UKRAINIAN CATHOLIC UNIVERSITY

MASTER THESIS

# Context Independent Speaker Classification

*Author:*
Borys OLSHANETSKYI

*A thesis submitted in fulfillment of the requirements for the degree of Master of Science*

*in the*

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2020

# Declaration of Authorship

I, Borys OLSHANETSKYI, declare that this thesis titled, "Context Independent Speaker Classification" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

*"In A.I., the holy grail was how do you generate internal representations."*

Geoffrey Hinton

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Master of Science

**Context Independent Speaker Classification**

by Borys OLSHANETSKYI

## *Abstract*

Speaker classification is an essential task in the machine learning domain, with many practical applications in identification and natural language processing. This work concentrates on speaker classification as a subtask of general speaker diarization for real-world conversation scenarios. We research the domain of modern speech processing and present the original speaker classification approach based on the recent developments in convolutional neural networks. Our method uses a spectrogram as input to the CNN classifier model, allowing it to capture spatial information about voice frequencies distribution. Presented results show beyond human ability performance and give strong prospects for future development.

# Contents

**Declaration of Authorship** **ii**

**Abstract** **iv**

**1 Introduction** **1**
1.1 Motivation . . . 1
1.2 Research Goals . . . 1
1.3 Industry Related Goals . . . 1
1.4 Challenges and Limitations . . . 2

**2 Background and Related Work** **3**
2.1 Speech Representation . . . 3
2.1.1 Voice Sound . . . 3
2.1.2 Fourier Transform (FT) . . . 3
2.1.3 Spectrogram . . . 3
2.1.4 Mel Frequency Cepstral Coefficients (MFCC) . . . 3
2.1.5 I-Vector . . . 5
2.1.6 D-Vector . . . 7
2.2 Neural Networks in Audio Processing . . . 7
2.2.1 DNN . . . 8
2.2.2 RNN and LSTM . . . 9
2.2.3 TDNN . . . 9
2.2.4 CNN . . . 10
2.3 ML-based Speaker Diarization . . . 11
2.3.1 Neural Networks in Speaker Diarization domain . . . 11
2.3.2 CNN over Spectrogram in voice and sound processing . . . 11

**3 Datasets** **13**
3.1 LibriSpeech ASR Corpus . . . 13
3.2 ICSI Meeting Speech Corpus . . . 13

**4 Experiments** **15**
4.1 Data Preprocessing . . . 15
4.2 First Stage Back-End . . . 15
4.3 Front-End . . . 16
4.3.1 MFCC . . . 16
4.3.2 Mel Spectrogram . . . 17
4.3.3 Spectrogram . . . 18
4.3.4 Training Details . . . 19
4.3.5 Front-End Experiments Summary . . . 19
4.4 Back-End Hyperparameters Tuning . . . 20
4.4.1 Loss Function . . . 20
4.4.2 Overfitting . . . 20

4.4.3 Convolutional Kernels . . . . . . . . . . . . . . . . . . . . . . . . . 21
4.4.4 Automated tuning . . . . . . . . . . . . . . . . . . . . . . . . . . . 21
4.5 Model Evaluation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21
4.6 Conclusions . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22

**5 Future Work** **23**
5.1 Bridging the Diarization Gap . . . . . . . . . . . . . . . . . . . . . . . . 23
5.2 Extended Research . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23

**Bibliography** **24**

# List of Figures

2.1 Signal over time presented in frequency domain. [*Image 1*] . . . . . . . . 4
2.2 Waveform and spectrogram plots.[*Spectrogram wikipedia*] . . . . . . . . 4
2.3 Pitch mel scale versus Hertz scale. [*Mel scale*] . . . . . . . . . . . . . . . 5
2.4 Overlapping frames processing.[*Image 2*] . . . . . . . . . . . . . . . . . . 6
2.5 Mel-scale band-pass filtering.[*Image 2*] . . . . . . . . . . . . . . . . . . . 6
2.6 Flowchart for MFCC calculation.[*Image 3*] . . . . . . . . . . . . . . . . . 6
2.7 MFCC heatmap.[*Image 2*] . . . . . . . . . . . . . . . . . . . . . . . . . . 7
2.8 Typical classification pipeline.[Ibrahim and Ramli, 2018] . . . . . . . . 7
2.9 D-vector based model for speaker verification.Variani et al., 2014 . . . 8
2.10 Multilayer Perceptron architecture. [*Image 4*] . . . . . . . . . . . . . . . 8
2.11 An unrolled RNN.[*Image 5*] . . . . . . . . . . . . . . . . . . . . . . . . . 9
2.12 The repeating block in LSTM.[*Image 5*] . . . . . . . . . . . . . . . . . . . 9
2.13 Computation in TDNN.[Peddinti, Povey, and Khudanpur, 2015] . . . . 10
2.14 Example of CNN, LeNet-5 architecture. [Lecun et al., 1998] . . . . . . . 10

4.1 CNN architecture example for 128 x 128 input. . . . . . . . . . . . . . . 16
4.2 MFCC with 16 coefficients for 2 seconds audio. . . . . . . . . . . . . . . 16
4.3 MFCC with 32 coefficients for 2 seconds audio. . . . . . . . . . . . . . . 17
4.4 Mel-frequency spectrogram window size = 256, cepstral coefficients = 16. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17
4.5 Mel-frequency spectrogram window size = 256, cepstral coefficients = 128. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
4.6 Mel-frequency spectrogram window size = 512, cepstral coefficients = 128. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
4.7 Sprectrogam of 1 sec audio fragment, FFT window length = 128. . . . . 19
4.8 Sprectrogam of 1 sec audio fragment, FFT window length = 256. . . . . 19
4.9 Final model architecture. . . . . . . . . . . . . . . . . . . . . . . . . . . . 21

# List of Tables

4.1 Experimentation summary. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20
4.2 Evaluation summary. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 21

# List of Abbreviations

| | |
|---|---|
| **CNN** | **C**onvolutional **N**eural **N**etwork |
| **DFT** | **D**iscrete **F**ourier **T**ransform |
| **FT** | **F**urier **T**ransform |
| **FFT** | **F**ast **F**ourier **T**ransform |
| **LSTM** | **L**ong **S**hort-**T**erm **M**emory |
| **MFCC** | **M**el **F**requency **C**epstral **C**oefficients |
| **NN** | **N**eural **N**etwork |
| **RNN** | **R**ecurrent **N**eural **N**etwork |
| **TDNN** | **T**ime **D**elay **N**eural **N**etwork |
| **SD** | **S**peaker **D**iarization |
| **SVM** | **S**upport **V**ector **M**achine |

*Dedicated to my patient wife.*

# Chapter 1

# Introduction

## 1.1 Motivation

In the real-world scenarios, speech is not represented by well defined audio segments with a single-source speaker. This fact makes many practical applications effectively not feasible without speaker diarization. General Speaker Diarization problem could be defined as a task of speech categorization for unlabeled talk, with no prior knowledge of speakers count. The output of this task is labeled speech segments belonging to the particular speaker.

Speaker diarization is an integral part of many speech applications. In cases where single sound source incorporates multiple voices, e.g., round table talk or meeting room recording, distinguishing speakers is crucial for speech to text processing and further general text understanding. Misclassification of a spoken person decreases an automated understanding of conversation by a significant margin. For example, doing a speech-to-text translation of the meeting recording with the goal of extraction summary, action items, or other higher-level understanding would be inefficient without knowing who said what.

In this work, we focus on a speaker classification task, which is a narrower version of general diarization, where the count of speakers is known. The solution to this problem can be used as a step of the end-to-end diarization system in future work.

## 1.2 Research Goals

1. Provide an overview of the related work in the scientific field of the speaker classification/diarization problem.

2. Explore existing solutions to the problem.

3. Propose and evaluate our original approach to the speaker classification.

## 1.3 Industry Related Goals

In this work, we focus on the *speaker classification task*. However, it is intended to be a very first step in a journey of building a real-world diarization system suitable for use in a product.

## 1.4 Challenges and Limitations

The biggest challenge for us in this work was finding a relevant dataset of high quality. Most datasets cited in related papers are proprietary or not freely available. Since this work has no dedicating funding, we were limited to open-source options, some of them have mediocre quality. We did our best to overcome this obstacle and produce convincing results.

# Chapter 2

# Background and Related Work

## 2.1 Speech Representation

### 2.1.1 Voice Sound

Humans have an incredible ability to separate and perceive voice information in a mixed sound environment. Even though the research on the topic has been conducted for many years, it is unknown how sound is represented on a biological level inside the human brain. Most of the sound processing is concentrated within the human hearing range, which lies between 20 and 20kHz. In the speech understanding field, there are several key concepts, and sound representation approaches worth highlighting.

