People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering
Signals and Systems Research Laboratory

PHD DISSERTATION

PRESENTED BY:

Youssouf Ismail CHERIFI

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTORATE

FIELD : ELECTRICAL ENGINEERING
OPTION : TELECOMMUNICATIONS AND COMPUTER ENGINEERING

TITLE:

Improvement of Voice Based Biometric System

JURY MEMBERS:

• Mr.	Abdelhamid	DAAMOUCHE	*Prof*	University M'Hamed BOUGARA – Boumerdes, Algeria.	Jury president
• Mr.	Abdelhakim	DAHIMENE	*Prof*	University M'Hamed BOUGARA – Boumerdes, Algeria.	Thesis director
• Mr.	Abderrezak	GUESSOUM	*Prof*	University SAAD DAHLAB – Blida, Algeria.	Examiner
• Mrs.	Dalila	CHERIFI	*MCA*	University M'Hamed BOUGARA – Boumerdes, Algeria.	Examiner

Academic year: 2020/2021

ABSTRACT

Identifying the speaker has become more of an imperative thing to do in the modern age. Especially since most personal and professional appliances rely on voice commands or speech in general terms to operate. These systems need to discern the identity of the speaker rather than just the words that have been said to be both smart and safe.

Although speaker recognition is considered a Natural Language Processing (NLP) task, it has not received the intention that other tasks had received over the last decades. This in fact is counter-intuitive considering the numerous advanced methods that have been developed for the sole purpose of generating fake speech segments for a given person, such as WaveNet, GenSynth, and MelGAN for example. The objective of this work is to bring attention back to the speaker recognition task and what can be done to obtain a robust and more accurate voice-based biometric systems that could keep up with the wave of advanced speech synthesizers.

The methodological approach adopted throughout this work relies on detecting the slightest changes that could occur in the speech from person to person. To that end, we focused primarily on defining a novel and more adapted speaker-specific features by implying transfer learning and artificial neural network. This approach increases the level of details that are transferred from the speech to the modeling algorithm while taking into consideration speaker specificity. This approach works because existing features suffer from keeping only one level of information either how speech is produced (type 1 features) or how speech is being perceived (type 2 features), and disregarding the other which provides certain details that are crucial to the task. By fusing the two types of features using artificial neural networks and then implying transfer learning a new set of features were obtained, a set of features that provides sufficient amounts of details that yielded a better recognition rate.

The primary results confirm the ability for such an approach to be used in modern voice biometrics systems, regardless of whether the system is identifying a speaker or just verifying his\her identity.

RESUME

La reconnaissance vocale est devenue, de nos jours, un besoin d'une importance cruciale. En particulier depuis que la majorité des outils utilisés dans la sphère privée et professionnelle comprennent des fonctionnalités basées sur des commandes vocales. Ces appareils doivent être capable de discerner l'identité du locuteur en plus des mots prononcés pour être à la fois un outil intelligent et sécurisé.

Bien que la recherche en reconnaissance vocale fasse parti des taches Traitement Automatique des Langues, elle n'a pas reçu la même attention que d'autre domaine sur les dernières décennies. Ce fait est d'autant plus surprenant que de nombreuses méthodes, tel que "WaveNet", "GenSynth" et "MelGAN", ont été développé dans le seul but de générer d'enregistrements vocaux artificiels pour un individu spécifique. L'objectif de cette thèse est donc de donner à la reconnaissance vocale l'attention qu'elle mérite afin de développer un système de biométrie vocale plus robuste et précis capable de rivaliser avec les nouvelles vagues de synthétiseurs vocaux

La démarche méthodologique utilisée à travers ce travail repose sur la détection des subtiles variations présentes dans la manière de s'exprimer de différents individus. Pour cela, nous nous sommes concentrés sur la définition de nouvelles caractéristiques à même de mieux représenter un locuteur spécifique en utilisant un réseau de neurone et l'apprentissage par transfert. Cette approche augmente le niveau de détails transféré du signal vocal à l'algorithme de modélisation tout en prenant en considération les spécificités du locuteur. Cette approche fonctionne car les préexistante caractéristiques ne représentait qu'une dimension de l'information vocale, qui était soit de type 1 (comment le message vocal est produit) ou de type deux (comment le message est perçu). Négliger l'une des deux dimensions empêche l'acquisition d'informations indispensable à la bonne réalisation de la reconnaissance. En combinant ces deux types de caractéristiques en utilisant un réseau de neurone artificiel et en appliquant l'apprentissage par transfert, nous avons obtenues une nouvelle série de caractéristiques qui contient suffisamment de détails pour permettre un meilleur taux de reconnaissance.

Le résultat principal de ce travail confirme la capacité d'une telle approche à être utilisée un système biométrique vocale moderne, indépendamment du faite que ce système cherche à identifier un individu ou simplement vérifier son identité.

نبذة مختصرة

أصبح تحديد هوية المتحدث أمرًا ضروريًا للغاية في العصر الحديث. خاصة أن معظم الأجهزة، الشخصية منها أو المهنية صارت تعتمد في عملها على الأوامر الصوتية أو الكلام بشكل عام. ولكي تكون هذه العملية ذكية وآمنة، ستحتاج هذه الأنظمة في عملها إلى تمييز هوية المتحدث، بدلاً من مجرد ترجمة الكلمات التي قيلت الى أوامر مباشرة.

على الرغم من أن التعرف على المتحدث يعتبر من مهام البرمجة اللغوية العصبية (NLP)، إلا أنها لم تحظ بنفس القدر من الاهتمام مقارنة بالمهام الأخرى على مدار العقود الماضية. وهذا في الواقع يخالف المنطق نظرا إلى الأساليب المتقدمة العديدة التي تم تطويرها لغرض وحيد وهو إنشاء مقاطع كلام مزيفة لشخص معين مثل WaveNet و GenSynth و MelGAN. الهدف الأساسي من وراء هذا العمل هو إعادة تسليط الأضواء على خاصية تحديد هوية المتحدث وما يمكن فعله من أجل الحصول على نظام بصمة صوتية أكثر دقة وفعالية يكون بمقدوره مواكبة موجة آلات تصنيع الكلام المتقدمة.

المنهج المتبع في هذا العمل يعتمد على الكشف عن أدنى التغييرات التي يمكن أن تحدث في الكلام من شخص لآخر. لتحقيق هذه الغاية، ركزنا في المقام الأول على تحديد ميزات جديدة وأكثر تكيفًا خاصة بالمتحدث باستخدام نقل التعلم والشبكة العصبية الاصطناعية. يزيد هذا النهج من مستوى التفاصيل التي يتم نقلها من الكلام إلى خوارزمية النمذجة مع مراعاة خصوصية المتحدث. يفرض هذا النهج تفوقه لأن الميزات الحالية تحتفظ بمستوى واحد فقط من المعلومات، إما كيف يتم إنتاج الكلام (ميزات من النوع 1) أو كيف يتم إدراك الكلام (ميزات من النوع 2)، بينما يتجاهل المستوى الآخر الذي يحتوي على تفاصيل مهمة. من خلال دمج كلتا الميزتين باستخدام الشبكات العصبية الاصطناعية ثم تضمين التعلم بالنقل، تم الحصول على مجموعة جديدة من الميزات تحتوي على تفاصيل كافية والتي حققت معدل تشخيص أفضل.

تؤكد النتائج الأولية قابلية استخدام منهج مشابه في أنظمة القياسات الحيوية الصوتية الحديثة، سواء لغرض تشخيص المتحدث أو التحقق من هويته.

ACKNOWLEDGMENTS

First of all, I would like to praise almighty ALLAH for giving me the power to realize this work.

I would also like to thank my supervisor Dr. Abdelhakim DAHIMENE for his support, guidance, and help during the different phases of the project.

Many thanks are given to all the professors and workers at the institute of electrical and electronic engineering (IGEE). Especially all the fellow students who helped me with recording my primary corpus.

Additionally, my deepest gratitude for Dr. Angelo ARLEO for providing access to the HPCaVe supercomputing machine, that without I would've not been able to obtain the most recent results concerning transfer learning and advanced ANN structures.

Finally, I would like to thank the reviewers for taking the time and the effort to read my thesis and provide the necessary comments to make it better.

DEDICATION

I dedicate this work to my supportive and understanding parents, my cheerful sister, and to all my friends especially Abdo, Karim, and Khalid for always being there throughout my journey.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

ANN	Artificial Neural Network
CNN	Convolutional Neural Network
DF	Deep Features
DL	Deep Learning
DCT	Discrete Cosine Transform
FFT	Fast Fourier Transform
FC	Fully Connected Layer
GMM	Gaussian Mixture Model
GUI	Graphical User Interface
HMM	Hidden Markov models
LP	Linear Prediction
LPCCs	Linear Predictive Cepstral Coefficients
LPCs	Linear Prediction Coefficients
LSTM	Long Short-Term Memory
MFCCs	Mel-Frequency Cepstral Coefficients
NLP	Natural Language Processing
PR	Pattern Recognition - PR
PLP	Perceptual Linear Prediction - PLP
RNN	Recurrent Neural Network - RNN
SR	Speaker Recognition
SSE	Sum of Square Errors
SVM	Support Vector Machine
VQ	Vector Quantization
VBBS	Voice-Based Biometrics System
VCD	Voice Command Devices
VUI	Voice-User Interface

SYMBOLS

V	Vector of values
Hz	Hertz
F	Frequency
e	Error function
a	LP predictor coefficient
c	Feature's coefficient
P	Power spectrum
R	Reference vector
f	Probability distribution
μ	Mean
σ	Standard deviation
λ	Probability distribution parameters
K	Kernal function
W	Window function
H	Filterbank or critical function
s	Signal function
d	Distance function
β	Decision boundaries
Φ	Support vector machine mapping function
M	Mel-scaling function
Ω	Bark-scaling function
Θ	Bark wrapped power spectrum
x	Input vector
y	Output vector
t	Target vector
w	ANN weights
∂	Partial derivative
α, δ	ANN training parameters
f	Filter function

CHAPTER I
INTRODUCTION

1.1 General introduction

Modern age devices have become more user friendly and natural to interact with than ever, and perhaps there is no functionality that made this true then voice-user interface (VUI) [1]. VUI uses speech recognition technology to make spoken human interaction with machines a possibility. Generally, this is done to either understand spoken commands, answer a question, or convert speech into text for people with disabilities [2]. In fact, VUI has been added to a variety of devices making them voice command devices (VCD). Automobiles, home automation systems, computer operating systems, phones, and even home appliances such as washing machines [3], microwaves, and televisions have all become VCD. In fact, as of 2019, an estimated 3.25 billion digital devices were used across the world containing some sort of VUI and by 2023 this number will more than double, reaching approximately eight billion units [4]. Given the wide application of VUI, it has become imperative to identify not just the words that have been spoken but also who is issuing these commands to make modern VCD both smart and safe [5]

Identifying a speaker is considered a task of Natural Language Processing (NLP) field, but unlike the major commonly researched tasks (i.e., speech recognition, speech-to-text, speech segmentation...), this task has received little to no attention over the last decade. What makes the situation even worse is the quick and sudden rise in Deepfakes technology. Deepfakes has become a real threat since cybercriminals are now able to create believable fake audio by having access just to 5 seconds of their target's voice [6]. These circumstances could potentially cost businesses across the globe well around 250 million USD by the end of 2020 [7], as the number of CFOs and CEOs falling victim to successful deepfaked audio scams keeps on increasing [8], [9]. Thereby, developing a more robust and secure voice-based biometric system (VBBS) is beneficial not only to individuals but to businesses as well.

The methodological backbone of this work stems from the fact that for decades now we have been using the same kinds of features for both speech recognition and Speaker Recognition (SR) [10] without considering more adaptable and appropriate features. Features that will utilize all the fluctuation present in a regular speech recognition feature in order to generate speaker-specific features. Furthermore, in addition to the fact that the features that have been used so far are ill-suited for SR tasks, at the end of the extraction pipeline they provide only one level of information, although the original signal has multiple levels of information. These levels of information can be split into two large categories, either features that represent how speech is produced (i.e., type 2 features) or those who represent how speech

is perceived by the human ear (i.e., type 1 features) [5]. A key asset of our method is that it uses both levels of information. This is achieved by implying the flexibility that Artificial Neural Networks (ANN) offers in handling multiple inputs, to fuse the two categories. ANN achieve this by providing each input with its own set of weights and biases, this can be better viewed as a set of isolated regressors that intern will be connected to their own set of weights and biases. These interconnections are what transform the inputs to the desired outputs in a sort of black box system. However, what makes ANN very useful and skillful, is the fact that at each level of interconnection the network extracts its own set of intermediary features that are in a way simpler and contains all the necessary information from the previous level. These intermediary features are known as Deep Features (DF) and in addition to being simpler, they are more adaptable and more task-dependent compared to the original signal [11]. DF carries out knowledge about the input that can be transferred successfully to a different use case, or an entirely different field of application if trained with a sufficient amount of data [12], a practice that is often been used to optimize the learning process and avoid the need for huge data sets.

The work presented here extends on these findings by directly answering the question, what if transfer learning were to be used to develop a novel, improved, and better-suited speaker-specific features [13]. Features that in addition to being more task-related, contain both levels of information that the speech signals transmit (i.e., information about how speech is perceived and produced). To answer this question, a series of experiments were conducted over the course of 5 years, with the focus on feature extraction and how tuning this procedure's hyperparameters influences the speaker identification accuracy. This was conducted on varying corpus in terms of size and difficulty. The primary results are promising and confirm that these transferred features can be applied to achieve an improved VBBSs which will have great implications on the following fields.

Security VBBSs are used to provide transaction authentication, granting access control to computers and even facilities, monitoring, and telephone voice authentication for long-distance calls or access to banking.

Personalization VBBSs can help with the implementation of intelligent responding machines with a personalized caller greeting. This means building a personalized dialog system, a dialog system that can recognize the user, and greet him directly, and offer him a user experience that is much simpler, more optimized, and take into consideration the information that's provided by the user's profile.

Audio Indexing VBBSs can be used to automatically label speech segments in meeting recordings for speaker-dependent audio indexing.

Information Retrieval VBBSs can be used to retrieve information according to specific speakers this in a way can help with managing and accessing easily for multimedia databases.

Speaker Tracking with VBBSs it feasible to keep track of who is speaking in teleconferences. This becomes extremely useful when the number for attendees is high but at the same time, they are unfamiliar with each other.

1.2 Roadmap of this thesis

The second chapter of this manuscript introduce what is SR and review its state-of-the-art by crossing the research fields that are at the core of most VBBSs (i.e., speaker specific features, voice-prints modeling, and pattern recognition algorithms). At the end of this section, we present a set of objectives that stemmed from the reviewed body of work and that have been experimentally addressed by this work. Chapter 3 and 4 go about identifying the tools that are selected to meet the objectives that are put together in the proceeding chapter.

Next, we present the main experimental contributions of this work through chapter 5. This chapter describes the methodology that was followed to address all experimental questions (e.g., the use of feature fusion and transfer learning for robust SR). This was done through 4 major steps (each is covered in a dedicated section), recording of the benchmark corpus, implementing the MATLAB toolbox for SR, evaluating the effect of feature fusion on recognition rate, training time, and segment length. Finally, we describe the transfer from using feature fusion as a mean of improving SR to a method for extracting a novel, improved, and better-suited speaker-specific features. These features were tested with several top-performing ANN structures. Chapter 6 explore the results from each step in the proceeding chapter and dive into way certain outcomes are the way they are. The findings presented in Part II are from two original articles that originated directly from this work. These articles are presented in this format instead of the original published format for ease of read and to avoid the redundancy present in both articles.

Finally, we conclude the manuscript by providing a general discussion of all main contributions in light of what have been reviewed in chapter 2, as well as delving into what would be tackled in future works.

CHAPTER II
SPEAKER RECOGNITION LITERATURE REVIEW

This chapter is dedicated to the presentation of the classical concepts used throughout the SR field, in order to familiarize the reader with questions such as: what is a SR? How is speaker identity discerned? and what is the pipeline that needs to be followed in order to do so? The chapter proceeds by breaking down each of the required steps to discern someone's identity based on voice and provides insight on what could be done to improve the open said step.

The main reason behind revisiting the principles of SR is to address what steps have the potential of improving the recognition rate. This would help narrow down the amount of literature review that needs to be done to focus only on those areas with a substantial chance of improving VBBSs. The criteria for doing so were: i) the number of years since the last major contribution to said step, and ii) the amount of the work that has been carried out in terms of applicable technologies and algorithms. The objectives defined in this chapter set forth what steps would impact the overall SR and should be reconsidered.

2.1 Speaker Recognition Principles

It is a well-known fact that speech itself contains several levels of information conveyed to the auditor. Mainly, speech is used to communicate a message to the auditor, a message that is carried in words. However, the speech also contains information about the speaker himself, this is due to the way it is produced. This information could be for instance the speaker's gender, emotions, age, origin, and obviously identity. Recognizing a person's identity by analyzing a portion of his speech is known as SR [14]. Depending on the nature of the speech and the number for speakers involved, the general applications of SR can be categorized as one of three specific tasks: identification, detection/verification, and segmentation and clustering [14 – 16], as depicted in Figure 2.1.

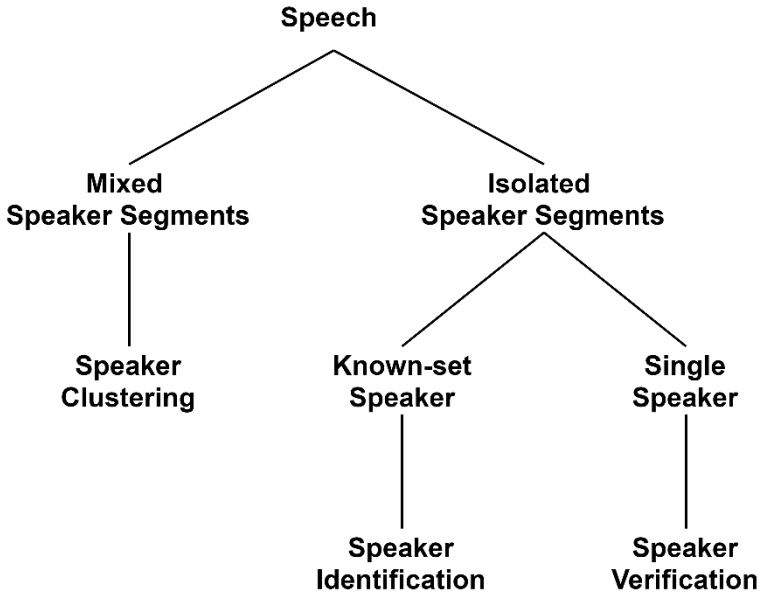

Figure 2.1- Speaker recognition general applications.

The aim of the speaker identification task as indicated by the task's name is to identify which speaker out of a known group of speakers produced the provided speech segments. There exist two modes of operation when it comes to the known-set of speakers, a closed-set mode, and an open-set mode. Closed-set systems assume that the to-be-determined speech segment must come from a set of known speakers. On the other hand, the open-set systems must take into consideration the fact that the speaker is not necessarily known, and that the to-be-determined speech segment could've originated from a completely new speaker, this speaker is referred to as an impostor. Closed-set identification systems can be seen as a multiple-class classification problem. Open-set systems are more of a clustering problem, and they are both perfect for forensic applications, i.e., using speech as evidence to discern the criminal's identity among several known suspects.

In speaker verification, the aim is to determine whether the speaker is who he/she claims to be according to his/her voice. The task is also known as speaker authentication, talker verification or authentication, voice verification or authentication, and speaker detection. verification is a true-or-false binary decision problem that can be considered as a single open-set speaker identifier since the system is required to distinguish a claimed speaker's voice known to the system from a potentially larger group of voiceprints that should be treated as unknowns. Verification is the most commercially viable task hence this task is at the core of most commercialized SR applications. Applications that have to do with security, such as, controlling or accessing banking services via telephone.

Speaker segmentation or clustering systems are useful when dealing with multiple-speaker scenarios. In most speech recognition and SR applications, the provided speech segments are often assumed to be from a particular individual. In reality, this is not often the case, and the speech of the speaker in question is entangled with speech segments from other speakers. In these scenarios, it is required to isolate the speech into separate segments that belong to individual speakers before proceeding to the recognition process. The aim of the clustering task is to achieve so by dividing the input speech into homogeneous segments and then label each segment with a speaker identity. The task has received a lot of attention recently due to the increased use of multiple-speaker audio such as meetings and recorded news shows, commonly present in web searches and consumer electronic devices. Speaker segmentation and clustering is one method of indexing audio archives to make for an easier retrieval if needed.

In addition to the first major categorization of SR, each of the aforementioned applications can be further split into text-dependent or text-independent tasks, according to the constraints placed on the speech segments used for training and testing the system. This relies on whether the user must speak a given sentence that is known to the system or no, having prior knowledge about the spoken sentence can provide details that yields in a better recognition rate, although this system has far fewer flexible applications, the reduced accuracy that is inherent in text-independent recognition make them less of liable choice for certain use cases.

Running any of the SR systems described above typically involves two phases. The first phase is enrollment or training. The user enrolls the speaker in the system by providing voice samples of him/her to the system. The system extracts speaker-specific features from the provided voice samples to build a voiceprint model for the enrolled speakers. The second phase is the recognition or testing. The user provides a different voice sample that is used cross-referenced in the system with all the previously enrolled voiceprints. A measure of similarity to each print is used to make a decision. The speaker associated with the print that is being tested is referred to as the target speaker or claimant. Depending on the task at hand the system might measure similarity to all the enrolled voiceprints (i.e., speaker identification task) or a single print (i.e., speaker verification task). The decision-making process can also differ across systems. For example, a closed-set identification task outputs the identity of the recognized user, an open-set identification task on the other hand is supposed to not only output the identity but also choose to reject the user's voice in case the sample does not belong to any of the stored voiceprints.

2.2 Structure of a Speaker Recognition System

SR is a pattern recognition (PR) problem, and like most PR problems, it consists of two main blocks: a feature extraction block and a classification block. The classification block consists of two steps: pattern matching and decision-making. Figure 2.2 depicts the general structure of a SR system regardless of the application.

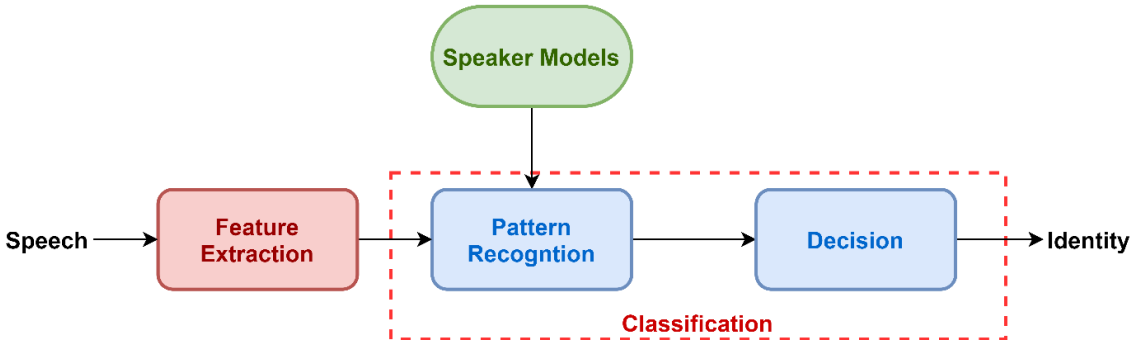

Figure 2.2 - The general structure of speaker recognition systems.

The role of the feature extraction block is to extract some speaker-specific information that is representative of the speech signal. The resulting speaker-specific information ideally should capture the complex transformations that the speech underwent for it to be produced. There are several ways for categorizing the extracted information, from a physical interpretation point of view features can be divided into semantic, phonologic, phonetic, and acoustic [16, 17]. Each level captures a certain aspect of the speech. First, the semantic level captures the transformation that affects the speech signal due to the speaker's intent during communication or interaction during a dialogue. In fact, this level of information allows you to know a lot about the speaker. For example, the implied vocabulary and sentence structure can be used to discern an individual's socioeconomic status and his/her educational background [18]. The phonological level captures all transformations that affect the vocalized phonemes due to communicative intent. This would be for example phonemes selection, duration, and intonation. This level is very useful for discerning the individual's native language or origin. On the other hand, the phonetic level captures transformation that went into producing the phoneme. This is represented by the vibration of the vocal cords and the movements of articulators (i.e., lips, jaw, tongue, and velum) within the vocal tract [19]. This level is not very reliable since a singular speaker can imply a different set of articulatory movements to vocalize the same phoneme [20]. Finally, the acoustic level captures the spectral parameters of the speech signal. This for example could be the mass and length of vocal folds or the dimensions

of the vocal tract which defines in a way the fundamental and resonant frequency. Although there exists a multitude of speaker-specific features, the information provided by said features should fulfill the following criteria [20, 21]:

- ease of extraction
- high inter-speakers' variability
- natural and frequent occurrence in speech
- intra-speaker consistency
- unaffected by aging or health issues
- robustness to noise and transmission artifacts
- mimicry-proof

Although not all of these criteria can be found in existing features, SR systems do exist and that is primarily thanks to pattern matching block.

The pattern matching block responsibility lies in cross-referencing the extracted features with the existing voiceprint models. There exist several types of pattern matching algorithms and corresponding models that are applied in SR [16]. Some of the early algorithms include Hidden Markov Models (HMM), Dynamic Time Warping (DTW), and Vector Quantization (VQ). For open-set systems (i.e., speaker verification and open-set speaker identification), the extracted features could additionally be cross-referenced to a voiceprint that models the unknown speakers. In verification systems, this block provides a similarity measure between the test sample and the claimed identity. In identification systems, the block provides a similarity measure for all stored voiceprints. The decision block analyzes the similarity measures (statistical or deterministic) and provides the user with a decision about the speaker identity, a decision that is heavily dependent on the system task. For a closed set identification task, the decision selects simply the identity associated with the voiceprint that has the highest similarity measure to the test sample. In an open-set identification/verification task, the decision block needs to have a threshold to verify whether the similarity measure is sufficient or not. Since open-set tasks are required to also reject speakers, the cost of committing an error needs to be considered when defining the threshold. For example, A bank might want to tolerate a few rejections of true bank customers than to provide imposter access to banking services given the stakes.

