STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

BIA 678

Team Project Report

Athanasios Ageridis

Iswariya Ravi

Kuntal Gupta

Venkat Vishal

Chendra Kousik Jillellamudi

Table of Contents

Introduction

Idea

For the implementation of the ASR[1] system, a CNN[2] was used; it was trained to recognize the MFCC[3] characteristics of sound after they are extracted through a pre-processing process from the dataset.

Data

Dataset

The project was based on two open-source datasets found on the internet. The first dataset consists of speakers with an American accent, while the second with English.

The dataset consists of 4500 recordings at different speeds, with a sampling rate of 22050 with 90-digit recordings for each speaker of both sexes; some are stereo and others monophonic.

Stereo recordings are converted to mono.

---

[1] ASR – Automatic Speech Recognition

[2] CNN – Convolutional Neural Network

[3] MFCC – Mel Furrier Cepstral Coefficients

Data Pre-Processing

For the recognition to start successfully, initially, in the processing stage, I prepared the input signals so that they could be appropriately modified and extracted from them all the necessary and fundamental characteristics of the sound.

Reading Data

The SciPy[4] library was used to read the dataset.

Recording Separation

For the separation of the intervals of all and the decomposition of the fundamental digits that are subjected, the following characteristics are detected:

- Minimum Silence Duration 175 milliseconds
- Silence Frame Rate 50 dBFS
- Tools: Pydub

Channel Altercation (Stereo to Mono):

Since some of the samples in the dataset were in stereo - two-channel recording, I proceeded to convert them to a single-channel – mono

Resampling

The sampling rate was changed using the following toolboxes.

Tools:

---

[4] SciPy: Free open-source library

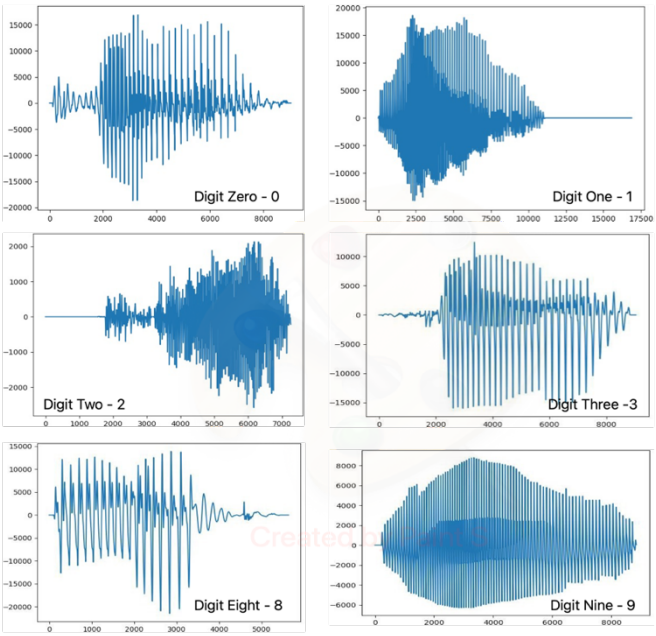
- SOX - (Butterworth Filter, n.d.) (Automatic Speech Recognition, n.d.)

  Dataset 1 / Architecture 1
- librosa - Dataset 2 / Architecture 2

Filtering

Butterworth[5]

To eliminate the DC - component, noise - buzz, of the signal, bi-directionally was applied a Butter, low pass, filter. More specifically, three Butter filters are used to extract the audio features. The results of the samples, the digits - [0-9], are shown in the following diagrams:



[5] Signal Processing Filter designed to have a frequency response that is as flat as possible in the passband

Padding

For the signals of the dataset to have the same length so that extracting the characteristics - stamp, sound - is more precise, I proceeded to separate the long-term signals and, respectively, to zero in, in the end, the shorter-duration signals.

Normalization

Since the recorded samples of the dataset do not follow a normal distribution, normalization was performed so that they are within the normal range distribution of the interval [0, 1].

Min-Max Strategy

More specifically, the Min-Max normalization[6] strategy was used.

Left-Right Padding

As the latest data that emerges does not follow a uniform distribution, we scaled by adding the appropriate Paddings, right and left to the samples of the dataset.