### 2.1.2 Fourier Transform (FT)

Fourier Transform is a cornerstone equation in the sound processing domain. In general, any wave signal could be presented as a sum of sinusoidal signals. FT allows us to find those signals and, therefore, decompose an incoming signal into frequencies spectrum, Figure 2.1. This ability, combined with inverse operation, brings many processing possibilities. For example, an incoming sound signal could be augmented in a way that all frequencies typical for a human voice are amplified for a better speech understanding. Alternatively, noise frequencies could be suppressed. The frequency spectrum as sound representation is also crucial for practical machine learning tasks.

### 2.1.3 Spectrogram

A spectrogram is a visual representation of the spectrum of frequencies of a signal as it varies with time. When applied to an audio signal, spectrograms are sometimes called sonographs, voiceprints, or voicegrams [*Spectrogram wikipedia*]. When the data is represented in a 3D plot, they may be called waterfalls. Spectrogram can be generated using FT. The input signal is split into overlapping chunks. Then FT is applied on each chunk, and resulting arrays are concatenated to produce a 3-dimensional surface. A spectrogram is usually presented as a heatmap, where the intensity of the color corresponds to the amplitude of the respective frequency, Figure 2.2. In our proposed method we used spectrogram as input into our model, see chapter 4.

### 2.1.4 Mel Frequency Cepstral Coefficients (MFCC)

MFCC is a biologically inspired sound representation. It is based on the **Mel scale** named by Stevens, Volkmann, and Newman, 1937, a perceptual log-like scale of

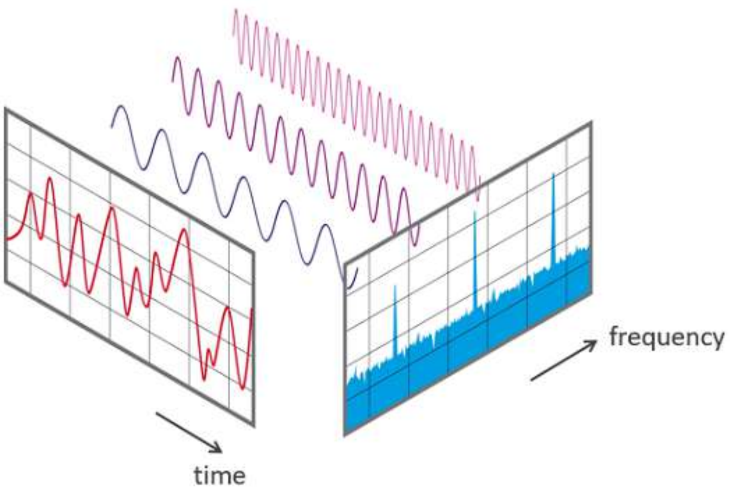

FIGURE 2.1: Signal over time presented in frequency domain. [*Image 1*]

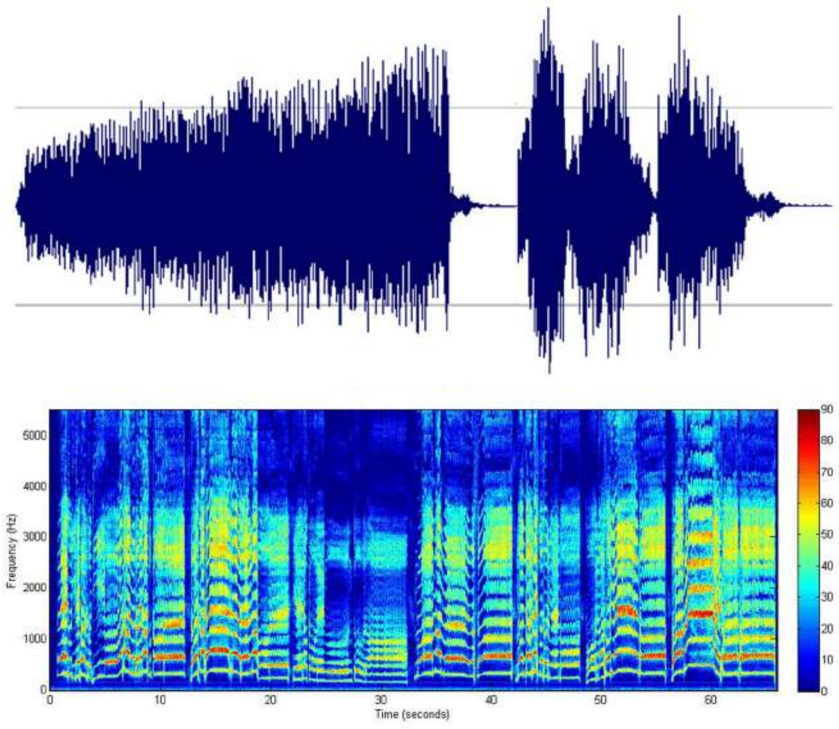

FIGURE 2.2: Waveform and spectrogram plots.[*Spectrogram wikipedia*]

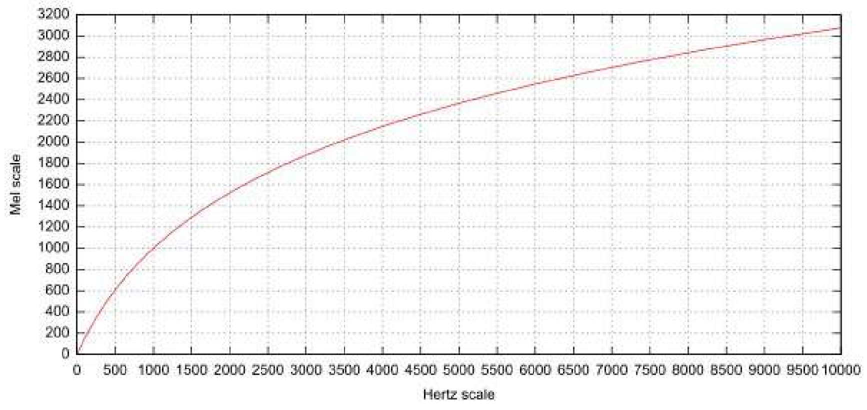

FIGURE 2.3: Pitch mel scale versus Hertz scale. [*Mel scale*]

pitches judged by listeners to be equal in distance from one another [*Mel scale*]. In other words, it is a frequency scale adjusted to empirically observed human perception through logarithmic transformation, Figure 2.3.

The most common formula to convert hertz(f) into Mel scale:

$$Mel(f) = 2595\log_{10}(1 + \frac{f}{700})$$

From a processing point of view calculation of the MFCC is done in several steps:

1. Break incoming signal into overlapping frames. It is a typical pattern in the algorithms using Fourier Transform, as integration in practice is always done on the specific time interval 2.4.

2. Apply Fast Fourier Transform(FFT) on each frame. FFT is essentially a computationally optimized Discrete Fourier Transform(DFT).

3. Compute the signal energy through a bank of filters adjusted to mel-scaled frequencies 2.5. As a result, we obtain a binned set of energy values and take the log of them.

4. Apply FFT on the previous step result, the top K (usually 12, but it can vary), coefficients would be the MFCC 2.7.

In machine learning terminology, we can call MFCC calculation a low-level feature extraction with hand-crafted parameters, like the mel scale filter bank. MFCC is very popular method and used in many related works. On Figure 2.7 example is depicted as a heatmap, where the color represents the intensity of each bin.

### 2.1.5 I-Vector

Identity vectors or i-vectors based models are often present in state-of-the-art in speaker verification solutions [Guo et al., 2018; Huang, Wang, and Qian, 2018; Rohdin et al., 2017]. The idea of the i-vector is to reduce high-dimensional sequential input data to a low-dimensional fixed-length feature vector. The speaker-dependant and channel-dependent factors are modeled using the total variability approach, as follows:

$$s = \mu + Tw$$

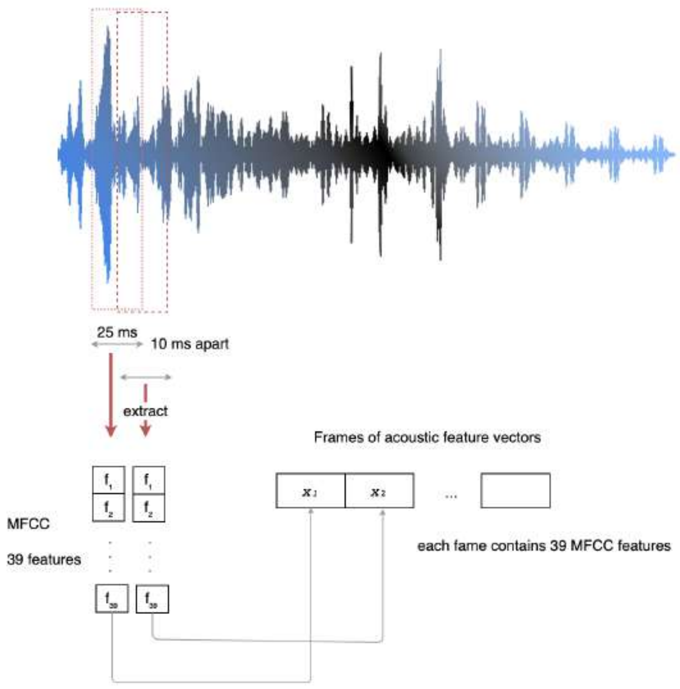
25 ms
10 ms apart
extract
Frames of acoustic feature vectors
MFCC
39 features
each fame contains 39 MFCC features

