Deep learning for stuttering detection

Shakeel Ahmad Sheikh

Thèse de Doctorat

Shakeel Ahmad Sheikh

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université de Lorraine
Mention Informatique

École doctorale:
Informatique, Automatique, Electronique-Electrotechnique, Mathématiques et Sciences de L'architecture

Unité de recherche:
Laboratoire Lorraine de Recherche en Informatique et ses Applications
UMR 7503

Soutenue le: Février 24, 2023
Thèse N°:

Apprentissage Profond pour la Détection du Bégaiement

Composition Du Jury

Rapporteur :	**Corinne Fredouille**, Professeure, Université de Avignon, LIA, France
Rapporteur :	**Benjamin Lecouteux**, Professeur, Université Grenoble Alpes, LIG, France
Examinateur (Présidente) :	**Armelle Brun** , Professeure, Université de Lorraine, LORIA, France
Invité :	**Fabrice Hirsch**, Professeur, Universite Paul-Valery Montpellier, Praxiling, France
Invité :	**Md Sahidullah**, Ex Research Scientist, Inria, France
Directeur de thèse :	**Slim Ouni**, Maître de Conférences, Université de Lorraine, LORIA, France

Février, 2023
Nancy, France

UNIVERSITÉ DE LORRAINE

Inria

Loria
Laboratoire lorrain de recherche en informatique et ses applications

Doctoral Thesis

Shakeel Ahmad Sheikh

Dissertation presented in order to obtain the
Doctoral Degree from the University of Lorraine
Computer Science

Doctoral School:
Computer Science, Automation, Electronics, Mathematics and Architectural Sciences (IAEM)

Research Unit:
Lorraine Laboratory for Research in Computer Science and its Applications
UMR 7503

Defended On : February 24, 2023
Thesis N°:

Deep Learning for Stuttering Detection

Jury Members

Reviewer :	**Corinne Fredouille**, Professor, University of Avignon, LIA, France
Reviewer :	**Benjamin Lecouteux**, Professor, University of Grenoble Alpes, LIG, France
Examiner (President) :	**Armelle Brun**, Professor, University of Lorraine, LORIA, France
Invitee :	**Fabrice Hirsch**, Professor, University of Paul-Valery Montpellier, Praxiling, France
Invitee :	**Md Sahidullah**, Ex Research Scientist, Inria, France
Director of thesis :	**Slim Ouni**, Associate Professor, University of Lorraine, LORIA, France

And Say My Lord, Increase Me in Knowledge.

بسم الله الرحمن الرحيم

In the name of Allah, the Entirely Merciful, the Especially Merciful.

I dedicate this thesis especially to my selfless brother **Mr. Tareq Ahmad Sheikh** who has been through a lot in life just to give me the best he could, & to my parents Mr. Bashir Ahmad Sheikh & Haleemah Akhter, & to my sister Ishrata Bashir, & to my nephew Mohammad Yahya.

Tareq Ahmad Sheikh

Acknowledgements

And Verily My Success is Only by Allah.

All glory to the ALLAH, Lord of all the worlds. Prayers & peace be upon our Prophet Muhammad, his family & all of his companions.

First & foremost, with great admiration, I would like to express my deepest gratitude to jury members, my supervisor Prof. Slim Ouni, & to my collaborators Dr. Md Sahidullah, & Prof. Fabrice Hirsch without whom this piece of work would not have been possible. To be sincere, I am unable to find words to describe my gratitude towards them. To continue this tradition, I try to make an attempt by following the verse *"Is there any reward for goodness except goodness?"* (Al-Quran 55.60) in thanking my supervisor & collaborators. With great humility & congeniality, they provide me such an excellent, constructive & encouraging environment which consistently pushed my motivation & confidence in this interdisciplinary stuttering detection research domain. Every now & then, they provide unbiased constructive critical reviews & recommendations, which has essentially helped me to improve more & more over the years. They not only trained me from the concerns of migrating to a new nation, but also taught me to be patient, endure, & lead good scientific endeavors, which is the quintessence of the *LORIA*'s & *INRIA*'s work culture.

With considerable reverence, I would also like to show my sincere & deepest prayers & gratitude to my sagacious & profound mentors especially Dr. Dawood Ashraf Khan, Mr. Khwaja Mohammad Shafi, & Dr. Ajaz Ahmad Bhat, whose role in my life has been phenomenal & inconceivable since I met them. They not only put me in the right direction, but rather, they molded me like their younger brother by guiding & advising me from time to time & by showing great empathy towards me. Ahh!, how can I forget the beautiful

moment of concern & care Dr. Dawood showed to me when I was going through tough time during the first year of my stay in France & during the first year of my PhD thesis. Recollecting my memorable moments with Mr. Shafi at the time of my admission in France, where he selflessly motivated me to explore the new educational adventures in France by offering financial support. Coming to Dr. Ajaz with whom there are no less memorable moments of my life especially in my first journal publication.

Working at Multispeech Inria/Loria, Nancy with such a supportive & pundit group is a once-in-a-lifetime opportunity. I am privileged to work with such a pleasant & courteous cohort. My colleagues at Inria with whom I form close bonds have been my stress relievers & constant companions, without them, I could not have completed my thesis. Special thanks to my lively *C137 office gang* - Louis Abel (the boss of C137 gang), Ashwin D'sa Geet (South Indian photographer), Michel Olvera (Mexican ping-pong champ), Joris Cosentino (French chess champ & an adorable enemy of the boss), Marie-Anne Lacroix (French computational ornithologist), & Mickaëlla Grondin-Verdon (French keyboard warrior), Nasser-Eddine Monir (Moroccan Tajine expert) and , Sewade Ogun (other peoples's food taster) for making it "the most exciting & amusing office", with white board discussions & games such as wor(l)dle & chess. Thanks to my *ping-pong gang* Ali Golmakani, Hugo, Vinicius Ribeiro, Prerak Srivastava, Sofiane Azzouz, Louis Delebecque, Hubert Nourtel. I would like to thank our senior scientists as well including Denis Jouvet, Emmanuel Vincent, Mostafa Sadeghi, Antoine Deleforge, Sheikh Imran, Paul Magron, Yves Laprie, Vincent Colotte, Romain Serizel & other colleagues including Nicolas Furnon, Pierre Champion, Théo Biasutto-Lervat, S&ipana Dowerah, Tulika Bose, Can Cui, & Marina Krémé.

A special thanks to my badminton mates Dr. Shoaib Khan, Dr. Mehdi & cricket mates Karamat, Nabeel, Ali, Talha Shahid, Aatiq Ovais, Khalil, Dr. Tareq Kaleemullah, Aadil Zargar, Shabir, Mohsin, Tatha, Yama, Asgar, etc.

I would like to express my gratitude to Brij Mohan Lal & Seyed Arash Hossein with whom I explored some new cities, had lengthy, sometimes heated, tea-time technical conversations, & even prepared meals together. I am indebted to Louis Abel (the boss of our C137 gang) with whom I share a special bonding and to Mickaëlla Grondin-Verdon who corrected French summary & abstract of this thesis. Thanks to Hélène Cavallini, Elise Urban, Sylvie Hilbert, Delphine Hubert, Anne-Marie Messaoudi, Souad Boutaguermouchet,

Aurore Tranchina, & Sabrina Ferry-Tritz for making my stay comfortable at Inria by providing an excellent administrative support.

I also offer my sincere gratitude to my European based Kashmiri brethren who made me feel home like Kashmir even though living thousands of miles away from our native place. Some of us traveled together to several places, had funny & serious political conversations over special Kashmiri meals such as Yakhni, Kahva, Noon chai, etc. These include but not limited to- Aamir Rashid Najar, Umar Bashir, Mohammad Mujtahid, Mir Junaid, , Saquib Lone, Dr. Anayat Bhat, Dr. Mudasir, Sahil Lone, Najmus Saqib, Zulkarnain Tantray, Furqaan Fayaz, Ashaq Hussain, Haroon Rashid, Masheer ul Haq, Moonis Ali, Mehran Jeelani, Aymen Mir, Jalees, Aadil Zargar, Shabir, Bilal, etc.

Penultimately, I would like to thank my *Khaleels* including Raashid ul Amin, Sheikh Mohsin, & Juanid Ahmad Dar, to my *Sadeeqs* Muneeb Ahmed, Kaiser Shabir, Imran Rather, Faisal Sofi, Abdul Haseeb Arif Mir, Sohaib Najat, Mustafa Rajab, Haseeb ullah, Zulfikar, Dr. Muzamil Wani, Dr. Arshad Hussain Zagoo, to my *Rafeeqs* Faisal Rashid, Mushtaq Ahmad Wani, Waheed Lone, Ijaz ul Haq Kanth, Rizwan uz Zaman Wani, to my *Najiyys* Dr. Ahmet Nicolas Cuny, Ahmet Yavuz hoca for their moral & emotional support. I would also like to offer my prayers & thanks to Mr. Nazir Ahmad Wani, Mr. Mohammad Amin Padder, & to my high school teachers Mr. Ghulam Mohammad & Mr. Abdul Aziz.

I unequivocally state that it won't be fair in expressing my prayers & gratitude to my *Qareen* in words.

Ultimately, this thesis would be incomplete without expressing prayers, devotion, gratitude & heartfelt thanks to my family members including my parents *Mr. Bashir Ahmad Sheikh, Mrs. Haleemah Akhter*, my brother *Mr. Tareq Ahmad Sheikh*, my sister *Mrs. Ishrata Bashir*, my newphew *Mohammad Yahya*, my grandmothers *Mrs Aisha Jan & Mrs Taja Bano* (I lost these two beautiful souls in this PhD journey), my grandfathers *Mr. Ghulam Ahmad Sheikh & Mr Ghulam Mohammad Sheikh*, my aunts *Mrs. Tasleema Akhter, Mrs. Yasmeena Akhter, & deceased Mrs. Maymoona Akhter* (whom I lost in 2014), my neonatal deceased nephew and niece. I also express my sincere thanks to my maternal uncle Abdul Waheed Sheikh, paternal uncles, cousin brothers including Bhat Altaf, Sheikh Ubaid, Sheikh Parvez, Bhat Tahir, Bhat Owais, & to my little cousin princesses Zehra Waheed, Safoora Zahoor, & Adeena Zahoor.

Shakeel Ahmad Sheikh

Résumé

Le bégaiement est un trouble de la parole qui est le plus fréquemment observé parmi les troubles de la parole et se traduit par des *caractéristique spécifiques*. La fastidieuse tâche de détection et d'analyse des schémas de parole des personnes qui bégaiement «*Persons who stutter (PWS)*», dans le but de les rectifier, est souvent traitée manuellement par les orthophonistes et est biaisée par leurs croyances subjectives. De plus, les systèmes de reconnaissance automatique de la parole, «*Automatic speech recognition (ASR)*», ne parviennent pas non plus à reconnaître les bégaiements. Cela empêche les PWS d'accéder à des assistants vocaux tels que Siri, Alexa, etc.

Cette thèse tente de développer des systèmes de détection du bégaiement «*Stuttering detection (SD)*», basés sur l'audio qui réussissent à capturer les différentes variabilités des énoncés de bégaiement telles que les styles d'expression, l'âge, les accents, etc., et apprennent des représentations robustes du bégaiement dans le but de fournir une évaluation juste, cohérente et impartiale de la parole bégayée.

Alors que la plupart des systèmes SD existants utilisent plusieurs classificateurs binaires pour chaque type de bégaiement, nous présentons un système multi-classe unifié nommé *StutterNet* capable de détecter plusieurs types de bégaiement. En abordant le problème du déséquilibre des classes dans le domaine du bégaiement, nous avons étudié l'impact de l'application d'une fonction de perte pondérée et nous avons également présenté une version du modéle *StutterNet* qui est multi-contextuel et multti-branches pour améliorer la détection des classes minoritaires.

En exploitant les informations sur le locuteur et en supposant que les modèles de bégaiement sont invariants par rapport aux méta-données telles que les informations sur le locuteur, nous présentons un système de détection du bégaiement multi-tâches «*Multi-task learning (MTL)*» qui apprend des représentations robustes discriminant le bégaiement et les invariantes par rapport au locuteur.

En raison de la rareté des données non étiquetées, la tâche automatisée de détection du bégaiement est limitée dans son utilisation des modèles d'apprentissages profonds de grande taille lorsqu'il s'agit de capturer différentes variabilités. Nous avons introduit la toute première architecture d'apprentissage auto-supervisé «*Self-supervised learning (SSL)*», dans le domaine de la détection du bégaiement. L'architecture SSL entraîne d'abord un extracteur de caractéristiques pour une tâche de pré-texte en utilisant une grande quantité de données audio non étiquetées et ne contenant pas de bégaiement pour capturer ces différentes variabilités. Puis elle applique l'extracteur de caractéristiques appris à une tâche SD en aval, en utilisant des données audio étiquetées limitées et contenant bégaiement.

Abstract

Stuttering is a speech disorder that is most frequently observed among speech impairments and results in the form of *core behaviours*. The tedious and time-consuming task of detecting and analyzing speech patterns of PWS, with the goal of rectifying them is often handled manually by speech therapists, and is biased towards their subjective beliefs. Moreover, the ASR systems also fail to recognize the stuttered speech, which makes it impractical for PWS to access virtual digital assistants such as Siri, Alexa, etc.

This thesis tries to develop *audio based SD systems* that successfully capture different variabilities from stuttering utterances such as speaking styles, age, accents, etc., and learns robust stuttering representations with an aim to provide a fair, consistent, and unbiased assessment of stuttered speech.

While most of the existing SD systems use multiple binary classifiers for each stutter type, we present a unified multi-class *StutterNet* capable of detecting multiple stutter types. Approaching the class-imbalance problem in stuttering domain, we investigated the impact of applying weighted loss function, and, also presented *Multi-contextual (MC) Multi-branch (MB)* StutterNet to improve the detection performance of minority classes.

Exploiting the speaker information with an assumption that the stuttering models should be invariant to meta-data such as speaker information, we present, an *adversarial MTL SD* method that learns robust *stutter discrimintaive speaker-invariant* representations.

Due to paucity of unlabeled data, the automated SD task is limited in its use of large deep models in capturing different varaibilities, we introduced the *first-ever* SSL framework to SD domain. The SSL framework first trains a feature extractor for a pre-text task using a large quantity of unlabeled non-stuttering audio data to capture these different varaibilities, and then applies the learned feature extractor to a downstream SD task using limited labeled stuttering audio data.

Contents

List of figures

List of tables

List of acronyms

ADV Adversarial learning
ANNs Artificial neural networks
ASR Automatic speech recognition
BN Batch normalisation
CNN Convolutional neural networks
ComParE Computational Paralinguistics ChallengE
CTC Connectionist temporal classification
ECAPA-TDNN Emphasized channel attention, propagation and aggregation TDNN
EMA Electromagnetic articulograph
ED Emotion detection
FC Fully connected
GMM Gaussian mixture models
HMM Hidden markov models
KSoF Kassel state of fluency
k-NN k-nearest neighbor
LDA Linear discriminant analysis
LPCCs Linear prediction cepstral coefficients
MB Multi-branch
MC Multi-contextual
MFCC Mel-frequency cepstral coefficients
MLP Multi-layer perceptron
MTL Multi-task learning
NBC Naive Bayes classifier
NNet Shallow three layered fully connected neural network
PWS Persons who stutter

RNN Recurrent neural networks
SD Stuttering detection
SID Speaker identification
SLP Speech language pathologists
SP Statistical pooling
SSL Self-supervised learning
ST Speech therapists
SVM Support vector machines
TDNN Time delay neural network
UAR Unweighted average recall
UCLASS University college London's archive of stuttered speech
VOT Voice onset time

1. Introduction

*"O my Lord! Open for me my chest. And ease my task for me. And loosen the knot from my tongue (**stuttering**), that they understand my speech."*

Prophet Moses (Taha [25-28])

1.1. Motivation

Over the past few decades, there has been a huge growth in interest in creating systems that mimic how humans comprehend and understand speech in terms of its linguistic content, personality, gender, age, health, emotional, and other paralinguistic signs such as *stuttering*[1] *or disfluency* (Huang et al., 2001; Guitar, 2013). The most practical and expressive methods of human communication are speaking and listening. During speech, the brain performs a series of cognitive processes in terms of selecting, activating, and organization of lexical units and sensorimotors that are required by the speech articulators to deliver the intended speech utterance (Harrington and Tabain, 2013). Normally, the speech utterance is governed by a number of nonlinear processes where the speakers continuously reshape the direction of their discourse. They could pause to consider what to say next, resume a sentence that was already spoken, add something new, or discard the ongoing utterance (Hardcastle et al., 2012). As a result, spoken utterances are often regulated by cycles of *fluent* (lexicaly and grammatically accurate) vs *disfluent* speech (non-lexical items that disrupts the forward flow of speech) (Guitar, 2013). Notice that normal disfluencies are useful in speech production since they can be considered in time during which the speaker can correct or plan the upcoming discourse, however, in cases, like *speech disorder/impairments*,

[1] Stuttering is also called stammering. In this manuscript, we will use stammering, stuttering or disfluency interchangeably

the normal disfluencies do not help the speaker to organize his/her discourse. Contrary to persons without any fluency disorder, persons who stutter (PWS) know what they want to pronounce but are momentarily unable to produce it because of the malfunctioning of their central nervous systems in generating the necessary patterns of motor commands for the production of fluent speech (Smith and Weber, 2017).

Speech impairments make it difficult for PWS to form correct and necessary speech sounds and impair a person's ability to produce these correct sounds that enable them to communicate with others (Guitar, 2013; Ward, 2017). Speech impairment can affect all age groups and usually take the form of *stuttering*, apraxia (the brain is unable to send an exact message to speech muscles), dysarthria (caused by muscle weakness), cluttering (speech being too jerky or too rapid), and so on (Guitar, 2013; Ward, 2017; Duffy, 2019). Among these speech impairments, *stuttering* is the most commonly experienced one (Minnis, 2020). The malfunctioning in sensorimotor regions which are responsible for speech production results in *stuttering* in the form of *core behaviours:* including *repetitions, blocks, and prolongations*, and its course is directed by lexical and psychological aspects (Smith and Weber, 2017; Guitar, 2013). In contrast to other speech impairments, stuttering can be treated if early intervention is provided to assist those PWS in achieving normal fluency (Guitar, 2013). Driven by smart home devices an estimate of 8 billion digital voice assistants in use is to be expected by 2030 (Smith, 2022). On one extreme, a speech technology like this enhances the use of gadgets on a daily basis and make their life easier in certain minority groups such as blind and physically challenged who are unable to access textual mode of devices, and on the other end, for the people with speech impairment problems such an influential technology remains inaccessible.

1.2. Objectives

In conventional stuttering detection (SD) and therapy sessions, the speech therapists or speech-language pathologists manually analyze the PWS' speech (Nöth et al., 2000). The speech therapists observe and monitor the speech patterns of PWS to rectify them (Guitar, 2013). This convention of SD is very laborious and time-consuming and is also inclined toward the idiosyncratic belief of speech therapists. Moreover, speech therapists might not be available all the time. In addition, the automatic speech recognition systems (ASR) are

working fine for normal fluent speech, however, they are unsuccessful in recognizing the stuttered speech (Mitra et al., 2021), which makes it impractical for PWS to easily access virtual assistants such as Cortona, Alexa, Siri, etc. As a result, interactive automatic SD systems that provide an impartial objective, and consistent evaluation of stuttering speech are strongly encouraged. We believe that these automatic stuttering detection systems will also provide a fair, consistent, and unbiased assessment of stuttered speech which later on can also be adapted towards voice digital assistants for PWS.

Despite the fact of having numerous potential applications, very little research attention has been given to the domain of SD and measurement which is the main focus of this thesis. The detection and identification of stuttering events can be quite a difficult and complex problem due to several variable factors including language, gender, age, accent, speech rate, etc. Most of the earlier work in SD methods are based either on ASR systems (Alharbi et al., 2018, 2020) or language models (Zayats et al., 2016a; Chen et al., 2020). These methods are two-stage approaches that first convert the acoustic speech signals into their corresponding spoken textual modality, and then detect stuttering by the application of language models. Even though this ASR-based two-stage approach for identifying stuttering has shown promising results, the dependence on the ASR unit makes it computationally costly and prone to error. Moreover, the adaption towards the ASR task results in the possible loss of stuttering relevant information such as prosodic, emotional, and other paralinguistic information. In summary,

We try to develop audio based stuttering detection systems which successfully capture different variabilities from stuttering audio utterances and helps to distinguish between fluent, and various disfluent speech categories such as blocks, prolongations, repetitions, and interjections.

1.3. Contribution summary

Stuttering can be expressed in multiple modalities including text, audio, visual, and bio-signals, we mainly focus on developing *acoustic* modality based stuttering classification systems by addressing some of the challenges stuttering domain is facing in terms of speaker variabilities, data availability, class imbalance, etc., when approached from deep learning paradigm. This PhD manuscript presents my contribution in proposing a new model called *StutterNet*, its variants, and their extensive evaluation on stuttering SEP-28k dataset curated

recently. This thesis also presents my contribution about the adoption of self-supervised learning (SSL) framework in stuttering detection domain by exploiting Wav2Vec2.0 model trained on large amounts of non-labelled data.

***StutterNet*:** The first main contribution of this thesis is the development of *StutterNet*, a time delay neural network based stuttering detection system, which is a multi-class SD classifier capable of detecting several stuttering types using one single system on a large set of speakers, unlike state-of-the-art SD systems which are mostly binary SD classification systems, and only considered few stuttered speakers. The proposed *StutterNet* first computes the Mel-frequency cepstral coefficient (MFCC) features from raw audio samples, which are then passed to the TDNN to learn and capture the temporal context of various types of disfluencies/stuttering classes. The proposed *StutterNet* has been evaluated on University college London's archive of stuttered speech (UCLASS) and SEP-28k datasets. Moreover, we, also consider *fluent* speech samples of PWS that greatly effects the stuttering classification systems unlike state-of-the-art SD systems, where they focus mostly within the stuttering classes.

Multi-branch (MB) *StutterNet*: Due to inherent nature of stuttering, obtaining class-balanced stuttering datasets is both extremely challenging and expensive. Deep neural network classifiers are typically biased towards the majority class when trained on highly class-imbalanced datasets, which leads to subpar modeling of minority classes. Moreover, the stuttering data is very limited, and, is collected usually in clean clinical environments, thus limits the adoption of advanced deep learning methods to stuttering detection, and SD can further be greatly effected by noisy environments. To contribute to the previously mentioned issues, we proposed MB *StutterNet* (addressing *class-imbalance*), which comprises a base encoder ($\mathcal{E}$) followed by two parallel branches referred as *DisfluentBranch* ($\mathcal{D}$) and *FluentBranch* ($\mathcal{F}$). The embeddings generated by ($\mathcal{E}$) are simultaneously passed to both the *FluentBranch* and *DisfluentBranch*, where the *FluentBranch* is trained to distinguish between fluent and disfluent samples, and the *DisfluentBranch* is trained to differentiate and classify the disfluent sub categories. Moreover, we also contributed by providing extensive analysis on employing class-balanced training by applying more weights to minority stuttered classes in the loss function, which in turn greatly improves their detection performance.

To further test our proposed *StutterNet* and MB *StutterNet* in real-life environments, we extensively evaluate them in simulated noisy environmental conditions by employing data augmentation techniques such as noise, babble, music, and reverberation to the stuttered speech samples. The advantage of applying data augmentation to stuttered speech is two-fold. First, it increases the amount of data in stuttered speech, thus makes it suitable for advanced deep learning methods. Second, it can simulate real-time noisy environmental conditions, thus making the proposed SD systems robust to such factors.

Multi-contextual (MC) *StutterNet*: In addition, we also found previously, that the context optimized for one category are not good for another stutter class, so exploiting this fact, we propose a MC StutterNet, which is a *multi-contextual* multi-branched time delay neural network based architecture for SD. The multi-contextual framework is based on the way humans perceive speech. In the *cochlea*, the input acoustic speech signal is partitioned into several frequency bands so that the *information* in each band can be *filtered* independently and thus processed in *parallel* by the human brain.

Multi-task and adversarial learning in stuttering detection: From recent years, the field of deep learning-based stuttering detection is evolving continuously. Yet all current systems focus only on the stuttering learning method without considering any auxiliary tasks. Unfortunately, this is not the way, how we humans perceive speech; instead, we simultaneously decode the main task with various meta contents like speaker traits, linguistic content, emotion, etc., in a multi-tasking fashion. Motivated by this, we exploited speaker characteristics in stuttering detection. The third contribution of this thesis is to learn and exploit these speaker traits in combination with the primary stuttering detection task in a multi-tasking learning paradigm. As the stuttering speech detection task should not be dependent on a specific set of speakers, an alternative approach, is to assume that a robust acoustic stuttering representation needs to be invariant to secondary tasks, particularly speaker recognition. We also contributed to this by proposing an adversarial multi-task (MTL) learning method that learns latent representations that are speaker invariant but at the same time are stutter discriminative.

Introducing self-supervised learning to stuttering detection: Due to a paucity of *unlabeled* data, the automated stuttering detection task is limited in its use of large deep models. As stated in Appendix A, several stuttering datasets, such as SEP-28k (Lea et al., 2021), UCLASS (Howell and Sackin, 1995), LibriStutter (Kourkounakis et al., 2020), and FluencyBank (Lea et al., 2021), have been curated throughout the time to examine various SD models, however, the improvement is bounded most likely because of the small sized and unlabeled stuttering datasets, which are unable to encapsulate several variable factors such as linguistic content, speaking style, gender, age, accent, speech tempo, recording conditions, and others. Our contribution to this challenge is that we introduced the *first-ever* self-supervised learning framework to stuttering detection domain. The self-supervised learning framework first trains a feature extractor for a pre-text task using a large quantity of *unlabeled* non-stuttering audio data to capture different speaking styles, accents, age group, linguistic content, paralinguistic content, prosody, environmental conditions, etc, and then applies the learned feature extractor to a downstream stuttering detection task using limited *labeled* stuttering audio data.

1.4. Publications

The below mentioned papers serve as a foundation of this thesis, and are presented in order in which they occur in this thesis.

1.4.1. Peer-reviewed

1. **Shakeel Ahmad Sheikh**, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. ". Advancing stuttering detection via data dugmentation class-balanced loss and multi-contextual deep learning". *Journal of IEEE Biomedical Health Informatics*, (2022).
2. **Shakeel Ahmad Sheikh**, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Machine learning for stuttering identification: Review, challenges and future directions". *Journal of Neurocomputing*, 514 (2022), pp 385-402.
3. **Shakeel Ahmad Sheikh**, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "StutterNet: Stuttering detection using time delay neural network". *In Proc. 29th European Signal Processing Conference (EUSIPCO)*, (2021), pp 426-430.
4. **Shakeel Ahmad Sheikh**, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Robust

stuttering detection via multi-task and adversarial learning ". *In Proc. 30th European Signal Processing Conference (EUSIPCO)*, (2022).

5. **Shakeel Ahmad Sheikh**, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "End-to-end and self-supervised learning for ComParE 2022 stuttering sub-challenge". *In Proc. 30th ACM International Conference on Multimedia*, (2022), pp 7104-7108.

1.4.2. Under revision

1. **Shakeel Ahmad Sheikh**, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Introducing ECAPA-TDNN and Wav2Vec2.0 embeddings to stuttering detection". *Under review in International Journal of Speech Technology*, (2022).

1.5. Thesis outline

This PhD thesis is situated at the intersection of deep learning applied to stuttering problem in the acoustic modality. The outline of this thesis is as follows: the first chapter discusses the speech disorder problem in general with a focus on *stuttering* from multiple views. The third chapter reviews the stat-of-the-art work in stuttering detection domain. The next three chapters details about the *StutterNet* and its variants. Finally, the chapter seven introduces the SSL framework to stuttering detection with an extensive evaluation on SEP-28k and KSoF dataset. The final thoughts towards the future research developments are presented in the last chapter. A more detailed summary of the chapters is given below and are based on the peer-reviewed and under revision papers highlighted in the box after the outline of each chapter:

Chapter 2: In this chapter, we discuss the general *speech disorder* problem with a focus on *stuttering*. We first begin by discussing the *stuttering* problem and its categories, and its impact on the real life of PWS. Then, we provide a discussion on the characteristics of stuttered speech from neurogenic, acoustic and phonetic point of views. Lastly, we give a detailed motivation about the need of *automatic stuttering detection systems* after pointing out the limitations and drawbacks of the conventional stuttering assessment.

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Machine learning for stuttering identification: Review, challenges and future directions". *Journal of Neurocomputing*, 514 (2022), pp 385-402 (peer-reviewed).

Chapter 3 In this chapter, we walk through the foundational ideas and fundamental concepts of the fields including datasets, speech processing and deep learning, which are essential to our work. Firstly, we present a brief discussion on speech production followed by acoustic features employed mostly in stuttering detection domain. Next, we present some *challenges*, the stuttering detection domain is currently facing, and, in the subsequent chapters, we try to address some of these challenges. Following the discussion on feature extraction, we provide a discussion on various deep learning methods that we adopted towards stuttering detection. Lastly, we review the state-of-the-art SD systems with their possible limitations.

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Machine learning for stuttering identification: Review, challenges and future directions". *Journal of Neurocomputing*, 514 (2022), pp 385-402 (peer-reviewed).

Chapter 4: This chapter gives a detailed discussion about our proposed *StutterNet* model, a time delay neural network single multi-class classifier capable of detecting stuttering and its types efficiently. Firstly, we discuss its architecture, followed by the dataset details used to evaluate the effectiveness of the proposed method. Finally, we comment on the results achieved by the proposed *StutterNet*.

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "StutterNet: Stuttering detection using time delay neural network". *In Proc. 29th European Signal Processing Conference (EUSIPCO)*, (2021), pp 426-430 (peer-reviewed).

Chapter 5: This chapter addresses the class-imbalance problem in stuttering detection domain via *weighted loss function* and *multi-branch* training schemes. In addition, this also investigates the effectiveness of *data augmentation* when applied to stuttering detection problem. Moreover, we found previously in Chapter 4 that the contexts optimized for one stutter class are not good for other class. Driven by this fact, this chapter also discusses the proposed MC *StutterNet*, which exploits *variable contexts* from the stuttered speech utterance. The recently released first-ever public labeled stutter dataset is used to assess the proposed

framework's overall performance.

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Advancing stuttering detection via data dugmentation class-balanced loss and multi-contextual deep learning". *Journal of IEEE Biomedical Health Informatics*, (2022) (peer-reviewed).

Chapter 6: In this chapter, we introduce and discuss an *adversarial* multi-task learning framework applied to stuttering detection problems to learn robust latent *stutter representations* that are stutter discriminative while being *speaker-invariant*. The overall methodology of the proposed framework is provided and evaluated on the newly released first-ever publicly labeled stutter dataset.

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Robust stuttering detection via multi-task and adversarial learning". *In Proc. 30th European Signal Processing Conference (EUSIPCO)*, (2022) (peer-reviewed).

Chapter 7: Finally, the last chapter discusses the employment of first-ever *self-supervised learning framework* to stuttering detection problem. We first, describe the motivation of applying contextual embeddings to SD, learned in an SSL setup on some pre-text auxiliary task. We mainly focus on speaker and contextual embeddings where the SSL models are pre-trained on massive datasets with the hypothesis in capturing various speaking styles and prosodic information which are extremely important to stuttering detection. We also discuss which layers of SSL model captures more stutter-related information. In addition, this chapter also discusses the evaluation of pre-trained SSL models in cross-language stuttering detection platform. To this, we evaluated the *English* language pre-trained SSL model on the *German* language stutter dataset provided to us in the ComParE challenge organised during ACM workshop (Schuller et al., 2022). Lastly, we discuss the results from the SSL based stuttering detection, and showed empirically that the contextual embeddings achieve a considerable gain and surpasses in all the disfluent classes over the state-of-the-art and proposed methods discussed in previous chapters.