The effectiveness of a SR system is evaluated differently for different applications. For instance, a closed-set speaker identification system provides the user with a speaker identity from a set of known speakers hence the most logical measure of effectiveness is identification accuracy. For open-set systems (i.e., open-set speaker identification or speaker verification

systems) there are two types of errors: accepting an impostor and rejecting a target speaker. The measure of effectiveness should somehow incorporate the cost associated with each error while considering the application, for example, a false acceptance is very costly in a telephone credit card purchase system, whereas a false rejection in a fraud prevention system can be daunting for costumers.

The objective of this work is to focus on the improvement of VBBSs, to achieve this the general structure of the SR system has to be considered. As shown in Figure 2.2, any SR system regardless of the underlying task consist of several blocks. However, based on the criteria set in the chapter's introduction, only two blocks could be of importance to this work, the feature extraction block and the pattern matching block. These two blocks play a major role in the recognition process and thus hold a substantial potential of improving the effectiveness of said systems. This assumption is based on the fact that the existing set of features are not ideal, coupled with the introduction of several novels and top-performing PR algorithms.

Several important aspects have to be taken into account in order to define what should be done at the level of each block. To that end, a full-body literature review needs have been carried out. This implied examining thoroughly what has been done so far and identifying for each method or approach what are its pros and cons.

2.3 Literature Review

2.3.1 Speaker Specific Features

This section investigates further the feature block and the role that it plays in the SR tasks. So far, the only thing that was covered is what features could be considered for a SR application, what information do they provide about the speaker, and ideally what characteristics should they have in order to function as a real speaker-specific feature. However ideal, the speaker-specific feature doesn't exist and that is what this chapter is all about. A search through decades of SR related works has been conducted in order to i) identify what features were the most prominent in term of effectiveness ii) how the information provided such features although not ideal was used iii) what's missing in terms for the characteristics set before [20].

Here we are only reviewing acoustic features and no detailed explanation on the full mathematics behind it is provided, as this would take hundreds of pages. A brief description of how to extract the chosen features will however be covered in chapter 3.

Speech signal contains several level information which is not all related to SR. However, as discussed briefly in 2.2, there are several features that could be considered speaker

discriminator. Our original dichotomy was based on the physical interpretation of the provided information. This divide features into either semantic, phonologic, phonetic, or acoustic. However, a fifth category will also be considered, which provides information about the temporal fluctuation in each one of these features, these are the Spectro-temporal features.

Semantic level

Speakers' differences are not always related to physical ones. In fact, some of the differences are related to the speakers' lexicons (i.e., the sort of words that a speaker might have the tendency to use more often in a conversation). The differences are captured by semantic features, which could also be considered ''high-level" conversational features [22]. The idea is to use idiolect (i.e., the speaker's characteristic lexicon) to discern his/her identity. Semantic features operate on the idea of associating each utterance with a sequence of tokens, each co-occurring pattern of tokens corresponds to a specific speaker. The modeled information by this level is a categorical or discrete one rather than being numerical or continuous. Although the original idea was about tokenizing words [22], the approach extended over the years to include phones [23, 24], prosodic gestures (i.e. rise and fall in pitch or energy) [25–27], place and manner in which articulation occurs [28] and even the highest-scoring Gaussian mixture component indices has also been tokenized [29–31].

Some works extended the use of tokenizers by incorporating several parallel ones [24, 29, 32]. This is inspired partially by the success of parallel phone recognizers in state-of-the-art speech recognition [33, 34]. This wave of work was driven by the assumption that different tokens (e.g., training phone recognizer with different phone models) would provide complementary information about the speaker. For example, the work done by [29] used a set of parallel tokenizers obtained by Gaussian Mixture Model (GMM) [30, 31]. Each set of tokens was trained using a different group of speakers provided by the GMM clusters.

The standard classification of tokenizers is based on N-gram modeling. This means if a sequence of speaker's tokens is denoted by $\{\alpha_1, \alpha_2 \dots \alpha_T\}$ where $\alpha_t \in V$ and V is the set of finite vocabulary. To construct the N-gram model, the joint probability of N consecutive tokenizers needs to be estimated. For example, if $N = 2$ constructs a bigram model for which the probability of all $\{\alpha_t, \alpha_{t+1}\}$ tokenizer pairs need to be estimated. A trigram model contains all triplets $\{\alpha_t, \alpha_{t+1}, \alpha_{t+2}\}$, and so forth. For instance, the bigram model of the sequence hello_world would consist of the following token pairs (h,e), (e,l), (l,l), (l,o), (o,_), (_,w), (w,o), (o,r), (r,l) and (l,d).

The estimation of each N-gram probability is carried out in a similar manner as N-gram in statistical language models used for automatic speech recognition [35], by using the maximum likelihood or maximum a posteriori to estimate the N-gram for the training segments [28]. This approach coupled with the use of vector space [24, 29] and entropy measures [28, 36] to evaluate the similarity between speakers.

Phonological level

The phonological level features capture non-segmental aspects of speech, which include aspects such as rhythm, syllable stress, intonation patterns, and speaking rate. A key about these features is that they span over long segments of speech, this could be syllables, words, and utterances. These features also reflect differences in emotional intent, speaking style, language background, and sentence structure. Capturing speaker differences is a challenging task using this level since the features need to be free of all effects of voluntary control.

One of the most prominent phonological parameters is the fundamental frequency (F0). F0-related features have been shown to be very effective when combined with spectral features, especially when there is inherent noise in the segments. F0 is not the only feature that provides phonological information, other features have included speaking rate, phone duration, pauses, and energy distribution among others [25, 27, 37, 38]. In their work [27] Shriberg et al conducted a comparison between the aforementioned features and they concluded that F0-related features outperformed the remaining features in terms of accuracy, followed by energy features and duration.

Practically, determining F0 reliably is a difficult task. For example, the quality of telephone speech is often operating within the narrowband of 0.3kHz to 3.4kHz a bandwidth that doesn't contain F0. Algorithms need to use the information provided by the upper harmonics of F0 for its detection [39].

When it comes to SR applications, F0 provides both physiological and learned information. For example, the mean value of F0 is related to the size of the larynx [40], the temporal variations of pitch on the other hand are more related to the manner of speaking. This made F0-related features suited for both text-dependent application [41] by implying temporal alignment of pitch contours, and text-independent application [20, 42–44] by implying long-term F0 statistics such as the mean.

In fact, the F0 mean has been combined with other statistics such as kurtosis and variance to model speaker difference [37, 42], although histograms [45], latent semantic

analysis [26] and support vector machines [27] provided better accuracy. It has also been established that log of F0 outperformed F0 itself through several experiments [44, 45].

F0 is a one-dimensional feature, therefore it can't be mathematically expected to be that discriminative toward speaker differences. That is why some works extracted Multidimensional phonological features such pitch- and voicing-related features through auto-correlation instead of passing by the F0 extraction as in [46–48] for example.

Phonetic level

Phonetic level features capture the glottal excitation signal related to voiced sounds, this means mainly glottal pulse related parameters. Theoretically, these parameters are more speaker-specific compared to previous parameters. For instance, the degree of which vocal fold opening and the duration of the closing phase could be considered phonetic level features. These features are perfect for describing for example the quality of the speech in models such as breath, creaky, or pressed [49].

Measuring glottal related parameters is not a direct task due to the presence of the vocal tract filtering. Assuming that each source (i.e., the glottal source and the vocal tract) is independent of the other, what can be done is estimating the vocal tract parameters using a linear prediction (LP) model, followed by inverse filtering of the original waveform to estimate the glottal source signal [50–54]. Alternatively, a closed-phase covariance analysis could be used for when the vocal folds are closed [51, 55, 56]. This has been proven to improve the estimated vocal tract, however accurate detection of duration for when vocal folds are closed which is a challenge in noisy conditions. This is better demonstrated by [57] through Figure 2.3.

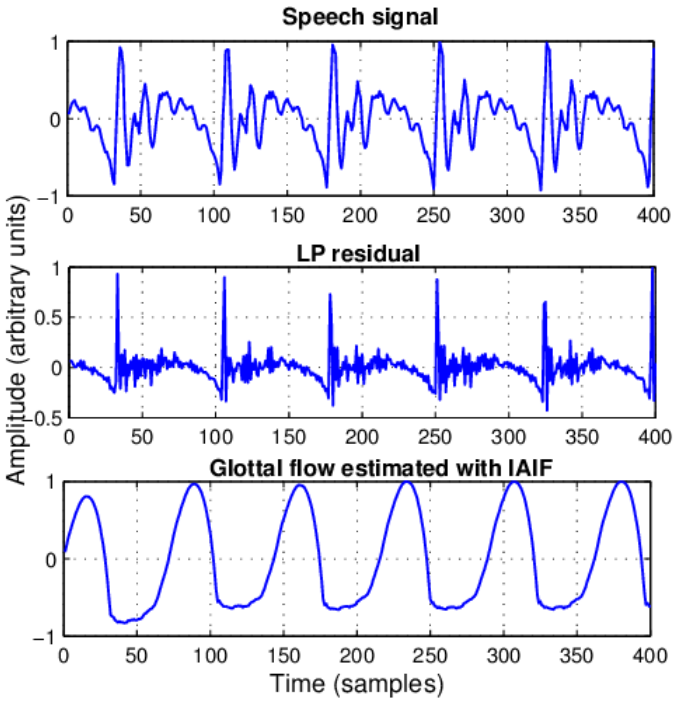

Figure 2.3 - Glottal feature extraction.

To mention a few of the methods used for extracting features of the inverse filtered signal, an auto-associative neural network has been used [52] as well as parametric glottal flow model parameters [51], wavelet analysis [54], residual phase [50], cepstral coefficients [55], [58] and higher-order statistics [58].

Based on the aforementioned literature, phonetic level features are not as discriminative to vocal tract parameters. This however could be improved upon by fusing the two complementary levels of features [50, 54]. In [52, 59] the amount of training and testing data required for both phonetic level features and vocal tract parameters. The amount was much less for the first feature compared to the later (4 times less, 10s for phonetic level features vs the 40s for vocal tract parameters).

Acoustic level

Acoustic level features are the most prominent features and the ones that have been used the most in the field of SR and perhaps no feature is as popular as the so-called Mel-frequency cepstral coefficients (MFCCs) [60]. These features were introduced in the early 1980s as a general method for audio processing and speech recognition and were adopted afterward for

SR. In practice, MFCCs are near impossible to beat as even when compared with various more recent features, such as spectral subband centroids (SSCs) [61] they performed better.

MFCCs are extracted with the aid of psychoacoustically (i.e., how sound or audio is perceived) inspired filterbanks, after which a logarithmic compression and a discrete cosine transform (DCT) is applied. The obtained MFCCs can be expressed as:

$$c_n = \sum_{m=1}^{M} [\log H(m)] \cos\left[\frac{\pi n}{M}\left(m - \frac{1}{2}\right)\right] \quad (2.1)$$

Where n is the index of the cepstral coefficient c and $H(m)$ is the M-channel filterbank. Finally, the lowest 12-15 DCT coefficients are retained to form the final MFCC vector. Alternative MFCC driven features have been developed and studied with the purpose of emphasizing speaker-specificity [62–64].

An alternative spectrum estimation method is the LP [65, 66]. Similar to MFCC, it uses DFT and it has a good intuition about the interpretation of both correlated adjacent samples (in the time domain) and resonant corresponding poles (in the frequency domain). LP predictor is defined as follow.

$$\tilde{s}[n] = \sum_{k=1}^{p} a_k \, s[n-k] \quad (2.2)$$

Where s is the observed signal, a_k are the LP predictor coefficients and $\tilde{s}$ is the predicted signal. The prediction error between the two signals as illustrated in the middle plot of Figure 2.3 is obtained by.

$$e[n] = s[n] - \tilde{s}[n] \quad (2.3)$$

In order to determine the predictor coefficients a_k this error need to be minimized by implying the so-called Levinson–Durbin algorithm [19, 67, 68]. In the frequency domain, the predictors are defined as.

$$\mathrm{P}(z) = \frac{1}{1-\sum_{k=1}^{p} a_k z^{-k}} \tag{2.4}$$

And it consists only of spectral poles (Figure 2.4).

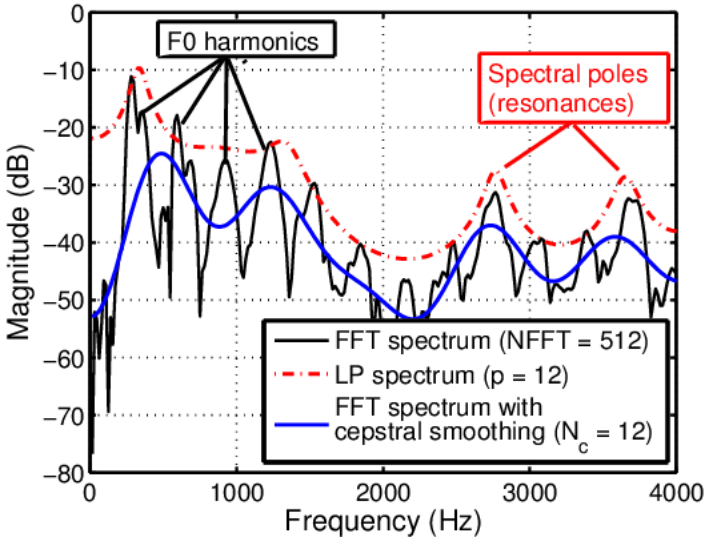

Figure 2.4 - Spectral envelope extraction with linear prediction (LP).

The predictor coefficients a_k themselves are not often used for SR. Instead, they are improved upon and transformed into a more robust and less correlated feature such as the linear predictive cepstral coefficients (LPCCs) [68], the perceptual linear prediction (PLP) coefficient [69], and the line spectral frequencies (LSFs) [68]. Of course, there are other less successful features in a sense such as the partial correlation coefficient (PARCORs), the log area ratios (LARs), and the formant bandwidths or frequencies [19].

The decision of choosing which acoustic features to be used for SR is not an easy task. Especially, since these features have several parameters that need to be tuned and offer two complementary information (i.e., psychoacoustics-related features or vocal track related features). However, the studies and comparisons conducted thus far indicated that in practice the following features: MFCC, LPCC, PLP, and LSF cannot be outperformed [70, 71].

Spectro-temporal features

The spectro-temporal features are not included in the original dichotomy as they do not provide independent information of the speech. Instead, they provide details such as energy modulations and formant transitions of an already extracted speaker-specific feature. A common way of incorporating such level is through the first and second-order time derivative,

also known as the delta D and delta-delta DD coefficients [68, 72, 73]. To compute these coefficients, it is sufficient to consider the time differences between adjacent coefficients of a said feature. Usually, this appended to the original coefficients for each frame, so if N MFCCs are used the resulting vector will contain 3 times N per frame if both D and DD are considered. Other alternatives include fitting a regression line [19] or an orthogonal polynomial [72] to the temporal signal, as well as time-frequency principal components [74] and data-driven temporal filters [75]. These studied features are potentially more robust compared to the delta coefficients, however in practice simple time differentiation appears to yield an equal or even better performance [76].

In [77, 78], Kinnunen proposed the modulation frequency from [79], [80] as a feature for SR (Figure 2.5). Modulation frequency contains potentially information about the speaking rate and other stylistic attributes. These speaker-related parameters are approximately within the range of 1–20 Hz [79, 80]. The dimensionality of the modulation frequency vector is dependent on the number of Fast Fourier Transform (FFT) points used and the number of frames spanning across the temporal axis of the signal. This means that for the best parameter combination (i.e. frequency range of 0-20 Hz with a frame width of 300ms) obtained in [77], the dimension of the feature vector was 3200. This was further reduced using the temporal discrete cosine transformation (TDCT) as illustrated in Figure 2.6. In addition to compressing the feature vector, this method has the advantage of retaining relative trajectories phases of the feature coefficient, hence preserving both speaker-specific and phonetic information.

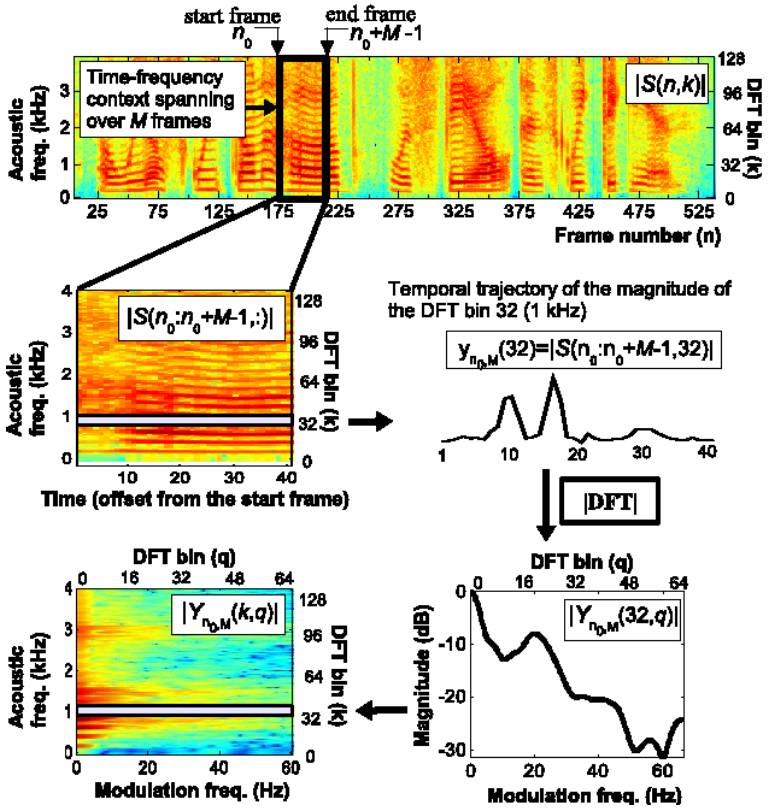

Figure 2.5 - Extracting modulation spectrogram features [79], [80].

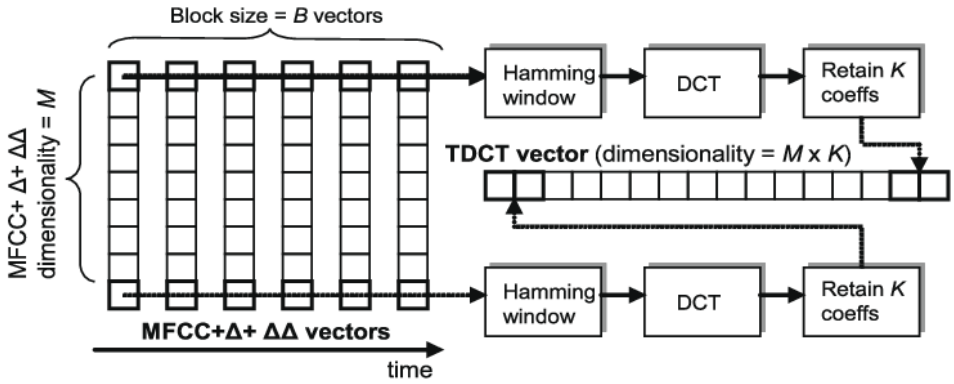

Figure 2.6 - Temporal discrete cosine transformation (TDCT) [77].

Even though some improvement was obtained compared to other features of this level, the obtained gain was rather insignificant, making it hard to use such features for practical applications without further research [77]. An alternative method for extracting spectro-temporal information is frequency modulations (FM)[81]. In this approach, the signal is first divided into subband segments using a bank of bandpass filters, with formant-like features being captured by the dominant frequency components in the subbands (e.g., frequency centroids). The method has been applied to SR [82], showed promising results when fused with conventional MFCCs.

2.3.2 Voice-prints modeling

Just as the previous section, here we are considering the state-of-the-art of a single block of SR systems (Figure 2.2), the modeling block. the modeling block is responsible for providing a measure of similarity with already existing voiceprints, acquired through training. Although the identity of the speaker is discerned through the decision block, it is the output of this crucial block that provides the decision block with the necessary information for it functionally properly. Disregarding the fact that SR deals with speech segments and the fact that these segments not to processed to obtain speaker-specific parameters, SR is a PR problem and that means that the modeling block has just to do detect a pattern from a provided set of signals from a specific speaker. This usually implies fitting this data point to a predefined model by optimizing for an error of sort. Hence, the objective of this subject it reviews what is the model used so far for SR? how where they used (i.e., model parameter, error function)? and what made them a good or bad candidate?

The last subsection is dedicated to reviewing what has been achieved over the last decade in the general field of PR. Although, there has been some improvement concerning the classical machine learning algorithm this subsection is dedicated mainly to Deep Learning (DL) and ANN, as they have been at the center of PR tasks and have reached new records in terms of both accuracy and flexibility.

Vector quantization

VQ model [73, 83–86] is one of the simplest and earliest models used in text-independent SR. The model is also known as the centroid model and was originally used for data compression [87] and was introduced in the field of SR in the 80s [83, 86]. VQ provides a competitive accuracy when applied together with background model adaption (details in next subsection about GMMs) [84]. Even though the model is very useful for lightweight practical implementations [88] and fast computational approaches [89, 90].

Considering that $X = \{x_1, x_2 \dots x_T\}$ is the test utterance feature vectors and $R = \{r_1, r_2 \dots r_K\}$ is the reference vectors, the likelihood that X has originated for the speaker with R is high when the average quantization distortion (2.5) is as small as possible.

$$D(X,R) = \frac{1}{T}\sum_{t=1}^{T} min_{1\leq k\leq K} d(x_t, r_k) \tag{2.5}$$

Where d is a distance measure such as the Euclidean distance $||x_y - r_k||$.

Theoretically, it is possible to compute (2.5) for all training vectors as R. However, computationally this is impractical, and instead, the number of vectors in R is often reduced through clustering methods such as K-means [91]. The reduced clustered set of vectors are known as codebooks (Figure 2.7).

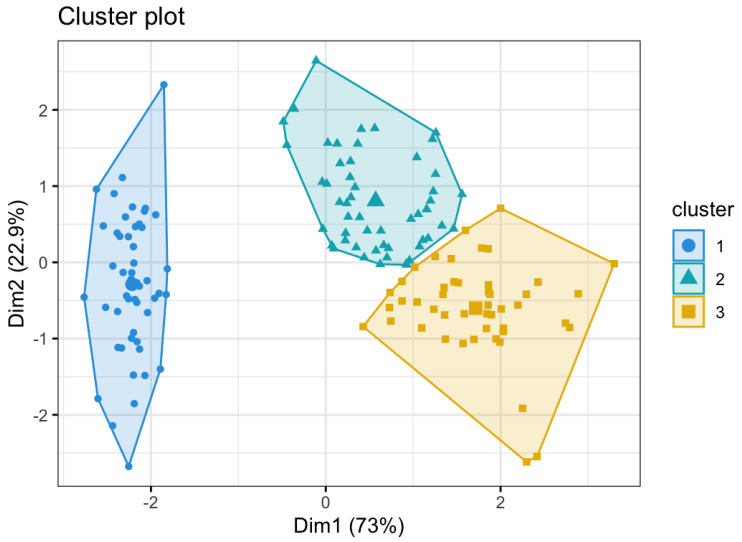

Figure 2.7 - VQ codebook construction via clustering.

Gaussian mixture model

Perhaps one of the reference models in SR is GMM [71, 92]. GMM is a stochastic model that can be viewed as an extension of the aforementioned VQ model, with the clusters being overlapping. This means that the feature vector is not assigned to the nearest cluster as in the K-mean, but instead has a nonzero probability of being originally from each cluster.

GMMs are composed of a finite number of multivariate Gaussian distributions. These distributions are characterized by the probability density function described below:

$$f(x|\lambda) = \sum_{k=1}^{K} F_k \, N(x|\mu_k, \sigma_k) \tag{2.6}$$

Some gaussian model uses a single Gaussian distribution with a full covariance matrix to model the speaker [16, 93–96]. Alternatively, some models use only the covariance matrix due to the presence of noise (inherent in the microphone/headset) in the cepstral mean vector.

These two approaches, although lagging in terms of accuracy compared to regular GMM, are computational efficient which made them very appealing for real-time applications.

Training a GMM is a basic maximum likelihood estimation task. Given a training sample $X = \{x_1, x_2 \dots x_T\}$ what need to be done is estimate the parameters $\lambda = \{F_k, \mu_k, \sigma_k\}$, by implying the average log-likelihood of X with respect to a distribution, that is defined as follow.

$$LL_{avg}(X, \lambda) = \frac{1}{T}\sum_{t=1}^{T} log \sum_{k=1}^{K} F_k N(x_t|\mu_k, \sigma_k) \quad (2.7)$$

A higher likelihood is a good indicator that the unknown feature vectors originated from the reference distribution. The expectation-maximization (EM) algorithm [97] can be used in this case to maximize the likelihood with respect to a given data. The EM algorithm is an iterative one. However, the number of iterations can significantly be reduced if the initial parameters are set using K-means [91] resulting in some case in no iterations at all [98, 99].

Just like VQ, GMM accuracy can be further improved by taking into consideration background model adaption. This simply allows the models to take into account recording conditions, by developing a speaker-independent world model or what is known as a universal background model (UBM). UBMs are trained first with using the same EM algorithm but on data that contain speech gathered from a large number of speakers [92]. The UBMs are then adapted each time a new speaker is enrolled. This adapted model is then used as the model of that speaker. Using this method, model parameters are not estimated from scratch but instead developed from prior general knowledge about the speech data. Practice has also proven that training two separate UBM is more advantageous, with each model trained in speech segments from specific gender (female/male). Any new enrolled speaker is then adapted from the UBM that is more appropriate to his gender.