FIGURE 2.4: Overlapping frames processing.[Image 2]

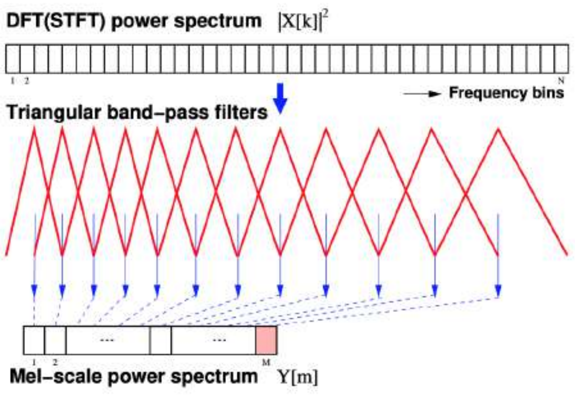
DFT(STFT) power spectrum |X[k]|²
Frequency bins
Triangular band–pass filters
Mel–scale power spectrum Y[m]

FIGURE 2.5: Mel-scale band-pass filtering.[Image 2]

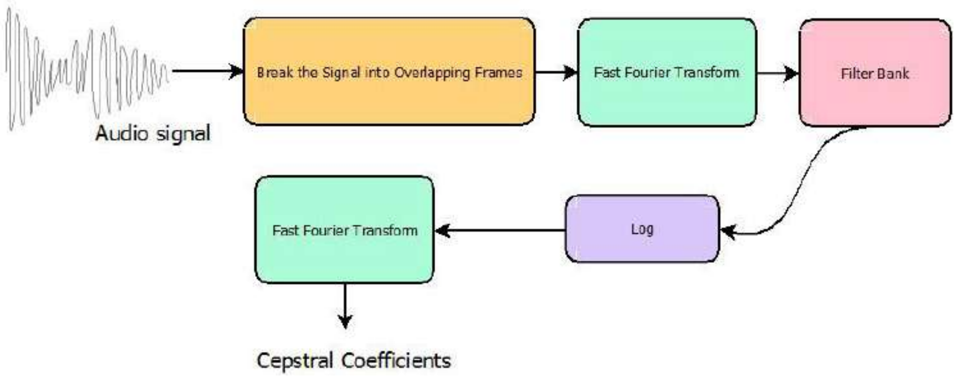

FIGURE 2.6: Flowchart for MFCC calculation.[Image 3]

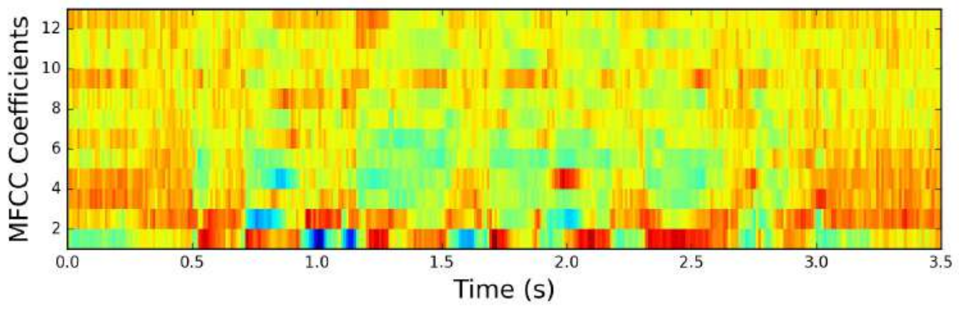

FIGURE 2.7: MFCC heatmap.[*Image 2*]

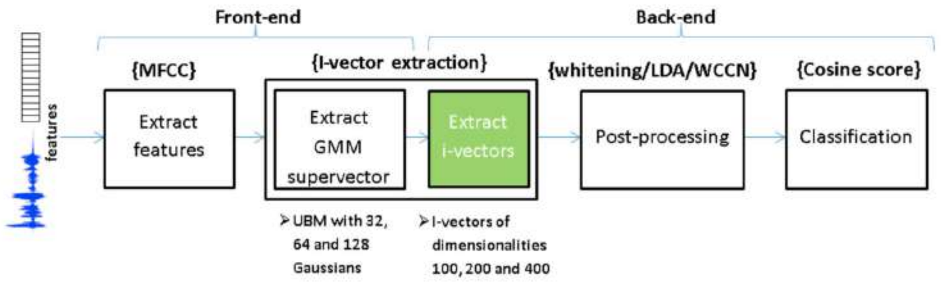

FIGURE 2.8: Typical classification pipeline.[Ibrahim and Ramli, 2018]

where $s$ is mean supervector, $\mu$ is the mean superverctor of a Universal Background Model (UBM), T is a low-rank matrix, and $w$ is the i-vector estimated using the Factor Analysis method. [Dehak et al., 2011; Malykh and Kudashev, 2017] Comparing to MFCC, I-Vector is a high-level representation of speech utterance. They are often used together, see Figure 2.8.

### 2.1.6 D-Vector

D-vector is a representation of a speaker identity embedding acquired through the deep neural network (DNN). In this approach, a feed-forward DNN is trained to classify speakers at the frame-level, then the last classification layer is removed, and d-vectors are obtained through averaging hidden layer activations [Variani et al., 2014 2.9]. D-vector approach has attracted attention over the previous couple years, as it utilizes neural networks for feature extraction. The idea has multiple proposed implementations and extensions, like proposed by Snyder et al., 2018, **X-vector**. These vectors are high-level generalized representations of the speaker voice features.

## 2.2 Neural Networks in Audio Processing

The usage of Neural Networks(NN) has grown tremendously for the last decade. NN showed their efficiency in sound data processing from the very beginning of this rise. Especially significant boost of accuracy was achieved in speech recognition [Hinton et al., 2012]. Since then, the research of NN based models in the sound processing field is steadily growing. Most of the recent state-of-the-art papers include some neural network in their model or pipeline.

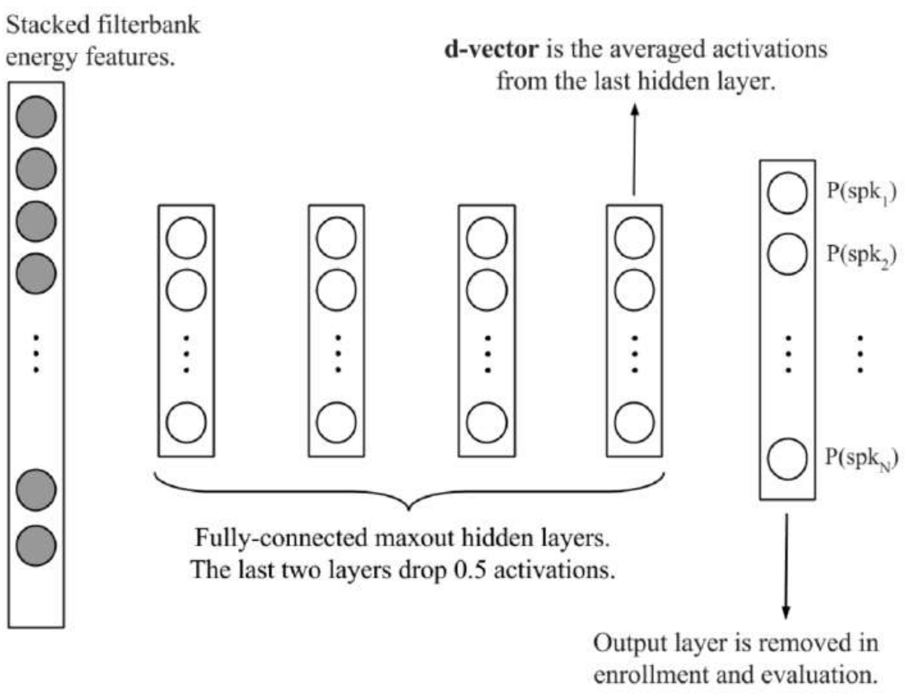
Stacked filterbank energy features.
d-vector is the averaged activations from the last hidden layer.
P(spk$_1$)
P(spk$_2$)
P(spk$_N$)
Fully-connected maxout hidden layers. The last two layers drop 0.5 activations.
Output layer is removed in enrollment and evaluation.