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Introducing ECAPA-TDNN and Wav2Vec2.0 embeddings to stuttering detection". *Under review in International Journal of Speech Technology*, (2022) (under review).
Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "End-to-end and self-supervised learning for ComParE 2022 stuttering sub-challenge". *In Proc. 30th ACM International Conference on Multimedia*, (2022), pp 7104-7108 (peer-reviewed).

Chapter 8: Finally in this chapter, we conclude our contributions and provide some of our reflections on the future directions that the studies conducted for this thesis has initiated.

2. Stuttering problem

"The earlier that the intervention starts for your child, the quicker that the disorder can be addressed and corrected."

Unknown

2.1. Speech disorders

Voice disorders[1] known as speech disorders or speech impairments occur when a person struggles to produce and articulate the typical spoken sounds needed to interact with others (Guitar, 2013; Duffy, 2019). These voice disorders can take the form of dysarthria, apraxia, stuttering, cluttering, lisping, and so on (Duffy, 2019; Ratner and MacWhinney, 2018; Guitar, 2013; Ward, 2008; Kehoe and Contributors, 2006). Dysarthria is defined as a speech disorder caused by the muscle weakness (including face, lips, tongue, and throat) controlled by the nervous system. The patients with dysarthria produce slurred or mumbled sounds with aberrant speech patterns, such as flat intonation or very low or fast speech rate, which makes their speech very difficult to comprehend (Duffy, 2019). Cluttering is characterized by a patient's speech being too jerky, too rapid, or both. Persons with cluttering usually exclude/collapse most of the syllables, or aberrant rhythms or syllable stresses, and also contain excessive amounts of interjections such as *so*, hmm, *like*, *umm*, etc (Ward, 2008). Apraxia is defined as a speech disorder when the neural path between the nervous system and the muscles responsible for speech production is obscured or lost. The persons with apraxia knows what they want to speak, but can not speak because the brain is

[1]This chapter is based on the following article:

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Machine learning for stuttering identification: Review, challenges and future directions". *Journal of Neurocomputing*, 514 (2022), pp 385-402 (peer-reviewed).

unable to send an exact message to the speech muscles which can articulate the intended sounds, despite the fact speech muscle movements are working fine (Duffy, 2019; Kehoe and Contributors, 2006). Lisping speech disorder is defined as the incapability of producing sibilant consonants (*z or s*) correctly. The sibilant sounds are usually substituted by *th* sounds. For example, the persons with lisping speech disorder would pronounce the word *lisp* as *lithp* (Duffy, 2019). Stuttering speech impairment is different than the other speech disorders because it can be cured if early intervention is being made to help the PWS develop normal fluency (Guitar, 2013). Of all these speech impairments, stuttering is the most common and predominant one[2].

2.2. Stuttering speech disorder

Stuttering is a neurodevelopmental speech disorder, caused by the failure of speech sensorimotors. Stuttering - also known by the name stammering/disfluency - commences when neural connections supporting language, speech, and emotional functions are quickly changing (Duffy, 2019; Ward, 2008; Smith and Weber, 2017), and is characterized by *core behaviors* that usually take the form of involuntary stoppages, repetition, and prolongation of sounds, syllables, words or phrases. Stuttering can be described as an abnormally and persistent duration of stoppages in the normal forward flow of speech (Guitar, 2013). These speech abnormalities are generally accompanied by unusual behaviors like head nodding, lip tremors, quick eye blinks, unusual lip shapes, etc (Riva-Posse et al., 2008). Fluency can be defined as the capacity to produce speech without any effort, at a normal rate (Starkweather, 1987). A fluent speech requires linguistic knowledge of the spoken language and a mastery of the message content. Concerning physiological aspects, a precise respiratory, laryngeal and supraglottic control movement is necessary to maintain fluency (Adams, 1974). When all these conditions are not met, speech disorder (stuttering) can emerge. They can take the form of silent or filled pauses, repetitions, interjections, revisions (content change or grammatical structure or pronunciation change), and incomplete phrases (Roberts et al., 2009). Generally, normal speech is made up of mostly the fluent and some disfluent parts. Notice that normal disfluencies are useful in speech production since they can be considered in time during which the speaker can correct or plan the upcoming discourse. In some cases,

[2] Speech disorders

like stuttering, disfluencies do not help the speaker to organize his/her discourse. Indeed, contrary to persons without any fluency disorder, PWS know what they want to pronounce but are momentarily unable to produce it (Organization et al., 1977; Guitar, 2013).

2.2.1. Stuttering categories

Stuttering can broadly be classified into two types (NIDCD, 2015):

- *Developmental Stuttering:* This stuttering is the most common one and it usually occurs in the learning phase of the language, *i.e. between two and seven* years of language development. Recent researches conclude that developmental stuttering is multifactorial trouble including neurological and genetic aspects (Etchell et al., 2018; Drayna and Kang, 2011). Indeed, functional magnetic resonance imaging studies show anomalies in neural activity before the speech, *i.e.* during the planning stages of speech production (Vanhoutte et al., 2016). Furthermore, an atypical activation in the left inferior frontal gyrus and right auditory regions has been highlighted (Neef et al., 2015; Belyk et al., 2015). Concerning the genetic aspects, Riaz et al. (2005) observe an unusual allele on chromosom 12 by PWS. Drayna and Kang (2011) identify 87 genes that could be involved in stuttering, including one called GNPTAB, which was significantly present in a lot of PWS.
- *Neurogenic Stuttering:* This stuttering can occur after the head trauma, brain stroke, or any kind of brain injury. This results in disfluent speech because of the incoordination of the different regions of the brain which are involved in speaking[3]. Even if neurogenic stuttering is rare, it can be observed in children and adults regardless of their age.

2.2.2. Impact of stuttering

Globally, stuttering concerns 1% of the world's population and its incidence rate is between 5% and 17% (Yairi and Ambrose, 2013). The difference between the prevalence and incidence rates can be explained by the fact that developmental stuttering disappears in 80% of the cases before adulthood either without any intervention or thanks to the speech therapy. Thus, about 70 million people suffer from this disorder which affects four times

[3] https://www.nidcd.nih.gov/health/stuttering

males than females (Yairi and Ambrose, 2013). As considered by the non-stuttering persons, the disfluency affects the flow of speech only, however for PWS, it is greater than that. Several studies show that PWS are ignored, teased, and/or bullied by normo-fluent people (Iverach et al., 2016). The PWS is usually rated less popular than their non-stuttering peers and less likely to be considered leaders (Iverach et al., 2016). According to the national stuttering association (NSA, 2009), 40% of the PWS have been repudiated school opportunities, promotions or job offers and it affects relationships. The data should be assessed in close conjunction with the fact that 85% of businessmen consider stuttering as a negative element during a job interview and prefer offering work that does not require a customer contact (Klein and Hood, 2004). All these elements explain that PWS develop social anxieties and negative feelings (fear, shame, etc.,) when they have to speak in public (Blood and Blood, 2016).

Stuttering appears to be complex and several factors that lead to stuttering include stress, delayed childhood development, and speech motor control abnormalities, as there is a strong correlation between stress, anxiety, and stuttering (Guitar, 2013). Indeed, disfluencies are more frequent in stress or anxiety conditions, in dual tasks including speech and another cognitive charge, and when they speak fast. Conversely, PWS produce fewer disfluencies when they sing in unison or speak with an altered auditory feedback (Antipova et al., 2008). In a recent study, Smith and Weber (2017) postulated the multifactorial dynamic pathways theory, where they asserted that stuttering occurs because of the failure of the central nervous system in generating the necessary patterns of motor commands for fluent speech. Thus, stuttering shows impairment in sensorimotor processes that are responsible for speech production, and its orientation throughout the life of PWS is strongly affected by the linguistic and emotional aspects.

2.3. Characteristics of stuttered speech

In this section, we discuss the stuttering problem from the neurogenic and acoustic point of view. This section also discusses the various acoustics properties of stuttered speech and describes how the stuttered speech impacts the various characteristics like formant transitions, pitch, and Voice onset time (VOT).

2.3.1. Neurogenic view

Most brain scan studies show that, during fluent speech and silent rest, there is no difference in the cerebral activity between persons who stutter and normal fluent speakers, however, during stuttering, there is a dramatic change in the cerebral activity (Ingham et al., 1996). The right-hemisphere of the brain areas which are normally not active during normal speech become active, and the left-hemisphere areas, which are active during normal speech become less active (Ingham et al., 1996; Kehoe and Contributors, 2006). It has also been found that there is an under-activity in the central auditory processing area of the brain. Kehoe and Contributors (2006) suggests that adult persons who stutter have an inability to integrate somatic and auditory processing. A brain study by Foundas et al. (2001) found that PWS has rightward asymmetry in planum temporale, i.e, their right planum temporale is larger than their left planum temporale, on the contrary, normal people's planum temporale is larger in the left side of their brains. Studies based on motor data have been carried out about stuttering. The PWS also show over-activity in the speech motor control area, in particular in the *left caudate nucleus* area (Smith and Weber, 2017; Kehoe and Contributors, 2006). Conture et al. (1977, 1985) observe inappropriate vocal folds abductions and adductions which lead to anarchic openings and closure of the glottis. Concerning the supraglottic level, Wingate (1969) hypothesizes that stuttering is not a sound production problem but a coarticulation one. He theorizes that disfluencies occur during a fault line, which corresponds to the interval when muscular activity due to a sound that has been produced is going off and muscular movements for the following sound are going on. However in a recent study, Didirková and Hirsch (2020) show, thanks to Electromagnetic articulograph (EMA) data, that stuttering is not a coarticulation trouble. They found correct coarticulatory patterns in the fluent and stuttered utterances. Furthermore, another study based on articulatory observation, notes that stuttered disfluencies and non-pathological disfluencies do have common characteristics. However, stuttered disfluencies and non-pathological disfluencies produced by PWS present some particularities, mainly in terms of retention and anticipation, and the presence of spasmodic movements (Didirkova et al., 2020). PWS tend to develop strategies allowing them to avoid sounds or words which can result in disfluency; such strategies consist in using paraphrases or synonyms instead of the problematic segment (Ward, 2008).

2.3.2. Acoustic and phonetic view

Concerning stuttering-like disfluencies such as blocks, prolongations, and repetitions as shown in Figure 3.2, some works try to link the locus of disfluencies and phonetic proprieties. Jayaram (1983); Blomgren et al. (2012); Didirkova (2016) indicate that unvoiced consonants are more disfluent than their voiced counterparts. Furthermore, Blomgren et al. (2012) notices that disfluencies are more frequent at the beginning of an utterance or just after a silent pause. Moreover, Didirkova (2016) observes an important inter-individual variability concerning sounds and/or phonetic features which are the most disfluent.

Acoustic analysis has been carried out about stuttering, including speech rate, stop-gap duration, vowel(V)-consonant(C) transition duration, fricative duration, VOT, CV transition duration, vowel duration, formants, glottis constriction, a sharp increase in articulatory power and closure length elongation before the speech segment is released (Zebrowski et al., 1985). Dehqan et al. (2016) studied the second formant (F2) transitions of fluent segments of Persian speaking PWS. They concluded that the F2 frequency extent transitions are greater in stuttering speakers than in normal fluent ones. The PWS takes longer to reach vowel steady state, but the overall F2 formant slopes are similar for both stuttering speakers and normal ones (Dehqan et al., 2016). The PWS generally exhibit slower speaking rates when compared to normal speakers. Several other studies have investigated the CV formant transitions in stuttered speech. Yaruss and Conture (1993) examined the F2 transitions of children who stutter on syllable repetitions and found no aberrant F2 transitions. However, Robb et al. (1998) analyzed the fluent speech segments of PWS and showed that F2 fluctuations are longer for voiced and voiceless stops than normal speakers. In a different study by Chang et al. (2002), where 3-5 year aged children were analyzed in the picture-naming task. The results showed that disfluent children produced smaller fluctuations of F2 transitions between the alveolar and bilabial place of articulations than did fluent children, and the overall degree of CV coarticulation is no different among stuttering and control groups. Subramanian et al. (2003) analyzed the F2 frequency fluctuations of voiceless stops and revealed that near the onsets of CV, the stuttering children exhibited smaller F2 changes than the normal speakers. Blomgren et al. (1998) found that PWS and normal speaker in /CVt/ token exhibit no differences in the F1 (average). The stutters show significantly lower F2 in /Cit/ tokens than the control groups. The formant spacing for

/i/ is significantly lower in PWS than fluent persons (Blomgren et al., 1998). Hirsch et al. (2008) conducted a study by analyzing the first two formants (vowel space) in CV sequences between the stuttered and normal groups. At a normal speaking rate, the stuttering group shows a reduction in the vowel space, contrary to the fast speaking rate, where, the latter shows no noticeable deviation.

VOT is the duration of time between the release of a stop consonant and the beginning of vocal fold vibrations (Guitar, 2013). Healey and Ramig (1986) showed that for voiceless stops, chronic stuttering exhibits longer VOT when compared with normal fluent persons. They showed that consonant and vowel duration were longer only in real-world phrases like *take the shape* in contrast with nonsense phrases like *ipi saw ipi* (Healey and Ramig, 1986). Hillman and Gilbert (1977) also found that the PWS reveals longer VOT for voiceless stops than for fluent persons. Adams (1987) found that not only voiceless stops exhibit longer VOT in PWS, but also, voiced stops display longer VOT than non-stuttering persons. No significant VOT differences have been found in control and PWS groups (Watson and Alfonso, 1982). In another study by Jäncke (1994), the PWS show strong variability in repeated production of VOT for voiceless stops, however, there is no significant difference between the two groups. In a study carried out by (De Nil and Brutten, 1991), it shows that the stuttering children exhibit more variability in VOT than their counterparts. Celeste and de Oliveira Martins-Reis (2015) also found that the stuttering group shows higher VOT for voiceless stops. Brosch et al. (2002) examined the VOT in stuttering children. They found that the children with severe stuttering have higher values of VOT. Borden et al. (1985) examined the VOT of fluent utterances from PWS. Their study showed that the fluent utterances of PWS exhibit no statistical differences in VOT and are within the limits of normal groups.

Fosnot and Jun (1999) examined the prosodic characteristics of PWS and fluent children. They found that the variability in pitch is greater in the stuttered group, but slightly differs from the normal group. In another study, it has been shown that the normal group and PWS show the same patterns in f0 deviation (Ramig and Adams, 1981). The stuttering occurs less significantly in low-pitched conditions as compared to high-pitched condition (Ramig and Adams, 1981)

2.4. Conventional stuttering assessment

In conventional stuttering assessment, the Speech language pathologists (SLP) or Speech therapists (ST) manually analyze either the PWS' speech or their recordings (Nöth et al., 2000; Guitar, 2013). The stuttering severity is usually measured by taking the ratio of disfluent words/duration to the total words/duration (Guitar, 2013). The most conventional speech therapy sessions involve helping the PWS observe and monitor their speech patterns to rectify them (Guitar, 2013). The speech therapeutic success rate recoveries have been reported to be 60-80% when dealt within the early stage (Saltuklaroglu and Kalinowski, 2005). However, this convention of detecting stuttering severity and its improvement due to therapeutic sessions is very demanding and time-consuming and is also biased and prejudiced towards the subjective belief of SLPs. Due to the nature of stuttering, its therapeutic sessions are very intense course, that usually, extend to several months (several years in some cases), which necessitates PWS to see the SLP regularly (Roberts, 2011). Usually, the speech therapy sessions are private and are very expensive, thus making it unaffordable to some PWS. Thus, it is important to develop interactive automatic SD systems.

Moreover, the ASR are working well for the fluent speech, but they fail to recognize the stuttered speech. So, it would not be feasible for a PWS to easily access virtual assistant tools like Alexa, Siri, etc[4]. The SD may help in adapting the virtual assistant tools for disfluent persons. Therefore, developing interactive SD systems are the need for an hour which provides an objective and consistent measurement of the stuttered speech. Consequently, with the recent developments in natural language processing, machine learning, and deep learning, it became a reality to develop smart and interactive SD and therapy tools. The stuttering detection systems can later on be used in an ASR pipeline to first detect the stuttered samples, and then either correct these stutter speech frames or removing them completely from an utterance.

2.5. Automatic stuttering detection

Stuttering speech disorder reflected by its *core behaviours*: prolongations, repetitions, and blocks impact the acoustic properties of speech which can help to discriminate from

[4] For people who stutter, the convenience of voice assistant technology remains out of reach

fluent voice as shown in Figure 3.2. Most of the computer based SD methods are based either on ASR systems (Alharbi et al., 2018, 2020) or language models (Zayats et al., 2016a; Chen et al., 2020). These methods are two stage approaches that first convert the acoustic speech signals into their corresponding spoken textual modality, and then detects stuttering by the application of language models. Even though this ASR based two stage approach of identifying stuttering has shown promising results, however, the dependence on ASR unit makes it computationally costly and prone to error. Moreover, the adaptation towards ASR task results in possible loss of stuttering relevant information such as prosodic and emotional content. In recent decades, the applications of deep learning have grown tremendously in speech recognition (Nassif and *et al.*, 2019), speaker recognition (Latif et al., 2020a), speech synthesis (Ning et al., 2019a), emotion detection (Akçay and Oğuz, 2020), voice conversion (Sisman et al., 2020), voice disorder detection (Verde et al., 2018) including Parkinson's disease detection (Almeida et al., 2019) and dysarthric speech detection (Narendra and Alku, 2019; Kodrasi, 2021). However, the application of deep learning in SD is limited. Due to the presence of acoustic cues in the stuttered embedded speech as shown in Figure 3.2, the deep learning models can be used to exploit these acoustic cues in the detection and identification of stuttering events. Most of the SD existing methods employ spectral features such as MFCCs and Linear prediction cepstral coefficients (LPCCs) or their variants that capture these formant-related information. Other spectral features such as pitch, zero-crossing rate, shimmer, and spectral spread are also used. Finally, those features are modeled with statistical modeling methods such as hidden Markov model, support vector machine, Gaussian mixture model, etc.

2.6. Summary

In this chapter, we discussed the underlying principles of stuttering, which makes it easier to grasp the subjects covered in the next chapters. First, speech impairments in general were discussed. Of these speech impairments, stuttering is the most predominant one. Next, we explained the characteristics of stuttering speech disorder from neurogenic, phonetic and acoustic point of view and its impact on PWS and SLP. Finally, we presented the conventional way of stuttering assessment and its limitations. This thesis provides a deep learning framework for SD, where we first describe the state-of-the-art deep learning

models followed by our contributions to automatic SD.

3. Background and state of the art

"Stuttering is no simple speech impediment. It is a complicated disorder which has both physical and emotional aspects."

Malcolm Fraser

There is a multitude of research content available in each of the domains pertinent to this thesis. It makes use of prior knowledge from speech processing, stuttering, and deep learning. In this chapter, we discuss specific background details of the state-of-the-art features and techniques proposed in this thesis, and we draw on existing literature to paint a complete picture of the problem statement and how the proposed solutions are considered by the speech disorder community in particular stuttering community[1].

3.1. Introduction

Research interest towards stuttered speech is becoming more and more prominent. Building a trustworthy stuttered speech classification system is still difficult owing to its complicated nature. Stuttering detection (SD) is a complex interdisciplinary problem that involves speech processing, signal processing, neuroscience, psychology, pathology, and machine learning. The recent advancements in the machine and deep learning have significantly transformed the speech domain. However, in SD, it has not been explored eminently. A myriad number of research works for SD (in terms of acoustic feature extraction and classification methods) have been carried out with a focus on developing automatic

[1]This chapter is based on the following article:

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Machine learning for stuttering identification: Review, challenges and future directions". *Journal of Neurocomputing*, 514 (2022), pp 385-402 (peer-reviewed).

tools for its detection and identification. Most of the earlier work detects and identifies stuttering either by language models (Zayats et al., 2016b; Chen et al., 2020) or by ASR systems (Alharbi et al., 2018, 2020), which first converts the audio signals into their corresponding textual form, and then by the application of language models, detects or identifies stuttering. However, in this chapter we focus on SD only from the acoustic modality point of view, and discuss the underlying speech processing principles, and state-of-the art SD classification methods.

3.2. Speech processing principles and tools

First of all let us go through some of the underlying principles and tools of speech processing that underpin the proposed methods for SD. We commence with a signal, a fundamental representation of speech, and how it is transformed to meaningful relevant feature space based on the phonetic and physiological factors.

3.2.1. Fundamentals of speech processing

Vocal Tract: The vocal tract as shown in Figure 3.1 is the physiological system that generates speech extended from the lungs to the lips and nose. The larynx divides the vocal tract into two physical regions: the lower sublaryngeal area and the upper supralaryngeal region (Huang et al., 2001; Kehoe and Contributors, 2006; Hardcastle et al., 2012). The diaphragm, lungs, and trachea (also called the windpipe) make up the vocal tract's sublaryngeal region. The air flows out of the lungs and encounters the vocal folds, which are a pair of flap-like structures in the larynx. When the vocal folds are kept at an intermediate tension, neither too close or too far apart, the movement of the air causes ripples along their length and makes them to vibrate, resulting in *voiced* speech. On the contrast, when the vocal folds are maintained sufficiently apart so that air can freely flow through them, they do not vibrate, resulting in voiceless speech. This is visible in the speech signal as aperiodic portions that appear to be random noise and are referred to as unvoiced areas. The supralaryngeal area, which includes the mouth and nasal cavities, is crucial in defining the exact nature and quality of the sounds emitted. Articulators comprise the numerous portions of the supralaryngeal area that contribute to the articulation of various vowels and consonants. The lips, teeth, tongue, alveolar ridge, hard palate, and velum are the principal

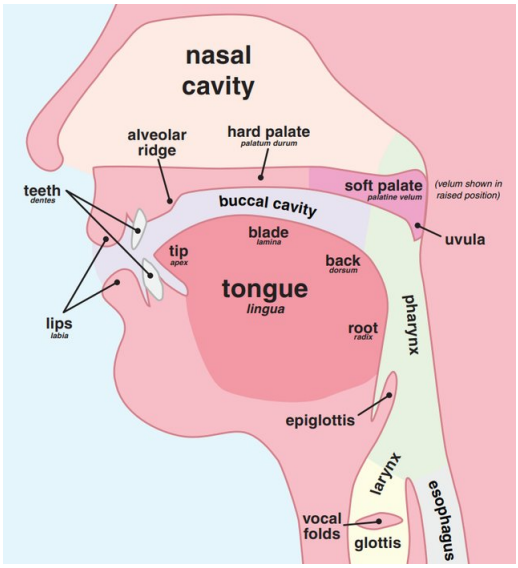


Figure 3.1.: A diagram of vocal tract with articulators [source: (Ahmed et al., 2022)].

articulators in the oral cavity. The tongue and lower lip are the active articulators, whereas the rest are passive and static. The intricate interplay of active and passive articulators to fully stop the airflow, constrict it into a restricted channel, or let it to flow through without restriction, produces the wide range of speech sounds present in languages throughout the world.

Speech is a temporal activity, and listeners are sensitive to changes in an utterance's temporal flow. An unlimited number of sounds called *phones* can be produced by the vocal tract which are independent of the language acquisition (Hardcastle et al., 2012). A given language may only have a limited number of perceptibly unique and distinct sound units called *phonemes* that can be utilized to build meaningful constructs which facilitate communication. These *phonemes* are studied under the branch of linguistics known as *phonology*. The continuous signal produced by the vocal tract is classified into distinct *phoneme* classes based on acoustic, articulatory, and perceptual aspects. Most languages have two types of *phonemes*: *vowels*, which are voiced sounds formed without restricting the airflow by articulators, and *consonants*, which are formed by restricting the airflow flowing through the vocal tract (Hardcastle et al., 2012).

Some studies try to connect the location of disfluencies with phonetic proprieties as discussed in Section 2.3.2. According to Jayaram (1983); Didirkova (2016); Blomgren

et al. (2012) unvoiced consonants are more disfluent than their voiced consonants. Wingate (1969) hypothesizes that stuttering is a coarticulation[2] difficulty rather than a sound production problem. He theorizes that disfluencies develop when muscle activity for a previously generated sound is turned off and muscular movements for the next sound are turned on. However, Didirková and Hirsch (2020) demonstrate in a recent study, using EMA data, that stuttering is not a coarticulation problem. They discovered accurate coarticulatory patterns in both fluent and stuttering speech.

Speech representation in time domain: Speech is a pressure wave that propagates across the medium of air. It can be measured over time by measuring the fluctuations in pressure of air molecules at a single location in space (Hardcastle et al., 2012). Figure 3.2 shows the time-domain signal, spectrogram, and MFCCs of a male speakers saying the word *"Communication Disorder"*. An amplitude with finite precision of bit depth, and the sampling rate/frequency F_s are required to digitally represent an acoustic speech signal. Mathematically, the digital audio speech signal can be represented as $\mathbf{y}[n]$, where $n = 0, 1, ..., N_s - 1$ denotes a sample of instantaneous amplitude.

Speech representation in spectral domain: The first step that takes us out of the time domain to its spectral domain is short-term analysis. A speech utterance is usually a sequence of phonemes. In each speech utterance, most of its properties such as amplitude, pitch, voicing, etc., fluctuate across time. Contemporary different articulator configurations make different sounds, therefore the mechanism that generates the signal changes with time. As a result, in order to analyse the speech utterance, it is divided into uniform segments known as *time frames* ($\vec{fr}$), each of which is independently processed and analyzed. The speech utterance $\mathbf{y}[n]$ is segmented into T time-frames (each of length L samples). An overlap of $L/2$ samples is used avoiding artifacts thereby obtaining a processed representation of whole speech utterance as [$\vec{fr}_0$, $\vec{fr}_1$, ..., $\vec{fr}_{T-1}$], where for each time stamp $t \in$ [0, 1, ..., T-1]

$$\vec{fr}_t = \mathbf{y}\left[t * \frac{L}{2} + i\right]_{i=0}^{i=L-1} \tag{3.1}$$

These small time frames $\vec{fr}_t$ are sliced using a small Hamming window $\psi[L]$ as shown

[2] We recall that the coarticulation refers to the influence of one speech segment on another, that is, the influence of a phonetic context on a given segment. The phonetic context is more varied in a connected segment speech than it is in isolated speech segments.

in Equation (3.2) (L is same length as time-frame length). A value of L is chosen 20 to 30 ms during which the system (vocal tract) is considered to be stationary throughout the window, hence the signal's spectral characteristics are assumed to remain constant throughout this region (Hardcastle et al., 2012).

$$\psi[i] = 0.54 - 0.46\, cos\left(\frac{i\pi}{L}\right) \tag{3.2}$$

3.2.2. Acoustic features

In developing any speech recognition or stuttering identification system, representative feature extraction and selection is extremely an important task that affects the system performance. The first common step in the speech processing domain is feature extraction. With the help of various signal processing techniques, we aim to extract the features that compactly represent the speech signal and approximate the human auditory system's response (Huang et al., 2001).

The various feature extraction methods that have been explored in the stuttering recognition systems are envelope parameters (Howell and Sackin, 1995), 1^{st} to 3^{rd} formant frequencies and their amplitudes (Czyzewski et al., 2003; Khara et al., 2018), spectral measure (fast Fourier transform (FFT) 512) (Szczurowska et al., 2014; Świetlicka et al., 2009), MFCCs (Chee et al., 2009; Khara et al., 2018; Hariharan et al., 2012; Esmaili et al., 2017), LPCCs (Khara et al., 2018; Hariharan et al., 2012), pitch, shimmer (López-de Ipiña et al., 2018), zero crossing rate (ZCR) (Khara et al., 2018), respiratory biosignals (Villegas et al., 2019b). Kourkounakis et al. (2020) exploited the use of spectrograms (as a gray scale image) as a sole feature extractor for stutter recognition and thus making it suitable for the convolutional neural networks.

For stuttering domain, MFCCs are the most suitable and are the most commonly used features in stuttered speech domain. Thus for all our experimental studies in the following chapters, we use MFCCs features extracted after every 10 ms on a 20 ms window for each speech sample.

MFCCs: In order to investigate the properties of speech sounds, spectral-domain representation has shown to be quite effective. The spectral envelope, for example, is the smooth curve that follows the peaks of the spectrum for every given analysis frame and

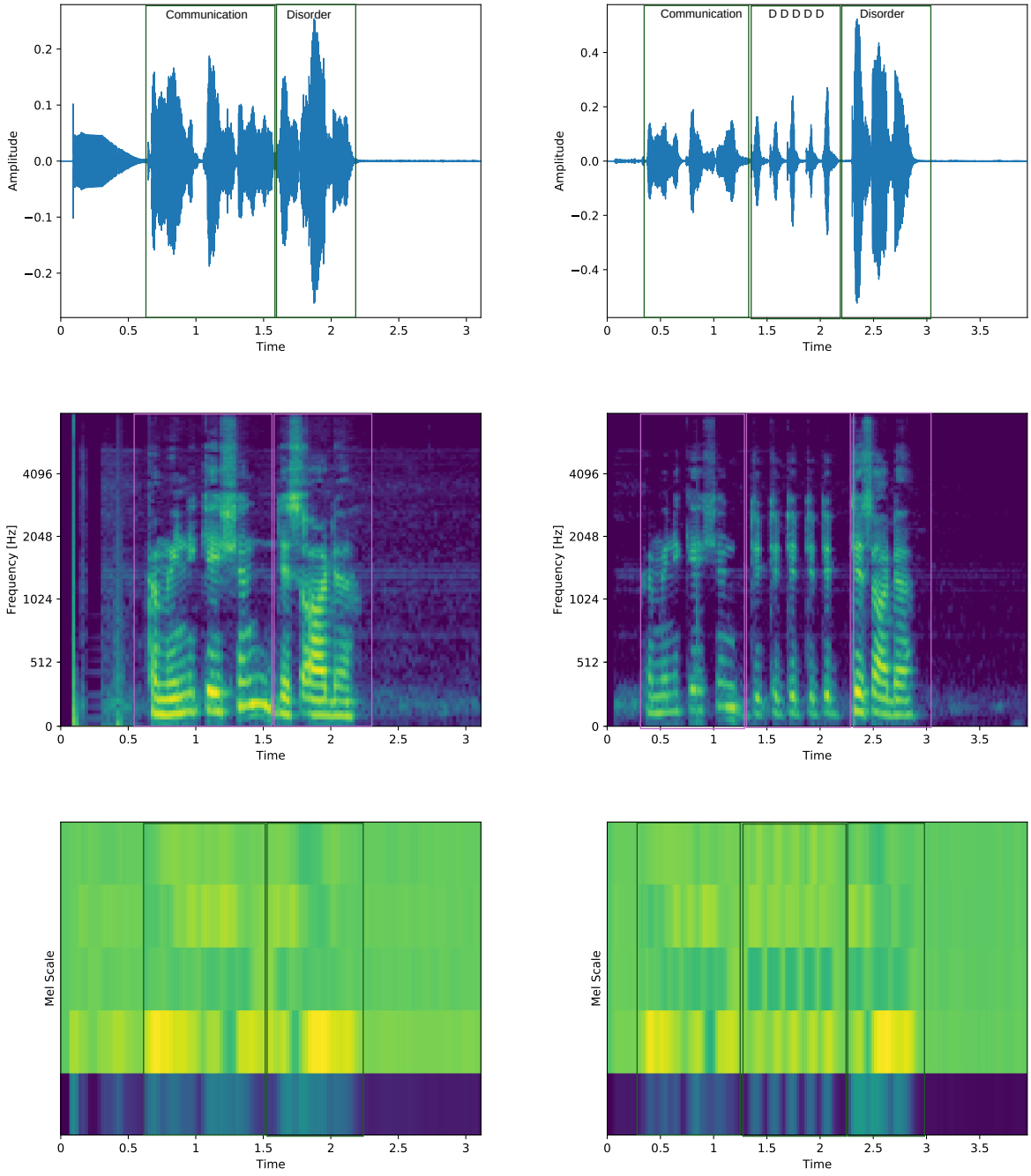


Figure 3.2.: Speech waveforms (top), spectrograms (middle), and MFCCs (bottom) of a male speaker saying *Communication disorder*. Left is fluent speech and right is stuttering (repetitions), occurring at the *D* (between 1.25 second to 2.25 second window) in the *disorder* utterance

.

is regulated by the configuration of the vocal tract. Each phoneme has its own unique spectral features that may be utilized as a blueprint for its identification (Huang et al., 2001; Hardcastle et al., 2012). The human auditory system, capable of perceiving sounds within a fixed range of 20 Hz to 20 kHz, is able to discriminate sounds at lower frequencies easily than the higher frequency region with the same frequency distance between the tones. Most people can recognize the difference between a 100 Hz and a 200 Hz sound. By the same argument, we should presume that we can distinguish between a 10000 Hz and 10100 Hz. Even if the distance between the two sets of sounds is exactly the same, our perception of the frequency range is not, thus, humans find it far more difficult to distinguish between tones at higher frequencies.