Adapting a UBM $\lambda = \{F_k, \mu_k, \sigma_k\}$ to a given training sample $X = \{x_1, x_2 \dots x_T\}$ is carried out by adapting the mean vectors (k) through the maximum a posteriori (MAP) method [92]. There are several implementations of the MAP adaption method, and to choose one implementation over the other falls to the amount of data available for training [100, 101]. For instance, if the enrollment utterances are very short (a few seconds), an implementation such as Maximum likelihood linear regression (MLLR) [102] is shown to be more effective than others [100, 101, 103, 104].

Due to the frame-by-frame matching process that is required, GMM is a computationally intensive modeling algorithm. Speed-up techniques include considering only the top-C when evaluation score for the GMM–UBM framework (Reynolds et al., 2000), reducing the numbers of speaker models, vectors, or GMM component evaluations [90, 105–111].

Another shortcoming of GMM is that they do not explicitly use any phonetic information, this is due to the fact that the training set for a GMM contains simply all spectral features from different phonetic classes. This causes the matching score to be biased due to the different phonemes' distribution between training and test. This issue has been addressed by implying phonetically-motivated tree structures [112, 113], a separate GMM for each phonetic class [114–117], a separate GMM for each class of syllables [118], or a neural network classifier for distinguishing between the phonetic classes, which is known as the phonetic GMM (PGMM) [114].

Support vector machine

Support vector machine (SVM) is one of the most powerful modeling techniques that has been also adopted for SR tasks. The model has been tested with several features including acoustic [119, 120], phonological [121, 122], and semantic features [24]. In fact, SVM is considered to be one of the most robust models for speaker verification, and it has also been proven that when combined with GMM the overall accuracy is improved [119, 120]. In addition to being robust, SVM offers a good generalization performance even when dealing with unseen data.

SVM models the decision boundary between two classes by a separating hyperplane as illustrated in Figure 2.8. This makes SVM the perfect model for speaker verification given the binary nature of the task. In speaker verification, one class contains training vectors for the target speaker labeled as +1, and for the non-target speakers or "imposters" labeled as -1. Using this labeled training data, SVM optimizer tries to fit a separating hyperplane by maximizing the separation margin between the two classes.

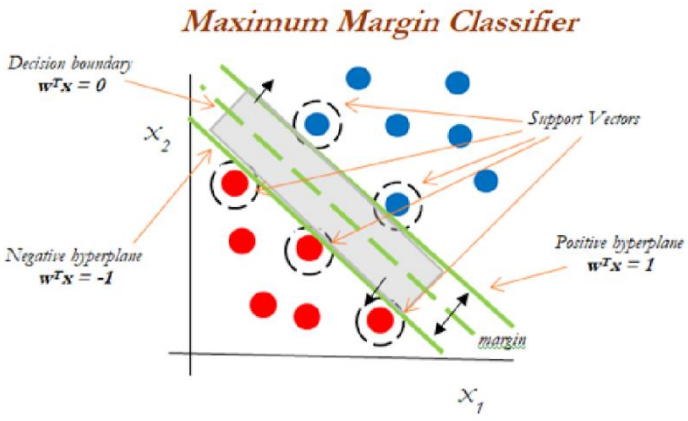

Figure 2.8 - Principles of support vector machine.

Formally, the boundaries β of the SVM are given by the following function [119],

$$\beta(x) = \sum_{i=1}^{N} \alpha_i t_i K(x, x_i) + \alpha_0 \tag{2.8}$$

Where $t_i \in \{-1, +1\}$ are the ideal output vector, $\sum_{i=1}^{N} \alpha_i t_i$ and $\alpha_i > 0$. Both the support vectors x_i, and their corresponding parameters (weights α_i and the biases α_0) are estimated through training using an optimization process. The kernel function K is expressed as $K(x, y) = \Phi(x)^T \Phi(y)$, here $\Phi(x)$ is a mapping from the original input space to a high dimensional kernel feature space. With high-dimensional spaces, it becomes easier to separate the two classes with a hyperplane. Intuitively, this means even if a nonlinear decision boundary is needed to separate the two classes in the original input space, in the high dimensional kernel feature space a linear hyperplane is sufficient to do the task.

Other technics

Other PR includes HMM, this model was very popular for speech analysis [123] and was adapted for SR [124]. ANNs have also been used for various SR applications [125–128].

Like in any other PR task, combining the score from multiple models has been widely used in SR, a technique that is known as fusion [129–131] Typically, different sets of features are extracted first from the original signal; then a model of choice is trained with each set of the extracted features; this is then followed by sub-scoring or a decision making of some kind which involves output from all models. For instance, this could be voting.

It is also possible to do the reverse, so instead of a single model, different classifiers are trained but using the same set of features [132–134]. Obtaining fusion through this mean require less memory, however, the input space is much limited in term of information.

2.3.3 Pattern recognition in the era of ANN

The era of ANN was a remarkable success in PR, it began with simplified applications in specific fields and had grown over the years to include big industrial procedures. ANNs are very appealing and powerful because unlike conventional PR approaches, they can easily model complex or multi complexes task [135, 136].

A typical model of that is often been used for the PR task is, convolutional neural network (CNN) or ConvNet. CNNs are considered to be deep feedforward ANNs, and they were originally developed to analyze visual imagery [137, 138]. Nowadays, CNNs enjoy huge success in PR. For instance, CNN structures have been applied for advanced experimental realization such facial expression [139–141] and emotion recognition [142]. Such applications require the use of novel structures of CNN (Figure 2.9).

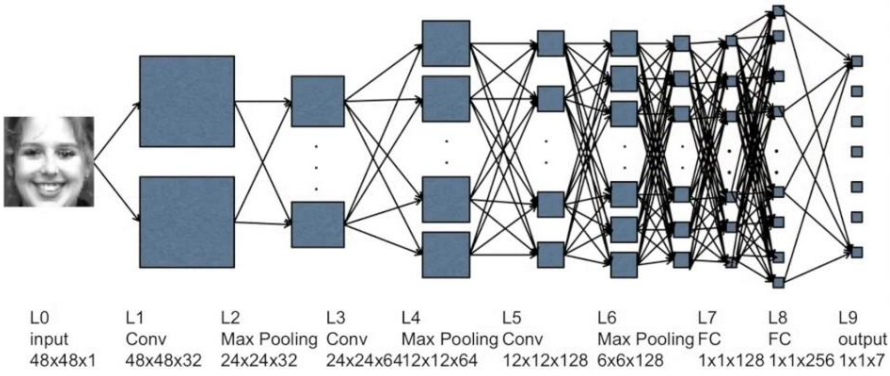

Figure 2.9 - A CNN structure designed for facial expression recognition [139].

CNN applications are not only limited to video and image recognition, recently the structure has been used for processing physiological signals [143] and as recommender systems [144]. In fact, CNN has become a very popular solution in the biomedical field and has produced better results than regular ANN. Although CNN has been shown to perform well for all sorts of data, they really excel when handling spatial data [145] and they have been demonstrated to provide the most accurate results when dealing with real-life problems [146].

In general, when dealing with data where the pattern is more present in a sequential manner CNNs are not a great choice. Nonetheless, CNN has been proven to achieve state-of-

the-art accuracy on applications such as text\document classification for sentiment analysis and related tasks.

Recurrent neural networks (RNNs) are better suited for handling PR tasks that require a form of sequence learning as demonstrated in [147]. This implies that RNNs is great for tasks that involve sequence machine learning such as predictive engines. Another area that RNNs excel at is predicting sequences and perhaps in terms of accuracy, no algorithm can even compete remotely with RNNs.

RNNs come with an inherent memory that helps them retain information for an extended period. However, this makes it also challenging to train a standard RNN to analyze signals that have patterns that extend over a long period of time. This is due to the exponential decay of the loss function gradient over time (this is also referred to as the gradient vanishing problem).

Long short-term memory (LSTM) networks are improved structures of RNN that solve the gradient vanishing problem by implying specific units called memory cells (details in section 4.3). The memory cells are able to keep information for a long time. LSTM has achieved state-of-the-art accuracy in sequence learning tasks. This includes accurate speech recognition, speech timing, and even industrial application such as analyzing the deterioration sequence of equipment as in [148], where LSTMs were trained using the status data of aircraft turbine engines. The obtained results demonstrated that the structure was able to perform excellently on the one-step, the long-term, and the remaining useful life prediction tasks.

2.4 Objectives

Although, the literature on SR is rich and spans over 70 years [10]. The approaches that have been adapted can be simply summarized in the following steps:

- Finding a speech analysis or compression method to extract features.
- Finding an appropriate modelling technique for the extracted features.
- Test the selected feature and modelling technique with an SR dataset.
- Optimize for robustness.
- Optimize for speed.

Of course, these steps are not completely useless as they are behind most VBBSs [61–64]. However, the first step on this list suffers from one major issue and is that the methods used for extracting speaker-specific features originate all from speech analysis or compression

methods. These methods can be very effective for speech recognition but the same doesn't hold for SR.

Based on the body for reviewed works and the identified issues with existing VBBSs, an approach has been put in place. This approach is based on three key ideas:

Fusion, Different features provide different type of information as indicated by the dichotomy introduced in 2.2. This implies that the dimension of the input space will increase as the number of features increases. The high dimensionality will provide the modeling technique with more space and hyper planes to separate the input points. This benefit comes at cost due to the inherited problems of high dimensional space:

- The need for higher parameters' models (high complexity)
- The need for bigger data set to avoid overfitting
- The need for longer training time

To avoid this, the features that are chosen for the fusion need to have high performance on their own, are completely uncorrelated from one another and are of same level. The only features that fits these criteria are acoustic features, as they provide the best reported accuracy [14]. Additionally, acoustic features also provide two level of complementary information that can be combined to enhance accuracy:

- Features that represent how speech is perceived, such as MFCC (type 1 features).
- Features that represent how speech is produced, such as LPC, LPCC and PLP (type 2 features).

Tokens, semantic features are often avoided since they can be easily mimicked. However, this depends entirely on the implied tokens. Tokens such as words, letters, or phonemes aren't really speaker specific. So instead of these common tokens, the idea is to develop a more appropriate token that are more speaker specific and then extrapolate to a higher level (i.e., semantics). The high-level will provide complementary information to the low-level tokens allowing for better recognition.

The low-level tokens are developed using ANN and their potential in combining extracting features and speaker modeling into a single model. This enables the model to optimize for both the (speaker-dependent) feature extractor and the speaker model. ANNs are also powerful tool for fusing different subsystems. So, to summarize the proposed tokens are developed using a speaker-specific mapping, the whole idea is to extract two parallel streams of features that are of the same nature (i.e., two short-term features, a type 1 feature paired with a type 2) and train an ANN to define a speaker modeling that will essentially find a mapping

from the acoustic space to the “speaker-specific space”. This mapping will also help recompress the original two streams into a single stream (dimensionality reduction) while retaining all of the speaker-related information.

Tokenizers, the newly developed low-level tokens contain on their own sufficient information for the recognition. However, there is a possibility also that these tokens might have different patterns of co-occurrence for different lectors. To capture these differences, sequence learning is required. To that end advanced neural network structures are implied, CNN and LSTM. Both structures offer sequence learning allowing them to outperform classical modeling algorithms. In addition, to their inability to detect sequences classical machine learning technique have their own shortcomings. For instance, GMM have an iterative approach that grows exponential as the number of data point increase and start with a very crude assumption that all data points of given subject have originated from a normal distribution. SVM on the other hand require finding a mapping to a higher feature space that will make the data points linearly separable, which becomes very difficult as the number of speakers increases.

CHAPTER III
FEATURE EXTRACTION

Based on the objectives set in the previous chapter, a decision of incorporating information from the most common acoustic features for the transfer learning. These features are perfect for defining a more adaptable speaker-specific features that contains both complementary information about how both the speech is produced and perceived.

This chapter explains how each of the selected acoustic feature is extracted.

3.1 MFCC

Ever since they were introduced by Davis and Mermelstein in 1980 [60], MFCC have been the state-of-the-art feature of choice. The main reasons for that lies in the fact that MFCC unlike regular cepstrum uses frequency bands that are equally spaced on the Mel scale, which approximates the human auditory system's response more closely than the linearly spaced frequency bands used in the normal cepstrum.

Deriving MFCCs commonly follows the setup illustrated in Figure 3.1.

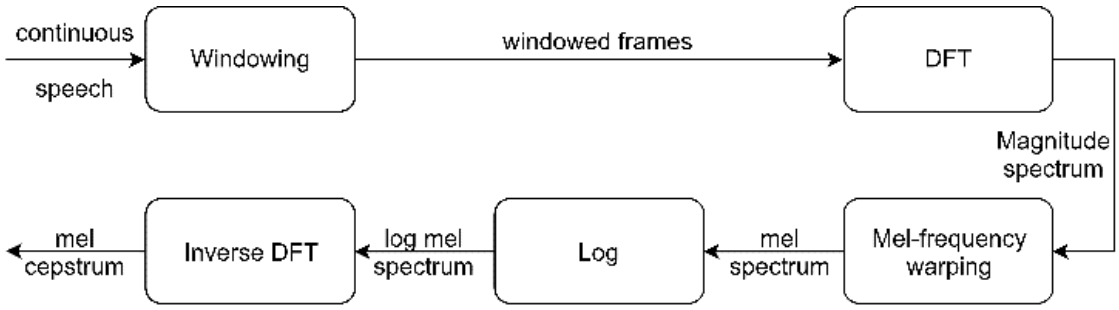

Figure 3.1 - MFCC derivation.

The first step to extracting MFCC is to frame the original signal into short time windows (generally of 20-40 ms). Windowing an audio signal is a common and recurring theme in all of the features used in both speech and SR since audio signals are constantly changing. The short time scale ensures that within that range of dozens of milliseconds the signal did not statistically change much (i.e., statistically stationary). Usually, hamming windows (3.1) are used to smooth out the signal and avoid any rippling effect in the spectrum, but what matters here is the width of the window. As a shorter window would completely miss the change and a longer window would result in the signal changing too much that it's no longer statistically stationary [16].

$$W(n) = 0.54 + 0.46\frac{2\pi n}{N-1} \tag{3.1}$$

The next step is to calculate the power spectrum of each frame using DFT. This is motivated by the cochlea (an organ presents in the human ear) which vibrates at different locations based on the frequency of the perceived sound. Each location is attached to a small hair which wobbles causing the nerves associated with it to fire, informing the brain that a certain frequency is present. The DFT is performed as follows [88].

$$S_i(k) = \sum_{n=1}^{N} s_i\,(n)w(n)e^{-\frac{j2\pi kn}{N}} \qquad 1 \leq k \leq N \tag{3.2}$$

Here w(n) is the N sample long framing hamming window, $s_i(n)$ is the input signal and K is the number of the DFT points. The outcome from (3.2) is a complex vector that contain both the power spectral and the phase. The phase information is generally not perceived by the human brain so only the power spectral is estimated from the speech frame $s_i(n)$ by the following equation.

$$P_i(k) = \frac{1}{N}|S_i(k)|^2 \tag{3.3}$$

The periodogram power spectral contains a lot of information that is not required for the general process of understanding speech. In particular the cochlea shows no difference between two closely spaced frequencies. This effect becomes even more pronounced with higher frequencies. This phenomenon has been modelled with what is known as the Mel scale [88].

$$M(\omega) = 1125\ln\left(1 + \frac{\omega}{700}\right) \tag{3.4}$$

This scale is used to exactly place a number of filterbanks (usually between 20-40) to represent exactly how the cochlea bins and sum the periodogram power spectral. The filterbanks give an idea of how much power is present within a specific frequency range with the first filter being very narrow and often discarded for containing powers that exist near 0 Hertz. The width of the filterbanks is linearly related to the frequency and the number of

implied filters. Figure 3.2 shows 20 Mel-filterbanks distributed among the speech frequency range (0 to 7 kHz).

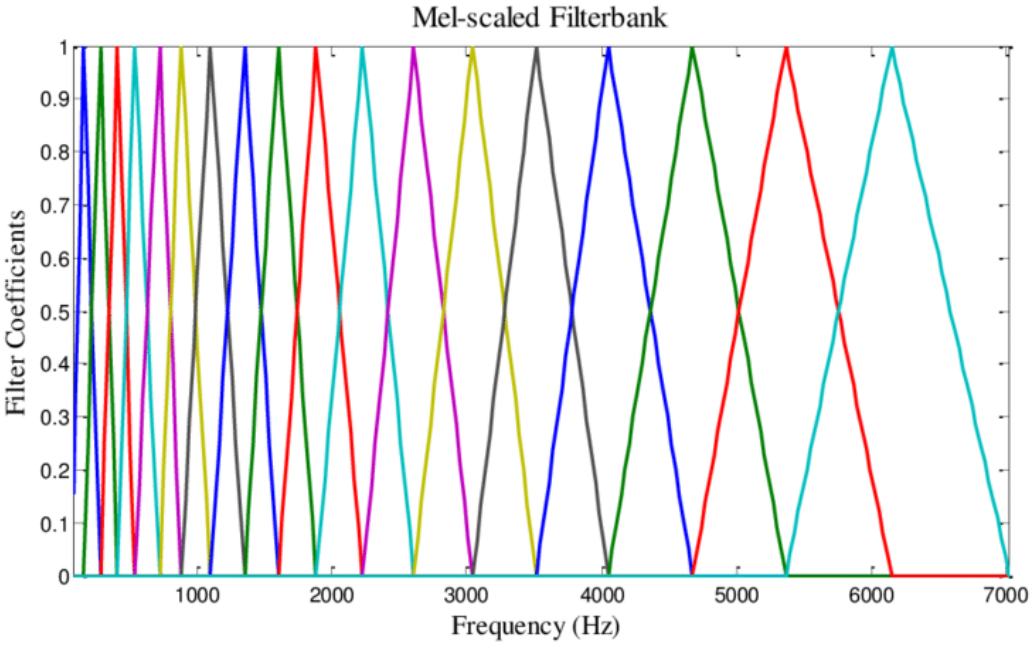

Figure 3.2 - Mel filter banks.

The equation required for obtaining these filterbanks $H_m(k)$ is the following.

$$H_m(k) = \begin{cases} 0 & k < \omega(m-1) \\ \frac{k-\omega(m-1)}{\omega(m)-\omega(m-1)} & \omega(m-1) \leq k \leq \omega(m) \\ \frac{\omega(m+1)-k}{\omega(m+1)-\omega(m)} & \omega(m) \leq k \leq \omega(m+1) \\ 0 & k > \omega(m+1) \end{cases} \tag{3.5}$$

Where m is the number of the filterbanks. Once the filterbanks are obtained, the filterbank energies can be computed by multiplying the each filterbank with the power spectrum (3.3) and stack up the obtained coefficients. Once these coefficients are computed only two steps remains (Figure 3.1), first to take the logarithm of them as the human ear doesn't perceive loudness on linear scales, and finally to compute the DCT from the log filterbanks energies to help disassociate the computed filterbank energies as they were computed using overlapping filterbanks.

3.2 LPC

LPC is a good candidate for speech analysis due to the nature of the speech generation process. The entire process can be represented by a digital filter as modelled in Figure 3.3 which makes LPC one of the most powerful type 2 features for speech analysis [149].

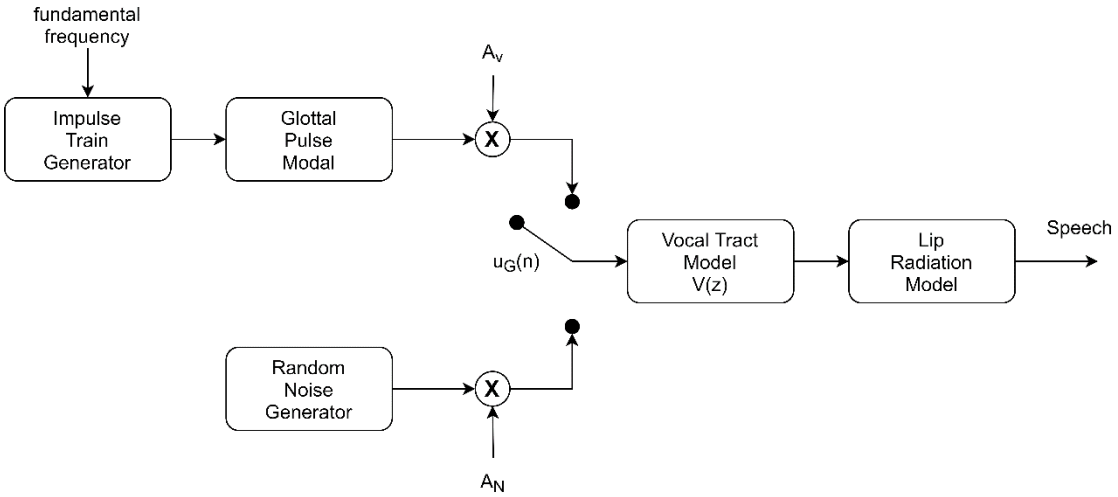

Figure 3.3 - Speech production model, $A_{\underline{v}}$ is the voiced sound gain, A_N is the unvoiced gain and $u_G(n)$ is voiced/unvoiced switching function.

The assumption for LPC is that speech is produced through a buzzer oscillating at a specific at the end of a tube, with the occasion of added hisses (sibilants sounds) or pops (plosive sounds). The assumption might seem crude, but it is actually a good approximation of how speech is produced in reality. The glottis (located between the vocal folds) can be considered as a buzzer characterized by an intensity (loudness) and a frequency (pitch). The vocal tract (combination of both the throat and mouth) is the tube that is characterized by resonances. These resonances produced formants (enhanced frequency bands) in the produced sound. Finally, the hisses and pops are given rise to through the action of the tongue, lips and throat during sibilant and plosive sounds.

LPC analyzes the speech signal by isolating the formants, and removing their effects from the produced speech, while also isolating the intensity and the frequency characterizing the glottis (buzz). The process of removing the formants effects is called inverse filtering, and what remains from this process is known as the residue.

The residue and the LP coefficients are what is transmitted for telecommunication purposes and it also what characterize the speech for SR application. Its computation requires first the original signal be framed with the help of a short time hamming window (3.1) and for

each isolated frame a number of formants are extracted. This ensures that both speech segments (voiced and unvoiced sounds) are covered and for all genders as well. Formants are extracted by computing coefficients that link the current speech sample with the previous samples with the same window. This have already been expressed in (2.2).

3.3 LPCC

Just like LPC, LPCC are a perfect for representing the speech production procedure. However, LPCC is a cepstrum and as all cepstrum it has that advantage over LPC for being more robust. This robustness is inherently in all cepstrum and can be simply obtained by applying a Cespral Mean Subtraction (CMS) on the extracted coefficients to remove any channel effects.

Extracting LPCC is fairly straightforward. First LPC need to be computed for each frame. Then, what is needed is a simple recursive formula for the coefficients from LPCs without the need to do any DFTs, the formula is shown below.

$$c(n) = \begin{cases} 0 & n < 0 \\ \ln(E) & n = 0 \\ a_n + \sum_{k=1}^{n-1} \left(\frac{k}{n}\right) c(k) a_{n-k} & 0 < n \leq p \\ \sum_{k=n-p}^{n-1} \left(\frac{k}{n}\right) c(k) a_{n-k} & n > p \end{cases} \tag{3.6}$$

Here c is the coefficients and a and E are defined in (2.2), (2.3). For a finite number of extracted LPC coefficients p, an infinite number of cepstral coefficients can be extracted. However, experiments have shown that only 12 to 20 cepstral coefficients suffice for NLP related applications.

3.4 PLP

This feature just as LPC is developed as a model to describe the psychophysics of the human speech production system in a more accurate way. PLP coefficients rely on the use of a chosen number of Bark-spaced filterbanks to cover the band ranges of the human speech (i.e., 0 to 8kHz). To specify PLP coefficients are computed by following the steps shown in the following block diagram.

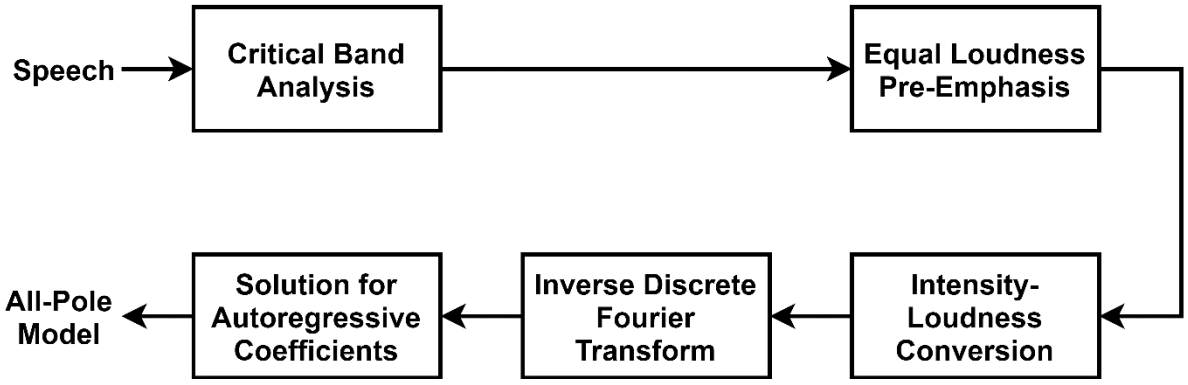

Figure 3.4 - PLP derivation.

First, just like MFCC and all short-term features the speech is segmented to short frames using for instance a hamming window (3.1). This is followed by computing the DFT to obtain the power spectrum $P(\omega)$. The spectrum is then warped using bark spaced filter banks.

$$\Omega(\omega) = 6\ln\left\{\frac{\omega}{1200\pi} + \left[\left(\frac{\omega}{1200\pi}\right)^2 + 1\right]^{0.5}\right\} \tag{3.7}$$

The resulting wrapped power spectrum is then convolved with the filterbanks. This step is a lot similar to the spectral analysis in MFCC (section 3.1), except for the shape of the filterbanks $H(\omega)$ which is defined using the following equation.

$$H(\omega) = \begin{cases} 0 & \omega < -1.3 \\ 10^{2.5(\omega+0.5)} & -1.3 \leq \omega \leq -0.5 \\ 1 & -0.5 < \omega < 0.5 \\ 10^{-(\omega-0.5)} & 0.5 \leq \omega \leq 2.5 \\ 0 & \omega > 2.5 \end{cases} \tag{3.8}$$

where ω in this case is the bark scaled frequency. The convolution of the filterbanks $H(\omega)$ and $P(\omega)$ yields samples of the power spectrum Θ that can be expressed as.

$$\Theta(\omega) = \sum_{\omega_i=-1.3}^{2.5} H(\omega_i)P(\omega - \omega_i) \tag{3.9}$$

The obtained power spectrum $\Theta(\omega)$ is much compressed compared to $P(\omega)$. The exact value of the sampling interval is selected based on the required number of coefficients. For

instance, if 18 coefficients are to be extracted for the range of 0 to 21.3 Bark (0-8kHz) the analysis would be carried out in steps of 1.25-Bark. $\Theta(\omega)$is then pre-emphasized using a simulated equal-loudness function $L(\omega)$. $L(\omega)$ is an approximation of the nonequal sensitivity present in the human speech for different frequencies [150].