FIGURE 2.9: D-vector based model for speaker verification.Variani et al., 2014

### 2.2.1 DNN

Deep Neural Network(DNN) is an artificial neural network with multiple layers between the input and output layer [*DNN definition*]. **DNN** propagates signal through layers modeling non-linear relationships between input and output. The most straightforward deep neural network is a multilayer perceptron. In this architecture, all the neurons of the previous layer are connected to each neuron of the next layer.( 2.10) These layers are often called *dense* or *fully-connected*. Training of the neural network consists of forward and backward passes. On forward pass, the input is propagated through layers, and the error function is calculated. On the backward pass, each weight is adjusted through the calculation of its impact on total error. This process is repeated until the error converges to zero or training is stopped by other conditions. DNNs allow modeling of sophisticated features from high-dimensional data, which

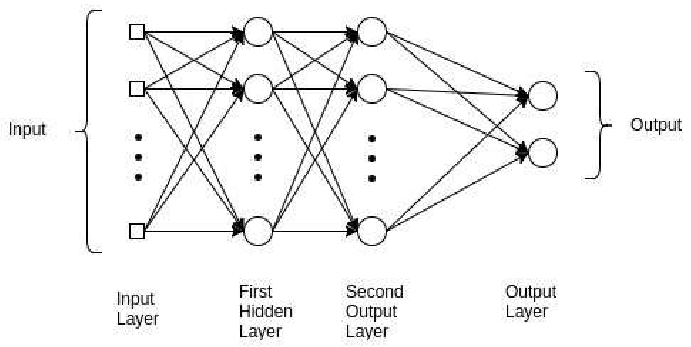

FIGURE 2.10: Multilayer Perceptron architecture. [*Image 4*]

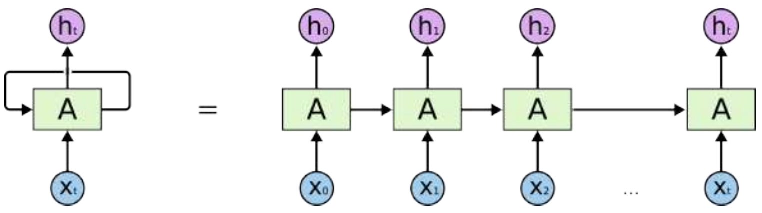

FIGURE 2.11: An unrolled RNN.[*Image 5*]

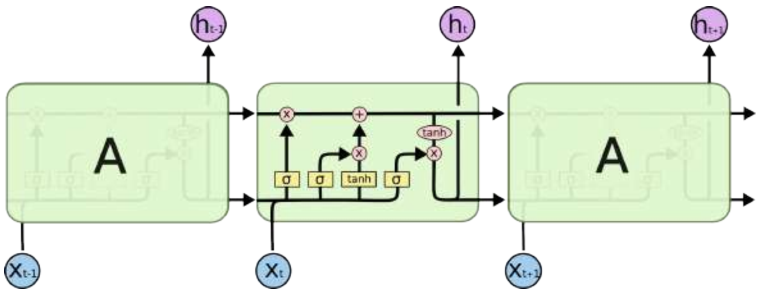

FIGURE 2.12: The repeating block in LSTM.[*Image 5*]

are hard to handcraft otherwise.

### 2.2.2 RNN and LSTM

*Recurrent Neural Networks(RNN)* address the problem of applying traditional DNN design on sequenced data. In traditional DNN, each input is treated independently, and the evaluation of one chunk of information has no impact on others 2.10. This approach is counterproductive when applied to data, where parts are naturally dependent or feed into the model as sequence, like text or speech. The main idea of RNN of any implementation is to pass previously accumulated data in some form to the next iteration 2.11. The early designs of RNN revealed the problem of long-term dependencies. As the state signal passes through iterations, it tends to decrease in power, in a sense the further dependency is, the lower is its impact in the present evaluation. This trait has a crucial impact on models where highly connected data points are split wide apart, e.g., language models.

*Long Short Term Memory(LSTM)* networks successfully approach this problem. They were introduced by Hochreiter and Schmidhuber, 1997 and improved by many contributions later on. The core idea of the LSTMs is to use a sophisticated neural network repeating block of several layers, designed to capture long term dependencies. The amount of "memory" information passed through this block depends on the input, Figure 2.12. As per Karpathy, 2015, the RNNs are "unreasonably effective" for a wide variety of tasks, and the LSTMs represent the majority of the successful RNN applications, including speech processing.

### 2.2.3 TDNN

Time Delay Neural Networks(TDNN) is another multilayer NN architecture often used for speech processing tasks. This architecture allows classifying patterns with shift-invariance and model context on each layer. Shift-invariant classification for speech means that TDNN does not need to define the beginning and end points of sounds before classifying them [*TDNN definition*]. The core idea of the TDNN is that sliding window with a fixed delay, generates overlapping input, and each

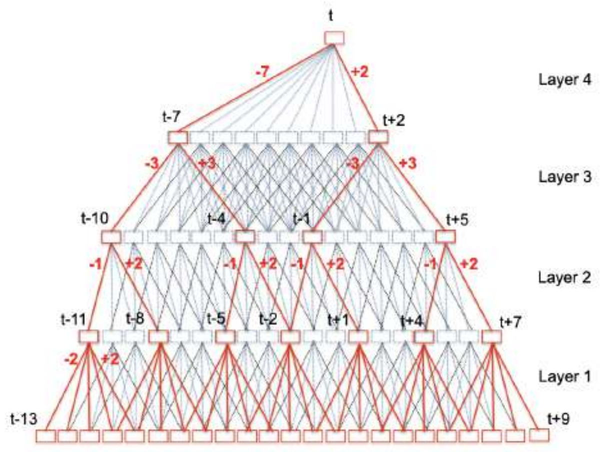

FIGURE 2.13: Computation in TDNN.[Peddinti, Povey, and Khudanpur, 2015]

layer has a connection to the time range of outputs from the previous layer. Thus, higher levels represent longer spans of speech. Shift-invariance is achieved during the backpropagation step, where the network is copied for each time-shifted instance and, then the error gradient is computed for all of these networks and averaged before weights update, see Figure 2.13.

### 2.2.4 CNN

Convolutional Neural Networks(CNN) are artificial neural networks where at least one layer is presented by convolution operations instead of traditional matrix multiplication. First introduced by Lecun et al., 1989, CNNs proved to be very efficient in the field of computer vision. They are capable of capturing spatial features like edges, lines, and corners of the image through local receptive fields. Through multiple layers, these features are combined to detect higher-level features. (2.14)

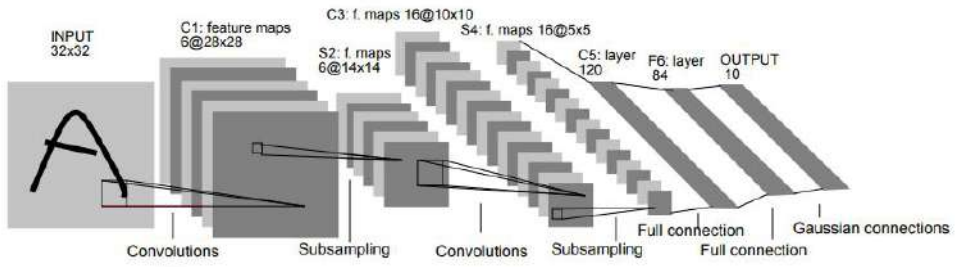

FIGURE 2.14: Example of CNN, LeNet-5 architecture. [Lecun et al., 1998]

As the speech utterance could be represented as a 2-dimensional data structure, e.g., MFCC or Spectrogram, we have explored the possible use of CNNs for speaker classification, see chapter 4.

## 2.3 ML-based Speaker Diarization

ML-based Speaker Diarization papers first started to appear in 2006 [Kenny, 2006; Kenny et al., 2007]. Early proposed solutions were based on Gaussian Mixture Models [Kenny et al., 2007], Hidden Markov Models [Kenny, 2006] or SVM(e.g. Kumar and Britto, 2012). Around 2012 they developed into complex diarization systems combining multiple models in the complex process but already with reasonable accuracy [Galibert and Kahn, 2013]. One of the noticeable such systems is the LIUM toolkit, an open-source tool that is developed and supported until today. [Meignier and Merlin, 2010] Since 2015 new wave of papers were published that proposed different approaches based on neural networks. To support the ideas of our research, we would split them into two categories:

- Papers presented successful usage of neural networks for Speaker Diarization
- Papers presented successful usage CNNs over spectrograms for other voice/sound processing tasks

### 2.3.1 Neural Networks in Speaker Diarization domain

In this category, we can highlight these recent works:

- *Speaker Diarization with LSTM*, Wang et al., 2018 - succsessfully used LSTM and D-vectors and achieved 12.0% diarization error rate, a state-of-the-art result.
- *Speaker Diarization Using Deep Neural Network Embeddings*, Garcia-Romero et al., 2017 - proposed to use LSTM neural network to generate low dimensional embeddings to represent voice utterances before clustering part. This approach showed 12.6% error rate. We are keen to reproduce this approach with modification in our research and build embeddings representation based on convolutional neural network.
- *Fully Supervised Speaker Diarization*, Zhang et al., 2019 - proposed new architecture - ubounded interleaved state RNN(UIS-RNN), which claim outstanding performance of 7.6% diarization error rate.