To tackle these issues, researchers suggested different spectrum modifications so that the obtained features adhere to a compressed representation driven by the perceptual process of the human ear (Jurafsky and Martin, 2018; Huang et al., 2001; Hardcastle et al., 2012). Thus, after computing the magnitude spectrum, each frame is transformed to a non-linear perceptual *Mel* scale using the Equation (3.3).

$$Mel(F) = 2595 * \log_{10}\left(1 + \frac{F}{700}\right) \tag{3.3}$$

where F is linear frequency. The *Mel* scale as shown in Figure 3.2 intends to emulate human auditory system's sensitivity by warping the frequency scale using closely packed narrow bandpass filters at lower frequencies and gradually broader and loosely spaced filters at higher frequencies.

3.3. Challenges in stuttering domain

Even though a handful number of stuttering datasets (for completed picture of the datasets, please see Appendix A) have been curated, however only three have been made public including UCLASS, LibriStutter, and SEP-28k datasets. Moreover, these datasets come with several flaws. The UCLASS lacks annotations and meta data information such as microphone used, etc. In SEP-28k stuttering datastet, the meta-data information such as speaker information, recording conditions, etc., is missing. This section discusses the various challenges faced by stuttering datasets and SD systems. In this thesis, we try to

address some of these challenges in the follow up chapters.

3.3.1. Data collection problem in stuttering domain

One of the most common barriers that need to be addressed is the issue of scarcity of data on stuttering identification research. There are only a few natural stuttered datasets as discussed in Appendix A. Usually, medical data collection is expensive and very demanding, and stuttering is no exception. Thus, having ample speaker and sentence variation adds more complexity to the stuttering domain. In order to make a fine analysis across several speakers, it is appropriate to have the same content (same list of sentences). Unfortunately, in practice, when a PWS is asked to read a list of sentences, the disfluency effects are greatly reduced. For this reason, more spontaneous speech is used in the hope to induce disfluencies. Moreover, depending on the speaker, the presence of disfluency in a recording is more or less important for several reasons: emotional state, speaking in public or alone, spontaneous or read speech, etc. This makes the collection of a corpus even more difficult and its size from one speaker to another can be variable if one aims at having a comparable number of examples of disfluencies. Moreover, it is extremely difficult, if not impossible, to collect a corpus that contains the same number of examples of each type of disfluency. It is even more challenging to achieve high variability in gender, language, and dialect. It should be noted that the recording of spontaneous speech must be well controlled to comply with the legislation. Due to the sensitivity of medical data and privacy concerns, it can not be applied at a large scale. Currently, we are not dealing with anonymization, as the voice could identify the speaker, but a minimum effort in this direction is required. In order to identify stuttering using deep learning models, the data must be properly labelled. Different background noises can corrupt the stuttered speech data. Likewise, the noise of recording equipment can also degrade the speech signal.

3.3.2. Stuttering data annotation ambiguity

It is no doubt that deep learning has led to enormous advancement in the speech domain, nonetheless, it demands a large amount of labelled data, and also, the dataset bias has plagued current SD methods. Annotating the stuttered speech requires expert SLP/ST, and are also inclined towards the idiosyncratic belief of SLP/ST, thus is expensive and laborious.

Unlike the other datasets such as ImgaeNet (Russakovsky et al., 2015), LibriSpeech (Panayotov et al., 2015), where it is easy to label a sample, however in stuttering datasets, it is not straightforward due to the fact that a stuttered speech sample can be labelled differently by different speech therapists/phoneticians.

3.3.3. Lack of evaluation protocol in stuttering detection

Different studies have used different evaluation metrics to test the performance of their SD systems. Accuracy (Kourkounakis et al., 2020), F1-score (Bayerl et al., 2022b; Lea et al., 2021), UAR (Schuller et al., 2022), etc., have been used widely in SD. However, there is no single consistent metric that is being used across the SD community which makes it hard to compare the newly developed SD systems to the existing ones. Even though the works mentioned above show promising results in SD, however, there are no consistent data selection (train/val/test) rules and no clear evaluation protocol for evaluating various SD methods which makes it extremely difficult to compare the proposed SD methods with the state-of-the-art SD techniques. Therefore, it is essential to have a unified data selection procedure for comparing the performance of different SD systems, as well as a single consistent evaluation metric, which considers the class count when evaluating the performance of the SD methods.

3.3.4. Class imbalance in stuttering detection

Stuttering datasets also suffer from class imbalance problems, i.e., the number of samples available for different disfluent categories is not uniform. It is mentioned that in stuttering, the repetitions are the most frequent ones followed by prolongations, and blocks (Guitar, 2013). However, in the UCLASS dataset, the block type is present in the majority followed by repetitions and prolongations. In SEP-28k, the majority class is interjections followed by repetitions, blocks, and prolongations respectively. The model trained on these type of imbalanced datasets is biased towards the majority class.

3.3.5. Lack of appropriate stuttering representations

Another issue in the stuttering speech domain is the need for hand-engineered features, which approximate the human auditory system. MFCCs are the principal set of hand-

engineered acoustic features that have been used mainly for stuttering identification tasks. The main drawback of this approach is that being manual is cumbersome and requires human knowledge. Over the past few years in the speech domain, the use of hand-engineered acoustic features is gradually changing and representation learning is acquiring recognition as an effective alternative to learn and capture task-specific features directly from raw speech signals, thus circumvents the hand-engineered feature extraction module from the pre-processing pipeline (Latif et al., 2020b).

3.3.6. Multimodal stuttering learning

In stuttering identification, deep learning has been successfully applied to single modalities like text and audio. Inspired by the human brain, where the perceptions are carried out through the integration of information from several sensory inputs including vision, hearing, smell, etc., Ngiam et al. (2011) proposed a multi-modal (audio visual) learning and showed how to train deep models that learn effective shared representations across the two modalities. The stuttering itself exhibits an audio-visual problem. Cues are present both in the visual (e.g. head nodding, lip tremors, quick eye blinks, and unusual lip shapes) as well as in the audio modality (Guitar, 2013). This multimodal learning paradigm could be helpful in learning robust stutter-specific hidden representations across the cross-modality platform, and could also help in building robust SD systems.

3.3.7. Domain adaptation in stuttering detection

Most of the existing SD techniques proposed so far are evaluated on a dataset of specific languages consisting of limited speakers. The existing SD techniques depend on a probabilistic model to capture domain-specific factors so that any alteration in the input speech domain could have a significant impact (in terms of language or speakers) at the time of inference. It is yet to be explored that, how well an SD technique performs in a cross-domain or cross-language environment. There could be two possible scenarios for the cross-language issue: the first is when the model is trained with a specific-language data, but tested in other languages; the other scenario could be, during training, a disfluent person registered in one language, but evaluated in a different language at the test time. Learning stutter-specific features that are invariant to variabilities in language, speakers, recording

conditions, etc., could improve the performance of SD systems.

3.3.8. Multi-stuttering identification

Most of the SD studies focus on utterances, which consist of only one type of stuttering. However, the speech utterance can contain a mix of stuttering such as *d-d-d—–dog dog is big*, which consists of syllable repetition, prolongation and word repetition types of disfluencies (Sawyer, 2010). There is a lack of studies in detecting multiple stuttering types if present in an utterance, and to the best of our knowledge, Ghonem et al. (2017) is the only study, that has been carried out to detect multiple stuttering types (*repetition-prolongation*) in an utterance.

3.4. Neural networks for stuttering detection

Artificial neural networks (ANNs) consist of several connected computing neurons that loosely model the biological neurons (Goodfellow et al., 2016). Like the synapses in biological neurons, each neuron can transmit a signal to other neurons via connections. A neuron receives a signal, processes it, and can transmit a signal to other connected neurons. The connections have weights associated with them which adjusts the learning procedure as shown in Figure 3.3 (Goodfellow et al., 2016). ANNs are trained by processing examples that maps a given input $\mathbf{x} = [x_0, x_1, x_2, ..., x_n]$ to its corresponding result by forming a probability-weighted associations between the two using the Equation (3.4), where $\alpha(.)$ is a non-linear activation function which transforms the weighted sum of inputs to the probability vector, $\theta = [w_0, w_1, w_2, ..., w_n]$ are the synaptic weights with a bias term w_0.

$$\begin{aligned} \hat{y} &= \alpha\left(\sum_{i=1}^{n} w_i * x_i\right) \\ &= \alpha\left(\theta^T \mathbf{x}\right) \text{ (\textit{in vector notation})} \end{aligned} \tag{3.4}$$

In practice, we employ neural networks with a more complex architecture characterized by numerous connections between thousands of neurons indicated by weights and biases than a simple perceptron model to learn sophisticated non-linear mappings required by the real-time applications such as emotion recognition, ASR, SD, etc. The input to the neural network is passed through a directed mechanism called forward propagation as

shown in Figure 3.4, which computes the output class probabilities of the network by processing the input via several layers with the first layer called input layer and the middle layers are called hidden layers.

Training: Give a stuttered dataset $\mathcal{D} = (X_i, d_i, s_i)_{i=1}^{N}$, with $X_i = [\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$ are speech frames, d_i is stutter label, and s_i is speaker label, the training is conducted with the help of backpropagation by optimizing the loss function by computing the error difference between the predicted output and its corresponding ground truth. Continuous weight adaptations will cause the ANNs to produce a similar output as the ground truth. After an adequate number of weight adjustments, the training can be terminated once the optimization criteria is reached (Goodfellow et al., 2016).

Loss function: One of the most common loss functions used in deep learning classification tasks is cross-entropy (Goodfellow et al., 2016). It is a metric used to assess how well a classification performs on a given task. The cross-entropy in a true sense measures the difference between the true probability distribution and the predicted distribution that can be defined for multi-class classification task as:

$$\mathcal{L} = -\sum_{c=1}^{C} y_{o,c} \log(p_{o,c}) \tag{3.5}$$

where, C is number of classes, y is a binary indicator (0 or 1), if the class label c is the correct classification for the observed sample o, and p is the predicted probability. A mapping is performed by optimizing the loss function using a gradient descent to update each of parameters $w_{ji} \in \theta$ by the following step:

$$\begin{aligned} w_{ji} &= w_{ji} - \eta \Delta_{w_{ji}} \\ \Delta_{w_{ji}} &= \nabla_{w_{ji}} \mathcal{L} \end{aligned} \tag{3.6}$$

where η is the learning rate $\Delta_{w_{ji}}$ is the gradient of the $\mathcal{L}$ with respect to the weight w_{ji} connecting the last hidden layer to the output classification layer.

Activation function: There are several possible choices in selecting an activation function for the neural network including sigmoid, rectified linear (ReLU), softmax, etc. The sigmoid non-linearity activation function $\sigma(x) = 1/(1 + e^{-x})$ saturate and kills the gradient thereby stopping the information to flow through the network and leads to vanishing gradient problem. Moreover, the $\sigma(x)$ is highly sensitive to the weight initialization

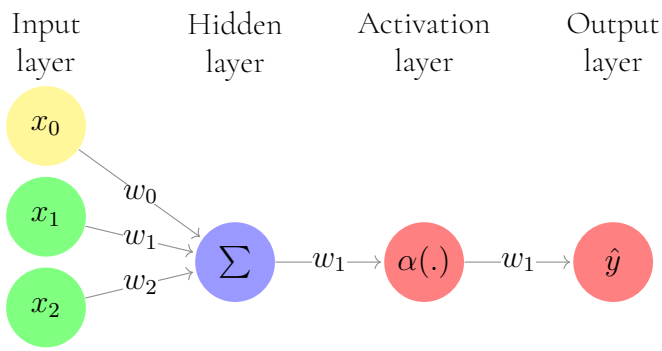


Figure 3.3.: Simple perceptron ANNs with one hidden layer.

with large initial weights forces most of the neurons to get saturated and in that case the neural network model will barely learn (Goodfellow et al., 2016). The ReLU non-linearity activation function has become the most adapted choice in the last few years in speech processing community (Nassif and *et al.*, 2019; Latif et al., 2020a; Ning et al., 2019b). In mathematical formulation, it is defined as:

$$\alpha(x) = \max(0, x) \tag{3.7}$$

with its derivative as follows:

$$\alpha^{'}(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x > 0 \end{cases} \tag{3.8}$$

Compared to the sigmoid and Tanh, the ReLU can simply be thresholded at zero, avoiding expensive operations such as exponentials resulting in faster training of the deep learning models (Zeiler et al., 2013). Another popular activation function is the softmax function, which ensures that the outputs are positive and add up to 1. In classification problems, it is commonly used as the activation for the output layer resulting in a probability distribution vector over the categorical classes.

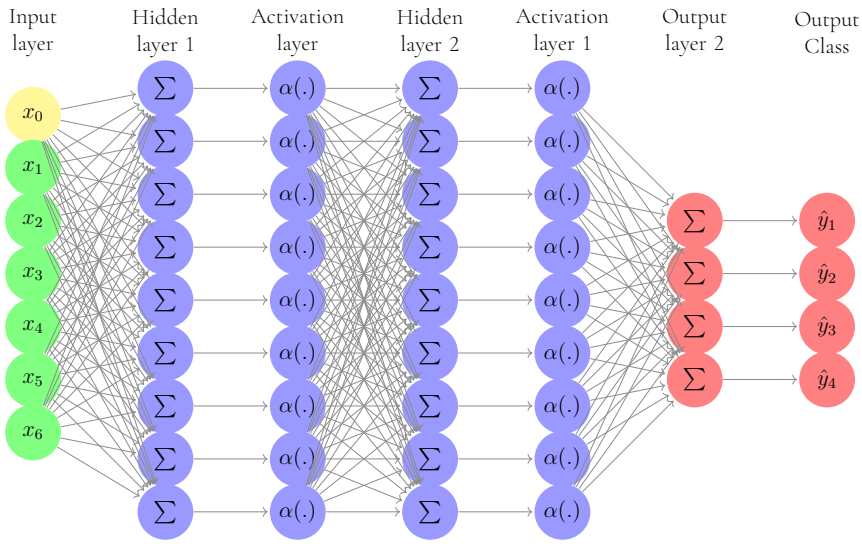


Figure 3.4.: Feed forward fully connected neural network.

3.4.1. Relevant artificial neural network models

Deep learning has allowed researchers to investigate a variety of complex network architectures that may be suitable for specific tasks. In this section, we give a brief overview of the models that are suitable for speech processing, where the frame level MFCCs (as described in Section 3.2.2) of the full speech utterance are fed to the network in learning the temporal behaviour of the speech signal. The most basic form of ANNs are the feed-forward networks as depicted in Figure 3.4, are the excellent function approximators to discover the relation between the independent and dependent variables of the data. However, they fail to capture the spatial information in images, and temporal information in speech data efficiently. In order to capture the spatial information in images effectively, Convolutional neural networks (CNN) inspired by the mechanism of visual cortex have been put forward (Goodfellow et al., 2016). A CNN's fundamental neuron behaves like a kernel with a predefined 2D receptive field, moving over the input image and performing a convolution operation with the area covered by it. To learn different spatial properties of an image, several kernels convolve with the same input image followed by a non-linear activation such as ReLU and a pooling operation to convert the kernel output to a feature map which represents discriminative features such as shapes, edges in the given image.

The time delay neural network TDNN is another feed-forward suitable architecture

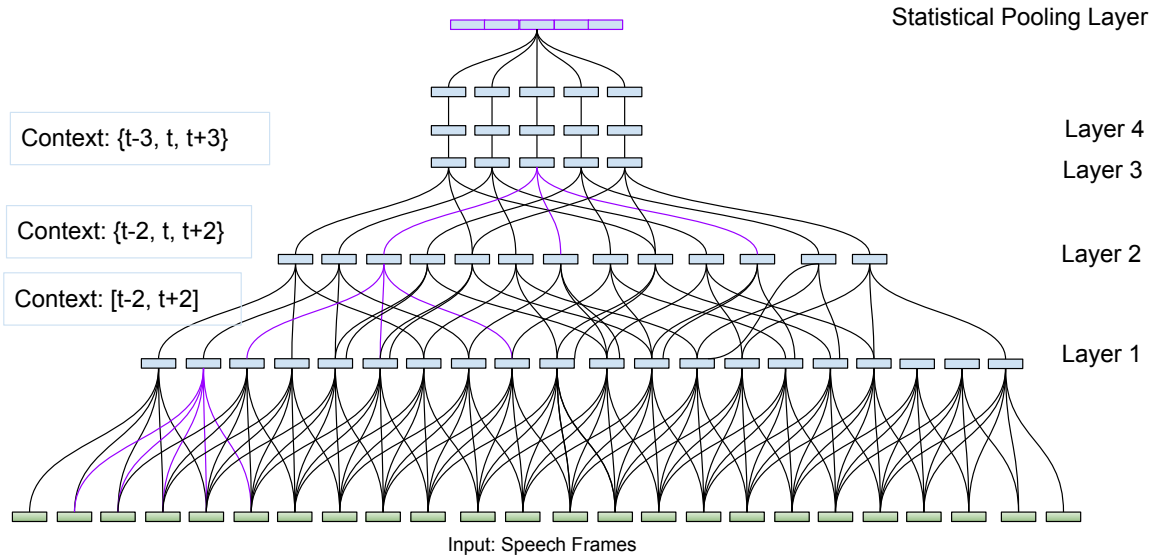


Figure 3.5.: Time delay neural network model. The first layers of TDNN model focuses only on the small context of speech frames. Layer 2 takes input as the sliced output of layer 1 which results in capturing a more temporal context at layer 2 with the help of the previous layer's context. The statistical pooling layer computes and concatenates the mean and standard deviation by aggregating all T output frames at the end of fifth layer.

to model the temporal information present in speech data. Usually, the initial layers in a standard deep neural network learn wider contexts when processing a temporal input signal. However in TDNN network as shown in Figure 3.5, the initial layers learn and capture only smaller contexts and the deeper ones compute the activations from a wider context, thus the deeper/higher layers are able to capture and learn the longer temporal contexts (Waibel et al., 1989). Because of the large overlap between neighboring contexts, TDNNs can be made computationally efficient by sub-sampling (called dilation) the activations that are passed on to the next layer.

Recurrent neural networks (RNN) belong to a family of deep neural architectures where connections between neurons/nodes form a directed graph along a temporal sequence, thus allowing it to show temporal dynamic behavior. RNN consist of an internal state (memory) that is used to process variable length input sequences. This structure makes RNN good for modeling sequential tasks like time series, connected handwriting, video, and various speech application tasks (Goodfellow et al., 2016).

Adversarial learning for speaker invariant stuttering detection: Domain adversarial training has been extensively applied in speech processing which aids in the adaptation of neural network classifiers to new domains such as different speakers, languages, environmental conditions, etc. (Ganin and Lempitsky, 2015a; Meng et al., 2018b). The main principle of adversarial training allows neural networks to learn latent representations in the hidden layers that are indifferent to data belonging to similar classes but coming from different domains (like speakers, language, microphone). For instance, the latent features for a particular stuttering class such as *prolongations* will be almost identically distributed when learned by adversarial training on various speakers. An *adversarial branch* that predicts the domain such as speaker is appended to a neural network classifier. This is accompanied by a *gradient reversal layer* which uses the negative scalar value to scale the backpropagated gradients as shown in Figure 3.6.

3.4.2. Self supervised learning

Supervised deep learning has been the cornerstone of speech transformation, which has provided remarkable advances for scenarios rich in labeled data. This high dependence on supervised learning paradigm, counter-intuitively, has hindered progress in other speech domains such as stuttering that are limited in resources and do not receive the same level of labeling effort. Inspired by the manner in which young children acquire their first language via involvement with the environment, researchers are attempting to learn speech representations in a self-supervised fashion drawn from the data itself that are able to capture low-level acoustic and lexical information in addition to semantic content (Mohamed et al., 2022). SSL leverages the information from the input data itself to learn latent speech representations that are useful for other downstream tasks (Mohamed et al., 2022). SSL usually is combined of two modes. In the first mode, SSL is used to learn good *latent speech representations* by pre-training the deep learning models on massive unlabeled datasets with some pre-auxilliary tasks such as predicting future speech frame based on the previous context frames. These learned speech representations are then used in the downstream tasks for low-resource settings either in the same frozen form or by fine-tuning the entire network in a supervised manner (Baevski et al., 2020; Pepino et al., 2021). The SSL framework has shown exceptional performance in the tasks like automatic speech recognition (Mohamed

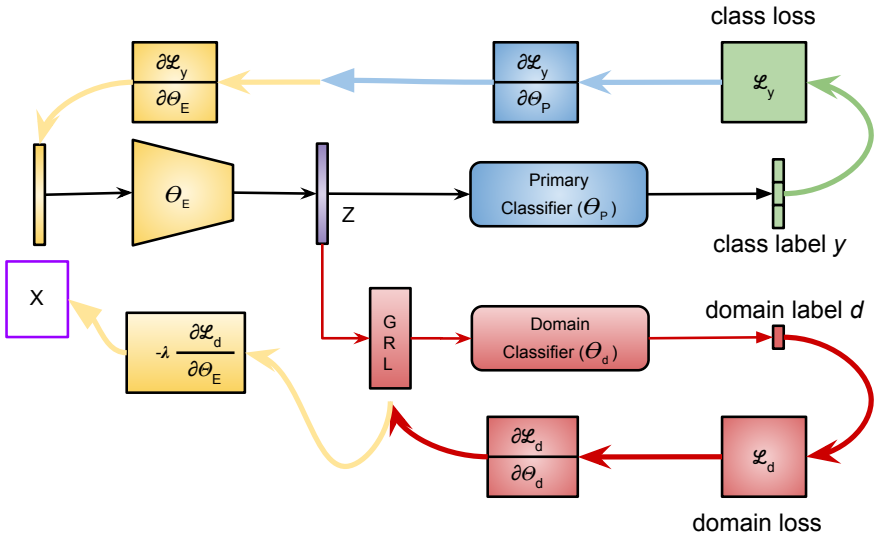


Figure 3.6.: General adversarial learning of neural network classifiers. The network includes a feature encoder $\mathcal{E}$ (yellow), with primary classifier in light blue, and domain classifier in red. The black and right red arrows show forward propagation of gradients in the network. All other backward arrows represent backward propagation of gradients with gradient reversal layer (GRL) scales the gradients by reversing using the coeffient λ.

et al., 2022), emotion detection (Pepino et al., 2021), etc. Motivated by this, we employ and evaluate the SSL framework in the stuttering detection domain.

Relevant self-supervised learning architectures: There are several state-of-the art SSL models, however we choose Wav2Vec2.0 for our experimental study in stuttering detection. The Wav2Vec2.0 has been proven effective in ASR (Baevski et al., 2020), emotion detection (Pepino et al., 2021), etc. The Wav2Vec2.0 model is trained in two phases. The first phase takes raw audio as an input and is pre-trained using large unlabeled data and aims at learning good high-level speech representations (Baevski et al., 2020). After the first pre-trained task is finished, the model is fine-tuned in a supervised fashion, during which the limited annotated data is used for downstream tasks such as ASR, Emotion detection (ED), Speaker identification (SID) etc. Exploiting the large volume of unlabelled speech data enables the model to learn robust speech representations. The Wav2Vec2.0 model is a three-module-based self-supervised representation learning framework for raw audio, pre-trained

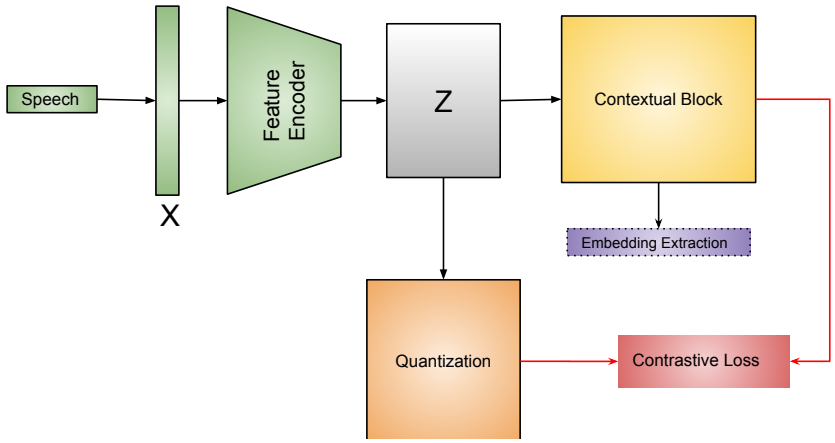


Figure 3.7.: Block diagram of Wav2Vec2.0 SSL architecture. We extract embeddings from the contextual block which is trained in a self-supervised fashion using quantisation module (Baevski et al., 2020).

on LibriSpeech dataset followed by fine-tuning towards automatic speech recognition task using connectionist temporal classification loss function. The three modules consist of feature encoder $\mathcal{E}$, contextual block $\mathcal{C}$ and quantization block $\mathcal{Q}$. The $\mathcal{E}$ encodes the raw input signal X into local features. These encoded features are then passed to $\mathcal{C}$ and $\mathcal{Q}$ modules to learn good contextual speech representations (Baevski et al., 2020) as shown in Figure 3.7

3.4.3. State-of-the-art automatic stuttering detection

Stuttering identification is an interdisciplinary research topic that is now being addressed by a plethora of research studies (in terms of acoustical feature extraction and classification approaches) with an emphasis on creating automatic tools for its detection and identification. The majority of work detects and identifies stuttering using language models (Zayats et al., 2016b; Chen et al., 2020) or by ASR systems (Alharbi et al., 2018, 2020), which first translates audio signals into their appropriate textual form, and then by the application of language models, detects or identifies stuttering in it. Even though this method of detecting stuttering has achieved encouraging results and has been proven effective, the reliance on ASR makes it computationally expensive and prone to error. Moreover, the adoption towards ASR task results in possible loss of stuttering relevant information such as prosody and emotional content (Bayerl et al., 2022b). An alternative strategy of SD is to apply the machine/deep learning classifiers directly to the acoustic represented stuttered

speech signal without involving ASR and language models to reveal the stuttering type embedded in it. In this section, we discuss the various and state-of-the art classifiers used in SD domain.

SD systems process and classify underlying stuttering embedded speech segments. Including traditional classifiers, many statistical machine learning techniques have been explored in the automatic detection of stuttering. However, the studies are empirical, so no generally accepted technique can be used. The traditional classifiers that were explored in stuttering identification include SVM, Hidden markov models (HMM), Multi-layer perceptron (MLP), NBC, Gaussian mixture models (GMM), and k-NN.

Most of the SD existing methods employ spectral features including MFCCs and spectrograms (please refer Section 3.2.1) or their variants that capture the stuttering-related information. The earlier studies in neural network based stuttering detection explored shallow architectures. Howell and Sackin (1995); Howell et al. (1997a,b) employed two separate artificial neural networks for the identification of repetition and prolongation disfluencies. This work used autocorrelation features, envelope parameters, and spectral information input to the neural network. The experiments were conducted with a dataset of 12 speakers. Ravikumar et al. (2009) attempted MLP for the detection of repetition disfluencies. They used MFCCs as input features from 12 disfluent speakers. Szczurowska et al. (2014) employed Kohonen network and MLP for discriminating fluent and disfluent speech. The experiments were conducted with eight speakers. Villegas et al. (2019a) proposed a respiratory-based stuttering classifier. They trained MLP on the respiratory air volume and pulse rate features for the detection of block stuttering. The network is trained on 68 Latin American Spanish speakers and reported promising results with bio-respiratory modality.

In recent decades, the application of deep learning have grown tremendously in speech recognition (Nassif and *et al.*, 2019), speaker recognition (Latif et al., 2020a), speech synthesis (Ning et al., 2019a), emotion detection (Akçay and Oğuz, 2020), voice conversion (Sisman et al., 2020), voice disorder detection (Verde et al., 2018) including Parkinson's disease detection (Almeida et al., 2019) and dysarthric speech detection (Narendra and Alku, 2019; Kodrasi, 2021). Inspired by the human auditory temporal mechanism, Kodrasi (2021) recently proposed a convolutional neural network based dysarthric speech detection method

by factoring the speech signal into two discriminative representations including temporal envelope (stress, voicing, and phonetic information) and fine structure (breathiness, pitch, and vowel quality) and reported the state-of-the-art results in dysarthric detection. However, the application of deep learning in SD is limited. Due to the presence of acoustic cues in the stuttered embedded speech, the deep learning models can be used to exploit these acoustic cues in the detection and identification of stuttering events as the stuttering can be reflected in representative feature space as shown in Figure 3.2.

Kourkounakis et al. (2020) recently proposed residual network in conjunction with bi-directional long short-term memory (ResNet+BiLSTM) based binary deep learning classifiers for the detection of six different types of disfluencies including prolongation, word repetition, sound repetition, phrase repetition, and false starts. They used spectrograms as an input features and reported promising results on a small subset (25 speakers) from the UCLASS dataset. Lea et al. (2021) develop a large stuttering dataset (SEP-28k) and utilized convolutional long short-term memory (ConvLSTM) model for the detection and identification of six different stuttering types which include: blocks, prolongations, sound repetitions, word repetitions, and interjections. The model takes 40-dimensional input MFCCs, 8-dimensional articulatory features, 41-dimensional phoneme feature vector, and three dimensional pitch features. The model was trained using a cross entropy loss function with a batch size of 256. In another study by Jouaiti and Dautenhahn (2022), where they introduced phoneme based BiLSTM for SD by mixing the SEP-28k and UCLASS datasets. A comprehensive detail on state-of-the-art SD methods can be found in the review paper (Sheikh et al., 2022).

3.5. Summary

This chapter presents an overview of underlying speech processing principles which forms the base for extracting acoustic features. Moreover, this chapter also discusses the various challenges in SD task. In addition to speech processing fundamentals, we also give a brief overview about the relevant deep learning models employed and adapted for SD in this thesis.

4. StutterNet

"I may be deprived of eloquence, but my mind can never be dumb."

M. B. Johnson

4.1. Introduction

People with the stuttering problem face several difficulties in social and professional interactions. Due to the very limited research in the field of stuttering detection SD, the idea is to design a single network that can be used to detect and identify various types of stuttering disfluencies. This chapter[1] is about the automatic detection of stuttering with several important applications. For example, it could facilitate the speech therapist's work, since they have to carry out a manual calculation to evaluate the severity of stuttering; to give a feedback to PWS about their fluency. Most of the classifiers proposed for SD so far are either single binary classifiers (distinguishing between fluent v/s stutter) or multiple binary classifiers (requiring a separate model to train for each disfluency category, thus making it computationally expensive). Moreover, the *fluent* samples of PWS were not taken into consideration during training (Kourkounakis et al., 2020, 2021; Bayerl et al., 2022b) which can have a huge impact on other disfluent categories. In addition to this, most of the studies have used only a *small subset* of *speakers* in their training, which makes it hard to generalize on a large pool of speakers due to their variability in language and accent. The obvious question still remaining in the stuttering domain is whether the stuttering patterns are uniform and consistent across a large pool of speakers.