Finally, the first value (0 Bark) and the last one (Nyquist frequency) of the sampled power spectrums are made equal to the adjacent values. Thus, making the spectrum starts and ends with two equal-values. This is followed by applying the cubic root, which is inspired by the nonlinear relation between the intensity of the sound and its original loudness. Together with the loudness pre-emphasis step, this operation reduces the variation within the critical-band spectrum.

The final step in PLP analysis is approximating the processed spectrum using the autocorrelation method of all-pole spectral model. For that, the inverse DFT (IDFT) of $\Theta(\omega)$ is computed yielding in the autocorrelation function that is dual to it. Typically, 34-points IDFT is used and not the inverse FFT since the number of points is very small. The first M autocorrelation values are put into the Yule-Walker equations to solve for the autoregressive coefficients of the M^{th} order all-pole model.

3.5 Delta & Delta-Deltas

Also known as differential and acceleration coefficients provide the dynamic aspect of speech. Features such as MFCC feature vector describes only the power spectral envelope for a single frame, but speech is a continuous process, so it is important to consider the change in the trajectory of said feature. Estimating the differential (Delta) coefficients is fairly straightforward and is computed using the following formula.

$$d_t = \frac{\sum_{n=1}^{N} n(c_{t+n} - c_{t-n})}{2\sum_{n=1}^{N} n^2} \tag{3.10}$$

Where $d_t$is a delta coefficient, from frame t computed in terms of the static coefficients c_{t-n} to c_{t+n}. A typical value for N is 2. For the acceleration (Delta-Delta) coefficients the same formula is used but in this iteration the deltas are used as the static coefficients c instead of the extracted features.

CHAPTER IV
ARTIFICIAL NEURAL NETWORK

As mentioned in the objectives the features that were defined in the previous chapter are coupled with the use of two of the most powerful ANN structures, CNNs and LSTMs. Both structures perform excellently for sequential learning.

This chapter delve into the details of how each of the selected modeling algorithm operate, this includes both the intuition of a given model as well as the required math for that.

4.1 Basic structure

ANN is a network inspired by biological neural networks which are used to estimate or approximate functions that can depend on a large number of inputs which are generally unknown. ANNs in general are composed of an input layer which is connected to the inputs and an output layer connected to the outputs, between the two there is a number of hidden layers that can vary in structure, size and shape depending on the application.

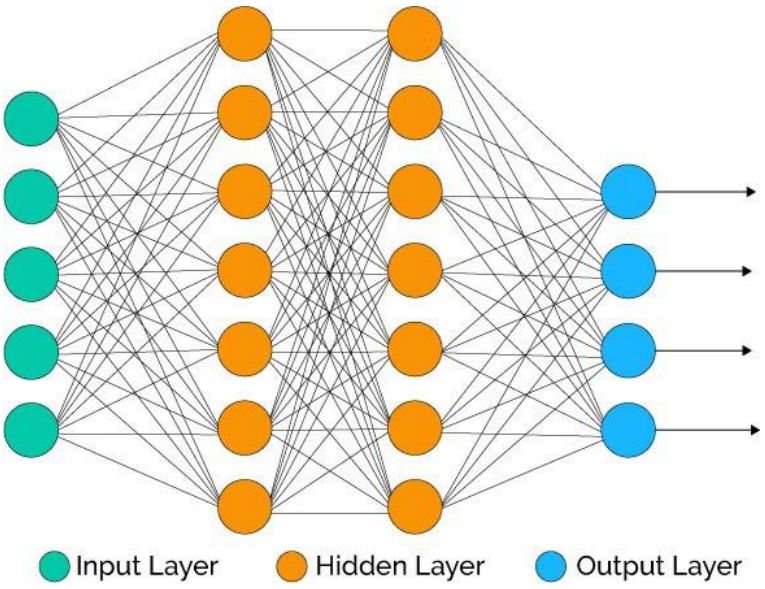

Figure 4.1 - Neural network structure.

Each layer is composed of one or more perceptrons (neurons). The inputs of any given perceptron could be the inputs or the outputs of perceptrons from the previous layer, current layer or even the perceptron itself. Given an input vector $X = \{x_1, x_2, \ldots, x_m\}$ the output of a perceptron is given by the following equation.

$$Y = f(w_0 + \sum_1^m w_i \times x_i) \tag{4.1}$$

Where x_i is i[th] value of the input, w_i is i[th] value of the weight matrix, w_0 is the bias, Y is the output and f is the activation function. This is better visualized by the diagram below.

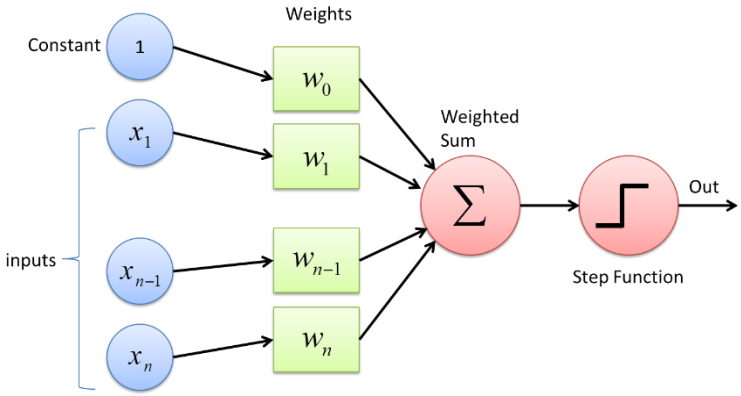

Figure 4.2 - Perceptron diagram.

The objective here is to make all perceptrons learn a mapping that associates a given input with required output. This is achieved by finding the corresponding weights for each perceptron that would cause the error between the current output and the required output to be as low as possible (i.e., ideally the global minimum).

4.2 Back propagation

To reduce the error E between the actual output and the required output, the weight change of the weight connecting a node at the i[th] layer with a node in the j[th] layer w_{ij} need to be considered.

$$\Delta w_{ij} = -\alpha \frac{\partial E}{\partial w_{ij}} \tag{4.2}$$

Here α is a free parameter (the "learning rate") that should be set prior to the training; a scale out the step size according to the problem at hand. In regular structures, the adjustments of α will be adhoc; favoring a more intuitive explanation that can be obtained by the following expansion of the partial derivative by the chain rule.

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial w_{ij}} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial \sum_j}\frac{\partial \sum_j}{\partial w_{ij}} \tag{4.3}$$

$$\frac{\partial \sum_j}{\partial w_{ij}} = \frac{\partial}{\partial w_{ij}}\left(\sum_{k=1}^{n} w_{kj}x_k\right)\frac{\partial}{\partial w_{ij}}\, w_{ij}x_i \tag{4.4}$$

Here $\sum_j$ represents weighted input of neuron j. Now, the derivative of the output of neuron j with respect to its weight inputs is simply the partial derivative of the activation function f.

$$\frac{\partial y_j}{\partial \sum_j} = \frac{\partial f(\sum_j)}{\partial \sum_j} \tag{4.5}$$

In case f is a logistic activation function this becomes.

$$\frac{\partial y_j}{\partial \sum_j} = \frac{\partial}{\partial \sum_j} f(\sum_j) = f(\sum_j)(1 - f(\sum_j)) = y_j(1 - y_j) \tag{4.6}$$

The first factor in (4.3) is straightforward to evaluate and it depends on the implied loss function, which for a square error function (SSE) can be written as.

$$\frac{\partial E}{\partial y_j} = \frac{\partial}{\partial y_j} \frac{1}{2}(t - y)^2 = y - t \tag{4.7}$$

Where t is the target vector. This only applies for the last layer of network (output layer), in case of j being an arbitrary inner layer the derivative is less straightforward and would require considering the weighted inputs $\sum$ of all the set of neurons $L = \{u, v, \dots, w\}$ getting input from j.

$$\frac{\partial E}{\partial y_j} = \frac{\partial(\sum_u, \sum_v, \dots, \sum_w)}{\partial y_j} \tag{4.8}$$

$$\frac{\partial E}{\partial y_j} = \sum_{l \in L} \left(\frac{\partial E}{\partial \sum_l} \frac{\partial \sum_l}{\partial y_j} \right) = \sum_{l \in L} \left(\frac{\partial E}{\partial y_l} \frac{\partial y_l}{\partial \sum_l} \frac{\partial \sum_l}{\partial y_j} \right) = \sum_{l \in L} \left(\frac{\partial E}{\partial y_l} \frac{\partial y_l}{\partial \sum_l} w_{jl} \right) \tag{4.9}$$

Now by substituting each of the factors in (4.3) we obtain.

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial \sum_j} \frac{\partial \sum_j}{\partial w_{ij}} = \frac{\partial E}{\partial y_j} \frac{\partial y_j}{\partial \sum_j} y_i \tag{4.10}$$

$$\frac{\partial E}{\partial w_{ij}} = y_i \delta_j \tag{4.11}$$

Where δ is defined as:

$$\delta_j = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial \sum_j} = \begin{cases} \frac{\partial L(y_i,t)}{\partial y_j}\frac{\partial f(\sum_j)}{\partial \sum_j} & \text{if j is an output neuron,} \\ \left(\sum_{l\in L} w_{jl}\delta_l\right)\frac{\partial f(\sum_j)}{\partial \sum_j} & \text{if j is an inner neuron.} \end{cases} \quad (4.12)$$

The chain rule backpropagation is the most basic form of deriving weight changes, other advanced optimization techniques have been developed over the years such as momentum [151], Adagrad [152], RMSProp [153], and Adam [154]. These optimizers and others are covered in [155].

4.3 CNN

CNN is very powerful tool for dealing with image related applications (e.g., objects detections, face recognition, image classification...etc). However, in truth this structure is perfect for dealing with any input with 2 or more dimensions, this for instance could be a spectrum of speech signal or concatenated stack of acoustic features.

CNN classification takes an array of values (i.e., could pixels, powers or voxels) and depending on the dimension of that array a number of procedures is required. Technically, to train and test any CNN models each input array will pass through a series of convolution layers with filters (kernels), pooling, fully connected layers (FC) and final applying a SoftMax function to provide a probability (a value between 0 and 1) for all possible classes (states). Figure 4.3 demonstrates a complete flow of CNN for classifying images (i.e. a 3D array of pixels).

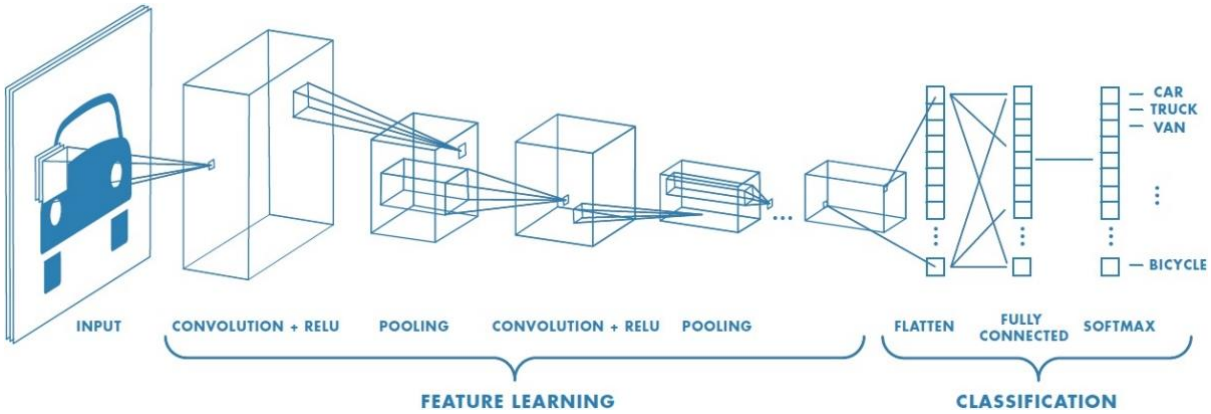

Figure 4.3 - Network with different convolutional layers.

Convolution Layer, this layer is always connected to the input and is responsible for extracting task related features. This is done through convolution (Figure 4.4), a mathematical operation that takes as input the input array and a smaller array of values that will represent the trainable weights, and preserves the relationship between values of the input. The array of weights is often referred to as a filter or a kernel.

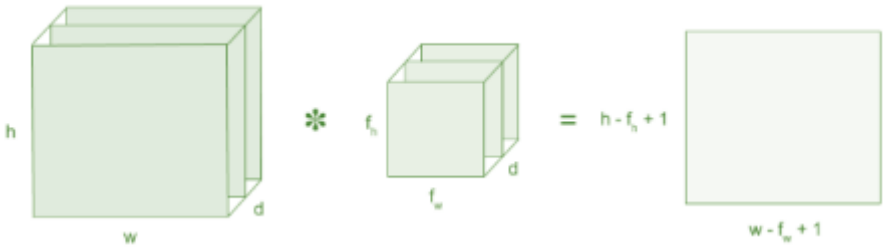

Figure 4.4 - Convolution between the input array (h x w x d) and a filter (f_h x f_w x d).

The output of this layer is basically a processed version of it. For instance, in image related applications the filters are performing processes such as edge detection, blur or sharpen. This is can be better explained by considering the example below, where different filters (kernels) are learned and we observe the output of each one of them one applied to a rodent image (the input).

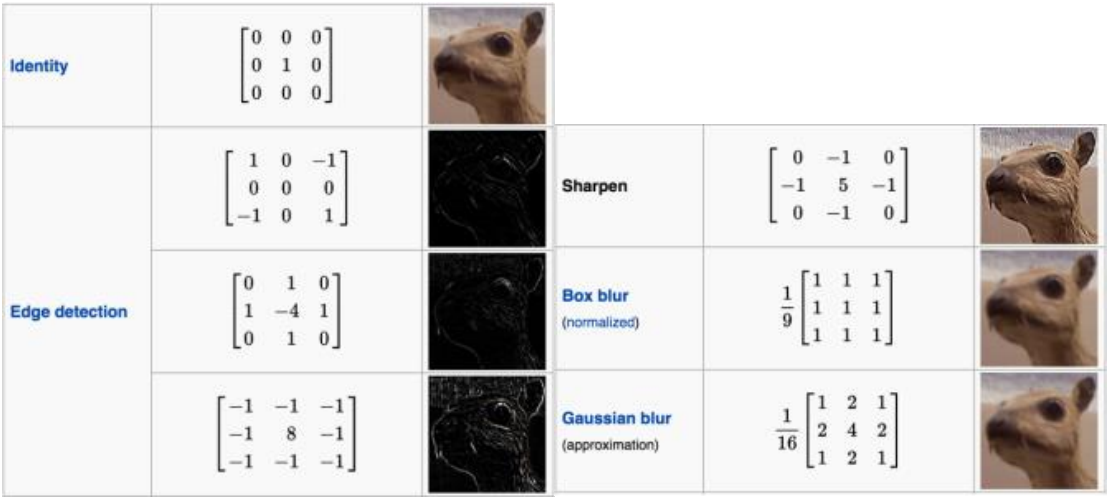

Figure 4.5 - Filter effect demonstration.

Ideally stride of convolution is usually a single unit. However, some applications could require a different stride in case where the input is very large in terms of dimension and requires fast method of compression (i.e., reduced size) or in the presence of redundant values in the input.

ReLU, stands for Rectified Linear Unit. This unit introduces non-linearity to the network since most real-world applications are so. The output of this unit is $f(x) = \max(0, x)$ and that's based on the fact most of the inputs that are used with such a network are non-negative linear values.

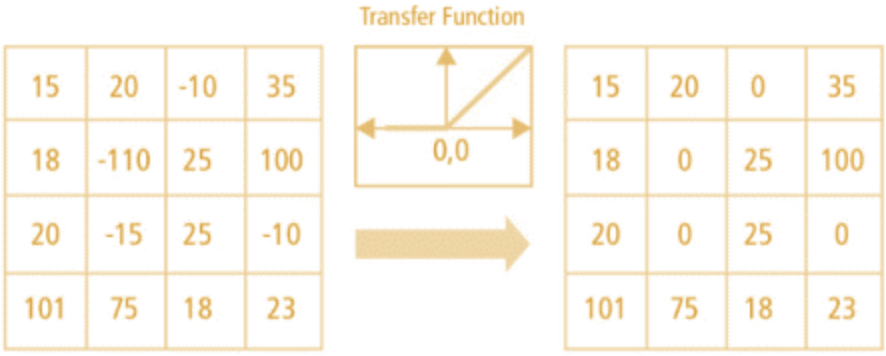

Figure 4.6 - ReLU demonstration.

Although, there are other non-linear activation functions (e.g., tanh or sigmoid) that can be used instead of ReLU. In practice ReLU offers the best performance compared to the rest.

Pooling, this layer introduces a mean of reducing the number of parameters when the output volumes are still too large to be flattened out. This layer is also referred to as the subsampling or down sampling layer, as it reduces the dimensionality of each output by retaining important information. This can be achieved through different means:

- Max pooling takes the largest element from the rectified feature map.
- Average pooling takes the average of the rectified feature map.
- Sum pooling which is similar to the average pooling in a sense and takes the sum of the rectified map.

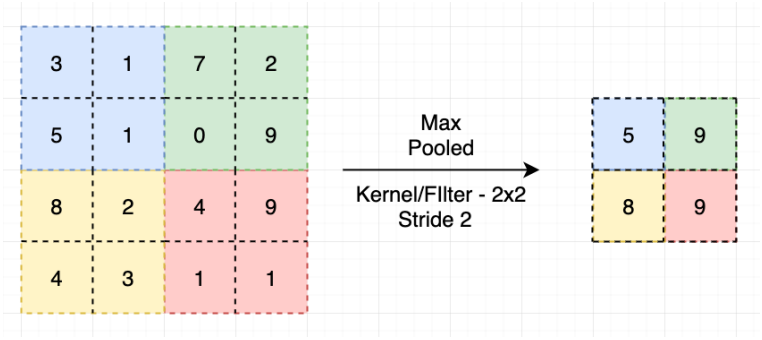

Figure 4.7 - Max pooling demonstration.

The sequence of (Conv x ReLU x Pooling) can be repeated as much as the application and the data require so. In fact, for some verification tasks this can be repeated till a 1x1 output is produced out it. This kind of structures are called pureCNN [156]. For most application however, the output of the pooling will be flattened to a vector $X = \{x_1, x_2, \dots, x_t\}$ and is then fed to a FC. Finally, a SoftMax layer is required to provide the classification output.

4.4 RNN & LSTM

RNNs are more general ANNs that have inherent internal memories. RNNs are recurrent in nature since the same function is performed for every input of data with the output of the current input depending on the past output computation. After the output is computed, it gets feedback back into the RNN. This means that the decision making is done by considering the current input and the output obtained from the previous input.

Unlike a regular ANN, RNNs are able to analyze sequences of inputs thanks to their memory (internal state). This makes them perfect for applications such as unsegmented/connected handwriting/speech recognition. In most neural networks, the inputs are independent of one another. However, in RNN all inputs are related as can observed below.

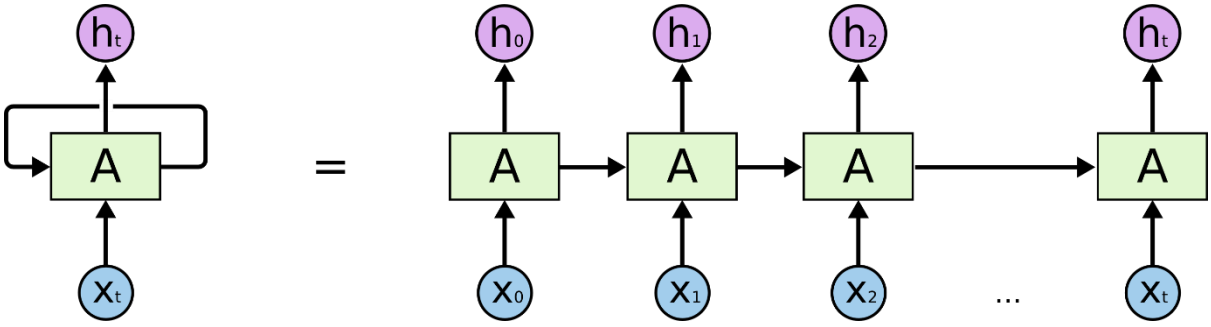

Figure 4.8 - An unrolled recurrent layer.

First, the network takes the first input X_0 from the sequence, it then outputs a value h_0 which is fed along with the next value of the sequence input X_1 for the next step. So, the input for the second iteration of the layer is h_0 and X_1. Similarly, h_1 and X_2 are the inputs for the third iteration and so on. This way, the network is able to keep the context remembered while training.

The current iteration can be expressed as.

$$h_t = f(h_{t-1}X_t) \tag{4.13}$$

By expansion this formula becomes.

$$h_t = \tanh(w_{hh}h_{t-1} + w_{xh}X_t) \tag{4.14}$$

Here $tanh$ is the activation function, that implements a non-linearity that squashes the activations to the desired range of {-1,1}. The last state is the output and it's equal to.

$$y_t = w_{hy}h_t \tag{4.15}$$

Unlike CNN layers here the output y_t is already flattened and can be fed directly to FC. The training is similar to 4.2 once the RNN layer is unrolled.

Although RNNs are very powerful tool they are not perfects as they suffer from the following issues:

- Gradient vanishing and exploding problems.
- Training RNNs is a very difficult task.
- Unable to process very long sequences.

To overcome these shortcomings, LSTM networks were developed as modified variation of RNNs, that can remember past data even for very long sequences. Here, the RNN vanishing gradient problem is solved making LSTM well-suited for classifying, processing and predicting sequences with unknown duration. Similarly, to RNN training a network with an LSTM layer involves enrolling it and applying back-propagation. The difference here is that the recurrent layer contains three gates that helps remember important inputs from the sequence and forgets those that are not, this is demonstrated in Figure 4.9.

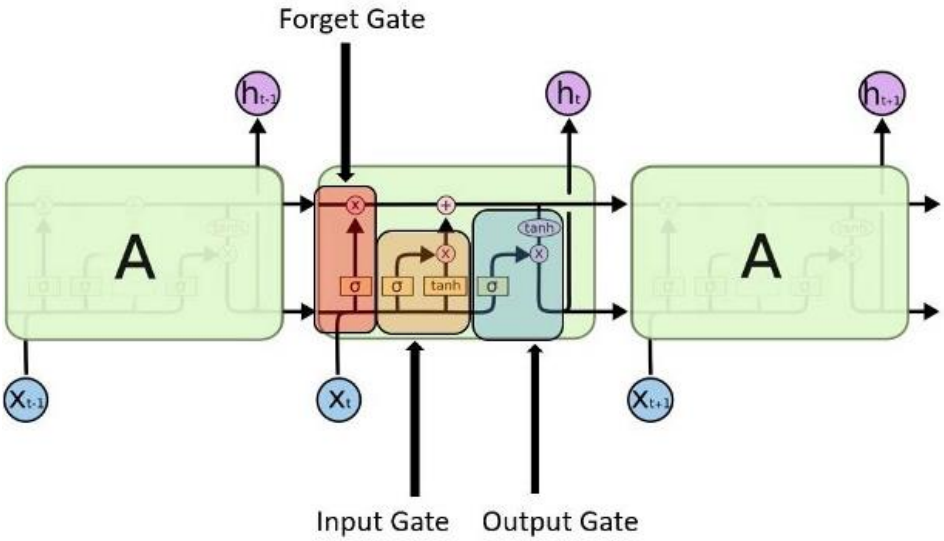

Figure 4.9 - LSTM layer diagram.

Input gate, controls the extent to which a new input is used. This is achieved by the sigmoid function which takes a value between 0 and 1. The tanh function assigns an importance weight (ranging from -1 to 1) to the amount of input that was passed.

Forget gate, controls the extent to which a value is remembered. This is achieved by the sigmoid function that considers the previous state h_{t-1} and the content input X_t and provides an output between 0 (for omitting the value) and 1 (for keeping the value).

Output gate, controls the extent to which the value affect the output activation of the unit. This is achieved by the sigmoid function which takes a value between 0 and 1. The tanh function assigns an importance weight (ranging from -1 to 1) to the amount of value that was passed.

CHAPTER V
METHODOLOGY

This chapter presents the methodological approach set forth throughout this doctoral work. First, the chapter focuses on the experimental tools that were developed as mean for testing the proposed approach with ease. The first phase of this project contributed to i) putting together a concise but sufficient corpus incorporating non-native English speakers ii) implementing a Graphical User Interface (GUI) use SR toolbox that would serve for quick prototyping and testing of ANN approaches for SR.

The second part of the chapter dives into the approach which was proposed based the body of literature review seen so far (section 2.3). The focus here is to tackle each of the key points defined in section 2.4 and explain in detail the methods that were implied to achieve a given point. First, the method for fusion is explained along with the pre-processing pipeline for cleaning up the signal, then how the fused features are mapped into speaker-specific features (tokens) using transfer learning. The chapter end with explaining the proposed CNN and LSTM structures, that are used to bring both low-level (acoustic) and high level (semantic) features for a better recognition.