### 2.3.2 CNN over Spectrogram in voice and sound processing

Next works fall into this category:

- *Complex Spectrogram Enhancement by Convolutional Neural Network with Multi-Metrics Learning*, Fu et al., 2017 – successfully enhanced speech through representation of it as spectrogram and applying CNNs for denoising.
- *SampleCNN: End-to-End Deep Convolutional Neural Networks Using Very Small Filters for Music Classification*, Lee et al., 2018 – proposed classification of the music styles based on spectrogram patches, which is similar to our approach.
- and many others Huang et al., 2014; Mao et al., 2014; Sainath et al., 2014; Abdel-Hamid et al., 2014; Ghahremani et al., 2016

Some interesting works combine multiple neural network architectures to improve accuracy results. For example, *Acoustic Scene Classification Using Parallel Combination of LSTM and CNN* [Hyun, Choi, and Soo, 2016] combined LSTM with CNN over spectrogram to denoise sounds and improve results for a scene classification task.

Speaker diarization as a research field is very active at the moment, with many works published in recent years. It worth to mention that a 10% diarization error rate is still not enough for many real-world applications, and accuracy is not the only metric that shall be taken into account. Computational and time efficiency are crucial for sound processing tasks. The field is advancing rapidly, and researchers seem to be close to a general solution to the problem.

# Chapter 3

# Datasets

Since our interest lies in speaker classification in the meeting room setting, one of the challenges of this research was finding the proper dataset. There are a few datasets often used in related works in the speaker classification domain, but most of them are not free or freely available. Furthermore, those that are available have somewhat questionable quality and require heavy preprocessing. This fact did not allow us to compare some of our models with other researches directly. As a result, we have used two datasets: LibriSpeech ASR and ICSI Meetings. The first one we used for most of our experiments because of its labeling quality. And the second we used as a reference to the real-life scenario, as we want to understand how our model performs in a realistic setting.

## 3.1 LibriSpeech ASR Corpus

LibriSpeech is a corpus of approximately 1000 hours of 16kHz English speech derived from audiobooks from the LibriVox project, prepared by Vassil Panayotov with the assistance of Daniel Povey. The data has been segmented and aligned.[Panayotov et al., 2015]. For the training purpose was used subset containing 100 hours of most "clean" speech from 251 different speakers. This dataset was chosen because of its overall quality and availability. Additionally, an important feature of this dataset is that all speakers are clearly separated since only one speaker reads each book. It allowed us to label all the data points with 100% precision and conduct clean experiments.

## 3.2 ICSI Meeting Speech Corpus

Since this work is seen as the first step of building applicable to product level speaker classification/diarization solution in a meeting setting, we used ICSI Meetings corpus additionally. This dataset is similar to real-life meetings data. The corpus is a collection of 75 meetings collected at the International Computer Science Institute in Berkeley during the years 2000-2002. The speech files have a length from 17 to 103 minutes but generally run just under an hour each, and the collection includes 922 speech files, for a total of approximately 72 hours of meeting room speech, with a total of 53 unique speakers in the corpus. The audio was collected at a 48 kHz sample-rate, downsampled on the fly to 16 kHz. Meetings range from 3 to 10 participants, with six on average.[Janin et al., 2003]

After several experiments and close examination of the dataset, we concluded that labeling is often misaligned. Also, overlapping speech causes severe drops in the accuracy of the models. For the training stage, all the overlapping speech was removed. Also, for both evaluation and training stages, the speech was cleaned up using labels from *VocalSounds* and *NonVocalSounds*, representing non-speech sounds from speakers and other noises, e.g., mic sounds.

# Chapter 4

# Experiments

Based on the research described in **??**, we conducted a series of experiments to find out the best performing architecture for the speaker classification task. The general idea of the experiments is to leverage the ability of convolutional neural networks to capture patterns in the multidimensional structured data. With this approach, we split the experimentation into two logical parts: front-end and back-end, and it consisted of two stages. The back-end is a CNN with a 2-dimensional input and softmax layer as output. And the front-end is a range of different sound representations, where each is processed through its separate pipeline. The output of the front-end is input into the back-end. The goal of the first stage was to figure out the best performing front-end on a fixed back-end. And the purpose of the second stage was to fine-tune the CNN back-end for the chosen front-end and train the best possible model for the task. For all the experiments the LibriSpeech dataset was used (3.1).

## 4.1 Data Preprocessing

We believe that the optimal length of speech for speaker classification lies between 0.5 and 2 seconds for the real-life scenario, where speakers can have dialog and speech fragments are overlapping each other. Thus, for each of the 251 speakers, sound files are split into chunks of lengths 0.5, 1, 2 seconds during preprocessing step. Each chunk was cut with an overlap of 50% with the previous one to increase the size of the training dataset.

## 4.2 First Stage Back-End

The back-end of our experimental setup consists of CNN with 3 convolutional layers and 2 fully-connected layers. The size of the input layer varies from 512 x 128 to 16 x 16 and entirely depends on the respective front-end output pipeline (4.3). After each convolutional layer, we apply 2x2 max-pooling. Thus the size of the filter on the next layer is half size of previous for each dimension. The chosen dataset has 251 speakers. Therefore the very last fully-connected layer has a size of 1 x 251. After initial experiments, it became clear that if we connect directly final convolutional layer with the output layer, it creates a significant number of parameters that cause severe overfitting on the training stage. For example, the convolutional layer of 32 x 32 x 32 with a 1 x 251 output will result in around 2 million parameters. We found the original solution to this problem and added one narrow fully-connected layer before the output, of size 1 x 16. This measure reduces the count of parameters for the previous example to approx. 150 thousand and allows this model to generalize better.(Figure 4.1)

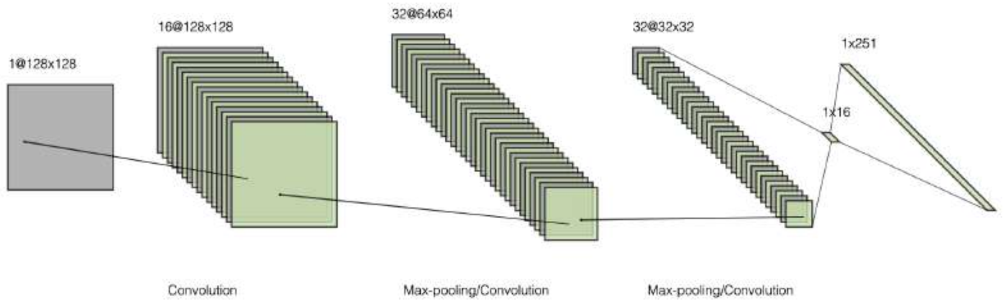

FIGURE 4.1: CNN architecture example for 128 x 128 input.

## 4.3 Front-End

For the front-end part of the research, we have chosen 3 audio representation options for testing: *MFCC, Spectrogram, and Mel Spectrogram*. Even though i-vectors show high performance in many applications, we intentionally left them outside of this research. We intend to train a neural network to extract high-level, complex features from the data. On the contrary, I-vectors present already extracted high-level features of a speech utterance. Also, i-vectors have 1 x *D* dimensionality that does not allow us to apply CNN to their processing and requires other architectures to be explored.

### 4.3.1 MFCC

MFCC were calculated as described in subsection 2.1.4 using *LibROSA package*. Two options were tested with 16 and 32 coefficients. These inputs were chosen to preserve as much information as possible from the original sound chunk. As it could be seen on Figure 4.2 and Figure 4.3, last 16 coefficients in the 32 variation does not contain much information, therefore we did not explore other options.

