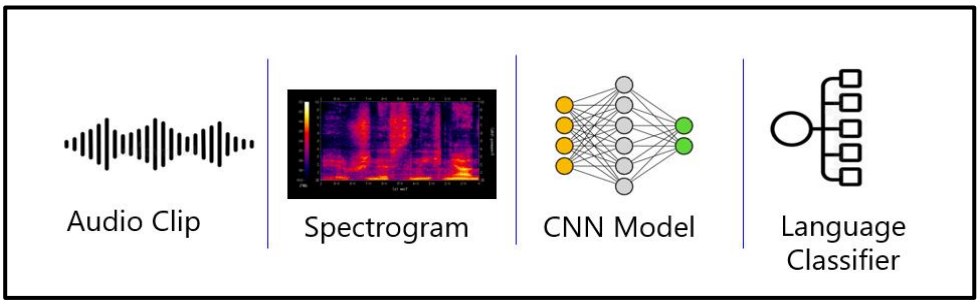


Fig-1 Speech Audio to the Spectrogram

3. Our Approach

The A Convolutional Neural Network(CNN) is a Deep Learning calculation or algorithm which takes an info picture and relegates significance to different objects in that picture and hence can separate it from other different pictures. CNN's, as neural systems, are comprised of neurons with customized weights and biases. Every neuron gets a few sources of info, takes a weighted total over them, gets it through an activation function, and reacts with a yield/output.



4. Model Implementation

We have implemented three Deep learning Models in this project Conv2D, InceptionV3, ResNet50. To make the datasets accessible for the model training algorithm, we first need to instantiate generators that iterate the datasets in a memory-efficient way. This is important because we have a large number of images for both training and evaluation data, and we cannot just load it into memory. Fortunately, Keras, which is part of TensorFlow, has amazing functions to deal with image datasets. We are using the Image Data Generator as a data input source for the model training. The suitable batch size is 128, which is big enough to help to reduce over-fitting and keeps the model training efficient.

4.1 Inception V3

A huge CNN such as Inception V3 is recommended to solve such a complex task. Spoken language recognition is a very complex task that demands a model with a big capacity. Smaller networks perform worse because they are not capable of handling the complexity of the data.

The original Inception V3 model, which is already available from Keras Applications, must be slightly adapted to process our dataset. It is expected to process images with 3 different color layers, but our images are grayscale. The following lines of code copy the single image color layer to all three channels for the input tensor for Inception V3.

In our evaluations, the RMS-Prop optimizer yielded good results. Adam is suggested. Training can be speeded up by automatically stopping when the learning finishes. We use Keras Early Stopping method to do so. We implemented an exponential learning rate decay.

The number of epochs is very high, but since we are using early stopping, fortunately, the algorithm stops after certain epochs. When looking at the training and evaluation accuracies, we can see that there is overfitting involved, which means that the model is much more accurate in classifying the training dataset than the evaluation dataset.

This is because the Inception V3 model is huge and has an enormous capacity. Using such a big network means that the amount of diverse training data should be very large, as well. So, to overcome this issue, we increased the amount of training data up to the point where overfitting does not occur anymore. In the plot, you can also see the effect of the learning rate decay.

4.2 Model Conv2D

A Basic CNN Network has also been implemented to verify the Model performance.

Convolution Operation The goal of the Convolution Operation is to separate the abnormal state highlights, for example, edges, from the information picture. CNN builds up different component locators and utilizes them to build up a few element maps which are alluded to as convolution layers. The significance of the component locator is to recognize features in the info picture and channel the parts that are basic to it and avoid the rest. Additionally, the component map acquired is smaller than the information picture in size.

The Rectified Linear Unit (RELU) The purpose of the rectifier function(RELU) is to increase the non-linearity in the pictures. This is because images are naturally non-linear.

$$f(x) = \max(x, 0)$$

Pooling The pooling layer is in charge of lessening the spatial size of the convolved map. This lessens the computational multifaceted nature to process the information. It removes prevailing highlights which are rotationally and positionally invariant, hence, keeping up the procedure of viable preparing of the model. Pooling jams the highlights and records for their conceivable spatial or different sorts of distortions (spatial invariance).