Feature Scaling

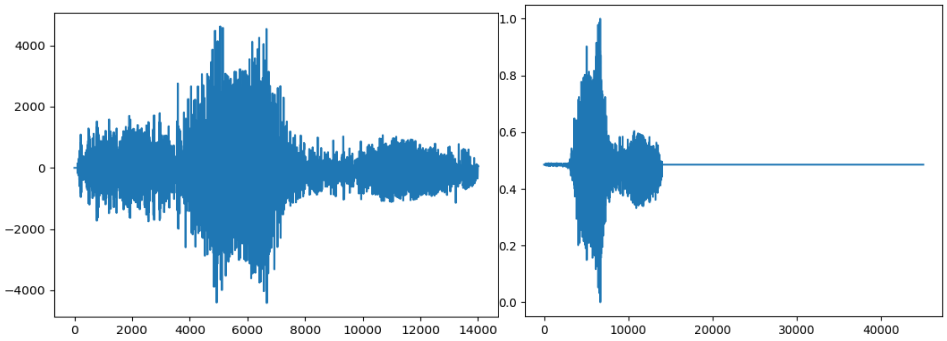
For identical digit signals to gain more accuracy - uniformity, but also to multiply the dataset's data, due to lack of diversity, we proceeded to filter, using three butter filters of similar digits in one. This resulted in the resulting signals being filtered around the

---

[6] Min-Max Normalization: For every feature the min value of the feature gets transformed into a 0 , the max value gets transformed into a 1, and every other value between 0 and 1

average and the labeling of the numbers being more accurate. The results of the above process are shown in the following diagrams:

Finally, the normalized and padded samples of the resulting spoken digit seven – 7 are shown in the following diagrams:



Feature Extraction

MFCC[7] - Mel Frequency Cepstral Coefficients, after pretreatment, Mfcc's were exported the Cepstral representation for digits two and seven

[7] MFCC: They are derived from a type of cepstral representation of the audio clip, a nonlinear spectrum of a spectrum

One Hot Encoding

The targets of the entered data have been converted to a one-hot encoded format with the following values.

0	[1,0,0,0,0,0,0,0,0,0]
1	[0,1,0,0,0,0,0,0,0,0]
2	[0,0,1,0,0,0,0,0,0,0]
3	[0,0,0,1,0,0,0,0,0,0]
4	[0,0,0,0,1,0,0,0,0,0]
5	[0,0,0,0,0,1,0,0,0,0]
6	[0,0,0,0,0,0,1,0,0,0]
7	[0,0,0,0,0,0,0,1,0,0]
8	[0,0,0,0,0,0,0,0,1,0]
9	[0,0,0,0,0,0,0,0,0,1]

Pre-Processing

The selected file undergoes the same processing as the dataset. First, resampling is performed and then split after segmentation of the spoken digits. Next, the model is built from the weights given by the user.

Tools

Keras

The Keras library was used for the network's training, the construction of the model, the fitting of the target data, and the extraction of the weights.

Machine Learning Network

Convolutional Neural Network

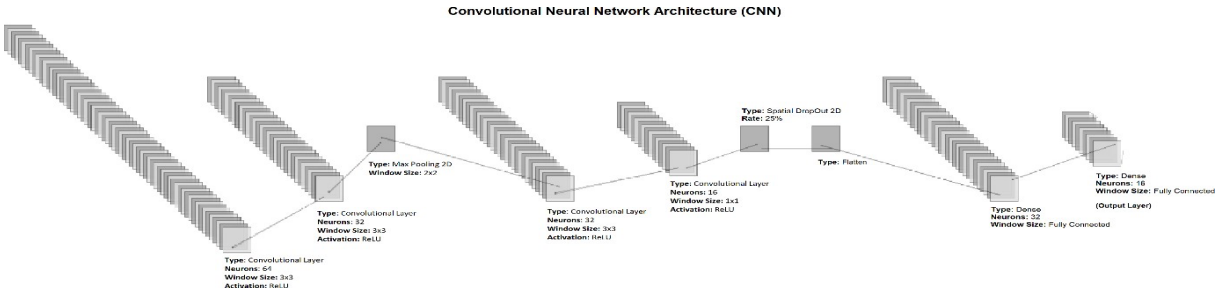
The neural network was built to recognize the features and differences between these images to recognize the sound signals.

To recognize the processed signals, the appropriate NN model was created.

The fully connected layers produce a vector [1 x 10] for each possible digit of the set; this architecture is since the Convolutional Layers will recognize features, i.e., the extracted Mel Frequency Cepstral Coefficients from the input images, and then the complete layers will learn to categorize based on these extracted features.