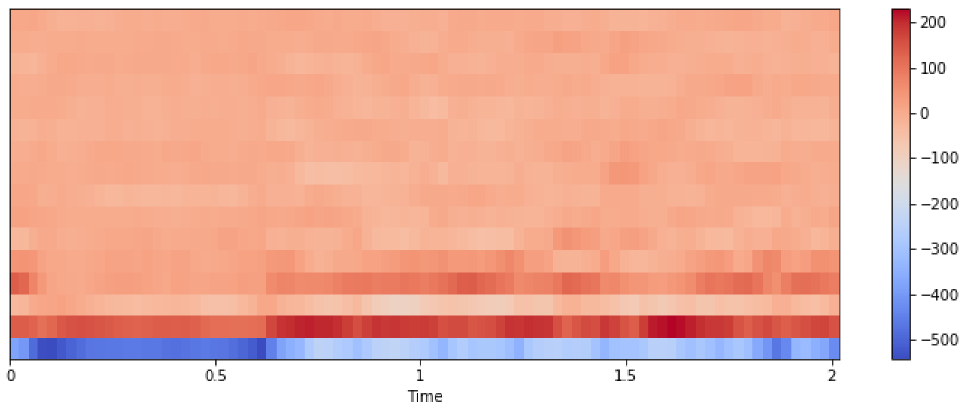

FIGURE 4.2: MFCC with 16 coefficients for 2 seconds audio.

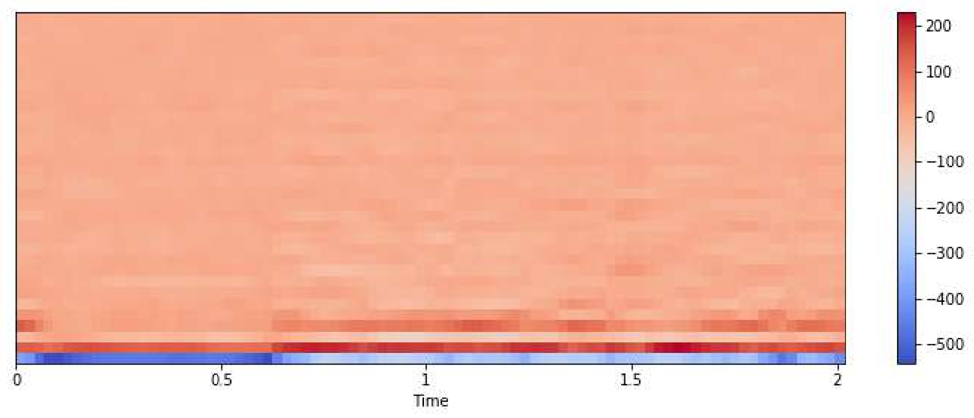

FIGURE 4.3: MFCC with 32 coefficients for 2 seconds audio.

### 4.3.2 Mel Spectrogram

Mel Spectrogram is obtained through the application of the bank of filters adjusted to mel-scaled frequencies, see. subsection 2.1.4 and Figure 2.3. As a result, the values are binned according to mel-scale, and frequencies important to human hearing perception are emphasized. Two option were tested with FFT sliding window of 512 (4.6) and 256 (4.5), and 128 filter bank applied. 128 filters were chosen in order to keep frequencies resolution high. It could be seen on Figure 4.4, how a decrease in filter quantity impacts the granularity in the frequencies domain. The 512 and 256 windows were chosen to keep the output tensor in a reasonable size for the back-end input.

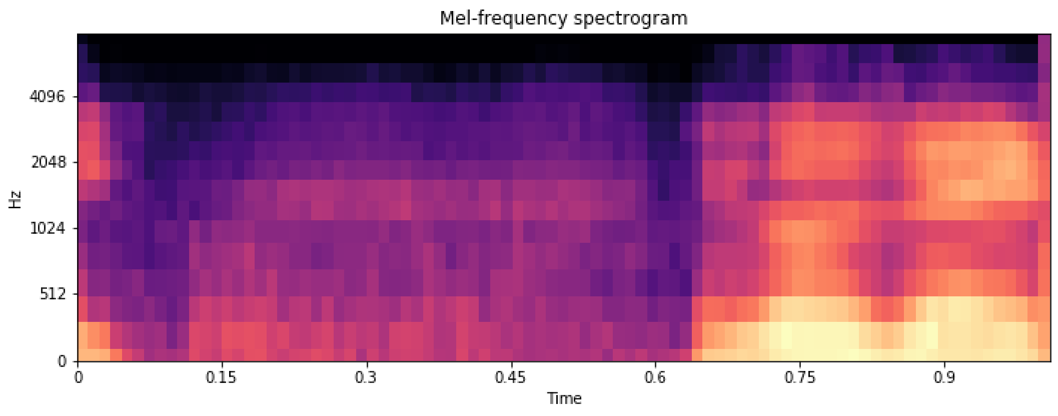

FIGURE 4.4: Mel-frequency spectrogram window size = 256, cepstral coefficients = 16.

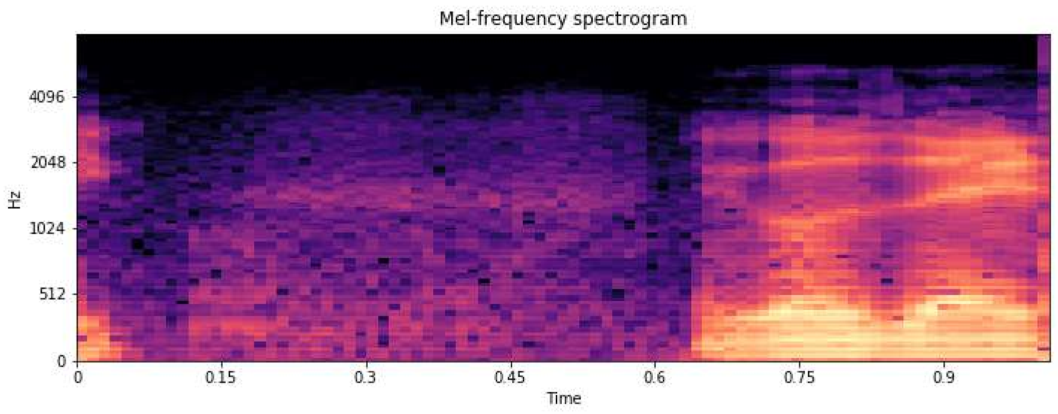

FIGURE 4.5: Mel-frequency spectrogram window size = 256, cepstral coefficients = 128.

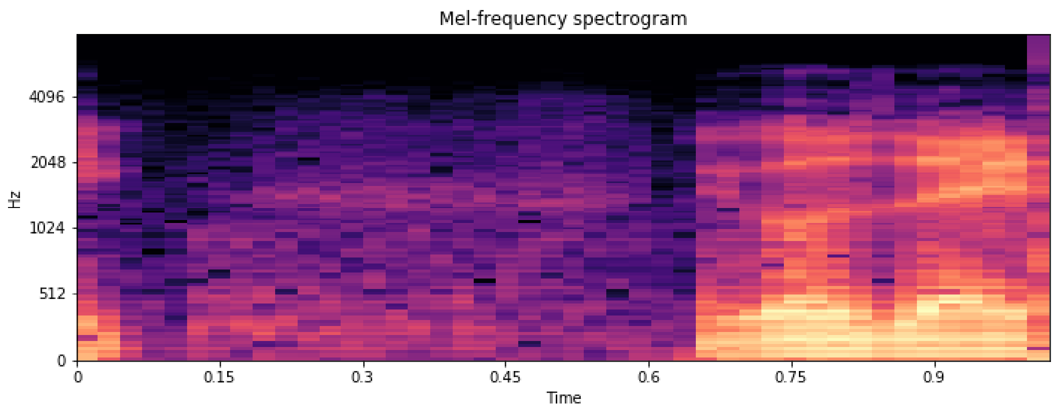

FIGURE 4.6: Mel-frequency spectrogram window size = 512, cepstral coefficients = 128.

### 4.3.3 Spectrogram

All spectrograms were calculated using FFT applied to the raw digital signal, with a sliding window of different length and 50% overlapping. The length of the window has a significant impact on spectrogram properties. With an increase in windows size, the resolution on time axis is decreasing, and the resolution on the frequencies axis is increasing and vice versa. From a practical point of view, it impacts the amount of information preserved in each pixel and the size of the pipeline output. The spectrogram of 1-second audio calculated with sliding window size *128* on Figure 4.7 has dimensions 64 x 256, while the audio fragment of the same length on Figure 4.8 with sliding windows *256* have dimensions 128 x 128. We tested spectrograms with 3 window sizes: 128, 256, 512.

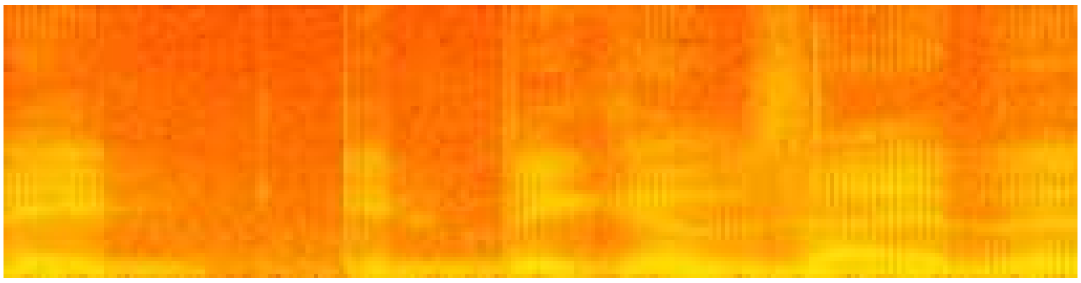

FIGURE 4.7: Sprectrogam of 1 sec audio fragment, FFT window length = 128.