5.1 Corpus recording

5.1.1 Rationale

In any speech related application, it's crucial to have data set of recorded speech segments often called the corpus. The corpus doesn't only provide the original signal from which streams of features are extracted, it also forces or favors the use of certain SR applications over others based on the recording setup, contents of each segment and the number of the implied speakers. This is better demonstrated in Figure 2.1 which shows the several SR applications based and how they differ from one another. Section 2.1 explains in detail how the layout of the corpus effects the task at hand, but what is worth mention is that out of all tasks the most common application is text-independent speaker identification. Based on that the proposed approach will be implemented, trained and evaluated for this specific task.

Text-independent speaker identification, requires the following:

- A closed set of speakers (i.e., more than one print).
- No sentence is said repeated by either the same or another speaker.

There several corpuses that can be used and fits these criteria such as the NIST Speaker Recognition Evaluation Benchmark [157], the VoxCeleb [158], and the ChiME-5 speech corpus [159]. However, all of these datasets are cleaned, have a large set of speakers and are expensive. In addition, the speakers that are recorded are native English speakers which is not

ideals given the geographical location (Algeria). Considering all this, a recording of a local corpus was carried out. The recorded corpus was used throughout the project and had a concise number of speech segments that could be of benefit to any team with limited computational resource but wants to test new methods for SR.

5.1.2 Subjects screening

To carry out the recordings, a methodological approach was set forth to screen out participants. In particular, the two following conditions need to be met:

- ❖ Sufficient level of English proficiency demonstrated by either the TOEFL or IELTS score
- ❖ Balanced population of male to female ratio to avoid having the model being biased toward one sex over the other.

With this in mind, sixteen participants were recruited from the Institute of Electrical and Electronic Engineering (IGEE) (except for one participant who was a relative) to carry out the recordings. The male to female ratio was 1 (i.e., 8 males and 8 females). The sixteen participants were all student at the time of the recordings and had an age ranging between 24 and 26 (age: 25.37 ± 0.88). As all participants were non-native English speakers, they had to show a high level of proficiency in English especially in speaking form. The criteria for each test were as follow:

- ❖ IELTS: total score $\geq$ 6.0 (IELTS: 6.78 $\pm$ 0.39) with a speaking section score $\geq$ 7.0 (IELTS_s: 7.71 $\pm$ 0.9)
- ❖ TOEFL: total score $\geq$ 80 (TOEFL: 88.78 $\pm$ 5.59) with a speaking section score $\geq$ 22 (TOEFL_s: 26.67 $\pm$ 2.92)

Although most of the participant had undergone one of the aforementioned English tests during their higher education years, some didn't have the means to carry out these tests. These participants had to undergo an online practice version of the tests. The test was assigned randomly, unless the participant demonstrated a high level of familiarity with it.

5.1.3 Recording procedures

All recordings were carried out in the same room located within the institute, (6.0m(L) x 3.5m (W) x 4m(H) as shown in Figure 5.1). The room in Figure 5.1 was regular and had no sort of sound isolation. This was to ensure the robustness of the approach and its ability to still recognize the speaker regardless of whether external noises were present. However, all

recording were controlled for health factor, so prior to the day of the recording the subject were contacted and ask whether or not they had any health issue that might affect the way in which they speak (flue, fever, coughing, sore throat…etc).

On the day of the recording subject were guided to the recording room (Figure 5.1) where they were briefed on the task at hand and put at a distance of 0.75m from the northern wall. In front of they had a monitor displaying the sentence to be read. The 40 to 43 first sentences were randomly selected out of a set of sentences scraped from the Wall Street Journal (WSJ) using WSJ-Scraper [160]. This was followed by 11 sentences dictating connected digits. The outgoing speech was recorded using the Honeywell CN51 Personal Digital Assistant placed between the subject and the monitor.

Since the aim is to tackle text-independent speaker identification, the sentences that were selected differ from one student to the other to ensure a minimum collaboration of the speaker.

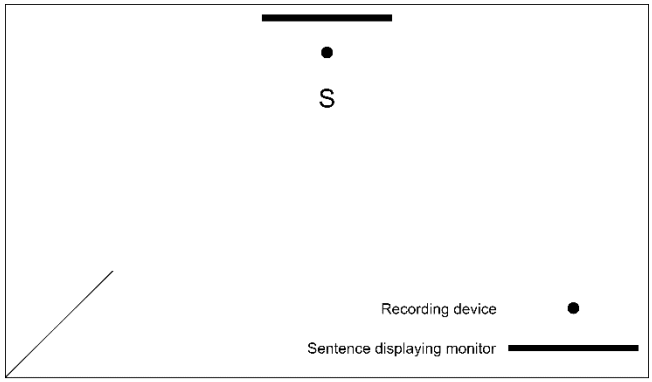

Figure 5.1- Recording room layout, S is the subject.

Figure 5.2 - Honeywell CN51 PDA.

To demonstrate that the approach is effective for native English speaker there was no effect of accent, the LibriSpeech ASR corpus [161] was implied. LibriSpeech ASR corpus contains around 921 reader which was ideal for confirming that our SR system is also functional for scenarios with a much wider range of voiceprints. However, this corpus was only used to confirm recognition results and was not used to investigate parameters tunning, synapsis, and speech duration effects due to hardware limitations.

5.2 Speaker recognition toolbox

5.2.1 Rationale

The language of choice for testing the points defined in section 2.4 was MATLAB. Another alternative would've been Python since it is very popular with researcher carrying out DL related work. Python however is lacking in terms of tools for processing audio signal and for extracting the desired acoustic features (Chapter 3). On the other hand MATLAB offer several open source tools for preprocessing audio signals [162] and for extracting acoustic features such as MFCC, LPCC and PLP [163, 164]. In addition, in recent versions (2017a and above), MathWorks developed a dedicated toolbox for DL, the NN toolbox [165].

The NN toolbox provides framework that is easy to use for implementing ANN with algorithms, pretrained models, and apps. This includes shallow ANN and more advanced structures such as CNN, RNN, and LSTM. It spears the user the trouble of going through the math behind back propagation (section 4.2) and other optimization algorithms such as Adam [154]. Additionally, it provides the user with speed up options such as training the network on a single- or multiple-GPU machines, or even scale up to a cluster of machines or to the clouds without the need to deal with issues that my result from synchronicity and compatibility. However, in its early releases (2017a) the toolbox was not very intuitive to use as designing a simple ANN involved several lines of code [166]. To avoid the process of writing several lines of codes each time a new configuration is needed, an SR toolbox was developed [167].

The SR toolbox make the process of redesigning an existing ANN or even creating a new one, reasonably easy even for the unexperienced programmers by providing a GUI (Figure 5.3). This GUI allow users to design, train, and evaluated ANN with the aim of performing SR.

5.2.2 Description

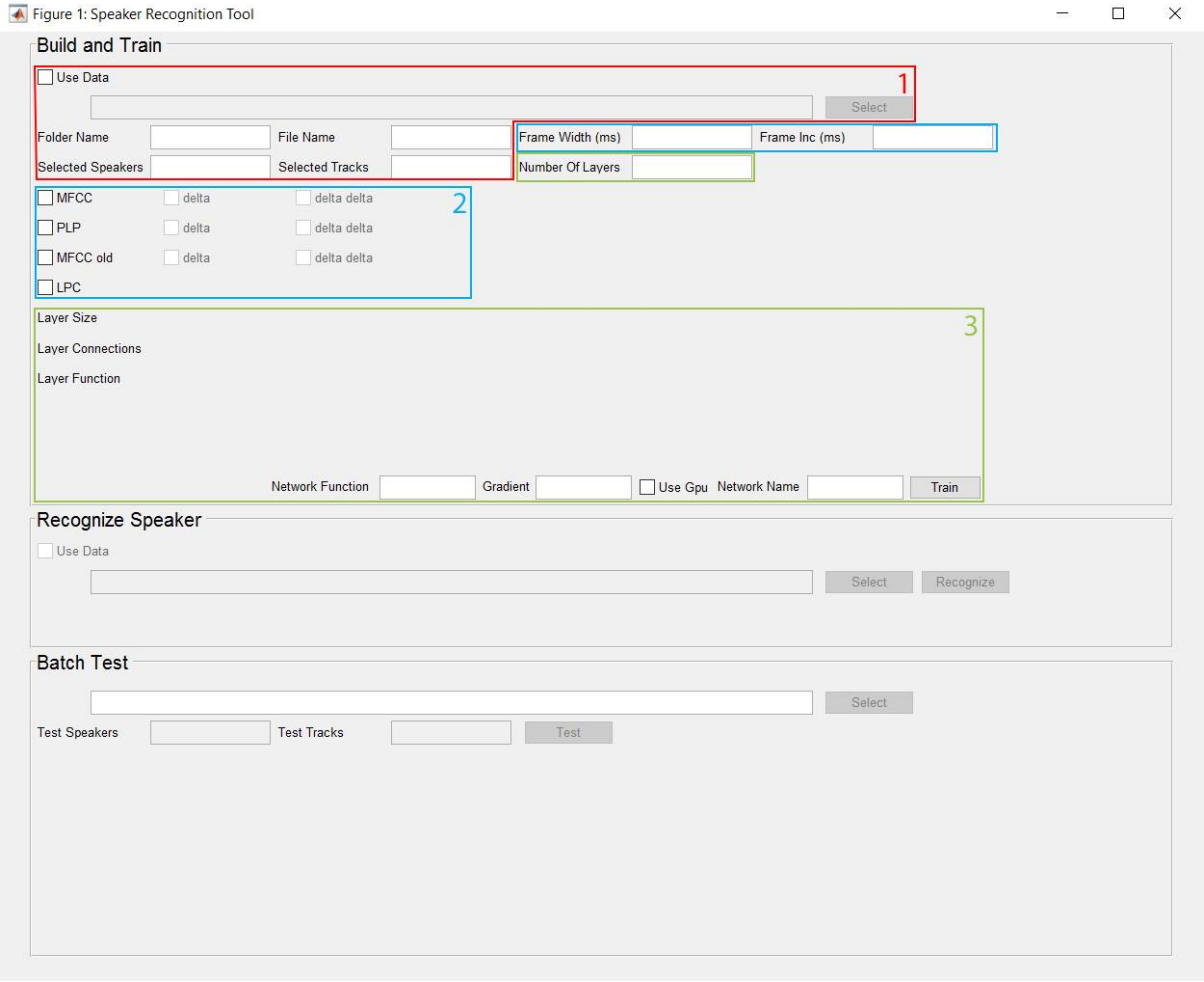

Figure 5.3 - SR toolbox GUI.

As seen in Figure 5.3 the GUI is split into 3 main sections:

Build and Train, this section is most important as it contains the elements for selecting the corpus (1), extracting the features (2) and implementing the ANN (3). The toolbox supports the use of both immediate recording and pre-recorded corpuses. This is decided by the "Use Data" check box, the "Folder Name", "File Name", "Selected Speakers", and "Selected Tracks" fields that follows are used to define the speaker naming convention, tracks naming convention, indices for training speakers and tracks respectively.

The toolbox allows for preprocessing the selected speech segments using a pipeline that will be covered in full length in the upcoming section 5.3. However, what worth mentioning is that using the GUI, specific features can be extracted with a parametrizable frame width and increments. The last component of this block contains the element for designing and training ANNs. The components are self-explanatory however what is worth mentioning is the

"Number of Layers" field. This field is an active component that create a corresponding number of fields for each parameter of a layer (size, connection, and activation function), this number should correspond to all layers include input layers and the output layer not just the hidden ones.

Once all parameters are defined, the model can be trained using one of the available optimizers ("Network Function" field) to reach a specific gradient ("Gradient" field) by pressing the train button. The training process can be accelerated by using a GPU if available ("Use GPU" box).

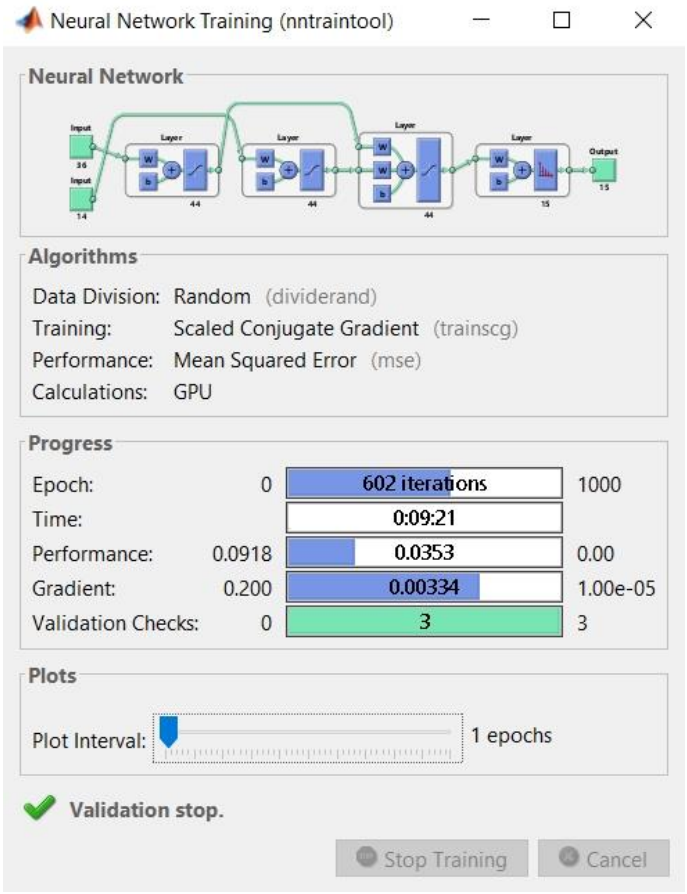

Figure 5.4 - Training progress display.

Recognize Speaker, this section contains only on functionality, the ability to classify a single speech segment. The speech segment could be either recorded immediately using the embedded microphone or prerecorded audio file. The output of this step a simple console message discerning the speaker identity with the level of conditionality (Figure 5.5).

```
Person 1 got 99.12
  Columns 1 through 10

   99.1200   10.2500   32.4500    5.1000   10.2000   36.8000   14.2500   23.0500   15.1000   12.0000

  Columns 11 through 16

   38.6000    0.7500   33.3300   41.2000   13.5000    2.2500
```

Figure 5.5 - Single speaker recognition display.

Batch Test, this section basically performs the same operation as the section before but on a speech segments batch. This section contains field for selecting the path to the corpus along with indices for the testing speakers and tracks. The obtained output is saved into a 2D Matrix (speaker x number_all_tracks) and is displayed to the user as demonstrated in Figure 5.6.

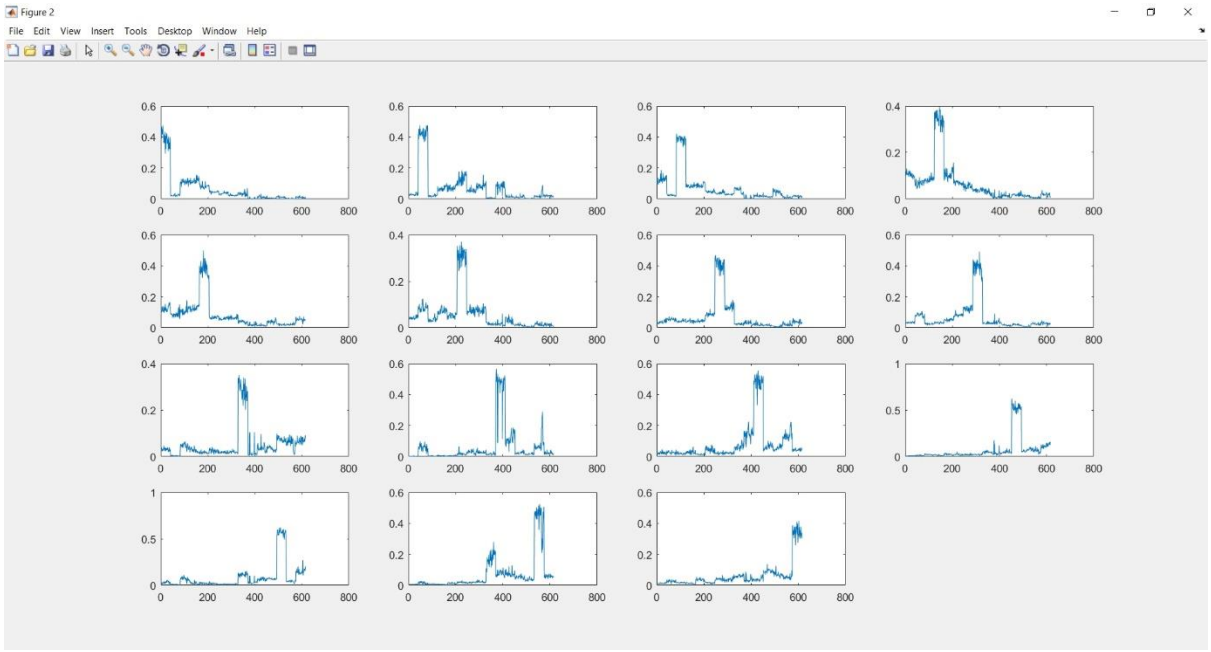

Figure 5.6 - Batch recognition display.

The toolbox is not perfect and requires a lot of adjustment before being able to cover a wide range of scenarios. However, there is no doubt that it has a lot of potential, therefore it is put on public domain [167] so that other researchers could make use and improve upon it.

5.3 Feature fusion & Tokens

5.3.1 Feature extraction pipeline

The core contribution of this work lies into the point that are covered in this section. In section 2.4 the need to develop a new and better speaker-specific tokens is emphasized, to achieve this three steps are required i) preprocessing the recorded signal ii) extracting the selected acoustic feature in an optimal way iii) train the mapping ANN using parallel stream of two features (Fusion). All three steps were implemented in the discussed SR toolbox [167].

The first step is to make sure the signal is preprocessed enough and contains only the desired information to be used for extracting the acoustic features. The speech cleaning pipeline contains the steps shown in Figure 5.7.

Figure 5.7 - Speech preprocessing pipeline.

The pipeline is straight forward and can be achieved with simple mathematical concepts. The DC component of any speech signal is its average, normally this value should be equal to zero. A different value then zero means that the signal is shifted along the amplitude axis and the average need to be subtracted from all value in the signal to solve that. The normalization steps ensure that the extracted features are not affected by any parameters such as distance and loudness of the speaker. A normalized signal would have a max value of 1 which is what is required given that the amplitude of the speech signal is a feature that is more attributed to the loudness of the speaker and his distance from the speaker. This level of information is misleading and is not speaker specific. To normalize the following formula can be used:

$$\hat{s}(n) = \frac{s(n)}{max(s(n))} \tag{5.1}$$

The last step on the cleaning pipeline is to remove silent parts although some paper recommends removing the unvoiced phonemes from the signal as well [162] here only the

silence segments are removed. Silence segments are decided based on the number of $y = 0$ intersections. Based on the used corpuses (i.e., recorded corpus and LibriSpeech) a value of x intersection is selected as the threshold for defining segments with silence, these segments were discarded before the feature extraction.

Overall, 3 set of features (Chapter 3) were extracted to test the effectiveness of the suggested approach. The extractions were carried out using the SR toolbox that made use of existing speech analysis toolbox on MATLAB, mainly Auditory Toolbox [164] and VOICEBOX [163]. The first feature was MFCC as it's the most common features for SR applications (section 3.1). MFCC extraction was carried out using VOICEBOX [163] employing the **melcepst** function, this implementation followed the same step as in Figure 3.1. Just like [88] 13 coefficients were extracted using for this work. The 0th coefficient is discarded since it does not contain any significant information. LPCC was extracted using the Auditory Toolbox [164] employing the **proclpc** function. For LPCC 14 coefficients were used based on [68]. Finally, PLP was also extracted using the Auditory Toolbox [164] but by employing a different function **rastaplp**. Here 12 coefficients were used [69].

To make the extracted feature more robust and minimizes distortion that occurs due different sources of noise contamination (section 5.1) a cepstral mean and variance normalization was applied. This a very common practice in speech recognition task and the same hold for SR. The **wcmvn** function from the MSR Identity toolbox [162] was employed to perform the normalization. The normalized coefficients are then used to compute both the delta and delta-delta which are added to the original vector feature. So, the number of coefficients for each feature became:

- 36 for both MFCC and PLP
- 42 for LPCC

Figure 5.8 illustrates better the feature extraction pipeline from the raw unprocessed signal to the implied features.

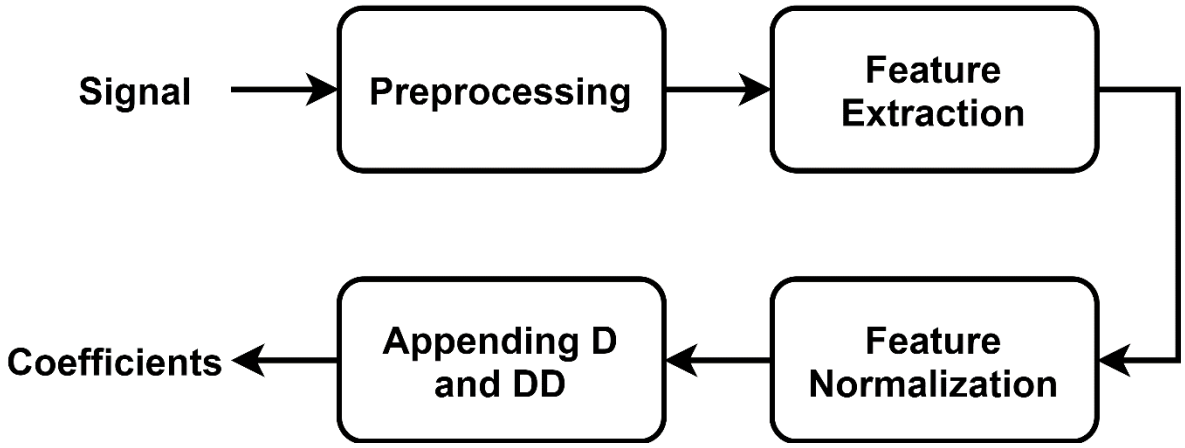

Figure 5.8 - Feature extraction pipeline.

5.3.2 Parameters tuning

Although the effect that the frame width is somehow debatable and has been pointed out to be unsignificant [10], the first step for improving the recognition rate was to adjust the frame width and increment for feature extraction pipeline described above. This improvement has already been shown [168, 169] which is compatible with the fact that these parameters determine whether the modeling algorithms are getting a sufficient level of information from the speech segment as input.

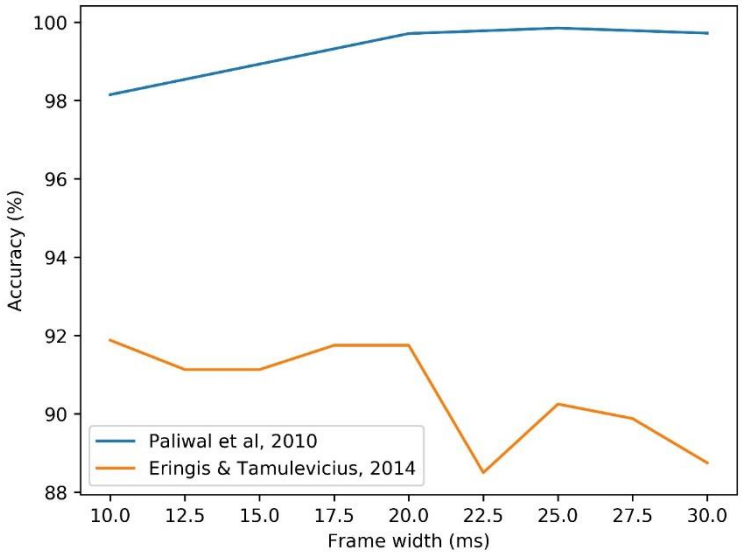

Figure 5.9 - The effect of adjusting the frame width during MFCC extraction on the overall accuracy, blue for the work done by [168] and orange for the work done by [169].

There is some discrepancy in the literature regarding the frame width values which optimize accuracy (Figure 5.9). To confirm which value to choose for the frame width, the SR

toolbox was used to design a single stream ANN structure as shown in Figure 5.10. The structure was trained using all 3 possible streams of features (i.e., MFCC, LPCC and PLP). The training was carried out using 10-fold cross-validation configuration, selecting each time thirty arbitrary segments for each speaker in the data set and the remaining 21 segments for reporting the results (testing). The results were averaged across folds and once this is done the frame width was increased by 5 ms. Overall, we started with a frame width of 10s and repeated the process until a frame width of 30.0 ms was reached.

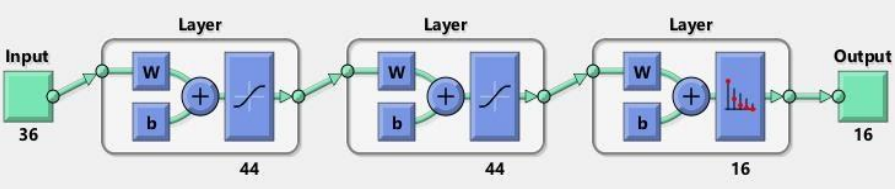

Figure 5.10 - The single feature stream ANN structure.

The second parameters to consider for tuning while extracting a feature was the frame increment. For this, 3 possible scenarios were considered: Overlapping (50%), Slightly Overlapping (75%) and Non-Overlapping (100%). To confirm the effect of each of these values, the value of the frame width was fixed to best reported value and in a similar fashion to the frame width tunning the single stream ANN (Figure 5.10) was trained (i.e., 10 folds cross validation, 30 segments for training and 21 for testing).

5.3.3 Implementation

The key idea for the proposed approach to improve VBBSs is through the definition of novel tokens. These tokens contain information from two different types of features fused together. The fused features are a combination of auditory system-based feature (type 1) and a speech system-based feature (type 2). To ensure that both features are kept isolated and not grouped under the same vector space, ANN were used to model the voice print. Using ANN, each set of features is connected to a dedicated input layer and are only fused/merged at the hidden layer (Figure 5.11).