[1]The following article forms the basis of this chapter:

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "StutterNet: Stuttering detection using time delay neural network". *In Proc. 29th EUSIPCO*, (2021), pp 426-430 (peer-reviewed).

In this chapter, we introduce and describe *StutterNet*, a novel deep learning based stuttering detection capable of detecting and identifying various types of disfluencies. Most of the existing work in this domain uses ASR combined with language models for stuttering detection. Compared to the existing work, which depends on the ASR module, our method relies solely on the acoustic signal. We use a TDNN suitable for capturing contextual aspects of the disfluent utterances. TDNN has been widely used for different speech classification problems such as speech and speaker recognition (Snyder et al., 2018), etc. We introduce this for stuttering detection task. The proposed method, referred to as *StutterNet* is a *multi-class* classifier with output as stuttering types and fluent. Our method achieves promising results and outperforms the state-of-the-art residual neural network based method.

4.2. StutterNet architecture

In developing any speech domain application, the representative feature extraction is the most important that affects the model performance. With the aid of signal processing techniques, several features of the stuttered speech signal can be extracted like raw waveform, spectrograms, mel-spectrograms, and or MFCCs. However, our aim is to compute and extract the features that compactly characterize the stuttering embedded in a speech segment and also which approximates the human auditory system's response. For stuttering domain, MFCCs as described in Section 3.2.2 are the best suitable and are the most commonly used features in stuttered speech domain, thus, we use MFCCs as the sole features to our *StutterNet* network. These features are generated after every 10 ms on a 20 ms window for each 3-sec (3-sec for SEP-28k and 4-sec for UCLASS dataset) [2] audio sample. This four-second window is used as stuttering lasts on average for four seconds (Guitar, 2013).

The proposed StutterNet first computes the MFCCs features from audio samples, which are then passed to the TDNN to learn and capture the temporal context of various types of stuttering disfluencies. Usually, the initial layers in a standard deep neural network learn wider contexts when processing a temporal input signal. However in *StutterNet* network as shown in Table 4.1 and Figure 4.1, the initial layers learn and capture only smaller contexts and the deeper ones compute the activations from a wider context. Thus the deeper/higher layers are able to capture and learn the longer temporal contexts. The network

[2] Each audio sample in UCLASS in of 4-seconds long and in SEP-28k, it is only 3-seconds long.

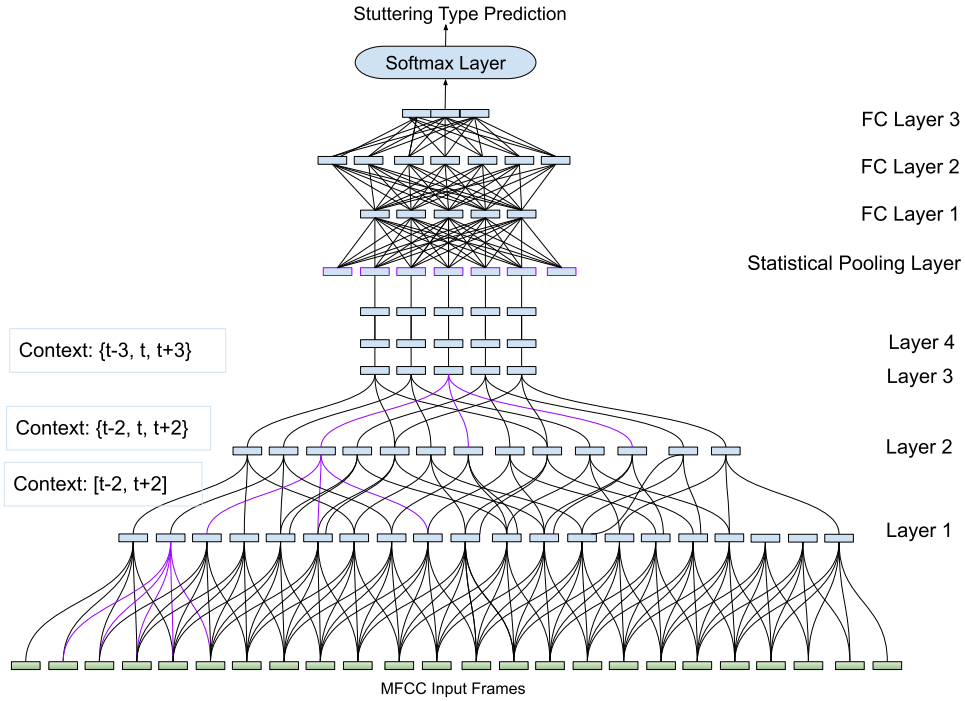


Figure 4.1.: *StutterNet* with layer based context-wise computation. The first five layers of *StutterNet* focus on the small context of speech frames. For example, layer 2 takes input as the sliced output of layer 1 at time frames of $\{t-2\,, t,\ t+2\}$, which results in capturing a total temporal context of 9 with the help of the previous layer's context of $[t-2,\ t+2]$. Similarly, layer 3 sees a total context of 15 time frames. The SP layer computes and concatenates the mean and standard deviation by aggregating all T output frames at the end of fifth layer. The SP layer accumulates the information across the temporal dimension which makes it suitable for subsequent layers to operate on the entire temporal speech segment.

consists of five time delay layers with the first three focusing on $[t-2,\ t+2]$ (dilation = 1), $\{t-2\,, t,\ t+2\}$ (dilation = 2), $\{t-3\,, t,\ t+3\}$ (dilation = 3), and the other two on $\{t\}$ contextual frames, respectively. This is followed by a mean and standard deviation pooling layer, three FC layers, and a softmax layer on top of the network that reveals the classification scores of stuttering disfluencies. A ReLU nonlinearity activation function and a 1D batch normalization are applied after each layer except the SP layer. The two FC layers are followed by a dropout of 0.3.

Consider an input speech sample with T frames. The first five layers of *StutterNet* focus on the small context of speech frames. For example, layer 2 takes input as the sliced

Layer	Output layer size	Layer context	TC
TDNN 1	512	$[t-2,\ t+2]$	5
TDNN 2	512	{t-2, t, t+2}	9
TDNN 3	512	{t-3, t, t+3}	15
TDNN 4	512	{t}	15
TDNN 5	3*512	{t}	15
Statistical Pooling	$3*512\times2$	$[0, T)$	T
FC1	512	-	T
FC2	512	-	T
FC3	NumClasses	-	T

Table 4.1.: StutterNet architecture. Except statistical pooling layer, each layer is followed by a ReLU activation function and a batch normalization (TC: Total context, TDNN: Time delay neural network layer, FC: Fully connected layer, T: Temporal dimensional of speech utterance, t: Current time frame).

output of layer 1 at time frames of $\{t-2,\ t,\ t+2\}$, which results in capturing a total temporal context of 9 with the help of the previous layer's context of $[t-2,\ t+2]$. Similarly, layer 3 sees a total context of 15 time frames. The SP layer computes and concatenates the mean and standard deviation by aggregating all T output frames at the end of fifth layer. The SP layer accumulates the information across the temporal dimension which makes it suitable for subsequent layers to operate on the entire temporal speech segment. The *StutterNet* is trained to classify different types of stuttering including *core behaviors* and the fluent part of the PWS.

4.3. Experimental evaluation

4.3.1. Datasets

UCLASS: We have used the UCLASS release 1 stuttering dataset that has been created by the Department of Psychology and Language Sciences, University College London (Howell et al., 2009). This dataset consists of monologue samples from 139 participants. Of these, 128 have been chosen in this case study with females 18 and males 110. Of these, 104 speakers were used for training, 12 for validation and 12 for testing. The audio samples were annotated manually by listening to the recordings of speech segments.

The annotations have been carried out into different categories of stuttering including: core behaviors (blocks, receptions, and prolongations), fluent, repetition-prolongations, blocks-prolongations, repetition-blocks, and blocks-repetition-prolongations. However, for UCLASS dataset, we are focusing only on the core behaviors as the dataset contains only few of the remaining ones. Each monologue audio clip was sliced into 4-second and down sampled to 16 *kHz* segments, resulting in a total of 4674 speech segment samples.

SEP-28k: The original SEP-28k dataset[3] contains 28,177 samples out of which, we used only 23573 segments, among which 3286 are repetitions, 1770 are prolongations, 2103 are blocks and 12419 are fluent segments, and 3995 are interjections. This results in a total of 19.65 hours of data which includes 2.74 hours of repetition, 1.48 hours of prolongation, 1.75 hours of block, 10.35 hours of fluent speech, and 3.34 hours of interjections. After labeling, each 3-sec sliced speech segment is downsampled to 16 kHz. We randomly selected 80% of podcasts (without mixing podcasts) for training, 10% of podcasts for validation, and the remaining 10% of the podcast for evaluation in a 10-fold cross-validation scheme. The speaker information is missing from the SEP-28k dataset, so we divided the dataset based on podcast ids (assuming each podcast is having a unique speaker). The SEP-28k dataset is publicly available, but the details about the train, validation, and test set splits are not provided. So, we decided to create a publicly available protocol that ensures no overlap of podcasts between train, validation, and test sets[4].

In order to evaluate the *StutterNet* method on the UCLASS and SEP-28k datasets, we adopted *K*-fold cross validation technique , where *K=10*. We conducted 10 experiments, each consisting of random sampling of 80% for training, 10% for validation and last 10% for testing[5]. The reported results are the average of 10 experiments. All experiments were trained with an early stopping criteria of patience 7 on validation loss.

4.3.2. Implementation

We develop *StutterNet* with PyTorch library in Python. We use a learning rate of 10^{-4}, amsgrad optimizer, and cross-entropy loss function. We use Librosa library for the feature

[3] For details, please refer to Appendix A
[4] For splitting of the dataset, please refer `https://shakeel608.github.io/protocol.pdf`
[5] Speaker disjoint split.

extraction. We select models using an early stopping with a patience of seven epochs on validation loss. We compare the results obtained by *StutterNet* to existing method (Kourkounakis et al., 2020) in the same experimental framework.

4.3.3. Evlauation metric

To evaluate the model performance, we use the following metrics: macro F1-score, and accuracy which is the standard and are widely used in the stuttered speech domain. The macro F1-score ($\mathcal{F}_1$) (which combines the advantages of both precision and recall in a single metric unlike unweighted average recall which only takes recall into account) from (Equation (4.1)) is often used in class imbalance scenarios with the intention to give equal importance to frequent and infrequent classes, and is also more robust towards the error type distribution (Opitz and Burst, 2019).

$$\mathcal{F}_1 = 1/C \sum_k F1_k = 1/C \sum_k \frac{2.P_k R_k}{P_k + R_k} \tag{4.1}$$

where C is the number of classes and P_k, R_k, and $F1_k$ denotes the precision, recall, and F1-score with respect to class k.

4.4. Results and discussion

The results of our *StutterNet* for different stuttering recognition are presented in Table 4.2, Table 4.3, Table 4.4, and, Table 4.5 where we compare our technique to (Kourkounakis et al., 2020). All the considered disfluencies and the fluent speech are recognized with good scores. As can be seen from the Table 4.4, F1-Scores show clearly the good performances of *StutterNet* method in comparison to residual network and bi-directional long short-term memory (ResNet+BiLSTM). Table 4.5 also shows that the *StutterNet* surpasses the state-of-the-art in most disfluency detection cases, but shows slightly lower performance in prolongation and block detection with an average accuracies of 17.13%, 42.43% in comparison to 23.17%, 53.33% average accuracies of ResNet+BiLSTM respectively. The *StutterNet* outperforms ResNet+BiLSTM in correctly detecting fluent speech with a difference of 11.63 points (66.63% for the *StutterNet* and 55.00% for ResNet+BiLSTM).

We separately optimize the *StutterNet* by varying the filter bank size (10, 30, 40, 50), context window (3, 7, 9, 11) and layer size (64, 128, 256, 1024). We found that context win-

	Precision			
Method	R	P	B	F
(Kourkounakis et al., 2020) (baseline)	0.33	0.42	0.43	**0.63**
StutterNet	**0.36**	**0.43**	0.42	0.59
***StutterNet* (Optimized)**	0.35	0.31	**0.47**	0.59

Table 4.2.: Results in recall on UCLASS dataset (B: Block, F: Fluent, R: Repetition, P: Prolongation).

	Recall			
Method	R	P	B	F
(Kourkounakis et al., 2020) (baseline)	0.20	**0.23**	**0.53**	0.55
StutterNet	**0.28**	0.17	0.42	0.67
***StutterNet* (Optimized)**	0.24	0.13	0.47	**0.70**

Table 4.3.: Results in precision on UCLASS dataset (B: Block, F: Fluent, R: Repetition, P: Prolongation).

	F1-Score			
Method	R	P	B	F
(Kourkounakis et al., 2020) (baseline)	0.22	**0.28**	0.44	0.52
StutterNet	**0.30**	0.23	0.42	0.62
***StutterNet* (Optimized)**	0.27	0.16	**0.46**	**0.63**

Table 4.4.: F1-score on UCLASS dataset (B: Block, F: Fluent, R: Repetition, P: Prolongation).

Method	Accuracy				Tot. Acc.	$\mathcal{F}_1$
	Rept	P	B	F		
Kourkounakis et al. (2020) (baseline)	20.39	23.17	53.33	55.00	46.10	0.36
StutterNet	27.88	17.13	42.43	66.63	49.26	0.39
***StutterNet* (Optimized)**	23.98	12.96	47.14	69.69	50.79	0.38

Table 4.5.: Results in accuracies and $\mathcal{F}_1$ score on UCLASS dataset (B: Block, F: Fluent, Rept: Repetition, P: Prolongation. $\mathcal{F}_1$: Macro F1 score).

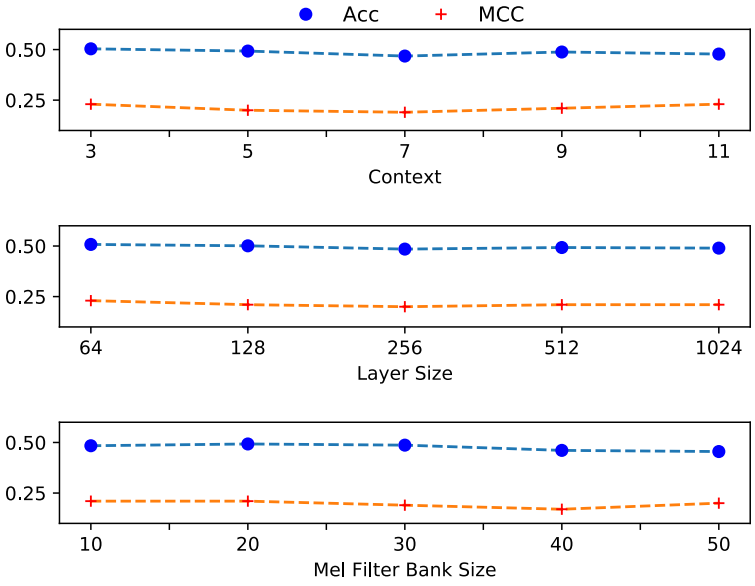


Figure 4.2.: Mathews correlation coefficient (MCC) and accuracy of *StutterNet* with varying context, layer size and mel filter bank size (Acc is normalized in [0,1]).

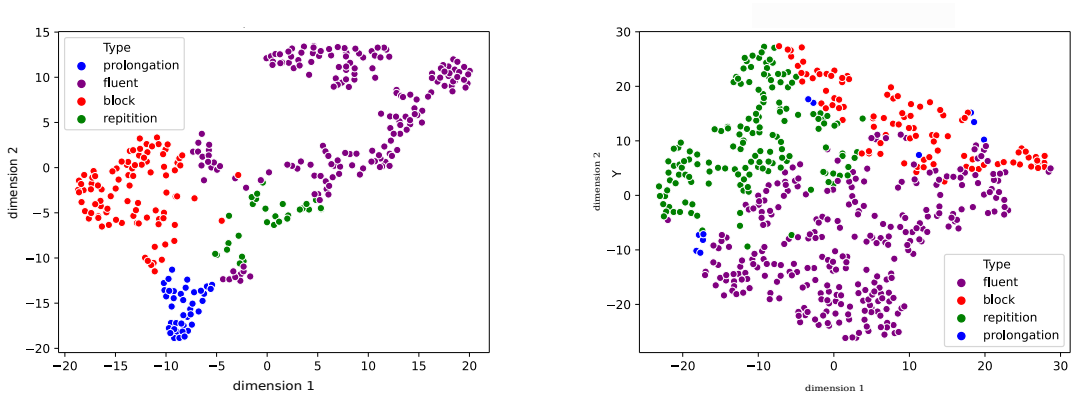


Figure 4.3.: t-SNE visualization of the output of last fully-connected layer for *ResNet + BiLSTM* (left) and *StutterNet* (right).

dow optimization improves the detection performance of prolongation and repetition type of disfluencies. As the context increases, the performance of *StutterNet* increases for the detection of prolongation and repetition stutterings, but decreases for the fluent segments, and it remains almost unchanged for the block stuttering. This makes sense because the repetition and prolongation disfluencies usually lasts longer and the longer context helps the *StutterNet* in improving the performance. The block disfluencies don't last longer and usually occur at the beginning of the speech segment and thus making it context independent. We also found that layer size optimization slightly improves the performance (overall accuracy and Mathews correlation coefficient) of the stuttering detection in block and fluent types of disfluencies as shown in Figure 4.2. This might be due to the possible reason that the baseline *StutterNet* is over-parameterized due to the limited size of the UCLASS dataset. We term the layer size optimized *StutterNet* as optimized *StutterNet* in Table 4.5 . Compared to (Kourkounakis et al., 2020), our optimized proposed method gains a margin of 4.69% in overall average accuracy. For detecting the *core behaviours* and the fluent part, the margins are also substantial (improvements of 7.49%, 14.69% in repetition and fluent speech segments, respectively). Most previous work tends to avoid block disfluencies because of their similar nature to silence and prolongation (blocks are prolonged without audible airflow). As shown in Table 4.5, the proposed *StutterNet* can detect and classify the block stuttering with an average accuracy of 47.14%. Moreover, our technique relies on the assumption that stuttering usually lasts for four-second window size (Guitar, 2013). Note that some of the stuttering (in particular prolongation and repetition) can exceed more than four seconds in speech (Guitar, 2013), thus causing those prolongation stuttering likely to be misclassified.

Figure 4.3 shows a visualization of the latent feature embeddings learned by *StutterNet* using t-SNE projection. Both ResNet+BiLSTM and *StutterNet* present very good discrimination of the different types of disfluencies. However, the latent feature embeddings learned by *StutterNet* are more distinctive for fluent and less distinctive for prolongations and accurately capture the stutter-specific information than the state-of-the-art ResNet+BiLSTM method. Interestingly for ResNet+BiLSTM, the fluent and repetition category's embeddings are widely spread and more overlapped with the other classes. The prolongations and blocks are well clustered in BiLSTM+ResNet as compared to *StutterNet*.

Method	Accuracy					Tot. Acc.	$\mathcal{F}_1$
	R	P	B	In	F		
(Lea et al., 2021) (baseline)	22.83	10.61	06.34	56.74	72.35	52.68	0.34
StutterNet	21.99	27.78	1.98	49.99	88.18	60.33	0.39

Table 4.6.: Results in accuracies and macro $\mathcal{F}_1$ score on SEP-28k dataset (B: Block, F: Fluent, R: Repetition, P: Prolongation, In: Interjection, $\mathcal{F}_1$: Macro F1 score).

We also perform experiments on SEP-28k dataset and compared our proposed *StutterNet* with the convolutional LSTM (ConvLSTM) model proposed by (Lea et al., 2021). From Table 4.6, it can be clearly seen that the proposed *StutterNet* outperforms ConvLSTM (Lea et al., 2021) in overall macro F1 ($\mathcal{F}_1$) score by a relative margin of 14.71%. The *StutterNet* is a single branched SD system which suffers from a class imbalance problem as can be seen from the major classes of interjection and fluent classes where the detection performance is greater as compared to minority classes. The ConvLSTM SD model performs better in repetitions and interjection disfluencies most likely because of the class balanced training using separate branching for each disfluency type.

4.5. Summary

This chapter addresses the stuttering detection as a multi-class classification problem and introduces *StutterNet* which is based on time delay neural network for stuttering detection. Next, this chapter discusses the *StutterNet* architecture in detail followed by experimental setup. Next, we give the details about UCLASS and SEP-28k datasets, and their protocol for splitting the data by ensuring there is no overlap of speakers across train, validation and test sets. This is followed by experimental results and discussion in comparison to the baselines.

5. Multi-branch and multi-contextual StutterNet

"I like to be multi-contextual, which is much more important than being uni-cultural."

Cornel West

5.1. Introduction

Most of the earlier work employed only a small set of disfluent speakers in their experimental studies, and has approached the stuttering detection problem as a binary classification problem: disfluent vs fluent identification or one vs other type[1]. The *StutterNet* we propose as discussed in previous Chapter 4, tackled the SD as a multi-class classification problem and evaluated the proposed methodology on a medium sized dataset (100+ speakers) as compared to the earlier work (Kourkounakis et al., 2020). Although this network has shown promising results in SD, it has several deficiencies. First, it doesn't generalize well on unseen data and leads to overfitting due to the limited amount of data available for training. In addition to data scarcity, the stuttering dataset was collected from podcasts in a clean environment, which makes the trained *StutterNet* difficult to generalize in other environmental conditions. Second, obtaining class-balanced datasets is extremely difficult and very expensive in the speech domain and the stuttering datasets are no exception. Deep learning classifiers trained on highly class-imbalanced datasets are usually biased towards the majority class, which results in poor modeling of minority classes (Fernández et al., 2018). In addition to the above deficiencies, we found in our previous Chapter 4 that the

[1]The following article forms the basis of this chapter:

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. ". Advancing stuttering detection via data dugmentation class-balanced loss and multi-contextual deep learning". *Journal of IEEE Biomedical Informatics*, (2022).

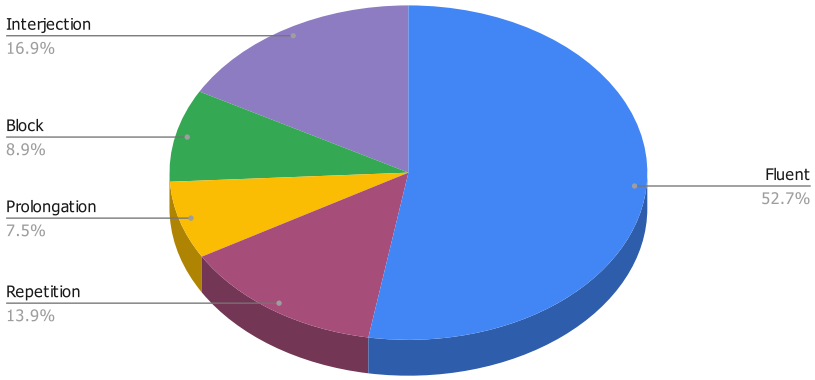


Figure 5.1.: Stuttering data distribution in SEP-28k dataset. In this thesis, we considered only those samples with single stuttering label.

context is very important in SD. A larger context increases the performance of prolongation and repetition types of disfluencies but decreases the recognition performance of fluent speech segments on the UCLASS dataset. Moreover, we found some annotation issues with the UCLASS dataset, so we have shifted to the newly created SEP-28k stuttering dataset for our experimental studies (SEP-28k is the largest dataset available in stuttering domain) (Lea et al., 2021).

5.2. Addressing deficiencies

Class Imbalance: In class imbalance learning, the distribution of samples across each class is not uniform. The deep neural network (DNN) classifiers, trained on highly imbalanced datasets generally perform poorly for the minority class and favor the majority class (Krawczyk, 2016). In the worst case, where, there is an extreme imbalance in the training set, the majority class dominates the learning, and samples among the minority class may go undetected, thus affecting the performance (Weiss, 2013).

The class-imbalance is one of the major problems in real-world applications, including multi-lingual speech recognition (Winata et al., 2020), and stuttering is no different as shown in Figure 5.1. In fact, stuttering is extremely imbalanced across fluent and other speech disfluencies. The Figure 5.1. shows that the interjections are the most common disfluency present in the SEP-28k dataset followed by repetitions, blocks, and prolongations and the overall distribution is approximately equivalent to the fluent distribution. Collecting a balanced dataset is difficult and expensive for the stuttering detection task. Other datasets such as Kassel State of Fluency (KSoF) (Schuller et al., 2022), FluencyBank (Lea et al., 2021) also suffer from this issue (discussed in the Appendix A).

Over the years, the class imbalance problem is one of the main concerns due to its prevalence, especially in the biomedical domain. Several methods have been proposed to tackle the class imbalance problem, which are mainly categorized into three groups: data-level, cost-level, and architecture-level (Fernández et al., 2018). Data-level approaches attempt to re-balance the class distribution by means of some re-sampling methods which include: under-sampling, over-sampling, or combined over-sampling and under-sampling (Fernández et al., 2018), (Krawczyk, 2016). The architecture-level approaches attempt to modify the existing algorithm or develop a new one to tune and adapt it for imbalanced datasets (Fernández et al., 2018), (Krawczyk, 2016),. Cost-level approaches attempt to influence the loss/objective function by providing comparatively higher misclassification cost penalties to the minority class in order to force the model to learn about minority classes (Krawczyk, 2016; Fernández et al., 2018). In DNNs, addressing the class-imbalance problem by re-sampling may either get rid of sensitive speech samples that are extremely important in training when under-sampling. It can also add numerous quantities of duplicated speech samples under the over-sampling strategy, which eventually makes the training expensive and makes the DNN model likely to overfit (Fernández et al., 2018). Because of these limitations, we investigate cost-level and architecture-level approaches in this work.

For the cost-based approach, we modify the standard cross entropy loss by assigning weights to different classes (Cui et al., 2019). We set the class weights inversely proportional to the number of samples. We define the weight for class i as

$$w_i = \frac{N}{C * N_i} \tag{5.1}$$

where N is the number of training samples, C is number of classes, N_i is the number of training samples for class i.

Therefore, the weighted cross-entropy (WCE) over the train set can be defined as,

$$\mathcal{L}_{WCE} = \frac{1}{\mathcal{B}} \sum_{b=1}^{\mathcal{B}} \frac{\sum_{i}^{M} w_i * \log(p_i)}{\sum_{i,\, i \in \mathcal{B}}^{M} w_i} \tag{5.2}$$

where $\mathcal{B}$ is the number of batches, M is number of stuttered speech samples in a batch b_i, $p_i = \left(\frac{e^{c_i}}{\sum_{j=1}^{C} e^{c_j}}\right)$ is the predicted probability of class c_i of sample i.

For architecture-level, we propose a multi-branch approach similar to the work by (Lea et al., 2021) and (Bader-El-Den et al., 2016) to address the class imbalance issue in SD task. Inspired by the fact that the number of fluent class samples is almost equal to the total number of samples in disfluent classes, we simultaneously classify fluent vs disfluent classes in one output branch and subcategories of disfluent classes in another output branch. The overall architecture with two output branches is shown in Figure 5.3 (For single contextual multi-branch (MB) *StutterNet*, only context = 5 is taken into consideration). This has one common encoder $\mathcal{E}$ (θ_e) section followed by two parallel branches referred as *FluentBranch* $\mathcal{F}$ (θ_f) and *DisfluentBranch* $\mathcal{D}$ (θ_d). The embeddings from the encoder are processed with both the branches parallelly, where the $\mathcal{F}$ is trained to distinguish between fluent and disfluent samples, and the $\mathcal{D}$ is trained to differentiate within the disfluent sub-categories.

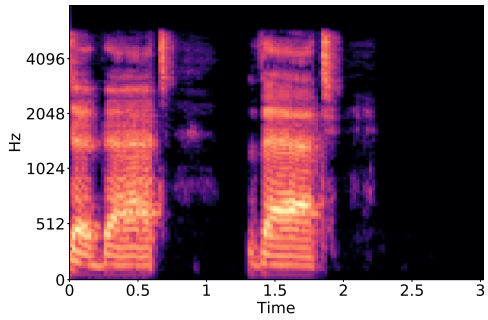
The objective is to optimize the sum of *FluentBranch* loss $\mathcal{L}_f$ and *DisfluentBranch* loss $\mathcal{L}_d$, resulting in an overall[2] objective function as below:

$$\mathcal{L}(\theta_e, \theta_f, \theta_d) = \mathcal{L}_f(\theta_e, \theta_f) + \mathcal{L}_d(\theta_e, \theta_d) \tag{5.3}$$

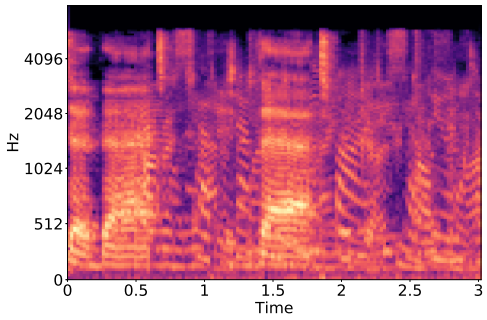
During the evaluation step, if the *FluentBranch* predicts the sample as fluent, then *FluentBranch* predictions are considered otherwise *DisfluentBranch* predictions are taken into consideration to reveal the stuttering category.

Data augmentation: Deep learning (DL) has achieved rapid progress in the domain of speech processing tasks including speech recognition (Nassif and *et al.*, 2019), speaker

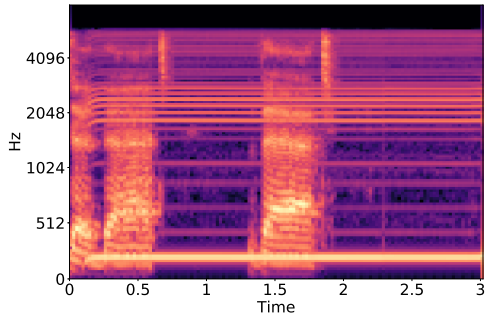
[2]For simplicity, a simple sum of the two losses has been taken into consideration.



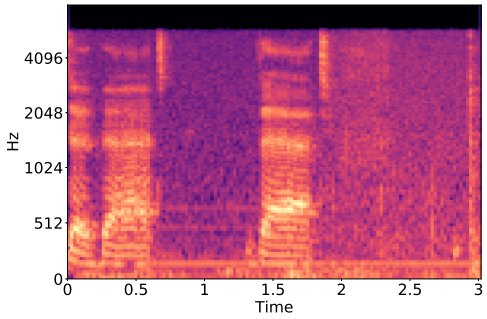
(a) Repetition (clean speech)



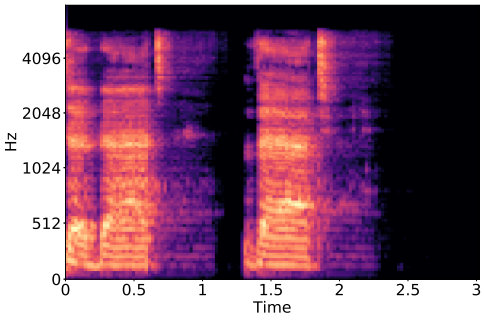
(b) Repetition with babble noise



(c) Repetition with white noise



(d) Repetition with music



(e) Repetition with reverberation

Figure 5.2.: Repetition stuttering with utterance *"said that, that"* and the effect of various data augmentations. From 0 to 0.25 seconds, the speaker is saying *"said"*, then followed by two repetitions *"that"*.

recognition (Latif et al., 2020a), emotion recognition (Akçay and Oğuz, 2020). However, as a drawback, DL based models are hungry for the data and require a substantial amount of annotated data, especially in the speech domain to address the over-fitting and robustness issues and *StutterNet* is no exception, and perhaps it needs more data than other speech processing areas, as the presence of examples of disfluencies in speech corpus is not frequent.