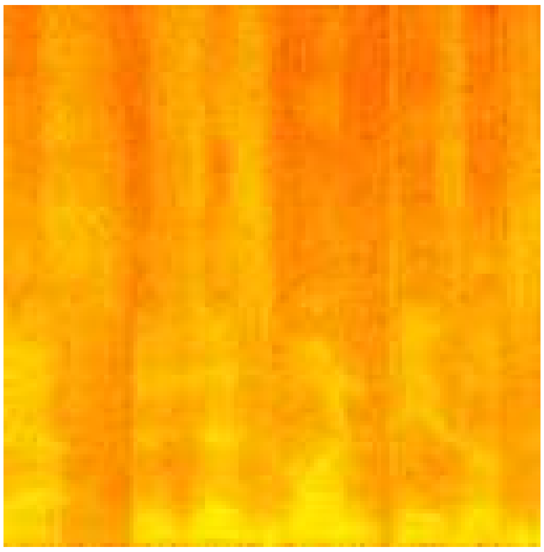

FIGURE 4.8: Sprectrogam of 1 sec audio fragment, FFT window length = 256.

### 4.3.4 Training Details

All the models were trained for 20 epochs and the best performing model is chosen for each front-end. The models were built using *Tensorflow package* back-end with *Keras package* as an interface. The training was conducted on NVIDIA GeForce GTX 1080 GPU.

### 4.3.5 Front-End Experiments Summary

The dataset is balanced, and false positive and false negatives have the same meaning in this comparison. Thus, *accuracy* was chosen as the primary metric for the models, see Table 4.1.

| Model | Accuracy | | |
|---|---|---|---|
| | 0.5 sec | 1 sec | 2 sec |
| MFCC 16 | 0.08 | 0.21 | 0.28 |
| MFCC 32 | 0.13 | 0.25 | 0.24 |
| Mel-Spectrogram 256 | 0.55 | **0.83** | 0.81 |
| Mel-Spectrogram 512 | 0.32 | 0.61 | 0.80 |
| Spectrogram 128 | 0.62 | 0.68 | 0.58 |
| Spectrogram 256 | 0.49 | 0.78 | 0.74 |
| Spectrogram 512 | 0.26 | 0.65 | **0.86** |

TABLE 4.1: Experimentation summary.

Summarizing obtained results:

- Spectrogram based model performed significantly better than MFCC ones. We explain that MFCCs preserve fewer data from raw input, and the spatial representation is lost in transformation, making it less beneficial to apply CNNs for high-level feature extraction.
- Longer sound fragments show generally better accuracy. It is not surprising as they contain much more data.
- The input with equal or closer to equal dimensions, generally performs better than rectangular with a high ratio between dimensions. We think it is connected with the form of the receptive field, as all the filters in CNN are square, they perform weaker on rectangular input. We tried to address this issue in final model, see section 4.4.
- **Spectrogram 512** model showed the best performance of accuracy = 0.86, so it was chosen as front-end for the final model.

## 4.4 Back-End Hyperparameters Tuning

After the the first stage our next task was to improve the result from baseline accuracy = 0.86 by tuning back-end part of the model. Each update was tested on short 10 epochs iterations, all positive changes were included into the next stages of testing.

### 4.4.1 Loss Function

*Multi-Class Crossentropy* is used as a loss function. Other losses like *Kullback Leibler Divergence loss* and *Hinge Loss* did not show any significant difference in training speed and model accuracy.

### 4.4.2 Overfitting

One of the encountered problems was overfitting. The accuracy of the model on training data was significantly larger than on validation data. In addition to a shallow layer before output described in section 4.2 and batch normalization[Ioffe and Szegedy, 2015], we applied L2 weights regularization for each layer and dropout between last two dense layers, with positive outcomes.

### 4.4.3 Convolutional Kernels

We learned from the first stage experiments that models with square inputs generally perform better, but our model has input $128 \times 256$. In order to mitigate this issue, we came up with the original solution of using rectangular kernels in convolutional layers, as we believe they have better receptive field coverage. In the final architecture, we have three convolutional layers. For them, we have used rectangular kernels, $3 \times 6$, $3 \times 6$, $2 \times 4$, resembling input dimensions.

### 4.4.4 Automated tuning

We used the automated hyperparameters tuner for finding the optimal CNN configuration [*Keras tuner*]. Optimizer was run against next parameters: $C \in \overline{2,5}$ - convolutional layers count, $F_c \in \overline{8,64}$ - count of filters for each layer, and $S \in \overline{8,64}$ - the size of the narrow layer. As a result, the final model consists of 5 layers, with three convolution layers having 16, 24, and 32 filters. The narrow layer has 14 neurons.

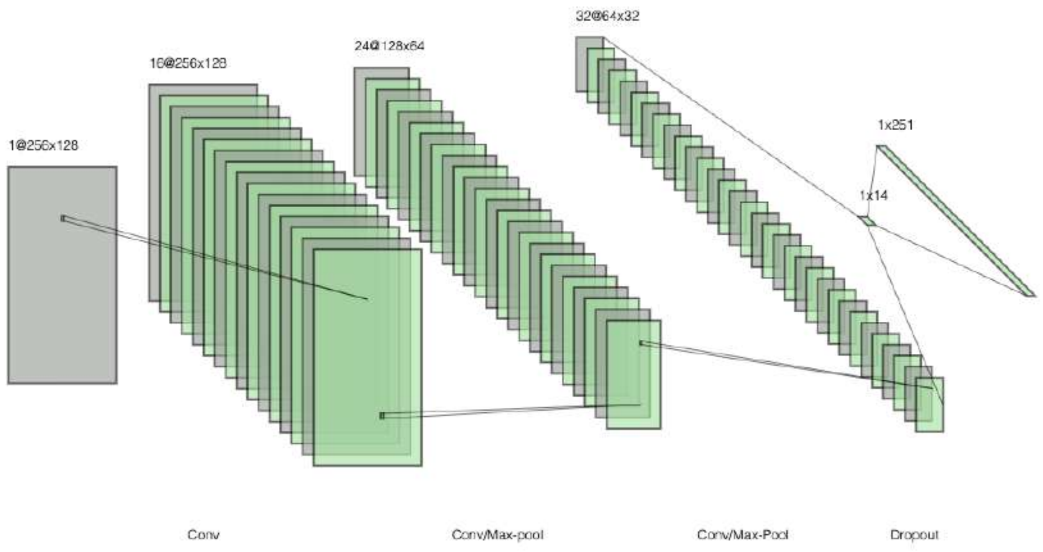

FIGURE 4.9: Final model architecture.

## 4.5 Model Evaluation

The model was evaluated against **LibriSpeech ASR Corpus**(3.1) and **ICSI Meeting Speech Corpus**(3.2). Testing on LibriSpeech corpus allows us to understand the full potential of the model since we know that labeling is 100% accurate, and speech quality is high. On the other hand, testing on ICSI Meeting corpus shows us a model performance on real-life data with a big amount of noise.(Table 4.2)

| Dataset | Accuracy |
|---|---|
| LibriSpeech corpus | **0.94** |
| ICSI Meeting corpus | **0.83** |

TABLE 4.2: Evaluation summary.

## 4.6 Conclusions

The proposed model showed performance comparable to state-of-the-art solutions. Given an accuracy of **0.94** on the clean dataset, it gives strong promise for future research. It worth to mention that the dataset was not cleaned from noise and silence fragments, therefore there is still space for accuracy improvements. Another fundamental feature of the proposed approach is **context independence**. As the model relies only on low-level features of the sound, it does not matter what kind of sound is classified, and it is not limited to speech in general. In the speech domain, it means that exact words or language do not impact model performance. The only condition is that training and evaluation data come from the same distribution.

The performance of **0.83** on the similar to real-life data refers to the fact that data quality matters. This dataset has a lot of overlapping speech and mediocre sound quality, but the accuracy is still significantly high, given that it contains 53 unique speakers. We came to the conclusion that any practical application should implement rigorous data preprocessing, including but not limited to denoising, removing non-speech sounds, dealing with overlapping speech, etc.

# Chapter 5

# Future Work

The presented model shows excellent potential, but the journey of building a product grade state-of-the-art diarization system has only started. We envision two main directions of development for this work.