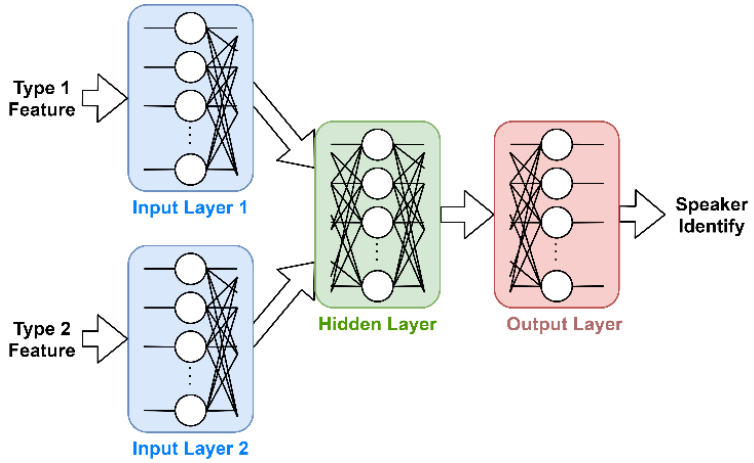

Figure 5.11 - The proposed system for feature fusion, with a type 1 feature being auditory system-based and type 2 being speech system-based.

The structure shown above was designed using the SR toolbox on MATLAB 2017a (Figure 5.12). This structure contains 4 layers: 2 input layers, a hidden layer, and an output layer. Each of the input layer as well as the hidden layer contained 44 perceptrons, to account for all possible phonemes in the English language. While the output layer contained a number of perceptrons equivalent to the number of implied speakers (16 for the recorded corpus, 921 for LibriSpeech).

To reduce the influence of extreme values or outliers in the dataset without having to remove them, a SoftMax function was used as the activation function for the output layer. For input layers and hidden layers perceptrons, the tangential sigmoid activation function was used. This function has a steeper derivative which make it a good candidate for extracting intermediary features [170].

To train the structures shown above, the conjugate gradient backpropagation algorithm is used to reduce the Sum of Square Errors (SSE) between the outputs of the network and a vector of desired targets. This algorithm has a better accuracy when compared with other algorithms as shown in Table 5.1 [171].

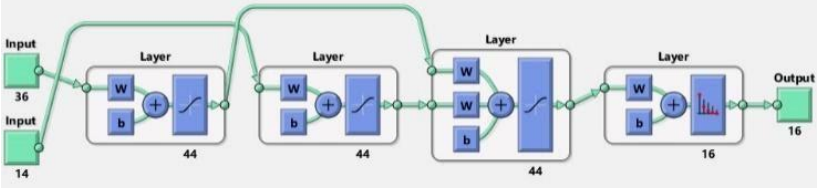

Figure 5.12 - Feature fusion ANN structure.

Table 5.1 - Performance of different training functions in MATLAB's NN toolbox [171]. where *trainlm* is levenberg-marquardt, *trainbr* is bayesian regularization, *trainbfg* is bfgs quasi-newton, *trainrp* is resilient backpropagation, *trainscg* is scaled conjugate gradient, *traincgb* is conjugate gradient with powell/beale restarts, *traincgf* is fletcher-powell conjugate gradient, *traincgp* is polak-ribiére conjugate gradient, *trainoss* one step secant, *traingdx* is variable learning rate gradient descent, *traingdm* is gradient descent with momentum and *traingd* is gradient descent.

Function name	training		validation		testing		time	
	mean	stdev	mean	stdev	mean	stdev	mean	stdev
trainlm	0.0065	0.0027	0.0199	0.0066	**0.0231**	0.0037	8.5762	3.494
trainbr	7.6088	3.5328	18.9761	10.219	149.829	32.2893	18.5063	8.927
trainbfg	0.0096	0.0032	0.0199	0.0084	**0.0209**	0.0046	7.3219	4.5702
trainrp	0.0137	0.0045	0.0207	0.0059	0.0229	0.0035	7.4954	3.8277
trainscg	0.0114	0.0035	0.0213	0.0109	**0.0218**	0.0073	4.3171	1.7394
traincgb	0.0102	0.0026	0.0193	0.0069	**0.0203**	0.0059	4.3389	1.886
traincgf	0.0112	0.0033	0.0199	0.0091	**0.0202**	0.0051	4.9752	2.4127
traincgp	0.0114	0.003	0.0213	0.0093	**0.0216**	0.0045	4.0544	1.9337
trainoss	0.013	0.0038	0.0204	0.0081	**0.0205**	0.0035	5.1703	2.8221
traingdx	0.0394	0.0312	0.0448	0.0317	0.0445	0.0274	5.4219	3.526
traingdm	0.5528	0.34	0.5556	0.3221	0.5592	0.3499	1.2875	0.3697
traingd	0.0265	0.0055	0.0332	0.0099	0.0323	0.0029	13.003	4.4432

Before applying transfer learning to define the mapping from the acoustic features vector space to the new token vector space, a study of the effect that feature fusion has on the overall recognition rate was conducted. For that the structure shown in Figure 5.12 was trained using two streams of features extracted at the optimal frame width and increment (5.3.2). The training was carried out using a 10-folds cross validation configuration. For each run 30 arbitrary segments were selected for each speaker while the remaining 21 segments are used to produce the evaluation results. This performance was then compared to that of training the structure in Figure 5.10 in a similar fashion using a single stream of all possible features (MFCC, LPCC and PLP) extracted at the same optimal parameters.

The steps that are discussed above were reapplied to the LibriSpeech corpus to confirm the approach usability for scenario involving a larger number of speakers (921) who are native English speakers. The corpus contains a varying number of speech segments for each speaker however for the sake of consistency only 51 segments were selected between training and testing similarly to the recorded corpus.

5.4 Tokenizers

The fusion steps provide a structure that has the potential of providing speaker-specific phonetic information for each frame, this is due to the fact at the output of the hidden layer in Figure 5.12 a mapping is being carried out intrinsically. This mapping takes the two streams of acoustic features and transform them into a single feature of 44 coefficients. We speculate that these 44 coefficients would correspond to the way in each speaker vocalize a specific phoneme, so coefficients should be different from a speaker to another and from a phoneme to another. To examine this the structure shown in Figure 5.12 was trained using the best performing pair of features using the recorded corpus. Once trained the last layer is dropped (the output or classification layer), and the output of the hidden layer is examined for the following phonemes: *"s", "ae", "iy", "ix", "aa", "n", "r", "l"*. This was carried out for all 16 subjects in the recorded corpus; however, these results were not reproduced for the LibriSpeech as examining synapsis for 921 speakers is an impossible task.

This obtained stream of coefficients can be viewed as a sequence of token that in addition to being phonetic and speaker specific, can also contain a co-occurring pattern of pronouncing certain words with a certain duration and certain configuration of phonemes. This should emphasize the speaker's print, so to that end, 2 additional structures were designed as a sort of advance tokenizers for detecting these patterns through the entire stream of the newly extracted feature. Unlike other structure this was not designed using the SR toolbox and instead they were implemented using a more recent version of MATLAB (R2020a) which contains some functionality that are yet to be integrated in the SR toolbox.

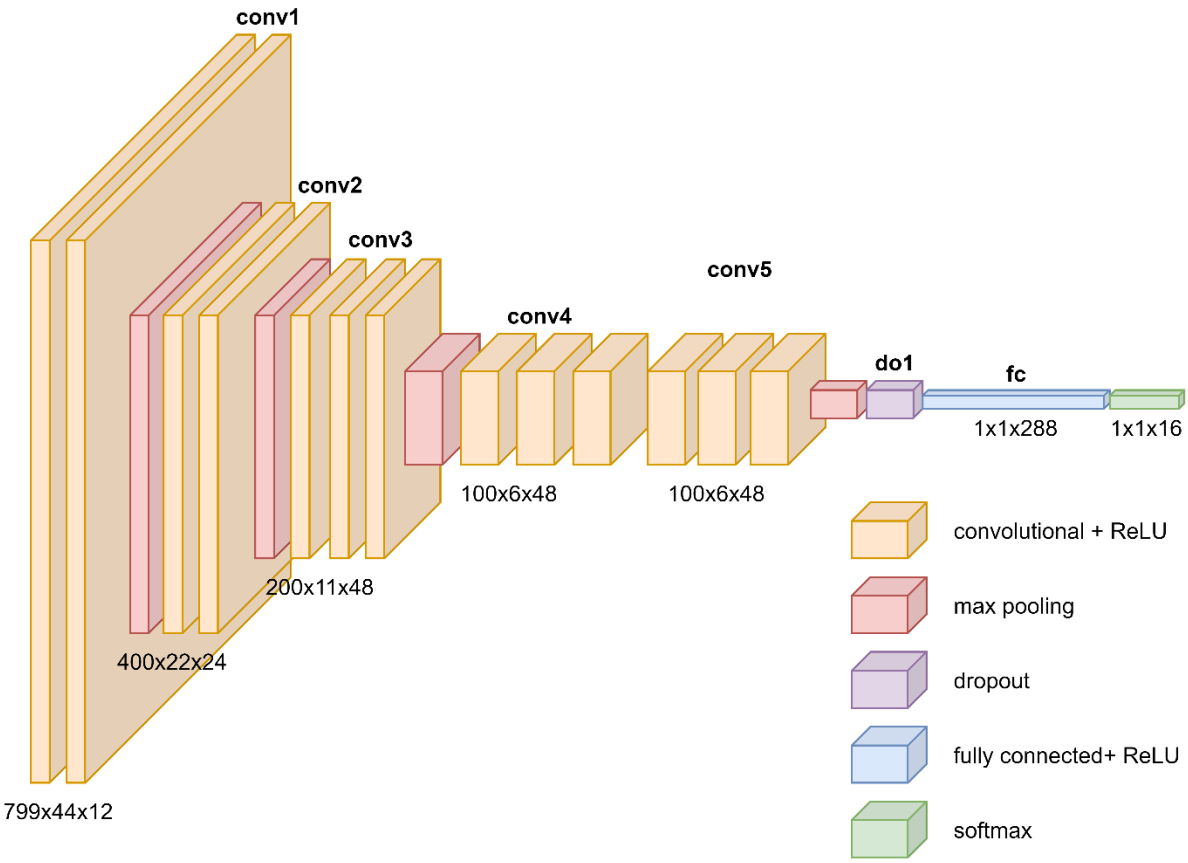

Figure 5.13 - CNN based structure for sequence learning and tokenization.

Table 5.2 - CNN based structure parameters

1	imageinput 799x44x1 images with 'zerocenter' normalization	Image Input	799×44×1	-
2	conv_1 12 3x3x1 convolutions with stride [1 1] and padding 'same'	Convolution	799×44×12	Weights 3×3×1×12 Bias 1×1×12
3	batchnorm_1 Batch normalization with 12 channels	Batch Normalization	799×44×12	Offset 1×1×12 Scale 1×1×12
4	relu_1 ReLU	ReLU	799×44×12	-
5	maxpool_1 3x3 max pooling with stride [2 2] and padding 'same'	Max Pooling	400×22×12	-
6	conv_2 24 3x3x12 convolutions with stride [1 1] and padding 'same'	Convolution	400×22×24	Weights 3×3×12×24 Bias 1×1×24
7	batchnorm_2 Batch normalization with 24 channels	Batch Normalization	400×22×24	Offset 1×1×24 Scale 1×1×24
8	relu_2 ReLU	ReLU	400×22×24	-
9	maxpool_2 3x3 max pooling with stride [2 2] and padding 'same'	Max Pooling	200×11×24	-
10	conv_3 48 3x3x24 convolutions with stride [1 1] and padding 'same'	Convolution	200×11×48	Weights 3×3×24×48 Bias 1×1×48
11	batchnorm_3 Batch normalization with 48 channels	Batch Normalization	200×11×48	Offset 1×1×48 Scale 1×1×48
12	relu_3 ReLU	ReLU	200×11×48	-
13	maxpool_3 3x3 max pooling with stride [2 2] and padding 'same'	Max Pooling	100×6×48	-
14	conv_4 48 3x3x48 convolutions with stride [1 1] and padding 'same'	Convolution	100×6×48	Weights 3×3×48×48 Bias 1×1×48
15	batchnorm_4 Batch normalization with 48 channels	Batch Normalization	100×6×48	Offset 1×1×48 Scale 1×1×48
16	relu_4 ReLU	ReLU	100×6×48	-
17	conv_5 48 3x3x48 convolutions with stride [1 1] and padding 'same'	Convolution	100×6×48	Weights 3×3×48×48 Bias 1×1×48
18	batchnorm_5 Batch normalization with 48 channels	Batch Normalization	100×6×48	Offset 1×1×48 Scale 1×1×48
19	relu_5 ReLU	ReLU	100×6×48	-
20	maxpool_4 100x1 max pooling with stride [1 1] and padding [0 0 0 0]	Max Pooling	1×6×48	-
21	dropout 20% dropout	Dropout	1×6×48	-
22	fc 16 fully connected layer	Fully Connected	1×1×16	Weights 16×288 Bias 16×1
23	softmax softmax	Softmax	1×1×16	-
24	classoutput Weighted cross entropy	Classification Output	-	-

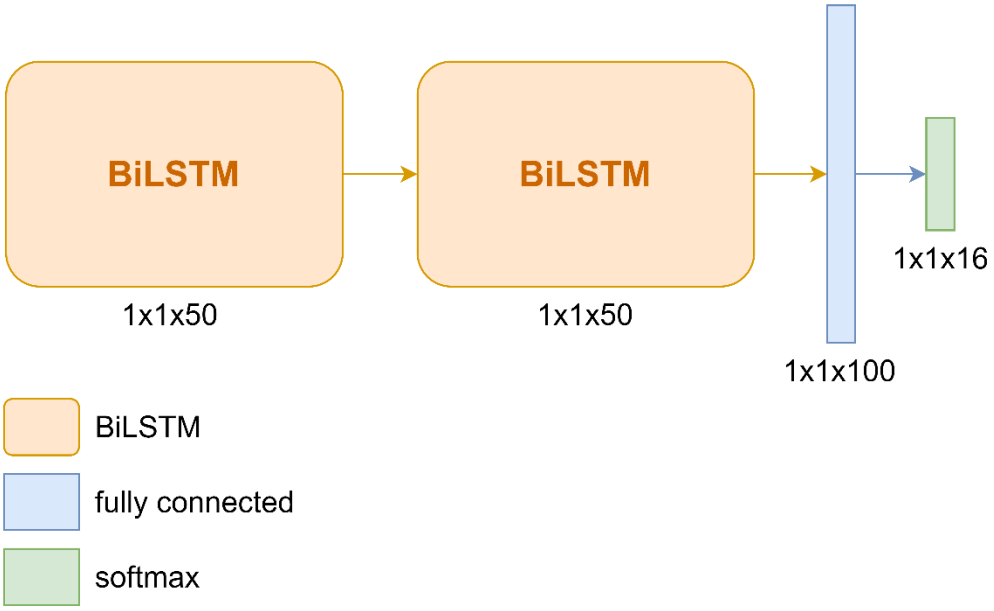

Figure 5.14 - BiLSTM based structure for sequence learning and tokenization.

Table 5.3 - BiLSTM based structure parameters

	Name	Type	Activa...	Learnables
1	sequenceinput Sequence input with 1 dimensions	Sequence Input	1	-
2	biLSTM_1 BiLSTM with 50 hidden units	BiLSTM	100	InputWeights 400×1 RecurrentWeights 400×50 Bias 400×1
3	biLSTM_2 BiLSTM with 50 hidden units	BiLSTM	100	InputWeights 400×100 RecurrentWeights 400×50 Bias 400×1
4	fc 16 fully connected layer	Fully Connected	16	Weights 16×100 Bias 16×1
5	softmax softmax	Softmax	16	-
6	classoutput crossentropyex	Classification Output	-	-

The hyperparameters for each structure are as shown in figures above (Figure 5.13 and 5.14). The first structure (Figure 5.13) is a CNN based network which is relatively small compared to the ones that are often been used for image classification tasks (subsection 2.3.3). However, for the specific task that is tackled in this work, 5 convolutional layers with their follow up filters were sufficient to downsample the original feature maps (remapped feature fusion) to a comprehensible level. The last filter that was used is the max pooling layer as this kind of pooling enforces time-translation symmetry in the input. The second structure (Figure 5.14) is a BiLSTM based structure. BiLSTM was used instead of a regular LSTM layer (which

was original intended) mainly because of the CNN based structures (Figure 5.13) and the fact that they are filtering the feature map in both direction (forward and backward). BiLSTM layer, unlike LSTM, have the ability to do just that. Here, 2 BiLSTM layers were implied, the first is for extracting intermediate features from the sequence of the single stream feature and the second provide interpretability of these hidden features to the FC layer that just follow. Both contain 100 value a decision that was made based on trial and error. To evaluate the loss of both structure the categorical class entropy was used as it is better suited for this compared to SSE. Since both structures contain relatively large number of parameters to be adjusted Adam optimizer was used instead of the conjugate gradient backpropagation algorithm as it is more appropriate.

To validate the effectiveness of the examined tokens, both structures were implied. First the ANN structure shown in Figure 5.12 was trained using a pair of type 1-2 features similarly to subsection 5.3.3. However, instead of using all of the existing speakers, only 12.5% from each corpus (i.e., 2 speakers for the recorded corpus and 115 speakers for LibriSpeech) were used for obtaining the pretrained fusion network (Figure 5.12). There was no need to reevaluate the network, so all segments of the selected speakers were used in training. This procedure defines the mapping from regular features to the speaker and phonetic specific ones.

With the features' mapping defined the advanced tokenizers (Figure 5.13 and 5.14) were trained and evaluated in a similar configuration as in subsection 4.3.2 (i.e., 10-folds cross validation with 30 segments for training and 21 for evaluation) using the remaining 87.5% speaker for both corpuses. This performance was then compared to that of training both structures in a similar fashion using a single stream of all possible features (MFCC, LPCC and PLP) extracted at the same optimal parameters.

Given the shape of the input for CNN, the full length of segments was not utilized but instead they were all were kept at a fixed duration of 6 seconds by either cropping or zero-padding.

CHAPTER VI

RESULTS & DISCUSSION

This chapter presents all of the results for the methods describes in the previous chapter. First, part present the results from section 5.3 which are related to [5] this includes the accuracy plots for the parameter tunning as well as the feature fusion. Each accuracy plot is followed by a discussion explaining the results. Then we explore the effect that the approach had on the training time and what is the minimum segment length for obtaining an acceptable recognition rate.

The second part explore the results from section 5.4 which are related to [13]. Here the synapsis for ANN structure shown in Figure 5.12 are explored and discussed, followed by the accuracy plot for implying these synapses with the structure shown in Figure 5.13 and 5.14.

6.1 Feature fusion

6.1.1 Parameter tunning

When adjusting for the frame width during feature extraction, the best recognition rate was obtained for a frame width of 10 ms, which coincides with the results obtained by [169]. These results are depicted in Figure 6.1. This pattern of improvement was found across all three features that were extracted.

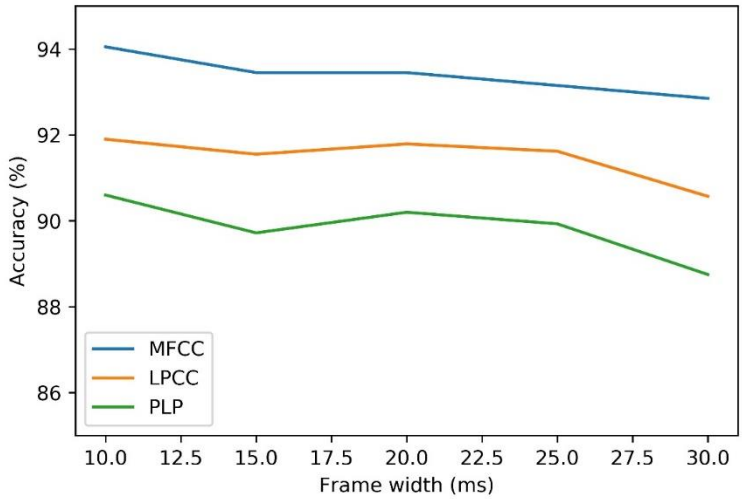

Figure 6.1 - The results of frame width tuning tasks.

A frame with of 10 ms is usually considered to be a relatively small window for the signal to be statically stable for classical machine learning algorithms. However, this does not to be the case for ANN structure due to the complexity of this model and its ability to find a

pattern if a sufficient amount of data points is provided. With the frame width fixed for the optimal value of 10 ms we Examined the 3 different scenarios for frame increments (subsection 5.3.2), here the best accuracy was reported for the slightly overlapping scenario (75% overlap) (Figure 6.2).

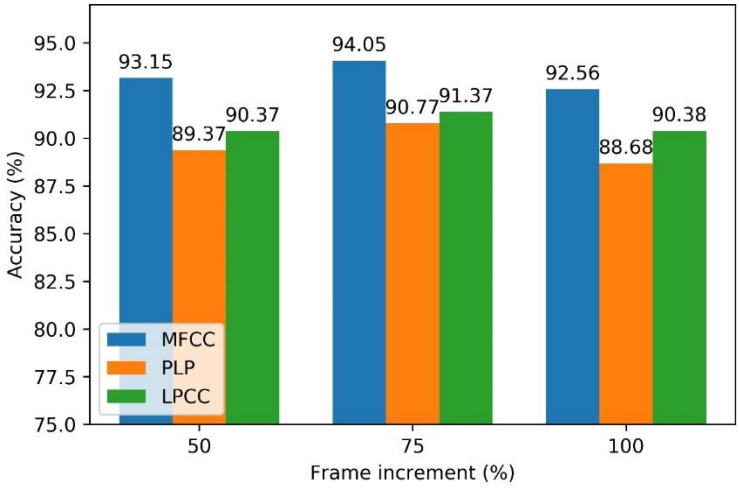

Figure 6.2 - The results of frame increment tuning task.

Unlike the value for the frame width this value coincides with other findings in the literature as 50% would result in more redundancy while the 100% increment could results in missing information, this is especially true for short duration window which is the case here. So, by just adjusting the extraction parameters to an optimal value, the ANN structure was able to correctly identify 3 additional speech segments compared to if it were trained with the default values for feature extraction (i.e., a frame width of 25 ms with 75% overlap).

6.1.2 Main results

The procedure described in subsection 5.3.4 resulted in the following accuracy (Figure 6.3).

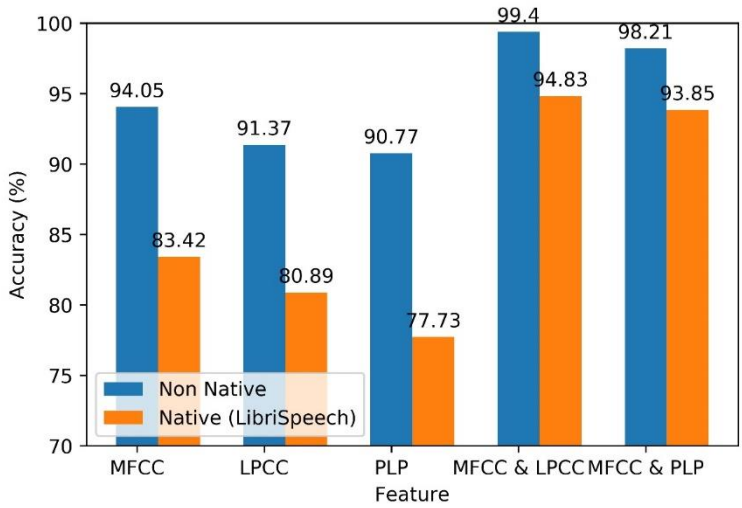

Figure 6.3 - Feature fusion effect on the overall accuracy for speaker recognition.

These results confirm that fusing a type 1 and type 2 features did improve the accuracy of the recognition tasks. In fact, as can be seen in Figure 6.3, any combination of type 1 and type 2 feature would result in a better recognition rate compared to when only a single acoustic feature is being used. The best result was obtained when combining MFCC and LPCC (99.4% accuracy), this is mainly due to the fact that these two sets of features are uncorrelated and that PLP are more speech recognition-oriented features then LPCC, as the pipeline for extracting its coefficient try to minimize the effect that the speaker has on the signal.

Figure 6.3 also demonstrates that the approach works well for native speakers and for a larger set of speakers. In fact, the improvement obtained when testing for the larger corpus was even more pronounced. This was due to the fact that this approach created higher dimensionality space that allowed for better discrimination. Of course, the approach is not perfect and resulted in a slight increase in the training time. Figure 6.4 shows the required training time for each of the trials described in section 2.5. These results were obtained using the 4710MG i7 CPU with 8Gb of RAMs.

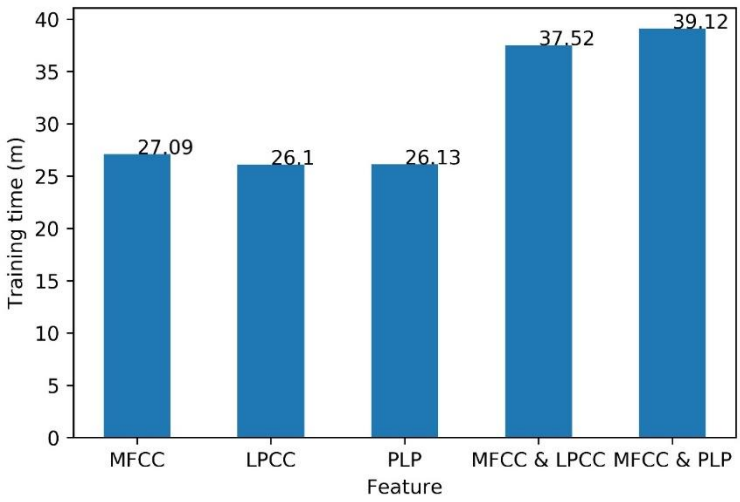

Figure 6.4 - Feature fusion effect on the required training time for SR.

The difference in training time between the best performing single feature structure and the best performing fused features structure is 10 minutes and 25 seconds. This means an increase of 38.50% in training time for an improvement of 5.35% in speaker-recognition accuracy if we consider only the effect of feature fusion without the parameter tuning and 6.55% when considering the parameters tuning. This is in fact very significant if we consider and application such as criminal investigation where 6.55% means that 22 cases or suspects are correctly being recognized using, for example, a phone call. Although, not tested we expect that this gain in accuracy will be even greater for scenario involving a much larger set of speakers such as the LibriSpeech Corpus.