Data augmentation is a popular technique that increases the quantity and diversity of the existing annotated training data, improves robustness, and avoids the overfitting of DNNs. For normal speech recognition, data augmentation demonstrates to be an effective approach for dealing with data scarcity and enhancing the performance of various DNN acoustic methods (Park et al., 2019). Several data augmentation techniques have been investigated including pitch adjust (Meng et al., 2021), spectral distortion (Kanda et al., 2013), tempo perturbation (Ko et al., 2015), speed perturbation, cross-domain adaptation (Bell et al., 2012), adding noise to clean speech (Ko et al., 2015), spectrogram deformation with frequency and time masking (Park et al., 2019), mixspeech (Meng et al., 2021), etc.

On the contrary, so far, very limited attention has been given to data augmentation targeting the speech disorder domain. In (Vachhani et al., 2018), speed and tempo perturbation based data augmentation were used to convert the normal speech to dysarthric speech impairment. In (Jiao et al., 2018), a voice conversion adversarial training based framework was used to simulate dysarthric speech from healthy speech. In (Christensen et al., 2013), normal speech samples (also called out-of-domain) were used as a data augmentation for dysarthric speech in the bottleneck feature extraction stage. In (Xiong et al., 2019), the speaker-dependent parameter was computed, which was then used for augmentation of scarce dysarthric speech in tempo adjustment. In (Woszczyk et al., 2022), several data augmentation techniques such as noise, time stretching, pitch shifting, time shift, masking, etc. were analysed in Dementia detection. Even though there are some studies on data augmentation targeting other speech dirsorders such as speech dysarthric detection, however, in the case of stuttering/disfluency detection, there is none.

In the stuttering domain, the employment of data augmentation is not straightforward, because most of the data augmentations like time stretch, speed perturbation, etc., alter the underlying structure of the stuttering speech sample completely. Our approach employs reverberation and additive noises because it reflects the real-world scenario and

doesn't change significantly the underlying stuttering in the speech sample as shown in Figure 5.2. Reverberation consists of convolving speech samples with room impulse responses (RIR). We utilize the simulated RIRs as described in (Ko et al., 2017). For additive noises, we utilize the MUSAN dataset, comprised of 60 hours of speech from 12 languages, 42 hours of music from various genres, and 900 hours of noises.

To augment the original speech samples, we combine the training "clean" set with below augmented copies from the following:

1. *music:* A single music sample file randomly chosen from MUSAN is added to the original clean stuttering speech sample (SNR: 5-15dB) (The music sample file is trimmed or repeated as required to match the duration of the clean stuttered speech sample).
2. *noise:* Throughout the stuttered speech, samples from MUSAN noises are added at 1 sec intervals (SNR: 0-15dB).
3. *babble:* Speech samples from randomly three to seven speakers are summed together, then added to the original clean stuttered speech sample (SNR: 13-20dB).
4. *reverb:* The "clean" train set is convolved with simulated RIRs.

All the data augmentation shown in Figure 5.2 were performed using Kaldi (Povey et al., 2011) tool.

Multi-contextual StutterNet: The multi-contextual framework is based on the way humans perceive speech. In the cochlea, the input acoustic speech signal is partitioned into several frequency bands so that the information in each band can be filtered independently and thus processed in parallel in the human brain (Allen, 1995).

For a long time, multi-contextual has been studied for action recognition in videos (Simonyan and Zisserman, 2014), robust ASR mostly in noisy environments (Han et al., 2021; Bourlard and Dupont, 1996; Bourlard et al., 1996; Hennansky et al., 1996; Naderi and Nasersharif, 2017), speech separation (Zhang and Wang, 2016), where the input speech signal is processed in multiple streams/contexts (multiple time or frequency resolutions), etc. Han et al. (2021) recently proposed a multi-stream[3] convolutional neural network for robust acoustic modeling. Chiba et al. (2020) recently proposed multi-stream attention-based BiLSTM network for speech emotion recognition. Li et al. (2018) extracted deep features by

[3]multi-stream, multi-scale, multi-resolution are the different names of multi-context.

training multi-stream hierarchical DNN for acoustic event detection. Moreover in Chapter 4, we found that settings like context frame size optimized for one stuttering class are not good for other stuttering types. Exploiting this fact, we also investigate how the multi-contextual neural networks will impact classification performance in the speech disorder domain, and in particular stuttering identification. In our preliminary study, we found that the context window improves the identification performance of two types of disfluencies on the UCLASS dataset. As the context frame size increases in the *StutterNet*, the performance detection of prolongation and repetition also increases, but decreases for fluent speech segments, and almost remains unaffected for a block type of stuttering. The prolongation and repetition last longer than other types of disfluencies. So to address this deficiency, we exploit the variable contexts of MB *StutterNet* by training the model jointly on different contexts as shown in Figure 5.3. The pseudo-code of MC *StutterNet* is provided in *Algorithm* 1.

5.3. Experimental setup

We evaluate our proposed architecture thoroughly on the newly released SEP-28k stuttered dataset (Lea et al., 2021), and for cross copora, we used the FluencyBank dataset (Lea et al., 2021).

5.3.1. Datasets

SEP-28k: In our case study, we evaluated SEP-28k dataset (for details, please see Appendix A) where out of 28,177 clips, we used only 23573 segments, among which 3286 are repetitions, 1770 are prolongations, 2103 are blocks and 12419 are fluent segments, and 3995 are interjections. This resulted in a total of 19.65 hours of data which includes 2.74 hours of repetition, 1.48 hours of prolongation, 1.75 hours of block, 10.35 hours of fluent speech, and 3.34 hours of interjections. After labeling, each 3-sec sliced speech segment is downsampled to 16 kHz. We randomly selected 80% of podcasts (without mixing podcasts) for training, 10% of podcasts for validation, and the remaining 10% of the podcast for evaluation in a 10-fold cross-validation scheme. The speaker information is missing from the SEP-28k dataset, so we divided the dataset based on podcast ids (assuming each podcast is having a unique speaker).The SEP-28k dataset is publicly available, but the details about

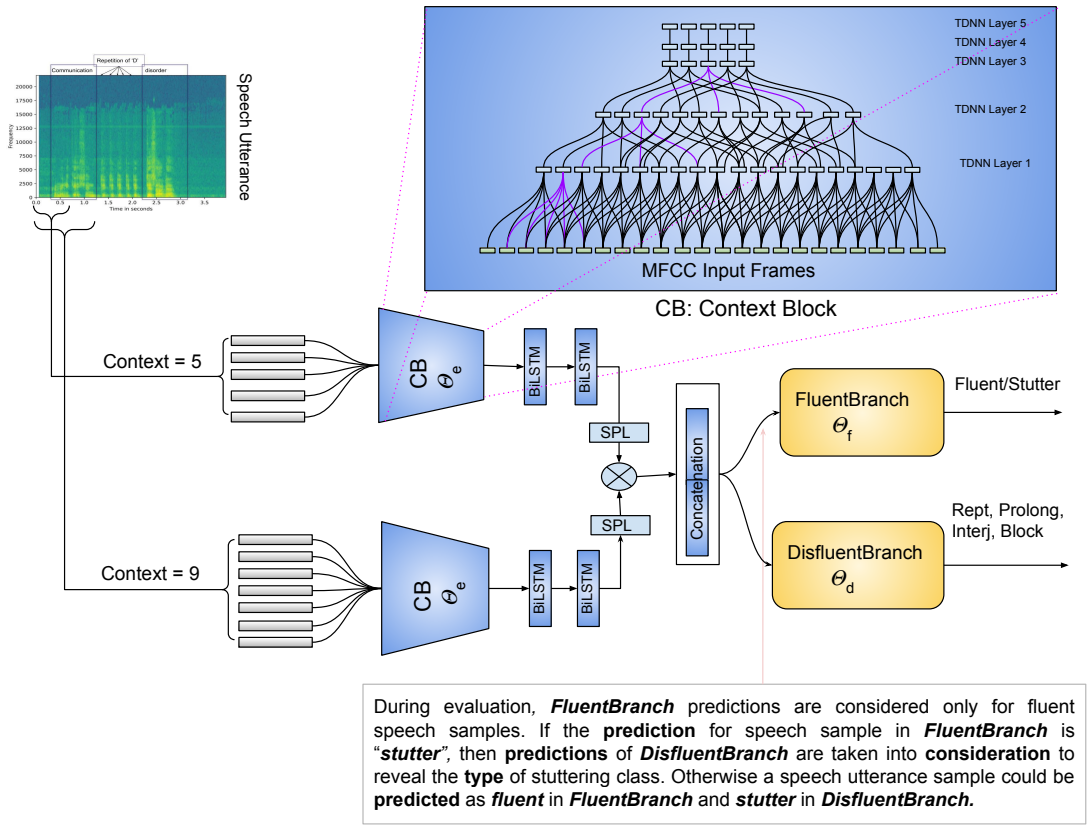


Figure 5.3.: A schematic diagram of multi-contextual *StutterNet*, which is a multi-class classifier that exploits different variable contexts of (5, 9) in SD. The FluentBranch and DisfluentBranch are composed of 3 fully connected layers followed by a softmax layer for prediction of different stuttering classes, CB: Context block, SP: Statistical pooling layer, TDNN: Time delay neural network layer.

Algorithm 1 Pseudo-code for MC *StutterNet*

Output: Predicted label set $\hat{y} \in (R, P, B, In, F)$
Input: Stutter dataset with $\mathcal{D} = (X_i, f_i, d_i)$, where each sample X_i is $\mathcal{R}^{20\times T}$ MFCC input sequence
d : Stutter label
f : Pseudo label for FluentBranch
K : Number of epochs
CE : Cross entropy loss function
p_f : Prediction of FluentBranch
p_d : Prediction of DisluentBranch
L_f : Loss of FluentBranch
L_d : Loss of DisluentBranch
L : Total loss
b : Sample batch
C : Context
η : Learning rate
$\theta = [\theta_e, \theta_f, \theta_d]$: Network parameters
θ_e : Encoder parameters
θ_d : DisluentBranch parameters
θ_f : FluentBranch parameters

Ensure: MC *StutterNet* weight initialization
for i in K **do**
  for b in $\mathcal{D}$ **do**
    Forward Pass:
    b = ApplyKaldiAugmentation(b)
    out_5 = StutterNetBase(b, C = 5)
    out_9 = StutterNetBase(b, C = 9)
    $stat_5$ = StatisticalPooling(out_5)
    $stat_9$ = StatisticalPooling(out_9)
    $stat_{embed}$ = Concat($stat_5$, $stat_9$)
    p_f = FluentBranch($stat_{embed}$)
    p_d = DisfluentBranch($stat_{embed}$)
    $L_f = CE(p_f, f_b)$ // f_b: Pseudo label of a batch
    $L_d = CE(p_d, d_b)$ // d_b: Stutter label of a batch
    $L(\theta_e, \theta_f, \theta_d) = L_f(\theta_f) + L_d(\theta_d)$
    Backward Pass:
    $\nabla\theta_b$ = backward_propagation($L(\theta)$, $\hat{y}$, d) // $\hat{y}$ is predicted label.
    Parameter Updates:
    $\theta_b \leftarrow \theta_b - \eta * \nabla\theta_b$
  end for
  Compute CE loss on validation set
  if $CE_{val} \leq CE_{val}^{prev}$ **then**
    if patience ≤ 7 **then**
      Continue training $\rightarrow$ Go to step 2
    else
      Stop training
    end if
  end if
end for

the train, validation, and test set splits are not provided. So, we decided to create a publicly available protocol that ensures no overlap of podcasts between train, validation, and test sets[4].

FluencyBank: For cross corpora, we used FluencyBank dataset (for details, please see Appendix A). Out of 4, 144 samples, we only use 3355 samples (ignoring the non-stuttering and multiple samples), among which 542 are repetitions, 222 are prolongations, 254 are blocks and 1584 are fluent segments, and 753 are Interjections. This results in a total of 2.80 hours of data which includes 0.45 hours of repetition, 0.19 hours of prolongation, 0.21 hours of block, 0.63 hours of interjection samples, and 1.32 hours of fluent samples.

We have considered those samples where at-least two annotators agree with the same labelling of stuttered speech sample.

LibriStutter: The original LibriStutter is having 6 classes including fluent, interjection, prolongation, sound, word and phrase repetitions. To make our experiments consistent with the SEP-28k dataset, we treated all repetitions as one class. After extracting samples based on the class label, we didn't find any interjection samples in the dataset, so we only trained with three classes including fluent, prolongation, and repetitions(for details of dataset split, please see Appendix A). We randomly selected 80% (40) speakers for training, 10% (5) speakers for validation , and remaining 10% (5) speakers for testing.

5.3.2. Training setup

We implement MC *StutterNet* using the PyTorch library. The acoustic input features used in this case study are 20-dimensional MFCC features, which are generated after every 10 ms on a 20 ms window size, extracted using the Librosa library. For 3-dim pitch and phoneme features, we use Pykaldi (Can et al., 2018) and Phonexia (Fer et al., 2017) tools respectively. For training, we use Adam optimizer and cross-entropy loss function with a learning rate of 10^{-2} and batch size of 128. All the results reported in this paper are the average of the 10-fold validation technique and all the training experiments were stopped by an early stopping criteria with a patience of 7 on validation loss.

[4]For splitting of the dataset, please refer https://shakeel608.github.io/protocol.pdf

5.3.3. Evaluation metrics

To evaluate the model performance, we use the following metrics: macro F1-score and accuracy which are the standard and are widely used in the stuttered speech domain (Schuller et al., 2022; Kourkounakis et al., 2020; Lea et al., 2021). The macro F1-score ($\mathcal{F}_1$) (which combines the advantages of both precision and recall in a single metric unlike unweighted average recall which only takes recall into account) from (Equation (5.4)) is often used in class imbalance scenarios with the intention to give equal importance to frequent and infrequent classes, and also is more robust towards the error type distribution (Opitz and Burst, 2019).

$$\mathcal{F}_1 = 1/C \sum_k F1_k = 1/C \sum_k \frac{2.P_k R_k}{P_k + R_k} \tag{5.4}$$

where C is the number of classes and P_k, R_k, and $F1_k$ denotes the precision, recall, and F1-score with respect to class k.

5.4. Results

In this section, we present the results with class balanced training, data augmentation, and multi-contextual learning. We propose two modifications to the vanilla *StutterNet* to address the class imbalance issue and one of them involves the loss function on the SEP-28k dataset. We further present the results of cross-corpora experiments on the FluencyBank dataset.

5.4.1. Baselines

In addition to our single branch *StutterNet* baseline (BL5), we first implement the state-of-the-art ConvLSTM and ResNet+BiLSTM as our baseline models for comparison purposes in the same settings. Our baseline model *StutterNet* is performing well in almost all the disfluent classes except blocks as compared to the baseline model used in SEP-28k paper (Lea et al., 2021) ConvLSTM (MFCC features) (we call it BL1[5]), ConvLSTM (phoneme features) (we call it BL2), and ConvLSTM (pitch+MFCC features) (we call it BL3). In addition, we also used one more model i.e, ResNet+BiLSTM classifier (we call it BL4) from (Kourkounakis et al., 2020). Comparing our single branch baseline *StutterNet* to BL4, the

[5]BL means baseline

Method	Accuracy						
	R	P	B	In	F	TA	$\mathcal{F}_1$(%)
				Baselines			
ConvLSTM + F_{MFCC} (BL1) (Lea et al., 2021)	22.83	10.61	06.34	56.74	72.35	52.68	34.00
ConvLSTM + F_{phone} (BL2) (Lea et al., 2021)	10.18	01.06	00.35	43.88	74.48	48.43	24.00
ConvLSTM + $F_{F0+MFCC}$ (BL3) (Lea et al., 2021)	19.28	09.55	08.51	51.78	66.60	48.47	30.80
ResNet+BiLSTM (BL4) (Kourkounakis et al., 2020)	18.76	41.24	5.47	57.18	88.19	62.36	43.12
StutterNet (BL5) (see Chapter 4)	27.14	32.55	2.96	57.74	87.60	62.57	42.84
				Class Imbalance			
ResNet+BiLSTM + WCE (Kourkounakis et al., 2020)	28.90	64.89	33.79	63.03	46.90	47.42	41.00
MB ResNet+BiLSTM (Kourkounakis et al., 2020)	34.79	30.19	5.92	49.26	75.47	55.62	39.20
StutterNet + WCE ($StutterNet_{WCE}$)	36.23	59.73	38.05	61.19	41.59	45.26	41.02
MB StutterNet ($StutterNet_{MB}$)	35.26	32.76	7.21	56.04	77.27	58.56	42.26
M_{enc}^{frz}	39.82	37.91	10.45	60.57	73.49	58.58	44.42
$M_{enc,disf}^{frz}$	29.25	45.85	18.11	56.88	74.49	58.18	44.80

Table 5.1.: Results using class imbalance learning (B: Block, F: Fluent, R: Repetition, P: Prolongation, In: Interjection, TA: Total Accuracy, $\mathcal{F}_1$: Macro F1- Score), BLX: Baseline and X is Number, F_{MFCC}: MFCC input features, F_{F0}: 3-Dim (Pitch, Pitch-delta, voicing) features, F_{phone}: Phoneme peatures, MB: Multi branch, WCE: Weighted cross entropy, M_{enc}^{frz}:Freezing encoder, $M_{enc,disf}^{frz}$: Freezing encoder and DisfluentBranch.

Method	Accuracy						
	R	P	B	In	F	TA	$\mathcal{F}_1$(%)
StutterNet + A4	28.67	32.53	3.69	64.24	88.70	64.69	45.30
$StutterNet_{WCE}$ + A4	43.06	61.43	38.18	65.33	43.73	48.30	44.34
MB ResNet+BiLSTM + A4	33.27	32.85	0.99	65.95	70.92	55.75	39.44
$StutterNet_{MB}$ + A4	34.05	33.93	4.32	68.74	78.88	61.35	44.03
M_{enc}^{frz} + A4	28.87	30.94	4.26	64.12	85.29	62.85	44.06
$M_{enc,disf}^{frz}$ + A4	27.63	36.33	11.52	62.31	83.05	62.14	45.76

Table 5.2.: Results using data augmentation (B: Block, F: Fluent, R: Repetition, P: Prolongation, In: Interjection, $\mathcal{F}_1$: Macro F1- Score), A4: Babble + Reverberation + Music + Noise augmentation, MB: Multi branch, WCE: Weighted cross entropy, M_{enc}^{frz}:Freezing encoder, $M_{enc,disf}^{frz}$: Freezing encoder and DisfluentBranch.

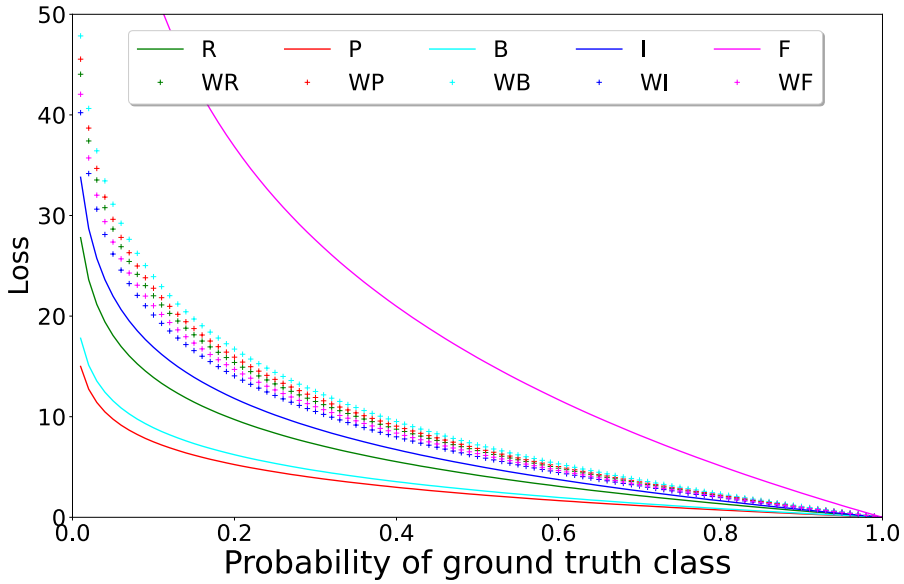


Figure 5.4.: Class-wise loss values for different probabilities of ground truth classes. Here block, fluent, repetition, prolongation, and interjection classes are correspondingly denoted by B, F, R, P, and I. The standard cross entropy (CE) is shown by solid curves and its weighted version is shown by dashed ones (WX represents the weighted loss of class X).

model performs well only in interjection and repetition classes as shown in Table 5.1. In order to evaluate our proposed methods, we choose BL1 (best among BL1, BL2, and BL3), and BL4 from baselines for comparison purposes.

5.4.2. Class balanced training

5.4.2.1. Weighted cross entropy

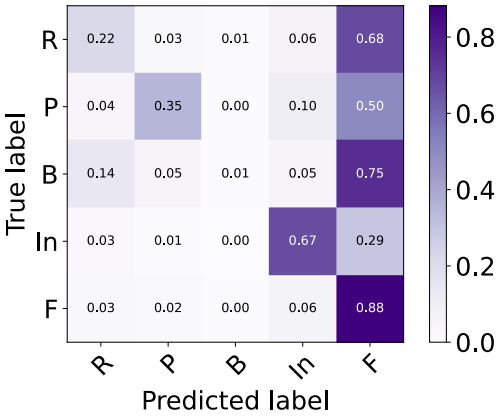
This section discusses the impact of applying weighted cross entropy to *StutterNet* (which we call $StutterNet_{WCE}$) in order to improve the detection performance of minority classes including repetitions, blocks, and prolongations. Figure 5.4. illustrates the class-wise training loss curves for normal and weighted cross-entropy loss functions. We can observe that for the standard cross entropy loss function, the majority class (i.e., fluents) exhibit

higher loss values and it dominates the overall loss.

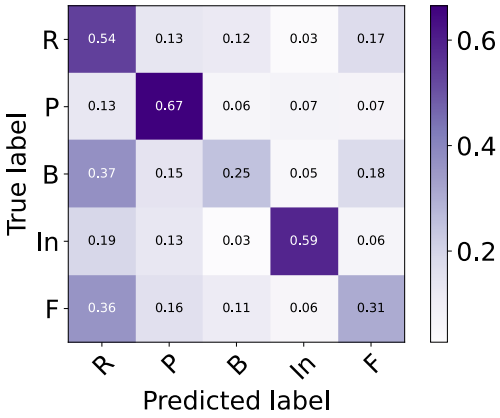
Therefore, during training with backpropagation, the number of updates for fluent class dominates the gradient values that in turn forces the model to focus mainly on correctly classifying/predicting the majority class. Thus, the minority classes including the blocks, prolongations, and repetitions are given less importance during training, which leads to their poor detection performance. Table 5.1 confirms that the detection performance of blocks and repetitions is very poor, as they are mostly predicted as fluent samples due to the class imbalance nature of the problem. In order to increase the detection performance of minority classes, we penalize the majority class by reducing its contribution to loss computation, and vice-versa for minority classes. The loss functions for the weighted cross entropy are shown by dashed curves in Figure 5.4. The figure indicates that applying weights to standard CE loss function proportional to the inverse of class frequency forces the model to give balanced importance to each disfluency class while optimizing the parameters of the network during backpropagation. This, in turn, helps in boosting the gradient updates for minority classes during training, and thus increases their detection performance in both the baselines as shown in Table 5.1. The *StutterNet* using the WCE gives a relative improvement of 33.49%, 83.50%, 1,185%, and 5.98% over BL5 and 58.69%, 462%, 500%, and 7.84% over BL1 for detecting repetitions, prolongations, blocks, and interjections, respectively. Table 5.1 also demonstrates the suitability of WCE with competitive ResNet+BiLSTM method and results in 54%, 57.35%, 517%, and 10.23% relative improvement in repetitions, prolongations, blocks, and prolongations respectively over BL4.

5.4.2.2. Multi-branch training

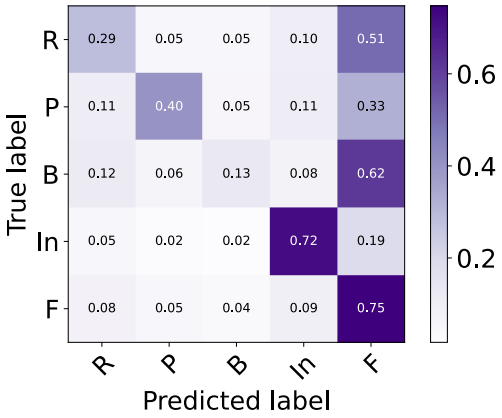
Moreover, we address the class imbalance problem via a multi-branch network (referred to as *StutterNet*$_{MB}$). This has two output branches: *FluentBranch* and *DisfluentBranch* as shown in Figure 5.3. This method improves the detection performance of repetitions, prolongations, blocks by a relative margin of 29.92%, 0.65%, 144% respectively over BL5, and 54.45%, 208%, 13.72%, 6.80% in repetitions, prolongations, blocks, and fluents respectively over BL1, and 88%, 31.81% in repetitions, blocks respectively over BL4, however, the BL4 is performing better in prolongations and interjection classes. Applying a multi-branch training scheme to our baseline BL4, we found that the MB ResNet+BiLSTM captures rich



(a) *StutterNet* (BL)



(b) *StutterNet* (WCE) (*StutterNet*$_{WCE}$)



(c) MB *StutterNet* (*StutterNet*$_{MB}$)

Figure 5.5.: Confusion matrices showing the repetitions and blocks are mostly confused with fluent speech utterances. F: Fluent, R: Repetition, B: Block, P: Prolongation, In: Interjection, BL: Baseline, WCE: Weighted cross-entropy, MB: Multi-branch.

stuttering information and improves the detection performance only in the repetition class by a relative margin of 84% over BL4.

In applying class balanced training, we found that there is drop in macro $\mathcal{F}_1$ score. Using WCE with *StutterNet* and ResNet+BiLSTM, the macro $\mathcal{F}_1$ score drops from 42.84% and 43.12% to 41.02% in *StutterNet* and 41.02% in ResNet+BiLSTM respectively. By employing multi-branch training, the macro $\mathcal{F}_1$ score drops very slightly from 42.84% to 42.26% in *StutterNet* but it drops remarkably from 43.12% to 39.20% in ResNet+BiLSTM.

5.4.2.3. Analysis of confusion matrix

Using the WCE training scheme, the detection performance of minority classes improved remarkably at the cost of fluent accuracy. From the Figure 5.5 (a), we analyze that most of the repetitions and blocks are being classified as fluent speech. Initially, we hypothesize that it is most likely because of the imbalanced nature of the problem. Despite addressing the class imbalance problem in the stuttering domain using WCE and multi-branched training, we found that the block and repetition type of stuttering disfluencies are the ones that are still getting confused with the fluent class as depicted in Figure 5.5 (b) and Figure 5.5 (c). This makes intuitive sense because the blocks are closely related with fluent segments having just different initial silence or gasp followed by fluent utterances like fluent speech. The repetitions, on the other hand, contain some word or phrasal repetitions, which are actually fluent utterances, if we carefully analyze their individual parts. Consider an utterance *he he is a boy*. The word he is being repeated twice but is a fluent part if two *he's* can be analyzed on an individual basis.

5.4.2.4. Exploiting advantage of WCE and MB *StutterNet*

Since *StutterNet*$_{WCE}$ and *StutterNet*$_{MB}$ address the class-imbalance issue differently, we combine them to exploit both of their advantages. We first pre-train the WCE based *StutterNet* (*StutterNet*$_{WCE}$) and used it as a *DisfluentBranch* in our multi-branched *StutterNet*. After the pre-training step, we applied freezing in two ways. Firstly, we freeze the parameters of the contextual encoder block basenet only and fine-tune only the two output branches. We label this training scheme as M_{enc}^{frz}. By exploiting this method, we achieved an overall detection improvement of 5.11% in $\mathcal{F}_1$ over MB *StutterNet* (*StutterNet*$_{MB}$). Secondly,

we apply freezing to the base encoder and *StutterNet*$_{WCE}$ (DisfluentBranch) and appended with one more FluentBranch (to distinguish between fluents and stutter samples). We label this training scheme as $M^{frz}_{enc,disf}$ and it results in an overall improvement of 6.01% in $\mathcal{F}_1$ over *StutterNet*$_{MB}$.

5.4.3. Data augmentation

The main experimental results obtained with various data augmentation techniques are shown in Table 5.2, where we compare the detection performance obtained with data augmented training to the baseline clean dataset. To the best of our knowledge, this is the first-ever study that investigates the impact of data augmentation in SD. We first separately train the *StutterNet*$_{MB}$ on different data augmentation techniques which are described in Section 5.2. We found that the training MB ResNet+BiLSTM and *StutterNet*$_{MB}$ with Kaldi augmentation increases the overall $\mathcal{F}_1$ performance. From Table 5.2, it can be seen that the data augmentation does help in improving the $\mathcal{F}_1$ in almost all the cases. Applying data augmentation with a single branched and weighted cross entropy *StutterNet*, there is a relative improvement of 5.74% and 3.50% in $\mathcal{F}_1$ respectively, over the clean versions of the training. When data augmentation is applied to MB training, there is a relative improvement of 4.17% and 0.61% in $\mathcal{F}_1$ using *StutterNet*$_{MB}$ and ResNet+BiLSTM over clean training, respectively.

Moreover, applying Kaldi data augmentation on top of the M^{frz}_{enc} and $M^{frz}_{enc,disf}$ training scheme, there is a relative improvement of 7.28%, and 6.81% in overall accuracy respectively, however, M^{frz}_{enc} based augmented training results in a slight drop of $\mathcal{F}_1$ score. In addition to Kaldi augmentation, we also applied pitch scaling and bandpass filter as data augmentation, however, we didn't achieve much improvement in SD.