To solve the problem, the appropriate model was created, depicted below

Architecture



CNN: Architecture 1

Layers

More specifically, the architecture of the model consists of the following interconnected layers:

Layer	**Type**	**Neurons**	**Filter Size**	**Activation**	**Rate**	**Pool**

				Function		Size
L1	Conv2D	64	(3,3)	ReLu		
L3	Conv2D	32	(3,3)	ReLu		
L4	MaxPooling2D					(2,2)
L5	Conv2D	120	(2,2)	ReLu		
L6	MaxPooling2D					(2,2)
L7	Conv2D	32	(3,3)	ReLu		
L8	Conv2D	16	(3,3)	ReLu		
L9	SpatialDropout2D				0.25	
L10	Flatten					
L11	Dense	32		ReLu		
L12	Dense	10		Softmax		

Max Pooling 2D

Like architecture 1, the specific sampling method was used.

Overfitting - Spatial Dropout

To avoid overfitting, i.e., the "perfect" training in the already existing data, which would result in the low-failed performance in the exercise of new audio sets introduced in the network, the specific layers were introduced, which reset neighboring pixels, which the more likely to be related to each other.

	Epochs	Batch Size	Verbose	Validation Split	Accuracy		Training CMD Output (PNG)
Training	10	1	1	0.1	92.4%		
	25	32	1	0.1	94.85%		docs/model_1/ 25_Epochs_94_8_Acc.png
	33	32	1	0.1	93.63%		docs/model_1/33_Epochs_93_64_Acc.png
	38	32	1	0.1	96.66	BEST	Docs/model_1/38_Epochs_96_66_Acc.png
	50	1		0.1	92.7%		

Compiler

Loss	**Optimizer**	**Metrics**
Categorical Cross entropy	Adam	Accuracy

Loss Function

Multi-Class Cross-Entropy

Optimizer

Adaptative Moment Estimation

This optimizer allows you to find different learning rates for each class with the highest and best possible performance.

Recognition

To identify an audio file, the pre-processing procedure is performed by dividing the file into spoken digits, as described in the pre-processing stage.

Using a file of the available weights that have been created, the model is built, and the predicted digits are extracted.

Evaluation

An algorithm was implemented to evaluate the created model that utilizes the one-hot encoded coding of the objectives.

Initially, the algorithm checks the network predictions for a set of inputs, compares them with the samples kept for testing, and returns the accuracy for each digit [0-9].

Finally, the vector [1x10] is compared to the vector [1x10] of the one-hot encoded format corresponding to the specific target. If the registers are the same, the prediction is successful.

Finally, one of the Dataset speakers, Ralph, was randomly selected to evaluate the model. It was observed that a higher yield of 97% is presented for the Testers when they belong to the dataset and lower, while diversity was presented in unknown files, using different weight files, with it ranging from 50% - 100%.

In dataset 1, where the accent tends toward the American, with difficulty recognizing English accents, it was deemed appropriate, as analyzed in TODOs, to unite the two different datasets. Finally, the performance when recognizing files created from the dataset is high and accurate. In contrast, for recordings that were not used in the model training, the performance ranges from 70% to 100%, depending on the weight file to be used.

It should be noted that the tests were performed with recordings in which the noise levels were as low as possible.

Performance - Cloud Training

The created model was trained in the Cloud on a 12Gb Tesla 80K GPU for greater performance than my computer CPU, which enabled training the dataset in less than 20 minutes, whereas on a personal computer took over 2 hours

Conclusion

The current network recognizes an unknown recording at a success of 97%-98%

Feature Improvements

1. Use Delta-Delta MFCCs for greater efficiency.
2. Integrate the two different datasets.
3. Complete the functionality of recordings directly from the Client
4. Client to support Dataset processing
5. The Client to support Data Visualizations, and Plotting
6. The creation architecture is to be built by JSON Payload through the Client.
7. Specify batch size by the user during the process
8. Take the application into production

Works Cited

Automatic Speech Recognition. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Speech_recognition

Butterworth Filter. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Butterworth_filter

Convolutional Neural Network. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Convolutional_neural_network

MFCCs. (n.d.). Retrieved from Wikipedia: https://www.google.com/search?client=safari&rls=en&q=mfcc&ie=UTF-8&oe=UTF-8

Normalization. (n.d.). Retrieved from Code Academy: https://www.codecademy.com/article/normalization

One Hot. (n.d.). Retrieved from Wikipedia: https://en.wikipedia.org/wiki/One-hot