In addition, this extra training time can be significantly reduced by using a better hardware such as the Nvidia Tesla K80 or any equivalent top performing GPU. What can also be done is to reduce the length of the implied speech segments for training and testing the model as feature fusion allows for a better recognition of the speaker at much earlier point in time compared to single stream approach as shown in Figure 6.5 and Table 6.1.

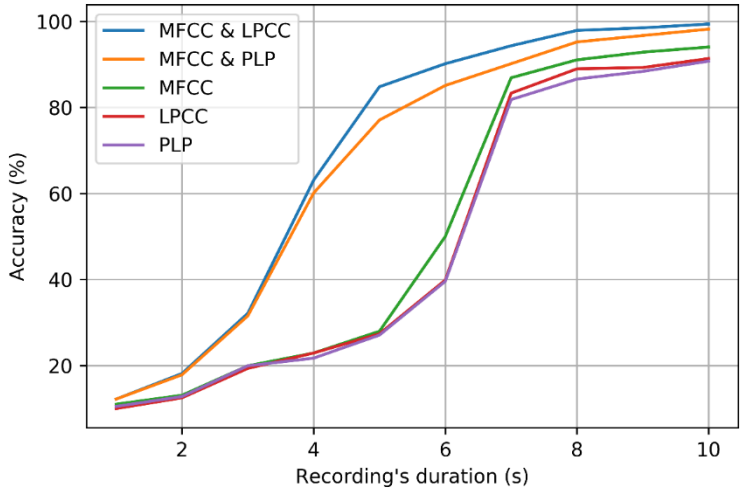

Figure 6.5 - Speaker recognition accuracy over time.

Table 6.1 - Speaker recognition accuracy over time.

Input	Time (s)									
	1	2	3	4	5	6	7	8	9	10
MFCC	11.01	13.09	19.94	22.91	27.97	50	86.9	91.07	92.85	94.05
LPC	10	12.5	19.34	22.91	27.38	39.88	83.33	88.89	89.28	91.37
PLP	10.5	12.79	19.94	21.72	27.08	39.58	81.84	86.60	88.39	90.77
MFCC & LPCC	12.20	18.15	32.14	63.09	84.82	**90.17**	**94.34**	97.91	98.57	99.4
MFCC & PLP	12.20	17.85	31.54	60.11	77.08	85.11	90.17	95.23	96.72	98.21

The results shown above proves that by using 6 seconds of a given speech segment the model was able to reach 90.17% when fusing MFCC and LPCC. What is even more impressing about these results is the fact that the fusion approach was able to outperform the single feature approach that while implying full length speech segments, by utilizing only 7 seconds out of the provided ~10 seconds of recording, this reduced training time in the fusion approach from 37 minutes and 32 seconds to 24 min and 41 seconds. This time is less than the time required to train the model for any of the features independently. These findings are very important when taking into consideration the applications of SR. For instance, in fields such as security, this means less data storage, and in crime investigation this means the ability to identify a suspect even if the provided audio is relatively short.

6.2 Advanced tokenizers

The results discussed above (section 6.1) are for when implying feature fusion in regular SR setup and not as mean of feature extraction or feature remapping, here the focus is the use of the ANN structure shown in Figure 5.12 as a token provider for future SR applications. First, the synapsis that are generated from this structure at the hidden layer need to be examined (section 5.3).

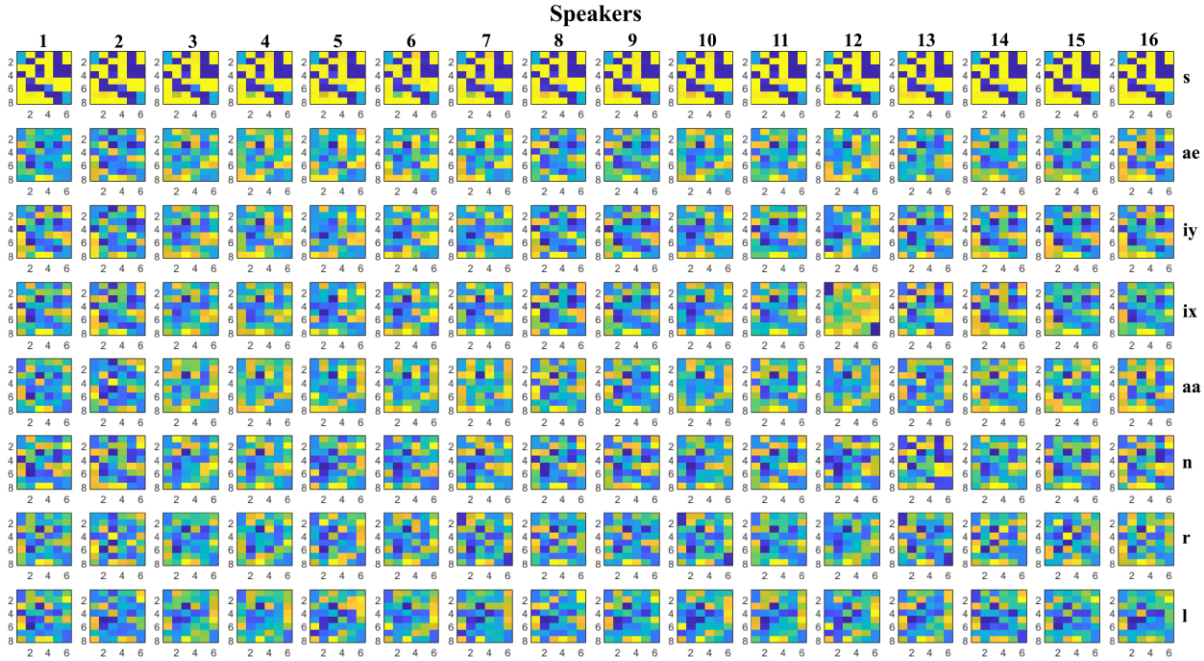

Figure 6.6 - Learned synapses using the feature fusion ANN structure, Y axis for phonemes and X axis for the speakers.

The synapses shown above are very much what was hypothesized in subsection 5.3.3 (specific to each speaker and each phoneme). What is interesting about these results is the fact that although the network was trained for discerning the speaker identity, the hidden layer intrinsically learned to distinguish between the different phoneme although no phonetic labeling was provided. In fact, the difference across phenome is even more pronounced than the difference across speakers, especially for unvoiced phonemes like *"s"* (1st line) where the output of the hidden layer looks almost identical. Nonetheless, even small there are difference which explain the high accuracy rate at the output layer. One synapsis that stands from the others is for the vocalization of the phoneme *"ix"* by 12th subject, this was a result of this phenome being underrepresented for this particular subject due to the random nature of sentence selection process during the recording.

Now that the speaker and phonetic specify of the newly developed tokens is confirmed, we are going to use them with the selected advanced tokenizer (Figure 5.13 and 5.14) to achieve boost the improvement that we have got so far from the approach. Both structures were trained using only 6 seconds of speech, a decision that was made based on the observation made in the previous section (section 6.1). The use of transfer learning to define a speaker-specific mapping for two streams of features worked and yielded in a huge boost the recognition rate as shown in Figure 6.7.

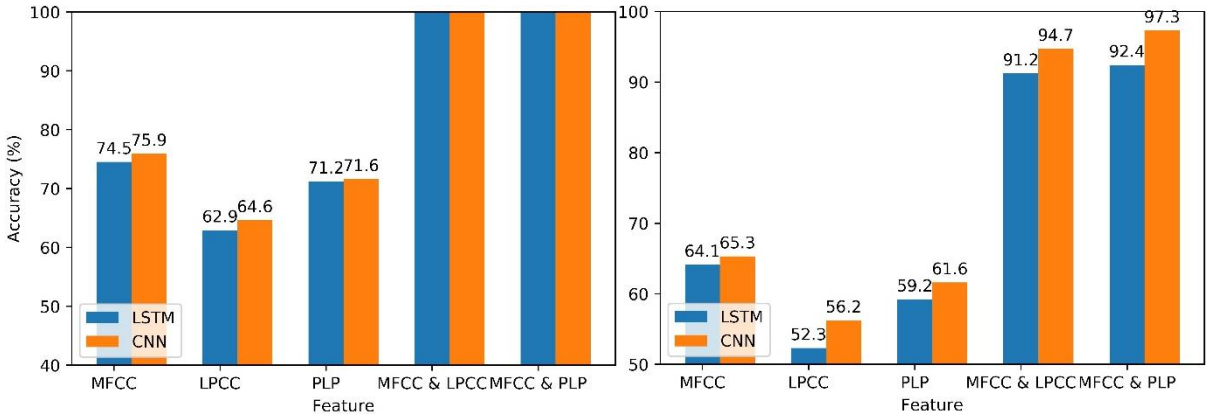

Figure 6.7 - The results of using the transferred feature on the overall accuracy, left is for the recorded corpus, and right is for LibriSpeech corpus.

Although the accuracy of the individual feature increased (i.e., compared to Figure 6.7) due to the use of advanced ANN structures, it is still not sufficient enough for this method to be considered for critical application such as security or crime investigation. However, the tokens that are defined through the use of transfer learning had a huge boost in performance leading even to saturation (100% accuracy) for the recorded corpus. In addition to this increase in performance by pre-training the mapping function beforehand, the training time for both fused features and regular acoustic feature, in this case, are identical.

What's worth noting is the fact that unlike the previous results, PLP performed better on its own as well as when paired with MFCC to make the remapped tokens, compared to LPCC. We believe that this is mainly due to the fact PLP although are less speaker-specific on a frame to frame basis, has the potential of providing clearer tokens for recognizing co-occurring patterns of phonetic information an effect that is further emphasized by the task being text-independent making it easy to detect differences in sequences. This implies that further investigation is required for text-dependent task especially text dependent verification as it is the norm for voice command bank services.

Another interesting observation is the fact that CNN slightly outperformed LSTM although it should be the other way around considering that LSTM are more sequence learning oriented. We believe that this is a result of CNN structure complexity and their ability to look for patterns through the entire spectrum and not just frame by frame as LSTM does. Of course, this means LSTM would be more practical for online applications. Regardless of what structure to use both are powerful tools and coupled with the newly defined speaker specific tokens a 65.3-97.3% increase in the performance have been obtained. This increase in performance is greater than what have been achieved so far, when considering the body of works that have been reviewed in chapter 2.

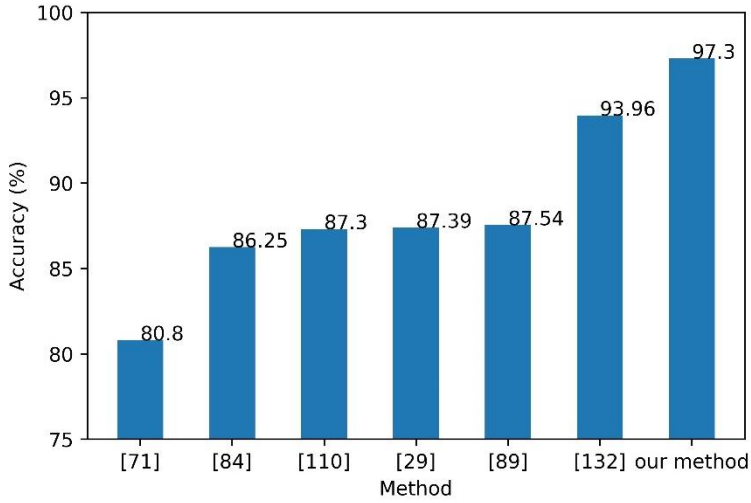

Figure 6.8 - A performance comparison between our final result and reviewed methods.

However, this comparison is not entirely just as each method was evaluated using a different dataset. This means different number of speakers, speech segments, and duration as well as different recording condition and audio quality. All of this information is shown in Table 6.2 and although some of used datasets contains a much larger number of speaker, segment and even longer duration, the indicated values are the ones used by the corresponding researcher team.

Table 6.2 - list of datasets implied by each of the compared methods. NIST SRE is the National Institute of Standards and Technology's Speaker Recognition Evaluation dataset.

Method	dataset	Number of speakers	Segments per speaker	Segment duration (s)
[29]	NIST SRE 2001	128	17	120
[71]	Local	49	16	15
[84]	NIST SRE 1999	30	80	60
[89]	NIST SRE 2004	820	50	10-120
[110]	NIST SRE 1999	480	70	15-45
[132]	NIST SRE 2006	640	78	>60
Our method	LibriSpeech	921	51	6

CHAPTER VII
CONCLUSION & FUTURE WORKS

7.1 General conclusion

The speech signal does not only convey a message, but it also conveys information about the speaker themselves, their gender, origins, health, and age. The aim of this work was to improve the task of recognizing a person based on speech segments.

In this work, we set to improve VBBSs through i) examining existing speaker specific feature ii) developing a new set feature that fits the speaker specific criteria iii) use the newly defined features with state-of-the-art PR algorithms. Overall, the obtained results suggest the effectiveness of fusing 2 or more complementary acoustic feature (MFCC, LPCC, and PLP) in obtaining a better accuracy for SR. Of course, in doing so new issues will be introduced to any VBBS. First issue is the increased training time due to the higher dimensionality resulting from concatenating multiple features. The second one has to do with need for additional storage unity for the additional features. The proposed approach solves these issues by remapping the selected features instead of just concatenating them. The remapping function that was suggested here didn't only reduce the dimension of the input vector but also extracted further speaker specific information which is key to any SR application.

Another important achievement of this work is the implication of state-of-the-art PR algorithms for SR. Modern algorithms are very complex and can learn from any data set as long as sufficient data and SR tasks are no different. In fact, it has been proven that even model that are originally designed for image related application can be implied and even outperform the structure that in theory should better suited for SR. The last points although unsignificant compared to the first is still important and it concern the feature extraction itself. This step has often been neglected in the literature which is not what was proven here, as adjusting the parameters for this step yielded in a significant increase in the recognition rate if it was not the main reason why the extracted features were transferable in the first place.

The presented work will set an era for new and various methods for speaker specific feature extraction, that will become the norm across all SR applications. The use of such features coupled with the ever-improving modelling techniques for PR tasks, will make voice-based application more secure and thus more popular. This of course will a huge impact on the following fields.

Crime Investigation, the most important achievement of this work is the fact that it performs reasonably well even with small duration segments (6 seconds) and for the scenario with 16 speakers the approach was 100% all the time (10-folds). This usually the case as in most investigations the list of suspects is narrowed down to a number between 1 to 10.

Banking services, if this approach is able to perform almost perfectly for text independent speaker identification (very small margin of misidentification or none in the case of the recorded corpus), performing a text independent speaker verification would be an even easier task for the approach. This implies that banks will no longer acquire of their subject to repeat a certain sentence to grant the user access to banking service, instead costumer will be able to issue any voice command and have their identity verified at the same time. This might be a trivial thing to do for a regular user as most prefer to use GUI elements present on the app, however for blind users this would be very convenient as phone assistance (e.g., Siri or "Ok, Google") usually has no control over such app.

Businesses, the approach was only tested for speaker identification, a task that is not generally used for businesses except for banks which is already discussed. However, the finding concerning the definition of novel features by remapping preexisting ones to amore speaker and phonetic specific space could be beneficial for speaker segmentation systems which is often used for parsing and labeling meetings. These features could also be used for speech analyzing these meetings as they are phoneme specific, but it is unlikely that they will outperform existing speech recognition systems.

Personal life, here the same implications that the approach has on the banking services are in effect. For instance, home appliances will not only use voice to recognize what is being asked of it, but also discern the identity of the speaker at the same time allowing for it to decide whether this user have access to the command in question and even provide a bit of recommendation based on the commander preferences. An additional application of this, could be the detection of intruders and signal the authority, however this feature will have ethical implications that would be controversial to the general population.

7.2 Raised question & future works

Is feature remapping effective against synthesizers?

It true that the feature remapping approach coupled with the use advanced ANN structure as tokenizers helped improve the recognition rate significantly, there is no grantee that the approach will function when tasked with verifying whether a speech segment is genuine or generated using a one of the synthesizers discussed in the abstract (WaveNet [172], GenSynth [173] and MelGAN [174]). So, in future work the approach will be used in a speaker verification task were containing both type of segments. This work will be a review article

including not only this approach, but others top performing or common techniques selected from the literature of the last decade.

Is feature remapping applicable to raw signals?

What was tested here is the remapping of existing acoustic feature, specifically two streams of complementary features (type 1 and type 2 features). Both features originated from the signal and the same spectrum, which contains not only the level information represented by the extracted features but other level as well that could be helpful for reproducing speaker specific features using the same steps. Of course, the original signal whether in temporal or spectral domains is still raw and the ANN structure shown in Figure 5.12 will not be sufficient for obtaining comprehensible DF. So instead, when using raw signal, a much deeper structure will be implied a structure that might even contain CNN or LSTM block just for the feature remapping.

Is feature remapping accessible and easy to implement?

The proposed approach relies on ANN structure (Figure 5.12) to function, and it is a well-established fact that implementing ANN structures on hardware is not an easy task. The difficulty of implementing ANN on hardware stems from the training procedure. So, what need to be confirmed is whether the transferred features can be fixed or reach a sort of steady state by including more subjects and more data to avoid retraining the remapping function?

What is missing in SR toolbox?

The SR toolbox was of the main contribution of this work. This toolbox is not perfect and is missing a lot of the functionality needed for it to be used an all sort of SR applications, including verification, identification, and segmentation. The first adjustment that would be added to this tool would be the steps that where manual made in this work. These are transfer learning, synapsis examination, advanced ANN structures designing, and training. Other features might be added afterward such as the possibility to perform verification and segmentation along with all the necessary function for implementing, evaluating, and visualizing both tasks.

What other PR algorithms could work for SR applications?

The two structures (Figure 5.13 and 5.14) implied in this work are not the only advanced ANN structure that could work for SR. Using the same logic implied here there are several

structures that in theory have the potential to work as good if not better than the implemented structures. Structures such as ResNet+BiLSTMs [175], Large-10h-LV-60k [176], BiT-L [177] and Branching CNN [178] could also be implied as tokenizers.

REFERENCES

[1] Syntellect Inc., “The Importance of Creating An Effective Voice User Interface,” 2003. .

[2] A. Abdolrahmani, R. Kuber, and S. M. Branham, “Siri talks at you: An empirical investigation of voice-activated personal assistant (VAPA) usage by individuals who are blind,” in *ASSETS 2018 - Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*, 2018, pp. 249–258.

[3] Donald R. Vander Molen, “Conversational voice command control system for home appliance,” US4520576A, Jun-1985.

[4] H. Tankovska, “Number of voice assistants in use worldwide 2019-2023,” *statista.com*, Sep-2020. .

[5] Y. I. Cherifi and A. Dahimene, “FEATURE FUSION BASED ON AUDITORY AND SPEECH SYSTEMS FOR AN IMPROVED VOICE BIOMETRICS SYSTEM USING ARTIFICIAL NEURAL NETWORK,” *Bull. Polytech. Inst. Jassy*, vol. 65 (69), no. 4, pp. 29–43, 2020.

[6] Y. Jia *et al.*, “Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis,” *Adv. Neural Inf. Process. Syst.*, vol. 2018-Decem, pp. 4480–4490, Jun. 2018.

[7] J. Pollard, “Predictions 2020: Cyberattacks Influence Society In a Broader Way,” *Forrester.com*, Oct-2019. .

[8] BBC, “Fake voices ‘help cyber-crooks steal cash,’” *BBC News*, Jul-2019. .

[9] N. Statt, “Thieves are now using AI deepfakes to trick companies into sending them money,” *The Verge*, Sep-2019. .

[10] T. Kinnunen and H. Li, “An overview of text-independent speaker recognition: From features to supervectors,” *Speech Commun.*, vol. 52, no. 1, pp. 12–40, Jan. 2010.

[11] R. Setiono and H. Liu, “Feature extraction via Neural networks,” in *Feature Extraction, Construction and Selection*, vol. 453, Springer US, 1998, pp. 191–204.

[12] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, 2010.

[13] Y. I. Cherifi and A. Dahimene, “Improved Voice-Based Biometrics using Multi-Channel Transfer Learning,” *IADIS Int. J. Comput. Sci. Syst.*, vol. 15, no. 1, pp. 99–113, 2020.

[14] D. A. Reynolds, “An overview of automatic speaker recognition technology,” in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2002, vol. 4.

[15] S. Furui, “Recent advances in speaker recognition,” *Pattern Recognit. Lett.*, vol. 18, no. 9, pp. 859–872, Sep. 1997.

[16] J. P. Campbell, “Speaker recognition: A tutorial,” *Proc. IEEE*, vol. 85, no. 9, pp. 1437–1462, 1997.

[17] B. S. Atal, “Automatic Recognition of Speakers from Their Voices,” *Proc. IEEE*, vol. 64, no. 4, pp. 460–475, 1976.

[18] F. J. D. Nolan, *The phonetic bases of speaker recognition*. University of Cambridge, 1983.

[19] L. Rabiner and B. H. Juang, *Fundamentals of speech recognition*. Englewood Cliffs N.J.: PTR Prentice Hall, 1993.

[20] F. Nolan, “The phonetic bases of speaker recognition,” *Speech Commun.*, vol. 6, no. 2, pp. 171–175, Jun. 1987.

[21] J. J. Wolf, “Efficient Acoustic Parameters for Speaker Recognition,” *J. Acoust. Soc.*

Am., vol. 51, no. 6B, pp. 2044–2056, Jun. 1972.
[22] G. Doddington, "Speaker recognition based on idiolectal differences between speakers.," in *Seventh European Conf. on Speech Communication and Technology (Eurospeech 2001)*, 2001, pp. 2521–2524.
[23] W. D. Andrews, M. A. Kohler, J. P. Campbell, J. J. Godfrey, and J. Hernández-Cordero, "Gender-dependent phonetic refraction for speaker recognition," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 1, pp. 149–152, 2002.
[24] W. M. Campbell, J. P. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek, "Phonetic speaker recognition with support vector machines," *Adv. Neural Inf. Process. Syst.*, 2004.
[25] A. G. Adami, R. Mihaescu, D. A. Reynolds, and J. J. Godfrey, "Modeling prosodic dynamics for speaker recognition," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2003, vol. 4, pp. 788–791.
[26] Z. H. Chen, Y. F. Liao, and Y. T. Juang, "Eigen-prpsody analysis for robust speaker recognition under mismatch handset environment," *Electron. Lett.*, vol. 40, no. 19, pp. 1233–1235, Sep. 2004.
[27] E. Shriberg, L. Ferrer, S. Kajarekar, A. Venkataraman, and A. Stolcke, "Modeling prosodic feature sequences for speaker recognition," in *Speech Communication*, 2005, vol. 46, no. 3–4, pp. 455–472.
[28] K. Y. Leung, M. W. Mak, M. H. Siu, and S. Y. Kung, "Adaptive articulatory feature-based conditional pronunciation modeling for speaker verification," *Speech Commun.*, vol. 48, no. 1, pp. 71–84, Jan. 2006.
[29] B. Ma, D. Zhu, R. Tong, and H. Li, "Speaker cluster based GMM tokenization for speaker recognition," *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, vol. 1, pp. 505–508, 2006.
[30] P. A. Torres-Carrasquillo, D. A. Reynolds, and J. R. Deller, "Language identification using Gaussian mixture model tokenization," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 1, pp. 757–760, 2002.
[31] B. Xiang, "Text-independent speaker verification with dynamic trajectory model," *IEEE Signal Process. Lett.*, vol. 10, no. 5, pp. 141–143, May 2003.
[32] Q. Jin, T. Schultz, and A. Waibel, "Speaker identification using multilingual phone strings," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2002, vol. 1.
[33] M. A. Zissman, "Comparison of four approaches to automatic language identification of telephone speech," *IEEE Trans. Speech Audio Process.*, vol. 4, no. 1, pp. 31–44, 1996.
[34] B. Ma, H. Li, and R. Tong, "Spoken language recognition using ensemble classifiers," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 15, no. 7, pp. 2053–2062, Sep. 2007.
[35] H. Ney, S. Martin, and F. Wessel, "Statistical Language Modeling Using Leaving-One-Out," Springer, Dordrecht, 1997, pp. 174–207.
[36] W. D. Andrews, M. A. Kohler, and J. P. Campbell, "Phonetic speaker recognition," *EUROSPEECH 2001 - Scand. - 7th Eur. Conf. Speech Commun. Technol.*, pp. 2517–2520, 2001.
[37] K. Bartkova, D. Gac, D. Charlet, and D. Jouvet, "Prosodic parameter for speaker identification," in *INTERSPEECH*, 2002.
[38] D. Reynolds *et al.*, "The SuperSID project: Exploiting high-level information for high-accuracy speaker recognition," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2003, vol. 4, pp. 784–787.
[39] W. Hess, *Pitch Determination of Speech Signals*, vol. 3. Berlin, Heidelberg: Springer Berlin Heidelberg, 1983.
[40] P. Rose, "Forensic Speaker Identification," 2002.