Layer (type)	Output Shape	Param #
conv2d_378 (Conv2D)	(None, 126, 498, 64)	640
conv2d_379 (Conv2D)	(None, 124, 496, 32)	18464
max_pooling2d_17 (MaxPooling	(None, 62, 248, 32)	0
flatten_1 (Flatten)	(None, 492032)	0
dense_1 (Dense)	(None, 5)	2460165

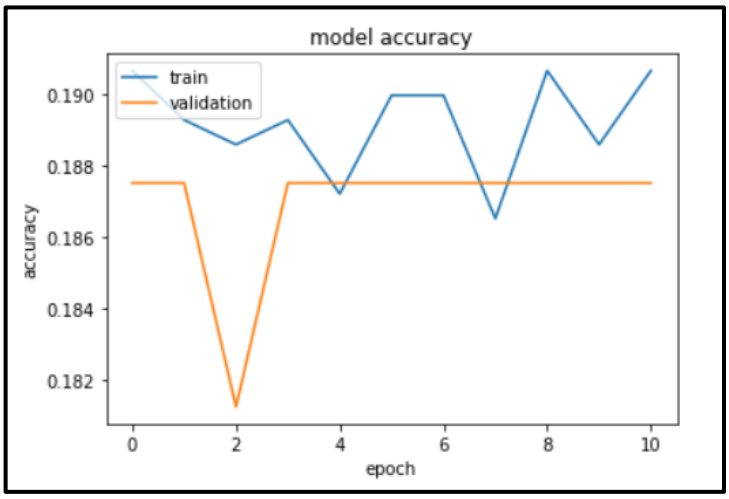
Total params: 2,479,269
Trainable params: 2,479,269
Non-trainable params: 0

There are two kinds of pooling utilized in the proposed methodology: Max Pooling and Average Pooling. Max Pooling restores the most extreme worthwhile Average Pooling restores the normal of the considerable number of qualities from the part of the picture

secured by the portion. Pooling serves to minimize the size of the picture and parameters which thus counteracts overfitting.

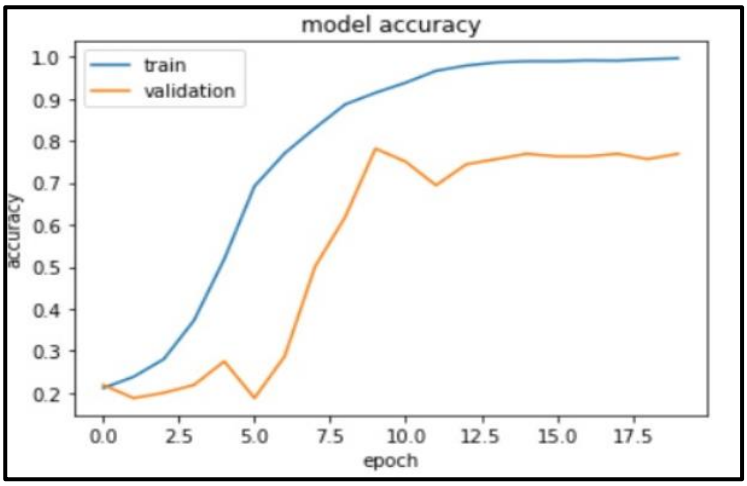
Flattening The pooled feature map is leveled into a column to input it into the Artificial Neural Network.

Full Connection This progression adds an Artificial Neural Network to the CNN which incorporates an input layer, a lot of completely associated concealed layers, and a yield/output layer. The job of the Artificial Neural Network is to take the information and consolidate the highlights into a more extensive assortment of properties that make the CNN progressively fit for ordering pictures. A Fully Connected Layer learns the straight mixes of the abnormal state as spoken to by the yield/output of the convolutional layer.



4.3 Model ResNet50

When we add more layers to our deep neural networks, the performance becomes stagnant or starts to degrade. This happens due to the vanishing gradient problem. When gradients are backpropagated through the deep neural network and repeatedly multiplied, this makes gradients extremely small causing a vanishing gradient problem. ResNet solves the vanishing gradient problem by using Identity shortcut connection or skip connections that skip one or more layers. Shortcut connections are connecting output on layer N to the input of layer N+Z. We use the ResNet50 deep learning model as the pre-trained model for feature extraction for Transfer Learning. The importance here is to learn about transfer learning and making robust models. We follow an example, but we can run with different approaches that we will discuss. There are two approaches we can take in transfer learning Feature extraction & Fine-tuning. layers of your trained model. We have already a very huge number of parameters because of the number of layers of the ResNet50 but we have calibrated weights. We can choose to 'freeze' those layers, so those values do not change, and by that way saving time and computational cost. We have regularized to help us avoid overfitting and optimizers to get a faster result.