5.4.4. Multi-contextual StutterNet

Different contexts show different optimized class accuracies. In order to exploit these different variable contexts and to improve the detection performance further, we propose a multi-contextual (MC) *StutterNet* for SD as shown in Figure 5.3. We jointly train and optimize the MC *StutterNet* on the clean and augmented data using variable contexts of 5 and 9. The embeddings extracted from each context as depicted by $\mathcal{CB}$ block are passed

Method	Accuracy						
	R	P	B	In	F	TA	$\mathcal{F}_1$(%)
MC *StutterNet* (Clean)	33.36	37.15	08.34	59.63	78.34	59.77	43.86
MC *StutterNet* + Bb	32.29	37.11	09.85	66.50	74.54	58.71	44.40
MC *StutterNet* + Mu	32.08	36.57	07.98	65.68	77.67	59.96	44.00
MC *StutterNet* + No	28.54	32.39	04.75	64.20	80.42	59.99	42.80
MC *StutterNet* + Rv	31.98	37.02	05.83	67.63	77.41	60.08	44.20
MC *StutterNet* + A4	34.65	39.75	10.12	67.91	77.89	61.72	46.00

Table 5.3.: Results using MC *StutterNet* with clean and data augmentation. The MC *StutterNet* also contains MB training. (B: Block, F: Fluent, R: Repetition, P: Prolongation, In: Interjection, TA: Total accuracy, Bb: Babble, Rv: Reverberation, Mu: Music, No: Noise, A4: Bb + Mu + No + Rv augmentation)

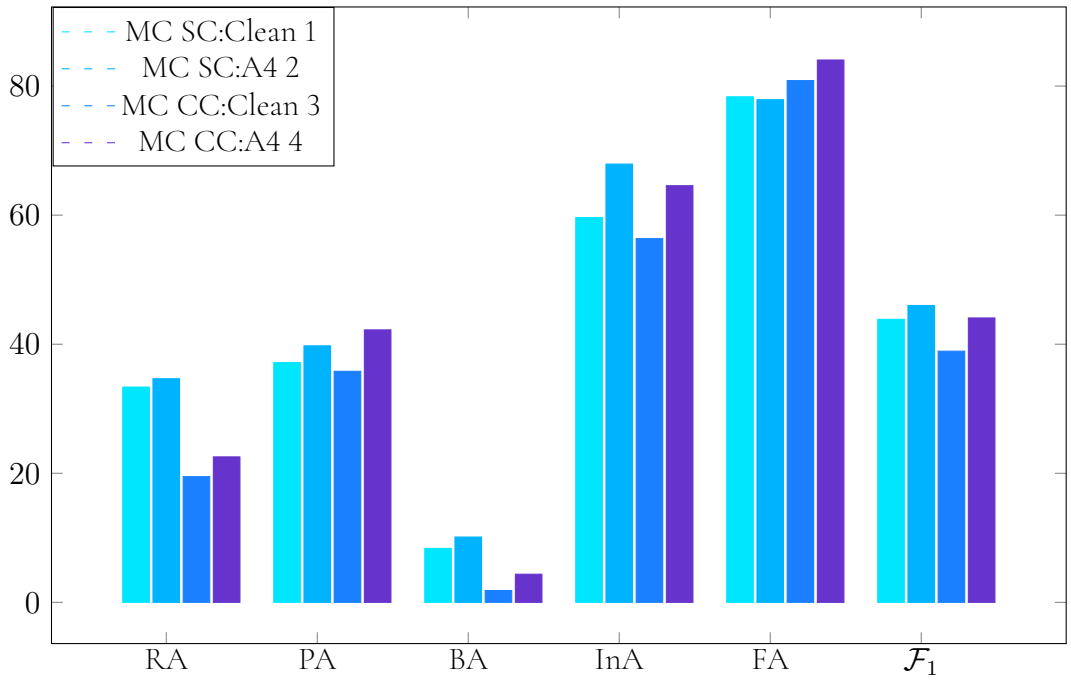


Figure 5.6.: Impact of data augmentation (A4) in cross corpora FluencyBank dataset with MC *StutterNet* (R: Repetition, P: Prolongation, B: Block, In: Interjection, F: Fluent, XA: X is Disfluency Class and A is accuracy, MC: Multi contextual *StutterNet*, SC: Same corpora, CC: Cross corpora, A4: Augmentation). The bar plot clearly shows that the model MC *StutterNet* trained on clean SEP-28k dataset fails to generalize on FluencyBank cross corpora data. Applying data augmentation improves the stuttering detection in cross domain corpora as shown by navy blue bars (1^{st} bar is clean training with SC, 2^{nd} is augmented training with SC, 3^{rd} is clean training with CC, 4^{th} is augmented training with CC).

to a two-layered BiLSTM unit, and then concatenated after applying SP, resulting in a $1 \times 2 \times (2 \times N)$-dimensional feature vector (where N is layer size), which is then fed parallelly to two different branches including *FluentBranch* and *DisfluentBranch* for class predictions. This results in a relative improvement of 13.40%, 15.67%, 6.41%, and 1.38% in prolongation, block, interjection, and fluent classes over *StutterNet*$_{MB}$ (clean), thus an improvement of 3.79% in macro $\mathcal{F}_1$ score, however, employing multi-contextual training, we see a drop from 35.26% to 33.36% in repetition accuracy over the *StutterNet*$_{MB}$. In comparison to the baseline BL5 (vanilla single branch *StutterNet*), there is a relative improvement of 22.92%, 14.13%, 181.81%, and 3.27% in repetition, prolongation, block, and interjection classes respectively. Thus a relative improvement of 2.38% in macro $\mathcal{F}_1$ score. The MC *StutterNet* also performs better in macro $\mathcal{F}_1$ score in comparison to state-of-the-art baselines BL1 and BL4. Applying data augmentation on top of the MC *StutterNet*, we found that except noise augmentation, all the other data augmentation helps in improving the macro $\mathcal{F}_1$ score in comparison MC *StutterNet* (clean) The noise augmented samples help in improving the detection accuracy of fluent class. For interjection, we found that all the data augmentation helps, and for block, only babble augmentation helps in improving their detection accuracy. We also found applying all four data augmentation in MC *StutterNet* results in an accuracy improvement in prolongation and repetition classes, however, with individual data augmentation, a drop in their accuracies can be observed in Table 5.3. By applying all four data augmentation in MC *StutterNet* training, there is an overall improvement of 1.76%, 17.15%, and 134% in repetitions, prolongations, and blocks respectively, which is a 4.48% relative gain in macro $\mathcal{F}_1$ score over the augmented *StutterNet*$_{MB}$ training.

5.4.5. Cross corpora evaluation

FluencyBank: By optimizing on SEP-28k dataset in terms of data augmentation and multi-contextual, we aim to evaluate our proposed methodology MC *StutterNet* on a cross corpora scenario. We trained MC *StutterNet* on SEP-28k dataset and evaluated it on *FluencyBank* dataset which comprises samples from 33 podcasts. We found that the model trained on one corpus (SEP-28k) fails to generalize and performs poorly on cross-domain corpora. As can be seen from the Table 5.4 and Figure 5.6, the $\mathcal{F}_1$ detection performance decreases remarkably from 43.86% to 38.92% employing clean training. The repetition and

			Accuracy						
Method	TrainSet	TestSet	R	P	B	In	F	TA	$\mathcal{F}_1$(%)
MC *StutterNet*	SEP-28k	SEP-28k	33.36	37.15	08.34	59.63	78.34	59.77	43.86
MC *StutterNet* (Clean)	SEP-28k	FluencyBank	19.48	35.80	01.83	56.36	80.85	56.48	38.92
MC *StutterNet* + A4	SEP-28k	FluencyBank	22.54	42.22	4.36	64.56	84.04	60.92	44.07

Table 5.4.: Results on cross corpora FluencyBank. The first row is from Table 5.3 which show the results on the same corpora SEP-28k dataset. The last two rows show the results, where the model is trained on SEP-28K and tested on FluencyBank in cross corpora setting. (B: Block, F: Fluent, R: Repetition, P: Prolongation, In: Interjection, $\mathcal{F}_1$(%): Macro F1 score), Bb: Babble, Rv: Reverberation, Mu: Music, No: Noise, A4: All four Bb + Rv + Mu + No augmentation.

block classes show more degradation in their performance in cross corpora scenarios. Furthermore, we applied data augmentation in cross corpora evaluation, and we found that it boosts the detection performance of all the classes, which results in an improvement of 13.23% in $\mathcal{F}_1$. The block samples from FluencyBank and SEP-28k dataset show very low detection performance and seem to be the hardest ones to detect.

			Accuracy				
Model	TrainSet	TestSet	RA	PA	FA	TA	$\mathcal{F}_1$(%)
MC StutterNet	LibriStutter	LibriStutter	93.36	76.26	98.19	96.11	91.0
MC StutterNet	SEP-28k (clean)	LibriStutter	0.65	0.48	99.95	77.25	30.0
MC StutterNet	SEP-28k + A4	LibriStutter	1.11	2.39	97.35	75.43	31.0
MC StutterNet	LibriStutter	SEP-28k	24.24	55.11	60.60	53.21	41.0

Table 5.5.: Results on simulated LibriStutter dataset with RA: Repetition accuracy, PA: prolongation accuracy, TA: Total accuracy, $\mathcal{F}_1$(%): Macro F1 score, A4: Babble + Reverberation + Music + Noise Augmentation.

Simulated LibriStutter: The experimental results on LibriStutter dataset are shown in Table 5.5. Experimenting and evaluating our MC *StutterNet* model on simulated LibriStutter dataset (Kourkounakis et al., 2021) results in a macro $\mathcal{F}_1$ score of $\approx$ 91%. However, the performance shows a considerable drop in macro $\mathcal{F}_1$ score when evaluated in cross corpora setting in comparison to real stuttering dataset such as FluencyBank. The MC *StutterNet* trained on SEP-28k dataset shows extremely poor performance when tested on simulated LibriStutter dataset. The simulated datasets such as LibriStutter looks like a choppy speech

and doesn't reflect the actual nature and characteristics of real world stuttered speech. So, a model that performs very well on simulated stuttered datasets can not be used in actual settings as such, as the speaker characteristics change to a very high extent in real clinical conditions.

Method	$\mathcal{F}_1$ (%)
StutterNet (baseline) (Clean) (Sheikh et al., 2021)	42.84
Class Imbalance	
StutterNet$_{WCE}$ (Clean)	41.02
StutterNet$_{MB}$ (Clean)	42.26
M$^{frz}_{enc}$ (Clean)	44.42
M$^{frz}_{enc,disf}$ (Clean)	44.80
Data Augmentation	
StutterNet + A4	45.30
StutterNet$_{WCE}$ + A4	44.34
StutterNet$_{MB}$ + A4	44.03
M$^{frz}_{enc}$ + A4	44.06
M$^{frz}_{enc,disf}$ + A4	45.76
Multi-Contextual	
MC StutterNet (Clean)	43.86
MC StutterNet + A4	**46.00**

Table 5.6.: Summary of macro $\mathcal{F}_1$ score of proposed methods (MB: Multi branch, WCE: Weighted cross entropy, M$^{frz}_{enc}$: Freezing encoder, M$^{frz}_{enc,disf}$: Freezing encoder and DisfluentBranch, MC: Multi-contextual, A4: All four augmentation (Bb + Mu + No + Rv).

5.4.6. Summary of proposed methods

This chapter discusses and addresses the deficiencies of *StutterNet* model that was presented in Chapter 4. In brief, we experiment with the following new modifications to our *StutterNet* baseline by first addressing the class imbalance problem in stuttering domain via weighted cross entropy and multi-branch training schemes. On top of that, we also investigated the impact of data augmentation in stuttering detection. And lastly, we exploited the different contexts for the identification of stuttering classes. The systematic improvement of all the proposed modifications can be seen in the Table 5.6, where the MC *StutterNet* outperforms against its single contextual framework in stuttering detection.

6. Multi-task and adversarial learning in stuttering detection

"Stuttering shows too much thinking. The human brain con only multitask efficiently at a certain level, beyond that the thoughts and signals get mixed up."

Roger W. Hancock

6.1. Introduction

The recent works have made significant strides in deep learning based stuttering detection. However, all the existing systems focus only on single task stuttering learning strategy without focusing on the auxiliary tasks. Unfortunately, this is not the approach that we human beings process the speech utterances, rather we simultaneously decode the primary task with other meta contents like speaker characteristics, linguistic content, emotion, etc. This multi-tasking (MTL) have been proven effective in computer vision, ASR, emotion detection, speaker recognition, etc., by jointly learning several tasks through a learned shared encoding (Vandenhende et al., 2021; Pironkov et al., 2016). In this chapter, we investigate the impact of MTL [1] approach in stuttering detection, by jointly learning stuttering and metada information. SD systems are further affected by source diversity, be that linguistic content, speaker, gender, accent, or encompassing acoustic conditions as discussed in Section 3.3. In addition to MTL, we also investigate an adversarial (ADV) framework with the aim to learn robust stuttering representations. Among the various source variabili-

[1] The following articles forms the basis of this chapter:

Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Robust stuttering detection via multi-task and adversarial learning". *In Proc. 30th European Signal Processing Conference (EUSIPCO)*, (2022) (peer-reviewed).

ties, we specifically focus on eliminating meta-data podcast (In this thesis, we assume each podcast is having a unique speaker and we use them as speaker ids, and we use the terms podcast and speaker interchangeably) information with the intention to improve the detection performance of stuttering and its types. Inspired by unsupervised domain adaptation training (Ganin and Lempitsky, 2015b) and speaker invariant affective learning (Li et al., 2020), we explore and provide the first ever deeper analysis by using MTL and ADV in the context of stuttering detection.

6.2. Motivation

MTL is basically an inductive transfer scheme which simultaneously learn multiple tasks by optimizing several objectives using a shared hidden representation (Caruana, 1997; Vandenhende et al., 2021). It provides an effective way of improving generalization. MTL aims to exploit the integrated knowledge across multiple domains with the intention of improving the performance of primary task (Crawshaw, 2020).

A contrasting approach is to presume that robust acoustic representation of stuttering should be invariant to secondary tasks, in particular speaker recognition, as stuttering speech task should not depend on a particular set of speakers. The way to achieve such robust invariances is ADV learning (Ganin and Lempitsky, 2015b). The ADV framework learns invariances to metada characteristics in a min-max fashion that the main branch is trained to maximize the metadata (podcast) classification loss and the speaker branch is trained to minimize its classification loss. Robust representation of speech signals have been studied via ADV training in various speech domains such as emotion detection (Li et al., 2020), acoustic matching (Su et al., 2020), and ASR (Meng et al., 2018a). However, there is no prior study on the application of ADV frameworks in the domain of acoustic SD.

6.3. Proposed framework

6.3.1. Multi-task learning

Within the MTL framework, given a training set of $D = (X_i, s_i, d_i, f_i)_{i=0}^{n}$, where each sample is comprised of three components: a sequence of $\mathcal{R}^{20\times T}$ acoustic MFCC fea-

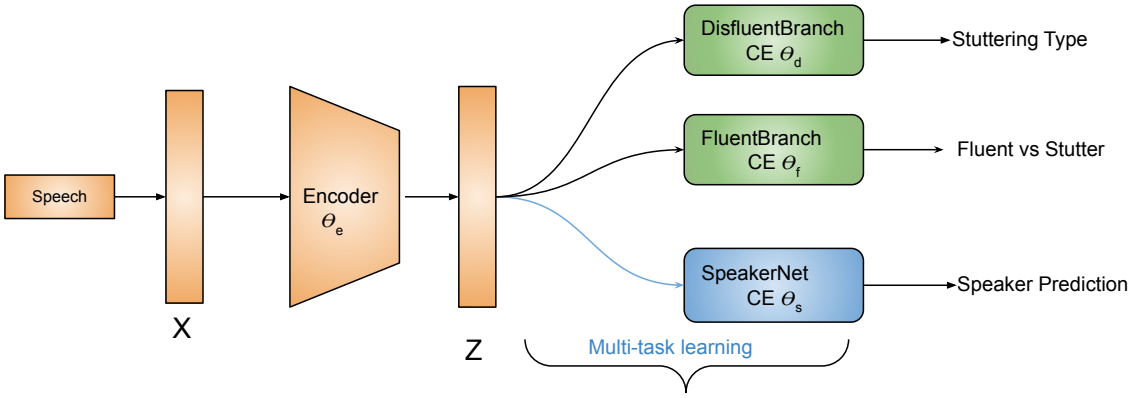


Figure 6.1.: Stuttering detection using MTL with light blue curve indicating secondary task which is speaker/podcast classification„ CE: Cross entropy, θ's are the parameters of each module, X: MFCCs input is $\mathcal{R}^{20\times T}$ dimensioanl features, Z: encoder output $= \mathcal{E}(X)$.

tures, a speaker label s_i, pseudo fluent label f_i (for *FluentBranch*, all different stuttering classes are treated as a single disfluent class by introducing a temporary pseudo-labelling scheme to solve the class imbalance problem) and a stutter label d_i, we consider a shared encoder $\mathcal{E}$ having parameters θ_e and three classifier branches S with parameters θ_s, $\mathcal{F}$ with parameters θ_f and $\mathcal{D}$ with parameters θ_d for podcast, fluent and stuttering sub type detections respectively. Our goal is to minimize both stutter loss $\mathcal{L}_{\text{stutter}}(\theta_\text{e}, \theta_\text{d})$ and speaker (podcast here) loss $\mathcal{L}_{\text{speaker}}(\theta_\text{e}, \theta_\text{s})$. Therefore, the total objective function in MTL as shown in Figure 6.1 can be represented as:

$$\begin{aligned}\mathcal{L}(\theta_\text{e}, \theta_\text{f}, \theta_\text{d}, \theta_\text{s}) =& (1-\lambda) * \mathcal{L}_{\text{stutter}}(\theta_\text{e}, \theta_\text{f}, \theta_\text{d}) + \lambda * \mathcal{L}_{\text{speaker}}(\theta_\text{e}, \theta_\text{s}) \\ \mathcal{L}_{\text{stutter}}(\theta_\text{e}, \theta_\text{f}, \theta_\text{d}) =& \mathcal{L}_{\text{fluent}}(\theta_\text{e}, \theta_\text{f}) + \mathcal{L}_{\text{disfluent}}(\theta_\text{e}, \theta_\text{d})\end{aligned} \tag{6.1}$$

where, $\lambda \in (0, 1)$, is a weighting parameter between the losses tuned manually.

6.3.2. Adversarial learning

In applying ADV framework to stuttering domain, we aim to learn acoustic representations which are robust and metada invariant, but at the same time are stuttering discriminative. For this purpose, similar to MTL, we consider an encoder $\mathcal{E}$ having parameters θ_e, which maps T input acoustic features to representation Z by $Z_i = \mathcal{E}(X_i)$. This Z_i representation is then fed to the *DisfluentBranch* (denoted by $\mathcal{D}$) and *FluentBranch* decoders

(denoted by $\mathcal{F}$) to output the stuttering and fluent class probabilities $p(d_i/Z_i;\theta_e,\theta_d)$ and $p(f_i/Z_i;\theta_e,\theta_f)$ respectively. In addition, this Z_i is fed to another branch speaker decoder (denoted by S), to output metadata (podcast) probabilities $p(s_i/Z_i;\theta_e,\theta_s)$. To learn metadata-invariant representations, our goal is to minimize stuttering classification loss, $\mathcal{L}_{\text{stutter}}(\theta_e,\theta_d)$ and to optimize θ_e to maximize the speaker classification loss, $\mathcal{L}_{\text{speaker}}(\theta_e,\theta_s)$ and, at the same time minimize the speaker classification loss by optimizing θ_s. The total objective loss becomes:

$$\begin{aligned}\mathcal{L}(\theta_e,\theta_f,\theta_d,\theta_s) &= \mathcal{L}_{\text{stutter}}(\theta_e,\theta_f,\theta_d) - \lambda * \mathcal{L}_{\text{speaker}}(\theta_e,\theta_s)\\ \mathcal{L}_{\text{speaker}} &= -\sum_{(X_i,f_i,d_i,s_i)\in\mathcal{D}} \log(p(s_i/Z_i))\\ \mathcal{L}_{\text{stutter}} &= -\sum_{(X_i,f_i,d_i,s_i)\in\mathcal{D}} \log(p(d_i/Z_i)) + \log(p(f_i/Z_i))\end{aligned} \tag{6.2}$$

where, $\lambda \in (0, 1)$, is a weighting trade-off hyper-parameter, and s_i, f_i and d_i are the labels. We optimize the parameters using Adam by applying the gradient reversal layer (Ganin and Lempitsky, 2015b) in speaker branch during backpropagation as shown by red curve in Figure 6.2.

6.3.3. Model architecture

Our stuttering detection models are based on time delay neural networks, which has been proven effective in various speech domains as discussed in Section 3.4.1. The model is fed with mean-normalized 20-dimensional MFCC input features extracted on a 20 ms sliding window with a hop length of 10 ms. The proposed adversarial model is based on MTL framework with four components including the encoder $\mathcal{E}$, which generates representations, the *FluentBranch* $\mathcal{F}$ classifier branch, the stutter classifier module named as *DisfluentBranch* D, and the speaker classifier S as illustrated in Figure 6.1 and Figure 6.2. The encoder module is composed of five time delay layers with the first three focusing only on small contextual frames of $[t-2, t+2]$, $\{t-2, t, t+2\}$, and $\{t-3, t, t+3\}$ respectively. The remaining two focuses on the $\{t\}$ contextual frames. The last layer of encoder is SP layer with concatenated mean and standard deviation. The fixed dimension representation from SP layer, which is the output of the $\mathcal{E}$, is further passed to the three sub-classifiers: *DisfluentBranch*, *FluentBranch* and *SpeakerNet*. These three sub modules are comprised of three fully connected layers and a softmax layer. A dropout of 0.2 is applied after first two fully

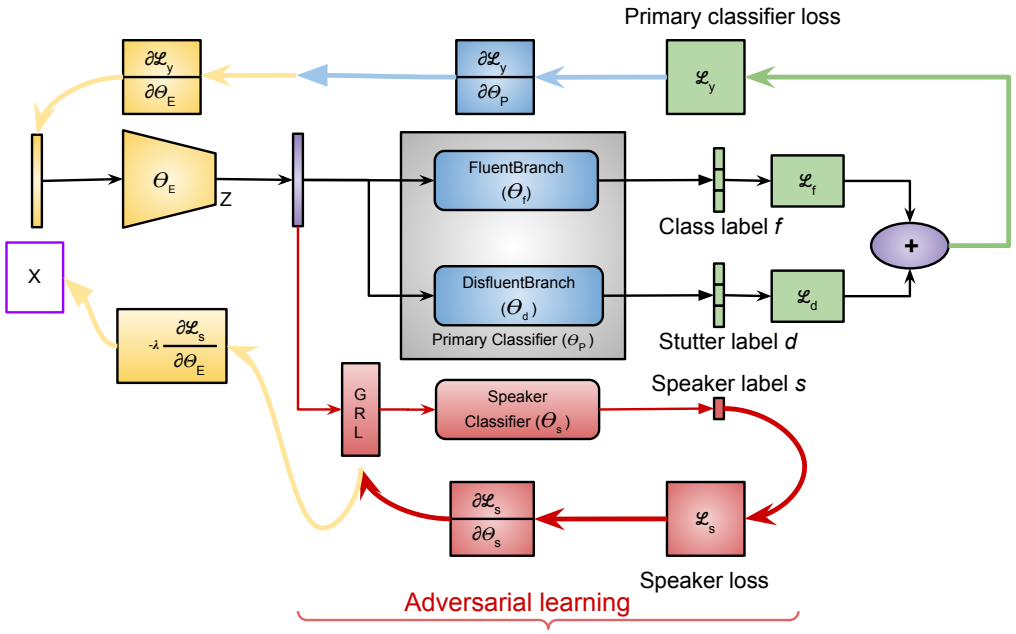


Figure 6.2.: Robust stuttering detection using adversarial MTL. Black arrows show forward propagation of primary task (stuttering detection). The top arrows including light blue, light yellow, and light green show backward propagation of primary classifier. The bottom red arrows show forward propagation in speaker classifier with bold ones are backward propagation, and GRL block indicates gradient reveresal CE: Cross entropy, θ's are the parameters of each module, X: MFCC input $\mathcal{R}^{20\times T}$, Z: encoder output, $\mathcal{E}(X)$. The primary classifier contains two sub-branches including *FluentBranch* ($\mathcal{F}$) and *DisfluentBranch* ($\mathcal{D}$).

connected layers in each branch. Except SP layer, each layer is followed by ReLU non linear activation function and a 1D batch normalization. To address the class imbalance problem, we use two branches *FluentBranch* and *DisfluentBranch* for SD. During inference time, speaker branch S_θ is discarded, and *FluentBranch* output is considered, if it predicts fluent, otherwise *DisfluentBranch* predictions are considered.

6.4. Experimental setup

Dataset: For our experimental studies, we have used the SEP-28k stuttering dataset discussed in Appendix A. Out of 28,177 samples, after leaving out non stuttering samples, we used only 23573 speech segments for our case study which constitutes of 19.65 hours of

		Accuracy							
Model		R	P	B	I	SA	F	TA	$\mathcal{F}_1$
StutterNet	SB	21.99	27.78	1.98	49.99	29.93	88.18	60.33	39.20
MB StutterNet	BL	28.70	37.89	9.58	57.65	37.72	74.43	57.04	41.20
MB StutterNet	MTL	31.59	31.62	10.23	58.92	38.32	72.14	56.09	40.60
MB StutterNet	ADV	27.24	32.89	8.33	56.36	35.96	77.10	57.51	40.80

Table 6.1.: Stuttering detection results (SA: Stutter accuracy, TA: Total accuracy, SB: Single branch, MB: Multi branch, $\mathcal{F}_1$: macro F1 score, BL: baseline, MTL: Multi-task learning, ADV: Adversarial learning)

data. The distribution of stuttering data is highly imbalanced, among which the majority of the samples (12419) are fluent segments constitutes 10.35 hours, 3995 are interjections constitutes 3.34 hours, 3286 are repetitions constitutes 2.74 hours, 2103 are blocks constitutes 1.75 hours, and 1770 are prolongations constitutes 1.48 hours of data. For our baseline, we randomly selected 80% of podcasts for training, 10% for validation and remaining 10% for test set which comprises of 309, 37 and 39 podcasts respectively. For MTL and ADV, we mix the train and validation sets from baseline, and then randomly selected 90% samples for train set and 10% samples for validation set from each podcast to have at-least each disfluency from each podcast in both the sets, and the test set remains same in all the cases. In this study, we use podcast ids as meta deta information to learn robust stutter representations.

Evaluation metrics: To test and evaluate the model performance, we use the widely and standard metrics that are used in stuttering and other speech tasks which include recall, precision, F1-score and accuracy. In addition, we also used macro F1 score as well. The models are evaluated and compared to our previous work *StutterNet* discussed in previous Chapter 2. The results reported are the average of 10 experiments with 10- fold cross validation technique. For ADV, we first train the *SpeakerNet* for 25 epochs by freezing the other two branches, then we freeze the *SpeakerNet* and train the *FluentBranch* and *DisfluentBranch* for 25 epochs. From epoch 50-75, we jointly train all the branches by applying gradient reversal layer in the speaker branch during backpropogation, and after epoch 75, we freeze all the weights and fine tune the *FluentBranch* and *DisfluentBranch* branches for the recovery phase until the stopping criteria is achieved.

Implementation: We have used PyTorch library for our implementation purposes.

		Precision				
Model	Method	R	P	B	I	F
StutterNet (Sheikh et al., 2021)	SB	0.22	0.28	0.02	0.50	0.88
MB StutterNet	BL	0.35	0.36	0.23	0.58	0.67
MB StutterNet	MTL	0.32	0.32	0.10	0.59	0.72
MB StutterNet	ADV	0.27	0.33	0.08	0.56	0.77
		Recall.				
StutterNet (Sheikh et al., 2021)	SB	0.42	0.42	0.25	0.69	0.62
MB StutterNet	BL	0.29	0.38	0.10	0.58	0.74
MB StutterNet	MTL	0.34	0.36	0.21	0.54	0.67
MB StutterNet	ADV	0.35	0.37	0.20	0.58	0.66
		F1 Score.				
StutterNet (Sheikh et al., 2021)	SB	0.29	0.33	0.04	0.57	0.73
MB StutterNet	BL	0.31	0.36	0.12	0.57	0.70
MB StutterNet	MTL	0.32	0.33	0.13	0.56	0.69
MB StutterNet	ADV	0.30	0.34	0.12	0.57	0.71

Table 6.2.: Stuttering detection results (B: Block, F: Fluent, R: Repetition, P: Prolongation, I: Interjection, MB: Multi branch, BL: Baseline, SB: Single branch).

We trained the models using Adam optimizer, cross entropy loss function with a learning rate of 10^{-2}. The training was stopped with an early stopping criteria having a patience of 7 on validation loss.

Baseline: Table 6.1 and Table 6.2 show the results of our baseline StutterNet along with the proposed methods. In comparison with the single branch (SB), multibranch (MB) outperforms in all the disfluency class predictions with an overall improvement of 26%. However, the SB shows better performance on fluent class because of the class imbalance issue. After addressing the class imbalance issue via two branch (*FluentBranch* and *DisfluentBranch*) training scheme, we found that the blocks and repetitions are still being confused with fluent speech as shown in Figure 6.3. This is intuitive because the repetitions are fluent speech with the same word being repeated more than once. Similarly, the blocks are fluent speech with some small initial pause followed by fluent speech. From the Table 6.1, we also observe that compared to the single task learning approach, our model based on simple MTL approach helps to boost the detection performance for stuttering types but

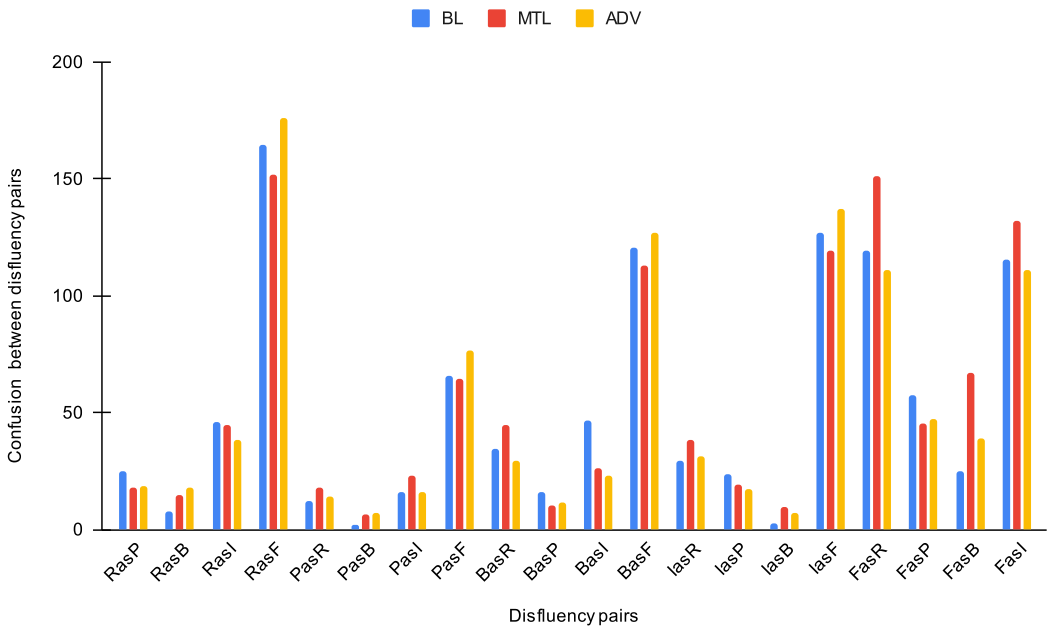


Figure 6.3.: Confusion in disfluency pairs in baseline (BL), MTL and ADV. XasY refers class X identified as class Y. For better visualization, we did not show in percent.

degrades for fluent and prolongation segments, as the confusion rate of fluent and prolongation samples increases with other disfluency pairs in the MTL framework as shown by the red bar plots in Figure 6.3. The MTL -based model increases the detection performance of R, B and I's by a relative margin of 10%, 6.78%, and 2% respectively, which results in an overall improvement by 1.6% in the stuttering classes over MB *StutterNet*. More interestingly, we found that the ADV improves the detection performance of fluent classes by 3.59% as compared to the baseline MB *StutterNet*. It can be clearly seen from the Figure 6.3, that there is a notable drop in the confusion rate of fluent category with other classes, but at the same time, the fluent false positive rate also increases. By adversarial training scheme, the model learns podcast-invariant representations which improves the performance of the fluent class.