[41] B. S. Atal, “Automatic Speaker Recognition Based on Pitch Contours,” *J. Acoust. Soc. Am.*, vol. 52, no. 6B, pp. 1687–1697, Dec. 1972.
[42] M. J. Carey, E. S. Parris, H. Lloyd-Thomas, and S. Bennett, “Robust prosodic features for speaker identification,” in *International Conference on Spoken Language Processing, ICSLP, Proceedings*, 1996, vol. 3, pp. 1800–1803.
[43] K. Sönmez, E. Shriberg, L. Heck, and M. Weintraub, “Modeling dynamic prosodic variation for speaker verification.,” in *Internat. Conf. Spok. Lang. Process. (ICSLP 1998)*, 1998, pp. 3189–3192.
[44] M. Sönmez, L. Heck, M. Weintraub, and E. Shriberg, “A lognormal tied mixture model of pitch for prosody-based speaker recognition.,” in *Fifth Eur. Conf. Speech Commun. Technol. (Eurospeech 1997)*, 1997, pp. 1391–1394.
[45] T. Kinnunen and R. G. Hautaṁaki, “Long-Term F0 Modeling for Text-Independent Speaker Recognition,” in *Proc. 10th Internat. Conf. Speech Comput.*, 2005, pp. 567–570.
[46] K. Laskowski and Q. Jin, “Modeling instantaneous intonation for speaker identification using the fundamental frequency variation spectrum,” in *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, 2009, pp. 4541–4544.
[47] B. Ma, D. Zhu, and R. Tong, “Chinese Dialect Identification Using Tone Features Based on Pitch Flux,” in *2006 IEEE Int. Conf. Acoust. Speed Signal Process. Proc.*, 2006, vol. 1, pp. 1029–1032.
[48] B. Wildermoth and K. Paliwal, “Use of voicing and pitch information for speaker recognition.,” in *Proc. Eighth Aust. Internat. Conf. Speech Sci. Technol.*, 2000, pp. 324–328.
[49] C. Espy-Wilson, S. Manocha, and S. Vishnubhotla, “A new set of features for text-independent speaker identification,” in *INTERSPEECH*, 2006.
[50] K. Sri Rama Murty, B. Yegnanarayana, K. S. R. Murty, and B. Yegnanarayana, “Combining evidence from residual phase and MFCC features for speaker recognition,” *IEEE Signal Process. Lett.*, vol. 13, no. 1, pp. 52–55, Jan. 2006.
[51] M. D. Plumpe, T. F. Quatieri, and D. A. Reynolds, “Modeling of the glottal flow derivative waveform with application to speaker identification,” *IEEE Trans. Speech Audio Process.*, vol. 7, no. 5, pp. 569–585, 1999.
[52] S. R. Mahadeva Prasanna *et al.*, “Extraction of speaker-specific excitation information from linear prediction residual of speech,” *Speech Commun.*, vol. 48, no. 10, pp. 1243–1261, Oct. 2006.
[53] P. Thévenaz and H. Hügli, “Usefulness of the LPC-residue in text-independent speaker verification,” *Speech Commun.*, vol. 17, no. 1–2, pp. 145–157, Aug. 1995.
[54] N. Zheng, T. Lee, and P. C. Ching, “Integration of complementary acoustic features for speaker recognition,” *IEEE Signal Process. Lett.*, vol. 14, no. 3, pp. 181–184, Mar. 2007.
[55] J. Gudnason and M. Brookes, “Voice source cepstrum coefficients for speaker identification,” in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2008, pp. 4821–4824.
[56] R. Slyh, E. G. Hansen, and T. R. Anderson, “Glottal modeling and closed-phase analysis for speaker recognition,” in *Odyssey*, 2004.
[57] P. Alku, H. Tiitinen, and R. Näätänen, “A method for generating natural-sounding speech stimuli for cognitive brain research,” *Clin. Neurophysiol.*, vol. 110, no. 8, pp. 1329–1333, Aug. 1999.
[58] M. Chetouani, M. Faundez-Zanuy, B. Gas, and J. L. Zarader, “Investigation on LP-residual representations for speaker identification,” *Pattern Recognit.*, vol. 42, no. 3, pp. 487–494, Mar. 2009.
[59] W. N. Chan, N. Zheng, and T. Lee, “Discrimination power of vocal source and vocal

tract related features for speaker segmentation," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 15, no. 6, pp. 1884–1892, Aug. 2007.
[60] S. B. Davis and P. Mermelstein, "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 4. pp. 357–366, 1980.
[61] N. P. H. Thian, C. Sanderson, and S. Bengio, "Spectral Subband Centroids as Complementary Features for Speaker Authentication," in *Biometric Authentication*, 2004, pp. 631–639.
[62] C. Charbuillet, B. Gas, M. Chetouani, and J. L. Zarader, "Filter bank design for speaker diarization based on genetic algorithms," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2006, vol. 1.
[63] C. Miyajima, H. Watanabe, K. Tokuda, T. Kitamura, and S. Katagiri, "A new approach to designing a feature extractor in speaker identification based on discriminative feature extraction," *Speech Commun.*, vol. 35, no. 3–4, pp. 203–218, Oct. 2001.
[64] Ö. D. Orman and L. Arslan, "Frequency analysis of speaker identification," in *Odyssey*, 2001.
[65] J. Makhoul, "Linear Prediction: A Tutorial Review," *Proc. IEEE*, vol. 63, no. 4, pp. 561–580, 1975.
[66] R. J. Mammone, X. Zhang, and R. P. Ramachandran, "Robust speaker recognition: A feature-based approach," *IEEE Signal Process. Mag.*, vol. 13, no. 5, pp. 58–71, 1996.
[67] J. Harrington and S. Cassidy, "Techniques in Speech Acoustics," *Comput. Linguist.*, vol. 26, no. 2, pp. 294–295, 2000.
[68] X. Huang, A. Acero, H. Hon, and R. Reddy, "Spoken Language Processing: A Guide to Theory, Algorithm and System Development," 2001.
[69] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *J. Acoust. Soc. Am.*, vol. 87, no. 4, pp. 1738–1752, Apr. 1990.
[70] B. S. Atal, "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification," *J. Acoust. Soc. Am.*, vol. 55, no. 6, pp. 1304–1312, Jun. 1974.
[71] D. Reynolds and R. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. Speech Audio Process.*, vol. 3, pp. 72–83, 1995.
[72] S. Furui, "Cepstral Analysis Technique for Automatic Speaker Verification," *IEEE Trans. Acoust.*, vol. 29, no. 2, pp. 254–272, 1981.
[73] F. Soong and A. Rosenberg, "On the use of instantaneous and transitional spectral information in speaker recognition," in *ICASSP '86. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 11, pp. 877–880.
[74] I. Magrin-Chagnolleau, G. Durou, and F. Bimbot, "Application of time-frequency principal component analysis to text-independent speaker identification," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 6, pp. 371–378, Sep. 2002.
[75] N. Malayath, H. Hermansky, S. Kajarekar, and B. Yegnanarayana, "Data-driven temporal filters and alternatives to GMM in speaker verification," *Digit. Signal Process. A Rev. J.*, vol. 10, no. 1, pp. 55–74, Jan. 2000.
[76] T. Kinnunen, "Spectral Features for Automatic Text-Independent Speaker Recognition. Licentiate's Thesis," University of Joensuu, Joensuu, Finland., 2004.
[77] T. Kinnunen, "Joint acoustic-modulation frequency for speaker recognition," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2006, vol. 1.
[78] T. Kinnunen, K.-A. Lee, and H. Li, "Dimension reduction of the modulation

spectrogram for speaker verification," in *Odyssey*, 2008.
[79] L. Atlas and S. A. Shamma, "Joint acoustic and modulation frequency," *EURASIP J. Appl. Signal Process.*, vol. 2003, no. 7, pp. 668–675, Jul. 2003.
[80] H. Hermansky, "Should recognizers have ears?," *Speech Commun.*, vol. 25, no. 1–3, pp. 3–27, Aug. 1998.
[81] T. Thiruvaran, E. Ambikairajah, and J. Epps, "Extraction of FM components from speech signals using all-pole model," *Electron. Lett.*, vol. 44, pp. 449–450, 2008.
[82] T. Thiruvaran, E. Ambikairajah, and J. Epps, "FM features for automatic forensic speaker recognition," in *INTERSPEECH*, 2008.
[83] D. K. Burton, "Text-Dependent Speaker Verification Using Vector Quantization Source Coding," *IEEE Trans. Acoust.*, vol. 35, no. 2, pp. 133–143, 1987.
[84] V. Hautamäki, T. Kinnunen, and P. Fränti, "Text-independent speaker recognition using graph matching," *Pattern Recognit. Lett.*, vol. 29, no. 9, pp. 1427–1432, Jul. 2008.
[85] E. Karpov, T. Kinnunen, and P. Fronti, "Symmetric Distortion Measure for Speaker Recognition," in *SPECOM'2004 9th Conf. Speech Comput.*, 2004, pp. 366–370.
[86] F. K. Soong, A. E. Rosenberg, B. -H Juang, and L. R. Rabiner, "Report: A Vector Quantization Approach to Speaker Recognition," *AT&T Tech. J.*, vol. 66, no. 2, pp. 14–26, 1987.
[87] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Springer US, 1992.
[88] J. Saastamoinen, E. Karpov, V. Hautamäki, and P. Fränti, "Accuracy of MFCC-based speaker recognition in series 60 device," *EURASIP J. Appl. Signal Process.*, vol. 2005, no. 17, pp. 2816–2827, 2005.
[89] J. Louradour and K. Daoudi, "SVM speaker verification using a new sequence Kernel," in *13th Eur. Conf. Signal Process. (EUSIPCO 2005)*, 2005.
[90] M. Roch, "Gaussian-selection-based non-optimal search for speaker identification," *Speech Commun.*, vol. 48, no. 1, pp. 85–95, Jan. 2006.
[91] Y. Linde, A. Buzo, and R. M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. Commun.*, vol. 28, no. 1, pp. 84–95, 1980.
[92] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digit. Signal Process. A Rev. J.*, vol. 10, no. 1, pp. 19–41, 2000.
[93] L. Besacier and J. F. Bonastre, "Subband architecture for automatic speaker recognition," *Signal Processing*, vol. 80, no. 7, pp. 1245–1259, Jul. 2000.
[94] L. Besacier, J. F. Bonastre, and C. Fredouille, "Localization and selection of speaker-specific information with statistical modeling," *Speech Commun.*, vol. 31, no. 2, pp. 89–106, Jun. 2000.
[95] F. Bimbot, I. Magrin-Chagnolleau, and L. Mathan, "Second-order statistical measures for text-independent speaker identification," *Speech Commun.*, vol. 17, no. 1–2, pp. 177–192, 1995.
[96] R. D. Zilca, "Text-independent speaker verification using utterance level scoring and covariance modeling," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 6, pp. 363–370, Sep. 2002.
[97] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006.
[98] G. Kolano and P. Regel-Brietzmann, "Combination of Vector Quantization and Gaussian Mixture Models for Speaker Verification with Sparse Training Data," in *Sixth Eur. Conf. Speech Commun. Technol.*, 1999.
[99] J. Pelecanos, S. Myers, S. Sridharan, and V. Chandran, "Vector quantization based gaussian modeling for speaker verification," *Proceedings-International Conf. Pattern*

Recognit., vol. 15, no. 3, pp. 294–297, 2000.
[100] M.-W. Mak, R. Hsiao, and B. Mak, "A Comparison of Various Adaptation Methods for Speaker Verification With Limited Enrollment Data," in *2006 IEEE Int. Conf. Acoust. Speed Signal Process. Proc.*, vol. 1, pp. I--929--I--932.
[101] J. Mariéthoz and S. Bengio, "A comparative study of adaptation methods for speaker verification," in *Internat. Conf. Spok. Lang. Process. (ICSLP 2002)*, 2002, pp. 581–584.
[102] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models," *Comput. Speech Lang.*, vol. 9, no. 2, pp. 171–185, 1995.
[103] Z. N. Karam and W. M. Campbell, "A New Kernel for SVM MLLR Based Speaker Recognition," in *INTERSPEECH 2007 8th Annu. Conf. Int. Speech Commun. Assoc.*, 2007, pp. 290–293.
[104] A. Stolcke, S. S. Kajarekar, L. Ferrer, and E. Shrinberg, "Speaker recognition with session variability normalization based on MLLR adaptation transforms," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 15, no. 7, pp. 1987–1998, Sep. 2007.
[105] R. Auckenthaler and J. S. Mason, "Gaussian selection applied to text-independent speaker verification," in *2001 A Speak. Odyssey - Speak. Recognit. Work.*, 2001, pp. 83–88.
[106] J. Louradour, K. Daoudi, and R. André-Obrecht, "Discriminative power of transient frames in speaker recognition," in *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, 2005, vol. I, pp. 613–616.
[107] J. McLaughlin, D. Reynolds, and T. Gleason, "A study of computation speed-ups of the GMM–UBM speaker recognition system," in *Sixth Eur. Conf. Speech Commun. Technol. (Eurospeech 1999)*, 1999, pp. 1215–1218.
[108] B. L. Pellom and J. H. L. Hansen, "An efficient scoring algorithm for Gaussian mixture model based speaker identification," *IEEE Signal Process. Lett.*, vol. 5, no. 11, pp. 281–284, 1998.
[109] R. Saeidi, H. R. S. Mohammadi, T. Ganchev, and R. D. Rodman, "Particle swarm optimization for sorted adapted gaussian mixture models," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 17, no. 2, pp. 344–353, Feb. 2009.
[110] B. Xiang and T. Berger, "Efficient text-independent speaker verification with structural Gaussian mixture models and neural network," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 5, pp. 447–456, Sep. 2003.
[111] Z. Xiong, T. F. Zheng, Z. Song, F. Soong, and W. Wu, "A tree-based kernel selection approach to efficient Gaussian mixture model-universal background model based speaker identification," *Speech Commun.*, vol. 48, no. 10, pp. 1273–1282, Oct. 2006.
[112] U. V Chaudhari, J. Navrátil, and S. H. Maes, "Multigrained modeling with pattern specific maximum likelihood transformations for text-independent speaker recognition," *IEEE Trans. Speech Audio Process.*, vol. 11, no. 1, pp. 61–69, Jan. 2003.
[113] M. Hebert and L. P. Heck, "Phonetic Class-Based Speaker Verification," in *EUROSPEECH 2003 - INTERSPEECH 2003 8th Eur. Conf. Speech Commun. Technol.*, 2003, pp. 1665–1668.
[114] F. Castaldo, D. Colibro, E. Dalmasso, P. Laface, and C. Vair, "Compensation of nuisance factors for speaker and language recognition," *IEEE Trans. Audio, Speech Lang. Process.*, vol. 15, no. 7, pp. 1969–1978, Sep. 2007.
[115] R. Faltlhauser and G. Ruske, "Improving Speaker Recognition Using Phonetically Structured Gaussian Mixture Models," in *EUROSPEECH 2001 Scand. 7th Eur. Conf. Speech Commun. Technol. 2nd INTERSPEECH Event*, 2001, pp. 751–754.
[116] E. G. Hansen, R. E. Slyh, and T. R. Anderson, "Speaker Recognition using Phoneme-Specific GMMs," in *ODYSSEY 2004 - Speak. Lang. Recognit. Work.*, 2004, pp. 179–

184.
[117] A. Park and T. Hazen, “ASR dependent techniques for speaker identification,” in *Internat. Conf. Spok. Lang. Process. (ICSLP 2002)*, 2002, pp. 1337–1340.
[118] T. Bocklet and E. Shriberg, “SPEAKER RECOGNITION USING SYLLABLE-BASED CONSTRAINTS FOR CEPSTRAL FRAME SELECTION,” in *Proc. Internat. Conf. Acoust. Speech, Signal Process. (ICASSP 2009)*, 2009, pp. 4525--4528.
[119] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff, “SVM based speaker verification using a GMM supervector kernel and NAP variability compensation,” in *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, 2006, vol. 1.
[120] W. M. Campbell, J. P. Campbell, D. A. Reynolds, E. Singer, and P. A. Torres-Carrasquillo, “Support vector machines for speaker and language recognition,” in *Comput. Speech Lang.*, 2006, vol. 20, no. 2-3 SPEC. ISS., pp. 210–229.
[121] E. Shriberg, L. Ferrer, S. Kajarekar, A. Venkataraman, and A. Stolcke, “Modeling prosodic feature sequences for speaker recognition,” in *Speech Commun.*, 2005, vol. 46, no. 3–4, pp. 455–472.
[122] L. Ferrer, E. Shriberg, S. Kajarekar, and K. Sönmez, “Parameterization of prosodic feature distributions for SVM modeling in speaker recognition,” in *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, 2007, vol. 4.
[123] M. J. F. Gales, “Maximum likelihood linear transformations for HMM-based speech recognition,” *Comput. Speech Lang.*, vol. 12, no. 2, pp. 75–98, Apr. 1998.
[124] T. Matsui, T. Kanno, and S. Furui, “Speaker Recognition Using HMM Composition in Noisy Environments,” in *EUROSPEECH ’95 Fourth European Conference on Speech Communication and Technology*, 1995, pp. 621–624.
[125] K. R. Farrell, R. J. Mammone, and K. T. Assaleh, “Speaker Recognition Using Neural Networks And Conventional Classifiers,” *IEEE Trans. Speech Audio Process.*, vol. 2, no. 1. pp. 194–205, 1994.
[126] L. P. Heck, Y. Konig, M. K. Sönmez, and M. Weintraub, “Robustness to telephone handset distortion in speaker recognition by discriminative feature design,” *Speech Commun.*, vol. 31, no. 2, pp. 181–192, 2000.
[127] I. Lapidot, H. Guterman, and A. Cohen, “Unsupervised speaker recognition based on competition between self-organizing maps,” *IEEE Trans. Neural Networks*, vol. 13, no. 4, pp. 877–887, Jul. 2002.
[128] B. Yegnanarayana and S. P. Kishore, “AANN: An alternative to GMM for pattern recognition,” *Neural Networks*, vol. 15, no. 3, pp. 459–469, 2002.
[129] H. Altinçay and M. Demirekler, “Speaker identification by combining multiple classifiers using Dempster-Shafer theory of evidence,” *Speech Commun.*, vol. 41, no. 4, pp. 531–547, Nov. 2003.
[130] M. W. Mak, M. C. Cheung, and S. Y. Kung, “Robust speaker verification from GSM-transcoded speech based on decision fusion and feature transformation,” in *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, 2003, vol. 2, pp. 745–748.
[131] L. Rodríguez-Linares, C. García-Mateo, and J. Luis Alba-Castro, “On combining classifiers for speaker authentication,” *Pattern Recognit.*, vol. 36, no. 2, pp. 347–359, Feb. 2003.
[132] N. Brümmer *et al.*, “Fusion of heterogeneous speaker recognition systems in the STBU submission for the NIST Speaker Recognition Evaluation 2006,” *IEEE Trans. Audio, Speech Lang. Process.*, vol. 15, no. 7, pp. 2072–2084, Sep. 2007.
[133] K. R. Farrell, R. P. Ramachandran, and R. J. Mammone, “An analysis of data fusion methods for speaker verification,” in *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, 1998, vol. 2, pp. 1129–1132.

[134] V. Moonasar and G. K. Venayagamoorthy, “A committee of neural networks for automatic speaker recognition (ASR) systems,” in *Proc. Int. Jt. Conf. Neural Networks*, 2001, vol. 4, pp. 2936–2940.
[135] M. Nikzad, K. Movagharnejad, and F. Talebnia, “Comparative Study between Neural Network Model and Mathematical Models for Prediction of Glucose Concentration during Enzymatic Hydrolysis,” *Int. J. Comput. Appl.*, vol. 56, no. 1, pp. 43–48, Oct. 2012.
[136] L. Lazli and M. Boukadoum, “Hidden Neural Network for Complex Pattern Recognition: A Comparison Study with Multi- Neural Network Based Approach,” *Int. J. Life Sci. Med. Res.*, vol. 3, no. 6, pp. 234–245, Dec. 2013.
[137] S. He, R. W. H. Lau, W. Liu, Z. Huang, and Q. Yang, “SuperCNN: A Superpixelwise Convolutional Neural Network for Salient Object Detection,” *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 330–344, Dec. 2015.
[138] J. Ruiz-del-Solar, P. Loncomilla, and N. Soto, “A Survey on Deep Learning Methods for Robot Vision,” Mar. 2018.
[139] S. Li and W. Deng, “Deep Facial Expression Recognition: A Survey,” *IEEE Trans. Affect. Comput.*, 2020.
[140] B. Hasani and M. H. Mahoor, “Facial Expression Recognition Using Enhanced Deep 3D Convolutional Neural Networks,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2017, vol. 2017-July, pp. 2278–2288.
[141] P. Khorrami, T. Le Paine, and T. S. Huang, “Do Deep Neural Networks Learn Facial Action Units When Doing Expression Recognition?,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, vol. 2015-Febru, pp. 19–27.
[142] S. A. Bargal, E. Barsoum, C. C. Ferrer, and C. Zhang, “Emotion recognition in the wild from videos using images,” in *ICMI 2016 - Proceedings of the 18th ACM International Conference on Multimodal Interaction*, 2016, pp. 433–436.
[143] A. E. Omolara, A. Jantan, O. Isaac Abiodun, K. Victoria Dada, H. Arshad, and E. Emmanuel, “A Deception Model Robust to Eavesdropping Over Communication for Social Network Systems,” *IEEE Access*, vol. 7, pp. 100881–100898, Jul. 2019.
[144] R. Collobert and J. Weston, “A unified architecture for natural language processing,” in *Proceedings of the 25th international conference on Machine learning - ICML ’08*, 2008, pp. 160–167.
[145] H. Li, J. G. Ellis, L. Zhang, and S.-F. Chang, “PatternNet: Visual Pattern Mining with Deep Neural Network,” *ICMR 2018 - Proc. 2018 ACM Int. Conf. Multimed. Retr.*, pp. 291–299, Mar. 2017.
[146] Bhandare A., M. Bhide, P. Gokhale, and R. Chandavarkar, “Applications of convolutional neural networks,” *Int. J. Comput. Sci. Inf. Technol*, vol. 7, no. 5, pp. 2206–2215, 2016.
[147] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks,” in *ACM International Conference Proceeding Series*, 2006, vol. 148, pp. 369–376.
[148] Q. Wu, K. Ding, and B. Huang, “Approach for fault prognosis using recurrent neural network,” *J. Intell. Manuf.*, vol. 31, no. 7, pp. 1621–1633, Oct. 2020.
[149] O. Buza, G. Toderean, A. Nica, and A. Căruntu, “Voice signal processing for speech synthesis,” in *2006 IEEE International Conference on Automation, Quality and Testing, Robotics, AQTR*, 2006, vol. 2, pp. 360–364.
[150] D. W. Robinson and R. S. Dadson, “Equal-Loudness Relations, and Threshold of Hearing for Pure Tones,” *J. Acoust. Soc. Am.*, vol. 28, no. 4, pp. 763–764, Jul. 1956.
[151] N. Qian, “On the momentum term in gradient descent learning algorithms,” *Neural*

Networks, vol. 12, no. 1, pp. 145–151, Jan. 1999.
[152] J. Duchi JDUCHI and Y. Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization * Elad Hazan," 2011.
[153] G. Hinton, N. Srivastava, and K. Swersky, "Neural Networks for Machine Learning Lecture 6a Overview of mini---batch gradient descent."
[154] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.
[155] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*. Cambridge: MIT Press, 2016.
[156] A. Kausar, M. Sharif, J. Park, and D. R. Shin, "Pure-CNN: A framework for fruit images classification," in *Proceedings - 2018 International Conference on Computational Science and Computational Intelligence, CSCI 2018*, 2018, pp. 404–408.
[157] NIST, "NIST 2019 Speaker Recognition Evaluation," 2019. .
[158] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild I," *Comput. Speech Lang.*, vol. 60, p. 101027, 2020.
[159] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' Speech Separation and Recognition Challenge: Dataset, task and baselines," *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, vol. 2018-Septe, pp. 1561–1565, Mar. 2018.
[160] Sunil Singh, "WSJ-Scraper: Scraping wall street journal news articles.," 2018. .
[161] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2015, vol. 2015-Augus, pp. 5206–5210.
[162] S. O. Sadjadi, M. Slaney, and L. Heck, "MSR identity toolbox v1. 0: A MATLAB toolbox for speaker-recognition research," *Speech Lang. Process. Tech. Comm. Newsl.*, vol. 1, no. 4, pp. 1–32, 2013.
[163] Mike Brookes, "VOICEBOX: Speech Processing Toolbox for MATLAB," 1997. .
[164] Malcolm Slaney, "Auditory Toolbox," 1998. .
[165] MathWorks, "Deep Learning Toolbox," 2017. .
[166] MathWorks, "Create custom shallow neural network - MATLAB network - MathWorks France." .
[167] Y. I. Cherifi, "Speaker-Recognition-with-GUI," 2017. .
[168] K. K. Paliwal, J. G. Lyons, and K. K. Wójcicki, "Preference for 20-40 ms window duration in speech analysis," in *4th International Conference on Signal Processing and Communication Systems, ICSPCS'2010 - Proceedings*, 2010.
[169] D. Eringis and G. Tamulevičius, "Improving Speech Recognition Rate through Analysis Parameters," *Electr. Control Commun. Eng.*, vol. 5, no. 1, pp. 61–66, May 2014.
[170] K. Meena, K. Subramaniam, and M. Gomathy, "Gender Classification in Speech Recognition using Fuzzy Logic and Neural Network," *Int. Arab J. Inf. Technol.*, vol. 10, no. 5, pp. 477–485, 2013.
[171] Vladimir Vacic, "Summary of the training functions in Matlab's NN toolbox," Riverside, 2015.
[172] A. Oord *et al.*, "WaveNet: A Generative Model for Raw Audio," *ArXiv*, vol. abs/1609.0, 2016.
[173] A. Wong, M. J. Shafiee, B. Chwyl, and F. Li, "Gensynth: A generative synthesis approach to learning generative machines for generate efficient neural networks," *Electron. Lett.*, vol. 55, no. 18, pp. 986–989, Sep. 2019.
[174] K. Kumar *et al.*, "MelGAN: Generative Adversarial Networks for Conditional Waveform Synthesis," *ArXiv*, vol. abs/1910.0, 2019.

[175] G. Saon *et al.*, “English conversational telephone speech recognition by humans and machines,” in *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, 2017, vol. 2017-Augus, pp. 132–136.
[176] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations,” 2020.
[177] A. Kolesnikov *et al.*, “Big Transfer (BiT): General Visual Representation Learning,” *arXiv Comput. Vis. Pattern Recognit.*, 2019.
[178] X. Zhu and M. Bain, “B-CNN: Branch Convolutional Neural Network for Hierarchical Classification,” *arXiv Comput. Vis. Pattern Recognit.*, Sep. 2017.