Batch size It is recommended to use several batch sizes with powers of 2 because it fits with the memory of the

computer. Learning rate for transfer learning is recommended a very low learning rate because we do not want to change too much what is previously learned.

Several layers This depends on how much you relay from the layers of the pre-trained model. We found that if we leave all the models for training just a flatten layer and a dense with SoftMax is enough but since we incorporated the feature extraction it was required more layers at the end. Optimization methods We tested with SGD and RMSprop. SGD with very low learning required more epochs to complete a reasonable training. Regularization methods to avoid overfitting we used Batch normalization and dropout in-between the dense layers. Callback in Keras, we can use callbacks in our model to perform certain actions in the training such as weight saving.

5. Results

This section presents results obtained by training and evaluating the proposed model for 20 epochs. The goal was to recognize language among 5 target languages. The accuracy generated when the trained model was evaluated against the testing set was found to be 94.20% for the ResNet50 Model. This model proves to be the best among all three models. Basic CNN Model has the least accuracy of 78.60% & InceptionV3 has an accuracy of 87.40%.

The following table shows the results obtained from the above discussed CNN Models and the Percentage variation in the Accuracy.

Models	Model Type	Epoch	Accuracy
Model-1	InceptionV3	20	97.99%
Model-2	Conv2D	20	17.05%
Model-3	ResNet50	20	98.80%

Based on the above results obtained on the training dataset, Model-3 provided us with the best results. Hence running test data on ResNet50 we obtained a test accuracy of 90.02.

```
1 print('Test accuracy: ' + str(round(test_accuracy * 100., 1)) + ' %')
2

Test accuracy: 90.2 %
```

Also, as a real-time implementation on Video Clips, we provided input as Japanese Video Clip on our model 3, Resnet50 was able to identify the language as Japanese correctly.

```
# pip install SpeechRecognition
preds = model.predict_generator(test_generator)
preds_cls_idx = preds.argmax(axis=-1)
```

```
import numpy as np

idx_to_cls = {v: k for k, v in train_generator.class_indices.items()}
preds_cls = np.vectorize(idx_to_cls.get)(preds_cls_idx)
filenames_to_cls = list(zip(test_generator.filenames, preds_cls))
```

```
filenames_to_cls
```

```
]: [('class/my_result.mp3.png', 'japanese'),
    ('class/my_result.png', 'japanese'),
    ('class/my_result1.mp3.png', 'japanese')]
```

6. Future Work

There is further scope for improvement, for example, we can add more classes and feed it with more data from other languages. Also, it is recommended to adapt the data and the augmentation to your application. We have an optimized version trained on media data. We are using two additional augmentation steps to improve results on double-talk (overdub) and background music. Also, the pitch can be altered speed to perform data augmentation. Moreover, it is recommended to use much more data. If the amount and diversity of the training dataset are not sufficient, overfitting might occur.

7. Conclusion

We have implemented various Deep Learning Models for identifying the spoken language directly from speech audio. The model accuracy for Inception V3 is 87.4%, 78.60% for Basic CNN Model & 94.20 % for ResNet50 Model, for the given datasets based on Common Voice. The idea is to analyze the Mel-scaled spectrogram of a 10-second-long audio segment using a CNN with a high capacity. Since language recognition is a complex task, the network needs to be large enough to capture the complexity and derive meaningful features during training.

8. References

[1] C. Bartz, T. Herold, H. Yang, and C. Meinel, Language Identification Using Deep Convolutional Recurrent Neural Networks (2015), Proc. of International Conference on Neural Information Processing (ICONIP)

[2] S. S. Sarthak, G. Mittal, Spoken Language Identification using ConvNets (2019), Ambient Intelligence, vol. 11912, Springer Nature 2019, p. 252

[3] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, Rethinking the Inception Architecture for Computer Vision (2015), CoRR, abs/1512.00567