Effect of lambda: Figure 6.4 (left) shows the result of MTL when the value of λ hyperparameter in the loss Equation (6.1) is varied from 0.1 to 0.9, and it includes the accuracy of all the classes. The results for all classes almost show a similar trend except for fluent

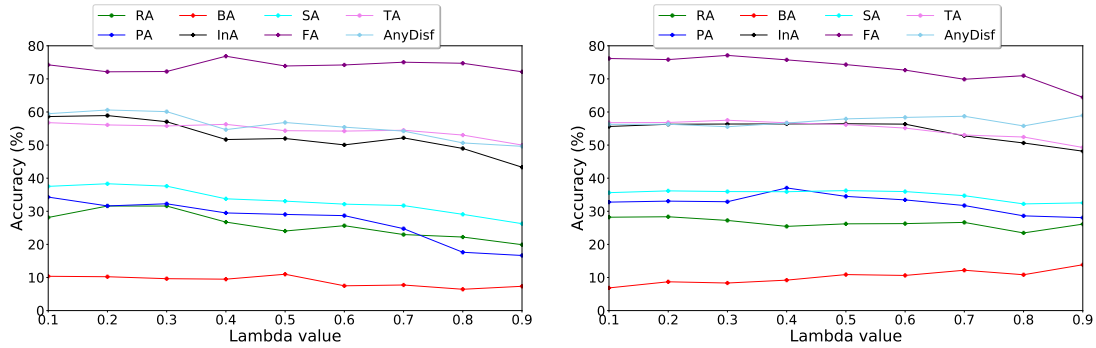


Figure 6.4.: Effect of lambda in MTL (left) and ADV (right) settings in SD with RA: Repetition accuracy, BA: Block accuracy, PA: Prolongation accuracy, InA: Interjection accuracy, SA: Average stutter accuracy, FA: Fluent accuracy, , AnyDisf: Any disfluent accuracy from two class *FluentBranch*.

class. The $\lambda = 0.4$ shows the best accuracy for the fluent class. As the λ value increases, the detection performance of all disfluent classes decrease as well as in fluent class, which is expected because the overall loss function assigns more weight to the podcast identification branch. In addition, we also tried to divide λ by 10 (starting with 1) at each epoch, and we found that it increases the detection performance of blocks slightly and repetitions considerably, with slight reduction in other classes. Figure 6.4 (right) shows the result of ADV training, with varying λ. The block class follows a trend that by increasing the λ, its detection performance also improves. For other classes, as the value of λ is increased from $0.1 \rightarrow 0.9$, the identification performance decreases. Following the work in (Ganin and Lempitsky, 2015a), we also tried varying λ by $2/(1+e^{p_i})-1$, where p_i is the scaled version of epoch. However, it degrades performance.

In addition, the two class accuracy of disfluent samples in the *FluentBranch* decreases as we increase the value of λ in the MTL framework. The λ acts as a control parameter for the podcast information to flow through the network. As λ increases, it is allowing more and more podcast-related information to flow through the network and thus decreasing the detection performance of stutter samples in the *FluentBranch*. To this contrary, the podcast identification branch in the ADV framework is trying to play a good adversary and tries to remove more irrelevant information as λ value increases. This, in turn, helps in boosting the detection performance of stutter samples in the *FluentBranch* branch. We can conclude that the podcast information is entangled with the stuttering characteristics, and by reducing

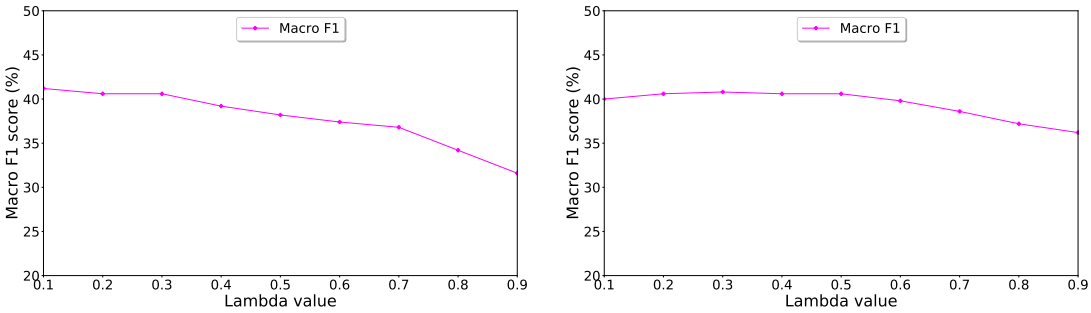


Figure 6.5.: Effect of lambda in MTL (left) and ADV (right) settings with overall macro F1 score in SD.

podcast-related information, the model is able to improve the discrimination between the fluent and disfluent classes, but it fails to generalize within the disfluent classes.

By analysing the macro F1 score ($\mathcal{F}_1$) from Figure 6.5, we can see that the $\mathcal{F}_1$ score decreases sharply in MTL stuttering detection setting as the λ value increases than in adversarial learning framework. As the λ value increases, the network allows more and more speaker information to pass through it during backward propagation, thus trying to dilute the latent stutter embeddings in feature encoder and results in decrease of $\mathcal{F}_1$ sharply. In contrast to MTL, with the increase in λ value, adversarial learning framework tries to unlearn more and more speaker information with an aim not to dilute the latent stutter embedidngs learned from *FluentBranch* and *DisfluentBranch* training setup.

Visualization: Figure 6.6 shows the embeddings computed from the statistical pooling layer. In MTL scheme, it is evident from the well formed podcast clusters that the model is trying to learn podcast dependent stuttering information. On the other hand, for the adversarial setting, the clusters are not visible and the model is trying to learn this meta-data invariant robust stutter features.

6.5. Summary

Due to the diversity and uniqueness of stuttering, it continues to be the most demanding and challenging to detect due to its inherent nature of huge class imbalance across different types of disfluencies. Training a model on such a type of dataset will bias the majority class. To this end, we employ an MB *StutterNet* (BL) in thic chapter, where, we introduce

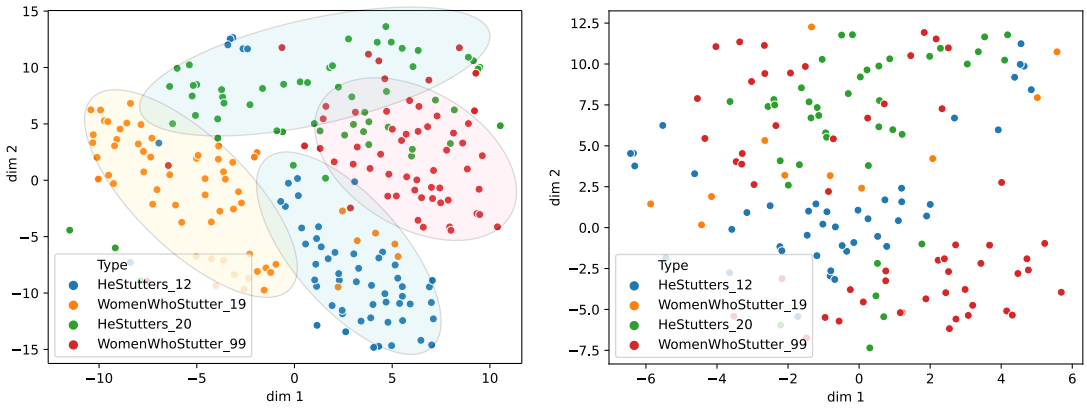


Figure 6.6.: t-SNE embeddings from the shared encoder MTL (left) and ADV (right). We selected four podcasts/speakers (type) randomly for visualisation purposes. Clear speaker clusters are visible in MTL framework where the adverarial MTL is trying to learn speaker independent stutter latent representations. The labels represent the speakers with *HeStutters* is a male speaker and *WomeneWhoStutter* is a female speaker.

the pseudo temporary labeling of all disfluent classes as one class. The MB *StutterNet* is a two-branch network, with one branch to differentiate fluents from disfluent and the other branch to differentiate different types of disfluent types if the prediction in the *FluentBranch* is non-fluent. On top of the baseline, we also investigate the MTL and ADV framework by adding one more speaker branch in the context of SD. From the results, we found that the MTL framework improves the detection performance of blocks, repetitions, and interjections. We also found that the confusion rate increases for the fluent and prolongation classes. With the ADV scheme, we found that the fluent detection performance increases with a relative improvement of 3.59%. In addition, we also found that by decreasing the λ hyper-parameter in both the frameworks, the detection rate of disfluent classes increases. There might be issues with the meta-information of the dataset, as there is a possibility of having the same speakers across multiple podcasts.

7. Self-supervised learning in stuttering detection

"Without big data, you are blind and deaf and in the middle of a freeway."

Geoffrey Moore

7.1. Introduction

The automated stuttering detection[1] task suffers from a lack of unlabeled data and thus limits the application of large deep models. Over the years, several stuttering datasets including: SEP-28k (Lea et al., 2021), UCLASS (Howell and Sackin, 1995), FluencyBank (Lea et al., 2021), and LibriStutter (Kourkounakis et al., 2021) have been developed for investigating different SD models. Due to a number of variable characteristics, such as language, speaker, gender, age, accent, speech rate, and others, the identification of stuttering is still a challenging task. We exploited the speaker variable and employed an adversarial multi-tasking learning framework to learn speaker invariant stuttering features (discussed in previous Chapter 6). Moreover due to this varying nature of stuttering from person to person, these small datasets are inclined to be biased towards these small pool of speakers (Guitar, 2013). Although deep learning has shown considerable improvement in several areas, this improvement in SD is nevertheless limited, most likely due to the limited size of stuttering

[1] This chapter is based on the following articles:
Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "Introducing ECAPA-TDNN and Wav2Vec2.0 embeddings to stuttering detection". *Under review in International Journal of Speech Technology*, (2022) (under review).
Shakeel Ahmad Sheikh, Md Sahidullah, Fabrice Hirsch, and Slim Ouni. "End-to-end and self-supervised learning for ComParE 2022 stuttering sub-challenge". *In Proc. 30th ACM International Conference on Multimedia*, (2022), pp 7104-7108 (peer-reviewed).

datasets, which are unable to capture different speaking styles, accents, linguistic content, etc. In addition, collecting medical data requires big-budget and is very taxing, and stuttering data collection is no exception. To address this issue, we introduce a self-supervised learning framework that first learns a feature extractor for a pre-text task on a large amount of non-stuttering audio data and then employs the learned feature extractor for the downstream stuttering detection task with limited audio data. We mainly focus on speaker and contextual embeddings where the self-supervised learning (SSL) models are pre-trained on massive datasets with the hypothesis in capturing various speaking styles and other paralinguistic information which are extremely important to stuttering detection.

7.2. Speaker and contextual embeddings

7.2.1. Speaker embeddings

Speaker embeddings are usually computed from trained neural networks to identify and classify speakers from a group of speakers (Desplanques et al., 2020). The pre-trained speaker embeddings have been successfully applied in speaker diarization (Dawalatabad et al., 2021). The (ECAPA-TDNN) emphasized channel attention, propagation and aggregation TDNN is the state-of-the-art for extracting speaker embeddings. As depicted in Figure 7.1, the ECAPA-TDNN is composed of 1D convolution followed by three 1D squeeze and excitation (SE) Res2Blocks, 1D convolution, attentive statistical pooling, and an FC layer. A non-linear ReLU activation and a batch normalisation is applied after each layer in SE-Res2Block. The model is fed with 80 dimensional mean normalized log Mel-filterbank energy features and is trained on a large Voxceleb dataset with additive angular margin (AAM)-softmax loss function using Adam optimizer having a cycling learning rate between 1e-8 and 1e-3. In this work, we use ECAPA-TDNN as a feature vector on SEP-28k stutter dataset to exploit it for SD. We extract $\mathcal{R}^{1\times 192}$ speaker embedding feature vector from the FC layer of ECAPA-TDNN as shown by the purple block in Figure 7.1.

7.2.2. Wav2Vec2.0 contextual embeddings

The WaveVec2.0 model is a self-supervised representation learning framework of raw audio, and is comprised of three modules including feature encoder $\mathcal{E}: X \mapsto \mathcal{Z}$, contex-

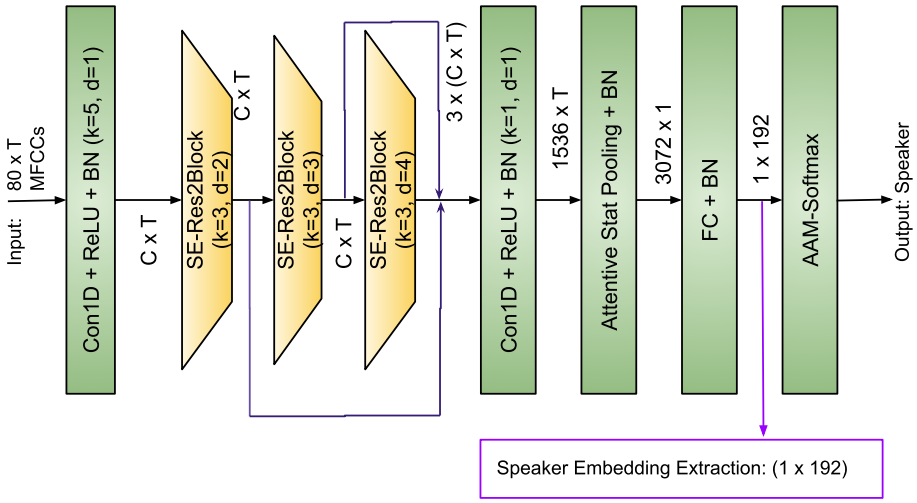


Figure 7.1.: ECAPA-TDNN block diagram, k is kernel size, d is dilation, C is channel, T is temporal dimension, FC is fully connected, BN is batch normalization, AAM is additive angular margin, input is 80-dim MFCC features, SE-Res2Block is squeeze-excitation block with variant of residual (Res2) dilated convolution in between two one-dimensional convolution layers (Desplanques et al., 2020).

tual block transformer $g: \mathcal{Z} \mapsto \mathcal{C}$ and quantization block $\mathcal{Z} \mapsto \mathcal{Q}$ as depicted in Figure 7.2 (Baevski et al., 2020). The feature encoder is comprised of multi-layer 1D convolution blocks followed by Batch normalisation (BN) and Gaussian error linear unit (GELU) activation functions, takes the normalized raw input X and encodes it into local feature representations $\mathcal{Z} = \mathcal{E}(X)$. These encoded feature representations of size $\mathcal{Z}^{T \times 768}$ are then fed to contextual transformer block to learn contextual speech representations $\mathcal{C} = g(\mathcal{Z})$. The paper uses two different transformer networks with the base model consisting of 12 blocks having eight attention heads at each block, and the large model is comprised of 24 blocks with 16 attention heads at each block. The feature representations $\mathcal{Z}$ are also fed to the quantization module which is comprised of two codebooks having 320 possible entries in each. For each vector representation $z_i \in \mathcal{Z}$, a logit of $\mathcal{R}^{2 \times 320}$ is chosen using (Equation (7.1)) by concatenating the corresponding entries from each codebook, which is then followed by linear transformation to produce the quantized vector q_i of the local feature encoder representation $z_i \in \mathcal{Z}$.

$$p_{g,v} = \frac{\exp(l_{g,v} + \eta_v)/\tau}{\sum_{k=1}^{V} \exp(l_{g,v} + \eta_v)/\tau} \tag{7.1}$$

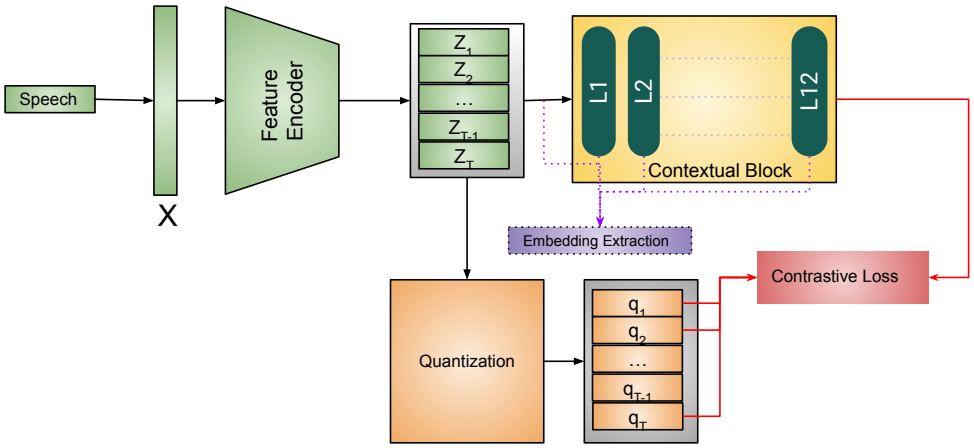


Figure 7.2.: Block diagram of Wav2Vec2.0 SSL architecture. We extract embeddings (shown in purple) from the contextual block which is trained in a self-supervised fashion using quantisation module with X is input raw signal.

where l is logit, v is v-th codebook entry, g is codebook group, $\eta = -\log(-log(u))$ with u are uniform samples from $\mathcal{U}(0,1)$, and τ is the temperature which controls the randomness.

Similar to the masked language modelling, the model is pre-trained in a self-supervised fashion using (Equation (7.2)) by randomly masking certain time stamp representation vectors of the feature encoder and the training objective is to reproduce the quantized $\tilde{q}_t$ latent speech representation from a set of K+1 distractors including candidate vector q_t and K distractors $\in$ Q for masked time stamp vectors at the end of contextual transformer block. The distractors are uniformly sampled from masked frames of the same speech utterance.

$$\mathcal{L}_{cont} = -\log \frac{\exp(sim(c_t, q_t)/\tau)}{\sum_{\tilde{q}\in Q} \exp(sim(c_t, \tilde{q})/\tau)} \tag{7.2}$$

where $sim(c_t, q_t) = \frac{c_t^T q_t}{\|c_t\|\|q_t\|}$ computes the cosine similarity between the quantized vector q_t and contextualized transformer vector c_t.

The authors have released several pre-trained feature embeddings with dimensions of 768 (base)[2] and 1024 (large) pre-trained on 960 hours of LibriSpeech dataset and then fine-tuned for ASR using Connectionist temporal classification (CTC) loss function by adding a linear layer on top of the contextual block.

The Wav2Vec2.0 showed remarkable improvement in ASR (Baevski et al., 2020),

[2]We are using (768-dimensional) base version of Wav2Vec2.0

emotion detection (Pepino et al., 2021). In the stuttering speech, most parts of the speech utterance are perturbed, and it seems a plausible way to employ and explore the role of the contextual and encoder representations in SD. In this work, without fine-tuning, we employ a total of 13 contextual embeddings extracted from a local encoder and 12 layers of the contextual transformer block as depicted in Figure 7.2.

7.3. Classifier description

7.3.1. k-nearest neighbourhood

A non-parametric supervised algorithm based on distance metric is used mostly for classification tasks. The prediction of the query sample depends on the voting majority of its k nearest neighbors (Murphy, 2012). In this work, we use the Minkowski metric distance from (Equation (7.3)) with $p = 2$ (Euclidean distance) to fit the k-NN on the SEP-28k dataset using embeddings computed from pre-trained ECAPA-TDNN and Wav2Vec2.0 models.

$$Dist = \left[\sum_{i=1}^{k}(\|x_i - y_i\|)^p\right]^{1/p} \tag{7.3}$$

7.3.2. Gaussian back-end

A NBC is simply a Bayesian network to handle continuous features by representing the likelihood of features using Gaussian distribution (Murphy, 2012). Given a data set $D = (X_i, d_i)$ of N samples with $\mathcal{R}^{1\times K}$ (K is 768 for Wav2Vec2.0 and 192 for ECAPA-TDNN) pre-trained features, NBC assumes that the likelihood of class conditional densities is normally distributed by

$$p(e|C = c, \mu_c, \Sigma_c) = \mathcal{N}(e|\mu_c, \Sigma_c) \tag{7.4}$$

where $e \in X$, is extracted pre-trained representation, μ and Σ are class-specific mean vector and covariance matrix respectively. The posterior probability for each target class is computed then by Bayes' formula:

$$\underbrace{p(C = c|e, \mu_c, \Sigma_c)}_{\text{class posterior}} = \frac{\overbrace{p(e|C = c, \mu_c, \Sigma_c)}^{\text{class conditional likelihood}} p(C = c)}{\sum_{i=1}^{K} p(e|C = i, \mu_i, \Sigma_c)p(C = i)} \tag{7.5}$$

and then the query sample e is classified by taking *argmax* over the classes.

$$\hat{y}(e) = \text{argmax}\, p(C = c|e, \mu_c, \Sigma_c) \tag{7.6}$$

7.3.3. Shallow neural network for stuttering detection

A NNet with just a few layers can also be applied on top of the pre-trained embeddings extracted from ECAPA-TDNN and Wav2Vec2.0 models. In this work, we use two-branched NNet with *FluentBranch* differentiating between fluent and stuttered utterances, and *DisfluentBranch* classifying stuttered speech utterances into several disfluency types. Each branch is composed of three FC layers with each layer followed by ReLU activation and 1D BN functions. A dropout of 0.2 is applied for first two FC layers. A softmax layer is used in the end to get the desired classes. Following the approach in (Lea et al., 2021), for *FluentBranch*, we employ a pseudo labelling scheme, where we re-label all different disfluent speech samples as one class and train the binary *FluentBranch* branch to differentiate between fluent and stuttered utterances. For the multi-class *DisfluentBranch* branch, we train it by penalizing the fluent class with zero loss. During evaluation phase, the outcome from *FluentBranch* is considered if the prediction is fluent class, otherwise predictions from *DisfluentBranch* are taken into consideration.

7.4. Experimental setup

7.4.1. Embedding extractors

ECAPA-TDNN: For each three-second speech utterance of the SEP-28k dataset, we extract speaker embeddings of dimension $\mathcal{R}^{1\times192}$ after the FC layer of ECAPA-TDNN as shown by the purple line in Figure 7.1, resulting in $\mathcal{R}^{N\times192}$ data (N is total samples). We use SpeechBrain toolkit (Ravanelli et al., 2021) for the extraction of 192-dimensional speaker embeddings. In addition, we also use Linear discriminant analysis (LDA) for dimensionality reduction with the component size of four resulting in $\mathcal{R}^{N\times4}$ dimensional data before passing it to the downstream classifiers.

Wav2Vec2.0: For contextual embedding extraction of each SEP-28k sample, we extract $\mathcal{R}^{T\times768}$ (T is a temporal dimension) dimensional representations from local feature encoder $\mathcal{Z}$ and from each layer of contextual transformer block $\mathcal{C}$, each of which yields differ-

ent contextual representation as shown by purple block in Figure 7.2. Before passing each layer representation separately to the downstream classifiers, we apply statistical pooling across the temporal domain and concatenated the mean and standard deviation resulting in a feature vector of $\mathcal{R}^{1\times768\times2}$. Moreover, we also concatenate the contextual embeddings of the local encoder (L0), L6, and layer L10 of $\mathcal{C}$ after applying the statistical pooling and LDA (with component size of four) which results in $\mathcal{R}^{N\times12}$ feature vector. We extract embeddings using the PyTorch version of Wav2Vec2.0.

7.4.2. Dataset description

In this case study, we use SEP-28k stuttering dataset (Lea et al., 2021) which consists of 28,177 speech samples from 385 podcasts. After removing non-stuttered samples, we are left with 23573 annotated speech segments. We randomly selected 80% podcasts (without mixing the podcasts) for training, 10% podcasts for validation and the remaining 10% of the podcast for evaluation in a 10-fold cross-validation scheme. The speaker information is missing from the SEP-28k dataset, so we divided the dataset based on podcast ids (assuming each podcast is having a unique speaker). This dataset contains two different types of annotations including stuttering and non-stuttering. We considered only stuttering annotations (repetitions, blocks, interjections, prolongations, and fluent speech) and avoided the non-stuttering annotations (unsure, no speech, poor audio quality, music, unintelligible and natural pauses) in this experimental study. The SEP-28k dataset is publicly available, but the details about the train, validation and test set splits are not provided. So, we decided to create a publicly available protocol that ensures no overlap of podcasts between train, validation and test sets[3]. The experimental results mentioned are the average of 10-fold cross-validation experiments, and are compared to the baseline results from Chapter 4 which is trained on MFCC features.

7.4.3. Implementation

For implementing the proposed pipeline for NNet, we use PyTorch library, and for LDA, k-NN, and NBC, we have used the Scikit-learn toolkit. We have chosen a value of $k = 5$ using the elbow method in the k-NN classifier. The downstream NNet is trained with

[3]For splitting of the dataset, please see protocol `https://shakeel608.github.io/protocol.pdf`

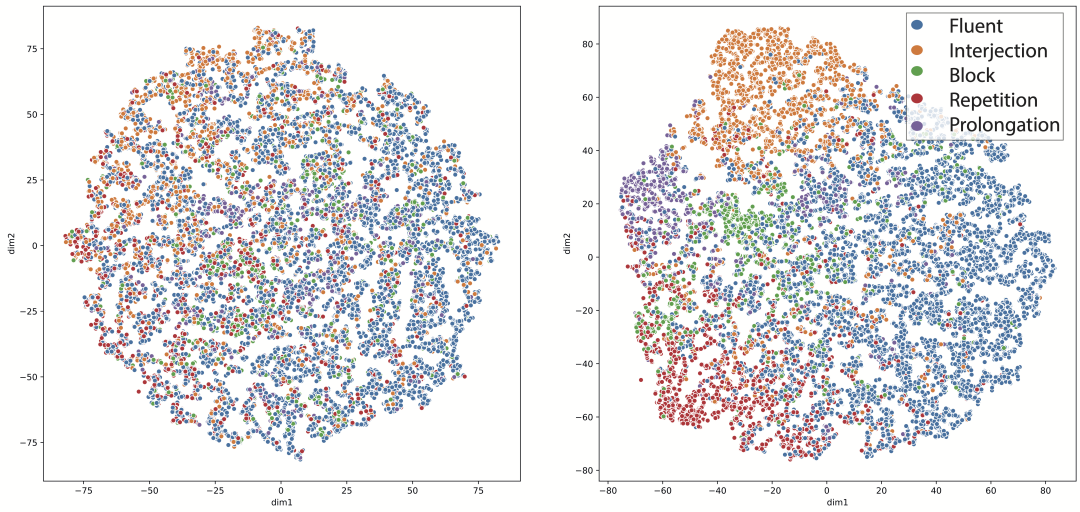


Figure 7.3.: t-SNE embeddings from the FC layer of ECAPA-TDNN (left) and layer L10 of the contextual block $\mathcal{C}$ of Wav2Vec2.0 (right). The Wav2Vec2.0 embeddings show clear visible clusters among disfluent classes.

a batch size of 128 using a normal sum of cross entropy loss ($L_{tot} = L_f + L_d$, L_f:*FluentBranch* loss, L_d:*DisfluentBranch* loss) function optimized by the Adam optimizer with a learning rate of 1e-2, and the training is stopped using an early stopping scheme on validation loss with the patience of seven.

7.5. Results and discussion

For thorough evaluation, we train each of the proposed models 10 times. Table 7.1. shows the macro $\mathcal{F}_1$, average UAR and accuracy results of different stuttering and fluent classes on exploiting different features extracted from ECAPA-TDNN and Wav2Vec2.0 models. We compare the proposed speaker and contextual embeddings based SD models to the work ResNet+BiLSTM (Kourkounakis et al., 2020), *StutterNet* (discussed previously in Chapter 4) (For a fair comparison, we trained these on the SEP-28k dataset with MFCC input features) and to the multi-branched *StutterNet* (discussed previously in Chapter 5) having two different branches with one branch differentiating between fluent and stutter utterances and the other branch distinguishing among different disfluency types. We use LDA for dimensionality reduction of features in each case before passing them to the

classifiers for prediction.

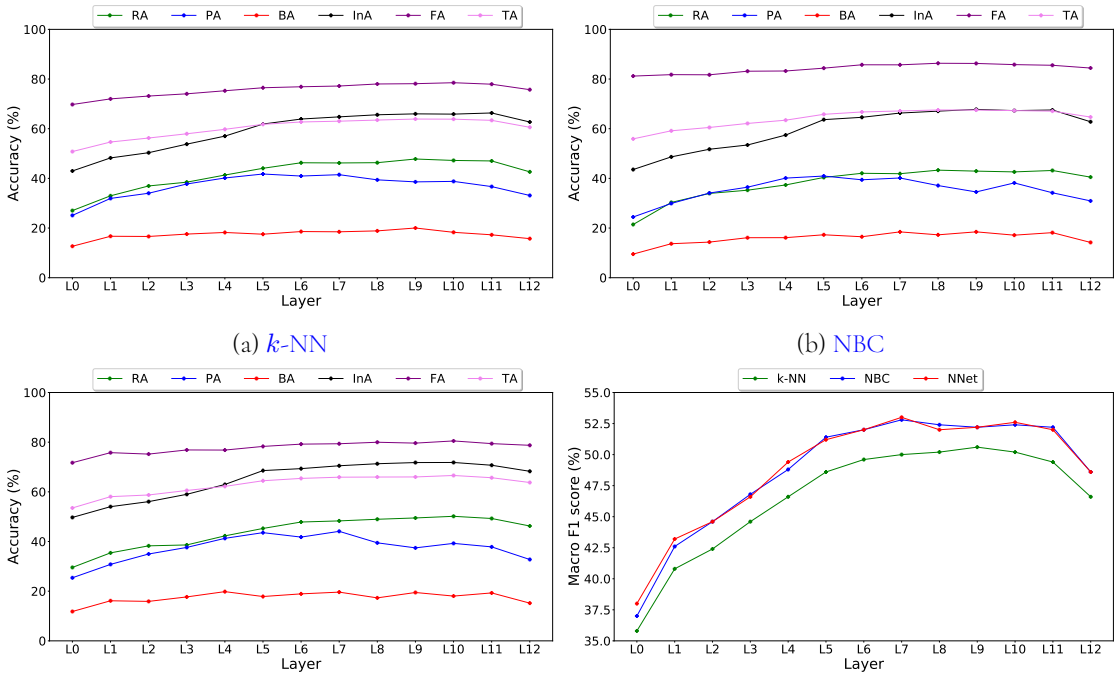
Speaker embeddings: We see from Table 7.1 (b), that the downstream classifiers trained on ECAPA-TDNN embeddings perform poorly in all the stuttering classes as compared to the baseline results from Table 7.1. This is evident from the Figure 7.3 as well, where the different stuttering type utterances are mixed and no clear cluster is visible among the disfluency classes. Furthermore, applying magnitude normalization on ECAPA-TDNN embeddings before passing them to the downstream classifiers, improves the SD performance marginally in the majority of classes. The ECAPA-TDNN is trained and adapted for the speaker identification task and it is likely possible that the information (such as linguistic content, prosody, and emotion state) which isn't essential for that task but could be crucial for SD gets removed from the latent embeddings.

Wav2Vec2.0 contextual embeddings: Table 7.1 (c) shows the results with the last but two-layer of contextual transformer block $\mathcal{C}$ of Wav2Vec2.0 with and without prior application of LDA. From the results, we can observe that for SD, the contextual embeddings from Wav2Vec2.0 outperform in all the classes with an overall relative improvement (TA) of 5.82% using k-NN, 11.54% using NBC, 10.38% using NNet over *StutterNet* and over MB *StutterNet* by 11.92% using k-NN, 17.97% using NBC and 16.74% using NNet downstream classifiers. Figure 7.4. shows the impact of different contextual embeddings in the detection of various stuttering classes. The plot shows almost a close trend in all the stuttering class accuracies. The detection accuracy of stuttering classes increases with the layer[4] number. We hypothesize that the lower layers including local encoder (L0) representations contain speech information only from the local window of size 25 ms, and, in addition, passing the representations to the downstream classifiers after applying the statistical pooling layer further restricts it in capturing more stutter specific patterns. In addition, the results show that the contextual layers from L5 to L11 of the Wav2Vec2.0 model trained in a self-supervised fashion are able to capture rich stuttering patterns as also depicted in Figure 7.3. As for the last layer (L12), it slightly degrades performance in SD in comparison to its previous layer due to the fact that the transformer block $\mathcal{C}$ was fine-tuned and adapted towards the ASR task. By fine-tuning towards ASR, it is possible that the Wav2Vec2.0 model has not focused on the information which is relevant to stuttering, resulting in the loss of rich stuttering

[4] L0 is a local encoder, L12 is the last layer of $\mathcal{C}$

information. Consider such an example of prosodic information, which is very essential in SD, but not that important for ASR. Using Wav2Vec2.0 embeddings with NNet in SD, there is an overall relative improvement of 61.36%, 47.91%, 14.65%, and 9.02% in repetitions, blocks, interjections, and fluents respectively over the MB *StutterNet*, thus outperforms over the state-of-the-art results.