## 5.1 Bridging the Diarization Gap

There is a gap between *speaker classification* and *speaker diarization* tasks. In our classification model, we assume that the number of speakers is known, and we knew each speaker before. As we try to build a real-life applicable solution, we would like to go to a solution where the speaker number is unknown. One of the possible solutions is building an ensemble model where before classification is done, speaker count is identified. Another possibility is to implement an end-2-end solution where both aspects are solved, as described by Zhang et al., 2019.

## 5.2 Extended Research

Taking into account the broadness of the field and accelerated activity of the scientific community, we can not claim that we explored all approaches to the problem. Some experiments were not conducted due to tight time constraints, some due to a lack of access to proprietary datasets. We believe that LSTMs (2.2.2) and TDNN(2.2.3), have great potential to be applied to the problem. Another interesting approach is to use CNN to extract the speaker's identity low-dimensional representation, similar to d-vectors (2.1.6). We plan to explore these possibilities in future works.

# Bibliography

Abdel-Hamid, Ossama et al. (Oct. 2014). “Convolutional Neural Networks for Speech Recognition”. In: *Audio, Speech, and Language Processing, IEEE/ACM Transactions on* 22, pp. 1533–1545. DOI: 10.1109/TASLP.2014.2339736.

Dehak, Najim et al. (June 2011). “Front-End Factor Analysis for Speaker Verification”. In: *Audio, Speech, and Language Processing, IEEE Transactions on* 19, pp. 788 –798. DOI: 10.1109/TASL.2010.2064307.

*DNN definition*. https://en.wikipedia.org/wiki/Deep_learning#Deep_neural_networks.

Fu, Szu-Wei et al. (Sept. 2017). “Complex spectrogram enhancement by convolutional neural network with multi-metrics learning”. In: pp. 1–6. DOI: 10.1109/MLSP.2017.8168119.

Galibert, Olivier and Juliette Kahn (Jan. 2013). “The first official REPERE evaluation”. In: 1012, pp. 43–48.

Garcia-Romero, Daniel et al. (Mar. 2017). “Speaker diarization using deep neural network embeddings”. In: pp. 4930–4934. DOI: 10.1109/ICASSP.2017.7953094.

Ghahremani, Pegah et al. (Sept. 2016). “Acoustic Modelling from the Signal Domain Using CNNs”. In: pp. 3434–3438. DOI: 10.21437/Interspeech.2016-1495.

Guo, Jinxi et al. (Oct. 2018). “Deep neural network based i-vector mapping for speaker verification using short utterances”. In: *Speech Communication* 105. DOI: 10.1016/j.specom.2018.10.004.

Hinton, Geoffrey et al. (Nov. 2012). “Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups”. In: *Signal Processing Magazine, IEEE* 29, pp. 82–97. DOI: 10.1109/MSP.2012.2205597.

Hochreiter, Sepp and Jürgen Schmidhuber (Dec. 1997). “Long Short-term Memory”. In: *Neural computation* 9, pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

Huang, Zhengwei et al. (Nov. 2014). “Speech Emotion Recognition Using CNN”. In: pp. 801–804. DOI: 10.1145/2647868.2654984.

Huang, Zili, Shuai Wang, and Yanmin Qian (Jan. 2018). “Joint i-vector with end-to-end system for short duration text-independent speaker verification”. In:

Hyun, Soo, Inkyu Choi, and Nam Kee Soo (2016). “Acoustic Scene Classification Using Parallel Combination of LSTM and CNN”. In:

Ibrahim, Salwani and Dzati Ramli (Jan. 2018). “I-vector Extraction for Speaker Recognition Based on Dimensionality Reduction”. In: *Procedia Computer Science* 126, pp. 1534–1540. DOI: 10.1016/j.procs.2018.08.126.

*Image 1*. https://www.nti-audio.com/en/support/know-how/fast-fourier-transform-fft.

*Image 2*. https://medium.com/@jonathan_hui/speech-recognition-feature-extraction-mfcc-plp-5455f5a69dd9.

*Image 3*. https://towardsdatascience.com/how-i-understood-what-features-to-consider-while-training-audio-files-eedfb6e9002b.

*Image 4*. https://medium.com/@AI_with_Kain/understanding-of-multilayer-perceptron-mlp-8f179c4a135f.

*Image 5*. http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Ioffe, Sergey and Christian Szegedy (Feb. 2015). “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In:

Janin, A. et al. (May 2003). “The ICSI meeting corpus”. In: pp. I–364. ISBN: 0-7803-7663-3. DOI: 10.1109/ICASSP.2003.1198793.

Karpathy, Andrej (2015). “The Unreasonable Effectiveness of Recurrent Neural Networks”. In:

Kenny, P. (2006). *Joint Factor Analysis of Speaker and Session Variability: Theory and Algorithms*.

Kenny, Patrick et al. (June 2007). “Speaker and Session Variability in GMM-Based Speaker Verification”. In: *Audio, Speech, and Language Processing, IEEE Transactions on* 15, pp. 1448 –1460. DOI: 10.1109/TASL.2007.894527.

*Keras package*. https://keras.io/.

*Keras tuner*. https://github.com/keras-team/keras-tuner.

Kumar, S. and J.G.M. Britto (Jan. 2012). “Speaker Change Detection- an Comparative Study using Support Vector Machines”. In: vol. 2012, pp. 161–165. ISBN: 978-1-84919-797-7. DOI: 10.1049/cp.2012.2208.

Lecun, Yann et al. (Jan. 1989). “Handwritten Digit Recognition with a Back-Propagation Network”. In: vol. 2, pp. 396–404.

Lecun, Yann et al. (Dec. 1998). “Gradient-Based Learning Applied to Document Recognition”. In: *Proceedings of the IEEE* 86, pp. 2278 –2324. DOI: 10.1109/5.726791.

Lee, Jongpil et al. (Jan. 2018). “SampleCNN: End-to-End Deep Convolutional Neural Networks Using Very Small Filters for Music Classification”. In: *Applied Sciences* 8, p. 150. DOI: 10.3390/app8010150.

*LibROSA package*. https://librosa.github.io/librosa/index.html.

Malykh Egor, Novoselov Sergey and Oleg Kudashev (May 2017). “On Residual CNN in Text-Dependent Speaker Verification Task”. In: pp. 593–601. ISBN: 978-3-319-66428-6. DOI: 10.1007/978-3-319-66429-3_59.

Mao, Qirong et al. (Dec. 2014). “Learning Salient Features for Speech Emotion Recognition Using Convolutional Neural Networks”. In: *Multimedia, IEEE Transactions on* 16, pp. 2203–2213. DOI: 10.1109/TMM.2014.2360798.

Meignier, Sylvain and Teva Merlin (2010). “LIUM SpkDiarization: An Open Source Toolkit For Diarization”. In:

*Mel scale*. https://en.wikipedia.org/wiki/Mel_scale.

Panayotov, Vassil et al. (Apr. 2015). “Librispeech: An ASR corpus based on public domain audio books”. In: pp. 5206–5210. DOI: 10.1109/ICASSP.2015.7178964.

Peddinti, Vijayaditya, Daniel Povey, and Sanjeev Khudanpur (2015). “A time delay neural network architecture for efficient modeling of long temporal contexts”. In: *INTERSPEECH*.

Rohdin, Johan et al. (Oct. 2017). “End-to-end DNN Based Speaker Recognition Inspired by i-vector and PLDA”. In:

Sainath, Tara et al. (Sept. 2014). “Deep Convolutional Neural Networks for Large-scale Speech Tasks”. In: *Neural Networks* 64. DOI: 10.1016/j.neunet.2014.08.005.

Snyder, David et al. (Apr. 2018). “X-Vectors: Robust DNN Embeddings for Speaker Recognition”. In: pp. 5329–5333. DOI: 10.1109/ICASSP.2018.8461375.

*Spectrogram wikipedia*. https://en.wikipedia.org/wiki/Spectrogram".

Stevens, S., J. Volkmann, and E. Newman (Jan. 1937). “A scale for the measurement of the psychological magnitude pitch”. In: *Journal of the Acoustical Society of America* 8, pp. 185–190. DOI: 10.1121/1.1915893.

*TDNN definition*. https://en.wikipedia.org/wiki/Time_delay_neural_network.

*Tensorflow package*. https://www.tensorflow.org/.

Variani, Ehsan et al. (May 2014). “Deep neural networks for small footprint text-dependent speaker verification”. In: pp. 4052–4056. ISBN: 978-1-4799-2893-4. DOI: 10.1109/ICASSP.2014.6854363.

Wang, Quan et al. (Apr. 2018). “Speaker Diarization with LSTM”. In: pp. 5239–5243. DOI: 10.1109/ICASSP.2018.8462628.

Zhang, Aonan et al. (May 2019). “Fully Supervised Speaker Diarization”. In: pp. 6301–6305. DOI: 10.1109/ICASSP.2019.8683892.