Moreover, the prior application of LDA on Wav2Vec2.0 representations further boosts the detection performance in repetitions by 8.29%, prolongation by 10.41%, blocks by 27.24%, and interjections by 8.72%.



(a) *k*-NN

(b) NBC

(c) Shallow NNet

(d) Average macro F1 score.

Figure 7.4.: Impact of various Wav2Vec2.0 contextual layers in stuttering detection with L0 is local encoder layer, L1 → L12 are contextual layers, RA is repetition accuracy, PA is prolongation accuracy, BA is block accuracy, InA is interjection accuracy, FA is fluent accuracy, and TA is total accuracy.

Fusion: In addition, we fuse the ECAPA-TDNN and Wav2Vec2.0 embeddings via score and embedding fusion schemes, the results of which can be seen in Table 7.1 (d, e). While computing the final score p from ECAPA-TDNN and Wav2Vec2.0 prediction probabilities, we empirically optimize the weighting parameter α on test set in $p = \alpha * p_{w2v2} +$

Model	Accuracy							
	R	P	B	I	F	TA	UAR	$\mathcal{F}_1$
a) MFCCs (Baselines)								
StutterNet	21.99	27.78	1.98	49.99	88.18	60.33	48.00	39.20
MB *StutterNet*	28.70	37.89	9.58	57.65	74.43	57.04	41.80	41.20
MB (Kourkounakis et al., 2020)	34.79	30.19	5.92	49.26	75.47	55.62	39.13	39.20
b) Embedding: ECAPA-TDNN								
k-NN	17.82	6.40	6.09	26.58	71.55	45.73	25.80	25.60
NBC	22.15	13.84	9.42	31.46	63.57	44.04	28.00	28.20
NNet	23.06	11.67	5.79	43.29	69.48	49.12	33.20	30.80
k-NN + LDA	21.92	11.93	8.56	26.6	66.77	44.37	27.40	27.40
NBC + LDA	11.99	7.24	2.52	25.43	88.77	53.53	27.20	26.80
NNet + LDA	24.51	10.33	5.03	44.49	68.73	48.81	32.40	30.20
c) Embedding: Wav2Vec2.0								
k-NN	24.48	8.88	11.33	54.02	84.10	58.85	36.40	37.60
NBC	41.62	17.33	35.75	36.88	60.96	48.74	38.60	38.00
NNet	46.31	35.55	14.17	66.07	81.14	64.69	52.20	49.00
k-NN + LDA	47.22	38.79	18.28	65.87	78.52	63.84	49.80	50.20
NBC + LDA	42.61	38.17	17.13	67.29	85.81	67.29	50.20	52.40
NNet + LDA	50.15	39.25	18.03	71.83	80.50	66.59	53.80	52.60
d) Score fusion: ECAPA-TDNN and Wav2Vec2.0								
k-NN + LDA	44.06	36.95	16.13	64.61	82.70	65.18	49.00	50.20
NBC + LDA	41.75	38.63	16.60	66.83	86.78	67.73	50.40	52.60
NNet + LDA	43.58	38.94	19.11	67.36	84.63	67.26	50.80	52.60
e) Embedding fusion: ECAPA-TDNN and Wav2Vec2.0								
k-NN + LDA	45.38	37.29	17.92	62.18	80.68	64.08	48.60	50.00
NBC + LDA	44.22	40.01	19.70	67.61	84.55	67.44	51.40	53.20
NNet + LDA	44.13	38.02	18.39	66.44	84.53	67.00	55.20	52.00
f) Embedding fusion: L0 + L6 + L10								
k-NN + LDA	46.98	42.18	20.96	66.28	82.24	66.30	51.60	53.00
NBC + LDA	48.46	47.84	28.51	70.41	80.33	67.45	55.00	55.60
NNet + LDA	46.79	40.79	23.86	69.54	84.32	**68.35**	**57.20**	54.60

Table 7.1.: SD results on SEP-28k stuttering dataset (TA: Total accuracy, B: Block , F: Fluent , R: Repetition , P: Prolongation , I: Interjection, UAR: Unweighted average recall, $\mathcal{F}_1$: Macro F1 score, BLs: Baselines, MB: Multi branch) for different methods. The results reported for Wav2Vec2.0 are from L10.

$(1-\alpha)*p_{ecapa}$ and we found $\alpha = 0.9$ gives the best results. The ECAPA-TDNN representations which contain rich information about speakers' identity further enhances the overall detection performance of 2.1% using k-NN, 0.65% using NBC, and 1% using NNet. However, it doesn't contain enough rich information about suprasegmental, emotional content, etc., which are incredibly important for disfluent classes, thus acting as a negative transfer for some of them. Moreover, concatenating the ECAPA-TDNN and Wav2Vec2.0 speech embeddings results in a further relative improvement of 8.66% in UAR using NNet.

Each layer of the Wav2Vec2.0 model contains different speech representations, exploiting this fact, we integrate the representations after applying LDA (of component size four) from the local encoder (L0), L6 and L10 from contextual block $\mathcal{C}$, resulting in a $\mathcal{R}^{N\times 12}$ dimensional data. Concatenating information from multiple layers further improves the minority class recognition including prolongations and blocks by a relative margin of 7.29% and 29.74% respectively using NNet. Moreover, the embedding fusion also enhances the UAR by a relative margin of 19.17%, 36.84%, 46.18% over the MFCC-based baselines *StutterNet*, MB *StutterNet*, and MB ResNet+BiLSTM (Kourkounakis et al., 2020) respectively mentioned in Table 7.1 (f).

We found that combining speaker embeddings with contextual Wav2Vec2.0 embeddings acts as a negative transfer for most of the disfluent classes except fluent category as shown in Table 7.1 (d, e). The stuttering characteristics vary from person to person and speaker embeddings in ECAPA-TDNN are trained in a manner to capture unique speaker attributes without focusing or giving much attention to other meta-data such as linguistic content and prosodic information. Usually, the Wav2Vec2.0 model captures acoustic information in the initial layers of contextual block, while the middle layers are more suitable for semantic, phonetic and class-related information (Mohamed et al., 2022; Baevski et al., 2021). Based on the results from Table 7.1 (f) and Figure 7.4 (d) where the middle layers show more $\mathcal{F}_1$ score for stuttering detection, it seems reasonable to hypothesize that the stuttering is mostly related to phonetic and semantic information than the acoustic information which are captured in initial layers of Wav2ec2.0 contextual embeddings. An interesting question might arise "finding the impact of finetuning Wav2Vec2.0 model in stuttering detection domain in the same language and cross language settings i.e., Is stuttering uniform across cross lingual settings or it does vary with the language"?. This is a very interesting

research question and it needs to be studied and evaluated deeply in collaboration with the speech therapists and phoneticians.

7.6. Evaluation on KSoF dataset in ComParE 2022 stuttering sub-challenge

The ACM Multimedia 2022 ComParE proposes four sub-competitions including Vocalisations, *Stuttering*, Activity, & Mosquitoes (Schuller et al., 2022). Among these, we only focus on the *Stuttering* (KSF-C) sub-challenge. In this sub-challenge, the task was to classify the speech segments into one of the eight categories including filler, garbage, prolongation, sound repetition, block, modified, word repetition, and no_disfluency (fluent).

7.6.1. KSoF

The KSoF is a German stuttered dataset collected recently and is comprised of 5597 three-second stuttered speech clips extracted from 214 recordings of 37 PWS with 28 males and 9 females (Bayerl et al., 2022a). This dataset contains a recording from three types of environments including telephonic conversation, read speech, and spontaneous speech. After collecting the stuttered recordings, the samples were downsampled to 16 kHz and were labelled by three annotators (please refer Appendix A). More than one label was assigned to some of the stuttered samples. However, in this ComParE challenge, after removing ambiguous samples, only 4601 stuttered samples were provided. For this task, we eliminated all parts with unclear labels, leaving only 4601 segments. The data is split into speaker disjoint sets with train set having 23 speaker, validation set having 6 speakers, and test set with 8 speakers.

7.6.2. Proposed framework

For this ComParE challenge, we use only the base pre-trained model of 768 embedding dimensions (trained on 960 hours of data) and we extract embeddings from the $\mathcal{E}$ block and from 12 layers of $\mathcal{C}$ block for SD. Moreover, we use statistical pooling over the temporal domain and concatenated the mean and standard deviation, resulting in a 768×2-dimensional feature vector before passing it to the downstream classifiers except

Class	Train	Val	Test	$\sum$
B	310	102	176	588
SR	169	038	132	339
Fi	205	104	083	392
G	052	033	016	101
M	687	185	331	1203
P	183	053	110	346
WR	053	023	018	094
ND	830	444	264	1538
$\sum$	2489	982	1130	4601

Table 7.2.: Distribution summary of KSoF ComParE stuttering dataset (G: Garbage, Fi: Fillers, P; Prolongation, SR: SoundRepetition, B: Block, M: Modified, WR: WordRepetition, ND: no_disfluencies).

multi-branched (MB) *StutterNet*[5], where we pass the $768 \times T$-dimensional speech embeddings directly. The block diagram of proposed framework is shown inFigure 7.5.

7.6.3. Training details

We train the MB *StutterNet* and a shallow NNet (discussed in section 7.3) with Adam optimizer on a cross entropy loss function ($L = Lf + L_d$, L_f : *FluentBranch loss*, L_d : *DisfluentBranch loss* similar to (Equation (7.7)) using a learning rate of 1e-2, with a batch size of 128 over 50 epochs. The shallow NNet is composed of two branches including *FluentBranch* and *DisfluentBranch* with each branch having three FC layers. Each FC layer is followed by a ReLU activation function and a 1D-batch normalization. A dropout of 0.3 is applied to the first two FC layers. A patience of seven is applied on a validation loss to stop the training criteria. We train the MB *StutterNet* with two input features including $20 \times T$ MFCCs and $768 \times T$ speech embeddings extracted from Wav2Vec2.0. For SVM (Murphy, 2012), we experiment with different kernels including linear, polynomial, radial basis function, and sigmoid, however, in this thesis, we report the results only with the linear kernel. For k-NN (Murphy, 2012), we choose the value of $k = 5$ empirically by grid search using the elbow method with $p = 2$ (Euclidean) distance metric. A statistical pooling (mean and standard deviation) is applied to Wav2Vec2.0 $768 \times T$ speech embeddings before pass-

[5]The MB *StutterNet* is discussed in Chapter 5.

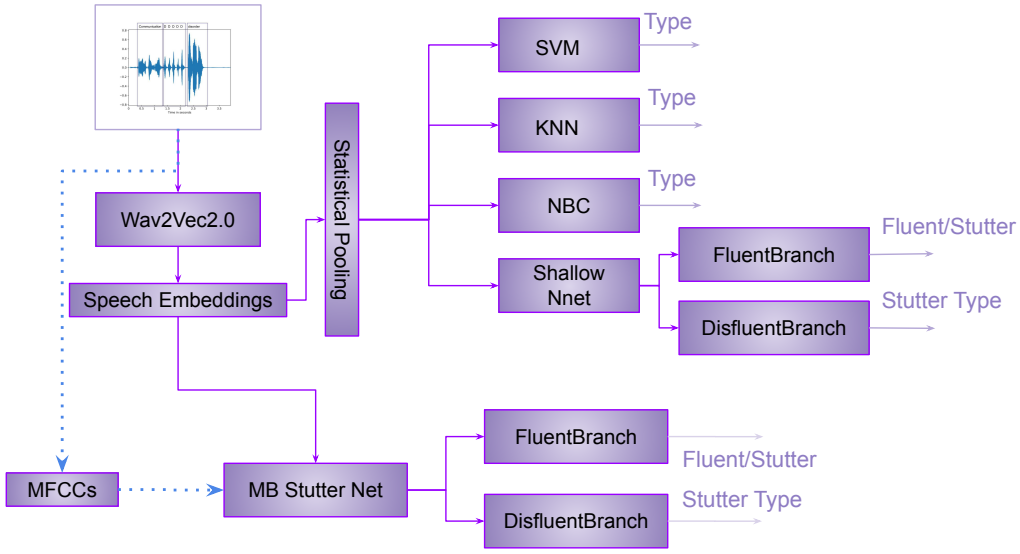


Figure 7.5.: Block diagram of the proposed pipeline for stuttering detection on KSoF ComParE stuttering challenge (NNet: Shallow neural network, SVM: Support vector machine, NBC: Naive Bayes classifier, KNN: k- nearest neighbor, MB: Multi branch). For shallow NNet, SVM, NBC, and KNN, the input is speech embeddings only extracted from Wav2Vec2.0.

ing them to k-NN, NBC (Murphy, 2012), SVM, and NNet downstream classifiers. We use PyTorch and Scikit-learn for implementation purposes.

$$\mathcal{L}_{\text{tot.}}(\theta_e, \theta_f, \theta_d) = \mathcal{L}_f(\theta_e, \theta_f) + \mathcal{L}_d(\theta_e, \theta_d) \tag{7.7}$$

where θ_e, θ_f, and θ_d are parameters of base encoder, *FluentBranch* and *DisfluentBranch* respectively.

7.7. Results and discussion

7.7.1. Evaluation

We experiment with different speech embeddings extracted from various layers of a Wav2Vec2.0 pre-trained model. The extracted speech embeddings improve the detection performance of almost all the classes over the baseline. Similar to (Schuller et al., 2022), we also found that the *word repetitions* are the most challenging ones to recognize. Among the various layers, layer five (L5) shows the best performance on UAR with all the downstream classifiers except for NBC, where, speech embeddings from layer nine (L9) perform better.

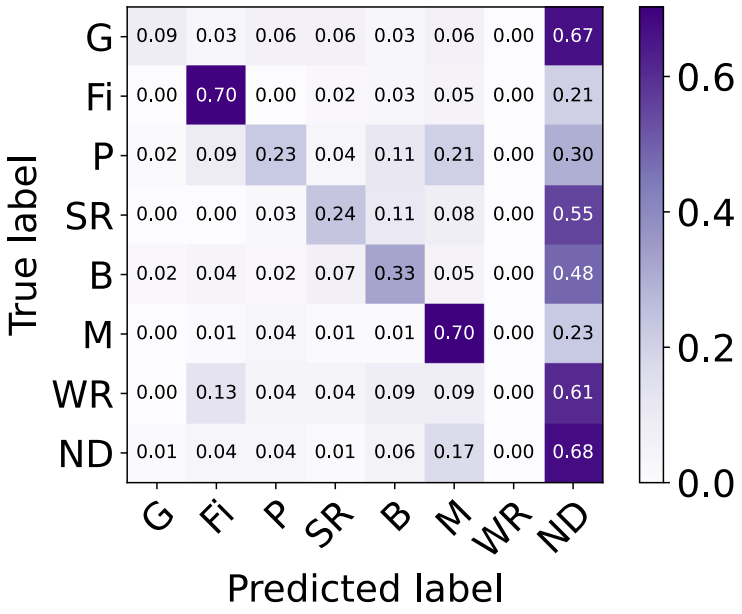


Figure 7.6.: Accuracy confusion matrix of SVM on Val. set with L5 Embeddings (G: Garbage, Fi: Fillers, P; Prolongation, SR: SoundRepetition, B: Block, M: Modified, WR: Word Repetition, ND: no_disfluencies, TA; Total accuracy)

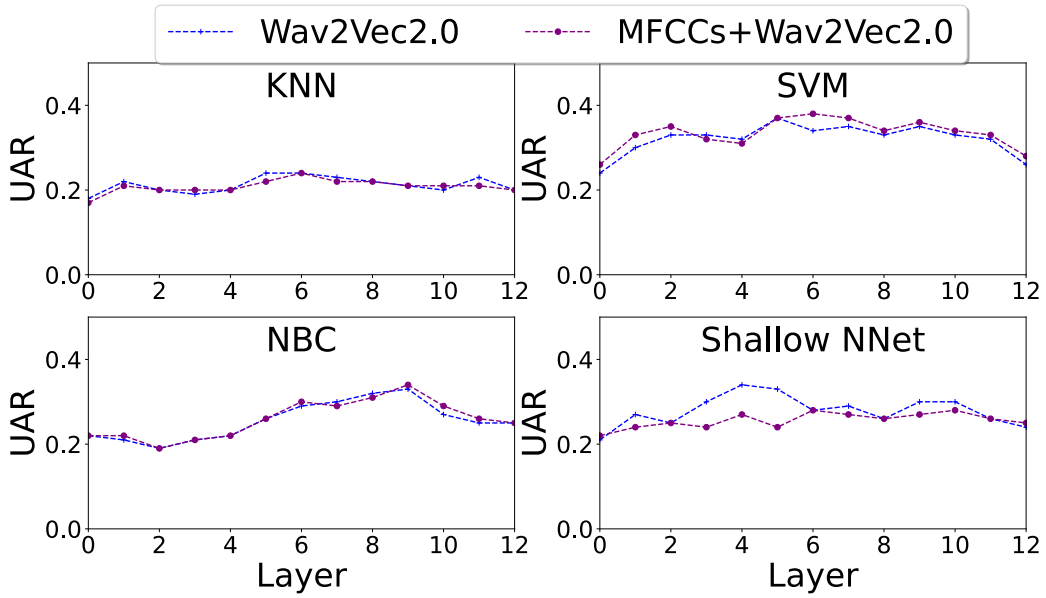


Figure 7.7.: Comparison of various Wav2Vec2.0 contextual layers and its concatenation with MFCCs in SD with k-NN, NBC, SVM, and shallow NNet slassifiers where Y-axis represents UAR:Unweighted average recall, and X-axis represents Wav2Vec2.0 layer embeddings.

Figure 7.7. shows the impact of various speech layer embeddings in SD with downstream classifiers. As presented by the plots, the first initial and last layers show lower UAR in comparison to the middle layers. We hypothesize that the lower layer speech representations contain information only from smaller contexts, and passing this information after statistical pooling to SVM, *k*-NN, NBC, and NNet further inhibits it in learning stutter-specific patterns. Furthermore, (Shah et al., 2021) have also shown that the middle layers are good at capturing fluency and pronunciation features which are extremely important for stuttering detection, while initial layers of Wav2Vec2.0 exhibit more deviation in learning fluency (e.g., speech rate, pauses, etc.,) and pronunciation (e.g., vowels, consonants, syllables, stress, etc.,) features. The trend also shows that the UAR curve decreases for the last layers. This is most likely due to the reason that the Wav2Vec2.0 model is fine-tuned and adapted towards the ASR task similar to SEP-28k dataset as dicusses previously, and it is quite possible that the information such as prosody, stress, and emotion state gets diluted, as also can be confirmed from the study by (Shah et al., 2021). We also observe that the improvement gap between MFCC-based systems and Wav2Vec2.0 embedding-based systems is lower than the study carried out above in Section 7.5. The most obvious reason seems the linguistic information between the two datasets. In previous Section 7.5, we used the SEP-28k corpus for the downstream SD task, which is also an English language stuttering dataset that coincides with the language of the LibriSpeech on which the Wav2Vec2.0 model was trained. Contrary to the SEP-28k dataset, there is a possibility of the loss of linguistic information while extracting embeddings of the KSoF German dataset (provided in this ComParE challenge) from the English-based Wav2Vec2.0 pre-trained model.

In addition, we experiment by concatenating the MFCC features to each speech embeddings layer of Wav2Vec2.0 after applying statistical pooling to both the features and using the downstream classifiers *k*-NN, SVM, NBC, and shallow NNet for SD, the plots of which are shown in Figure 7.7. We observe from the curves that the concatenation of MFCCs with layer embeddings only helps slightly for the SVM classifier while for *k*-NN and NBC, it remains almost unchanged. For shallow NNet, the UAR degrades while concatenating MFCC and speech embeddings. Over the CBL baseline, the SVM with speech embeddings and concatenated (MFCCs+speech embeddings) results in a relative improve-

		Accuracy										
Model	Feature	G	Fi	P	SR	B	M	WR	ND	TA	UAR(V)	UAR(T)
(Schuller et al., 2022) (CBL)	DS	NA	NA	NA	NA	NA	NA	NA	NA	NA	28.1	40.4
MB *StutterNet*	MFCCs	21.21	14.71	35.85	18.42	43.27	78.38	0.00	20.27	33.40	29.0	NA
SVM (Linear)	L5	9.09	69.61	22.64	23.68	32.69	70.27	0.00	67.57	56.93	36.9	41.0
SVM (Linear)	MFCCs:L5	21.21	59.80	26.42	26.32	39.42	71.35	0.00	56.08	52.34	37.6	42.6
NBC	L9	45.46	41.18	30.19	26.32	39.42	70.81	0.00	08.56	29.84	32.7	NA[7]
NNet	L5	6.06	59.80	26.42	10.53	0.00	75.14	4.35	**81.08**	**59.17**	32.9	NA
MB *StutterNet*	L5	15.15	71.57	32.08	15.79	39.42	76.22	0.00	57.43	54.79	38.5	40.3
MB *StutterNet*	L5:L9	03.03	81.37	28.30	**28.95**	50.96	74.60	4.35	54.50	55.40	40.8	42.6
MB *StutterNet*	$\sum_i L_i$	12.12	**78.43**	39.62	26.32	58.65	**80.00**	0.00	32.88	47.86	**41.0**	**42.7**

Table 7.3.: SD results in accuracy and UAR (G: Garbage, Fi: Fillers, P; Prolongation, SR: SoundRepetition, B: Block, M: Modified, WR: Word Repetition, ND: no_disfluencies (fluent), TA; Total accuracy, V: Validation set, T: Test set, CBL: Challenge baseline, CE: Cross entropy, WCE: Weighted cross entropy, UAR: Unweighted average recall, V: Validation set, T: Test set, DS: Deep spectrum features, L_i: Embeddings from layer i, ":" is concatenation, $\sum_i L_i$: Summing information over all embedding layers).

ment of 31.32%, and 33.81% on the validation set, and 1.49% [6] and 5.45% on the test set respectively. We make use of the fact that each layer of the Wav2Vec2.0 model has different information and passing the summation over all layers ($\sum_i L_i$) to MB *StutterNet* results in a relative improvement of 45.91% on the validation set and 5.69% on the test set in UAR over the challenge baseline (CBL) as shown in Table 7.3.

After analyzing various confusion matrices (one shown in Figure 7.6) from the SD systems mentioned in Table 7.3, we found the trend that the most of the disfluent classes are still being falsely classified as ND (*no_disfluencies* class), and confusion in *garbage* is maximum followed by *word repetitions*, *sound repetitions and blocks*. This makes intuitive sense because the *repetitions* contain certain words or phrasal parts that, when examined individually, reveal themselves to be fluent utterances. Consider the phrase, "*for for* the movement". The word *for* is repeated twice, however, it is a fluent component if two *fors* are examined separately. For *garbage*, it is hard to understand the reason since it is either unintelligible or contains no speech. Furthermore, we observe that switching from inputting speech embeddings from a single layer to summing information across all layers ($\sum_i L_i$) reduces the confusion rate of various classes for MB *StutterNet*.

[6] We were only allowed to submit five UAR results on the test set due to the restriction in this ComParE challenge.

7.8. Summary

The automated stuttering detection task suffers from a lack of unlabeled data and thus limits the application of large deep models. To address this issue, we introduced a self-supervised learning framework that first learns a feature extractor for a pre-text task on a large amount of audio data and then employs the learned feature extractor for the downstream stuttering detection task with limited audio data. We investigated ECAPA-TDNN-based speaker recognition and Wav2Vec2.0-based speech recognition as two separate pre-text tasks trained on VoxCeleb and LibriSpeech, respectively. Our study reveals that contextual embeddings associated with speech recognition tasks are more appropriate for SD than speaker embeddings. We found that Wav2Vec2.0-based contextual embeddings yielded at least 19.17% relative improvement in UAR over the competitive state-of-the-art systems trained only on limited labeled data. We further improved SD performance by combining embedding from different layers of the Wav2Vec2.0 model. We also found that post-processing the extracted embeddings with LDA improves classification performance. Our benchmarking experiments with three different classifiers for downstream tasks reveal that a simple MLP-based neural network performs best and it opens up opportunities for further advancements.

Even if the proposed self-supervised framework substantially improved the SD performance over baselines, the performance is still relatively low as compared to other speech applications, possibly due to the presence of a mismatch effect in podcast recordings and class imbalance during training. The future work includes further investigations of the proposed approach by compensating for those effects. In our present work, we did not utilize the stuttering labeled dataset for training the pre-text task. This work can also be extended by including this dataset in this stage which may mitigate the audio domain mismatch between pre-text and downstream tasks.

8. Conclusion and future research perspectives

We conclude this thesis by summarizing the key findings and overall contributions, followed by a list of open challenges in stuttering detection domain. Finally, we give a broader perspective for the practical application of the methods presented in this thesis from the clinical point of view.

8.1. Conclusion

The primary purpose of this thesis is to create deep learning based stuttering classification systems that might help speech pathologists and PWS in distinguishing fluent and disfluent speech. Recall the main question of this thesis that we highlighted in Section 1.1:

Can we develop audio based stuttering detection systems which successfully capture different variablities from scarce stuttering audio datasets ?

To address this question, firstly, we propose in Chapter 4, the first-ever *single* multi-class *StutterNet* classifier for stuttering detection. To learn and capture the temporal context of different types of disfluencies/stuttering classes, the proposed *StutterNet* first computes the MFCC features from raw stuttered utterances, which are then forwarded to the TDNN. The results show that the *StutterNet* achieves considerable gain in overall average accuracy compared to the residual neural network and convolutional long short-term memory based state-of-the-art methods on UCLASS and SEP-28k datasets. By optimizing the filter bank size, context window, and layer size, we discovered that context window optimization boosts the efficiency of detecting prolongation and repetition types. The performance of *StutterNet* improves with increasing context for the identification of prolongation and rep-

etition disfluencies, while it degrades for fluent segments and is remains almost unchanged for block stuttering. Even though the proposed model showed promising results in SD, but the improvement is bounded and limited due to class-imbalance problems, and the unavailability of large stuttering datasets. This limitation was eventually addressed by the use of class-balanced learning, data augmentation, and self supervised learning in Chapter 4, Chapter 5, and Chapter 7.

In Chapter 5, we consider MB *StutterNet*, a variant of above mentioned *StutterNet*, which addresses class-imbalance in stuttering detection domain via multi-branch training scheme. Inspired by the fact that the number of *fluent* class samples is almost equal to the total number of samples in *disfluent* classes as described in Figure 5.1, we simultaneously classify fluent vs disfluent classes in one output branch and subcategories of disfluent classes in another output branch. Furthermore, we contribute by offering thorough study on how to use class-balanced training by adding additional weights to minority stuttering classes in the loss function, which considerably enhances their detection performance. In addition, we thoroughly test our models under simulated noisy ambient conditions by applying data augmentation to the stuttered speech. The employment of data augmentation is not straightforward, because most of the data augmentations like time stretch, speed perturbation, etc., alter the underlying structure of the stuttering speech sample completely. Our approach employs techniques such as noise, babble, music, and reverberation reflecting real-world scenarios and doesn't change the underlying structure of the stuttering speech completely in order to further test SD methods in real-world settings. As discussed above, we further employ variable contexts, and presented MC *StutterNet* (please refer to Figure 5.3) to jointly learn different contexts of stuttered speech utterance, where a context optimised for one disfluent class is not good for another class. The experiments show $\approx$ 7% relative improvement in macro-F1 score over the vanilla *StutterNet*.

Chapter 6 presents MTL framework motivated by the way humans perceive speech and decode its meta characteristics such as emotional content, speaker traits, etc. In this thesis, we particularly exploit speaker characteristics as an auxiliary task in a MTL SD. In addition, this chapter also provides an adversarial learning based MTL framework for stuttering detection based on the hypothesis that stuttering is affected by various meta-data contents such as speaker information, recording conditions, age, accent, etc. We investigate

speaker information and try to learn speaker invariant stutter features.

In Chapter 7, we employ SSL to stuttering detection domain. We first use pre-trained Wav2Vec2.0 SSL model as a feature extractor, which is trained on massive amounts of non-stuttering unlabelled data with an assumption to capture different recording conditions, gender, age, linguistic content, paralinguistic information, and so on. These extracted features of stuttering labeled dataset from Wav2Vec2.0 model are then benchmarked with shallow neural network and MB *StutterNet*. We extensively evaluate our methodology and compared with the MFCC based input baseline features. We found that the features learned using SSL setup are rich enough to capture different types of stuttering and outperformed over the baseline MFCC based features. We also found that the middle layers of Wav2Vec2.0 model are good at capturing information such as paralinguistics, phonetic features, etc., which helps the model to capture more stutter related representations and its types. In addition, we found that the last layers of Wav2Vec2.0 model slightly degrades the stuttering detection performance due to the fact that the model was adapted towards ASR task, and there is a possibility of loss of prosodic, emotional content that are extremely important for stuttering detection.

Even though the proposed models in this thesis have shown great improvement in the stuttering detection domain, there are some limitations associated with these models. The performance of advanced *StutterNet* is yet to be investigated on the actual clinical stuttering datasets, the collection of which is under process. The issue with the current stuttering datasets is the noise in annotations, which are labeled by *phoneticians* not by *expert speech therapists*. So there is a possibility that a model such as *StutterNet* trained on noisy labels can perform poorly in actual clinical settings. Moreover, our methodology is based on an assumption that each stuttering speech sample is having a dominant unique disfluency in it, however stuttering rarely comes alone. This can cause variations in the outcome predictions of the trained models such as *StutterNet* and its variants, when fed with such kind of disfleunt speech samples having multiple disfluencies in them. In addition, models trained in one domain such as language, recording conditions, accents, speakers, linguistic content, etc., preform poorly in another domains such as cross corpora settings, etc. Analysing the performance of our methodology in cross corpora settings as discussed in Chapter 5, we observe a drop in the detection performance of stuttering and its types. The current thesis

focuses only on the global detection and identification of disfluencies, if embedded in a speech utterance, however, it didn't take into account where actually the stuttering occurs i.e., which frames in particular, in the stuttered speech signal are the disfluent ones.

Based on our initial investigations, we also found that some stuttering types are not clearly discernible in acoustic domain, and the proposed methodology in this thesis is likely to misrepresent those samples, However, those samples were clearly visible by the *visual cues* (e.g., head nodding, lip tremors, quick eye blinks, and unusual lip shapes) that can act as a *complementary information* to the acoustic domain, thus opens the door for *multi-modal* stuttering detection systems.

8.2. Future perspectives

The techniques presented in this thesis have a lot of potential for expansion. In this section, we discuss some possible future perspectives.

Overall, this thesis is one of the several potential beginning steps towards developing automatic stuttering detection systems for clinical settings. As discussed above, disfluencies rarely come alone, we can exploit multi-label training schemes by involving expert speech therapists to direct the *StutterNet* to make robust predictions in classifying the multiple disfluent samples into respective multiple labels. In addition, soft/fuzzy clustering algorithms can also be exploited for multiple disfluent samples. Exploiting speaker information, which we study in Chapter 6 in learning speaker-invariant stutter representations, additional meta-data such as language, linguistic content, age, recording conditions, etc., can also be explored in adversarial MTL setup to make stutter representations more robust and *invariant* to these meta-contents.

In learning frame-wise stuttering detection, the methodology proposed in this thesis can be adapted like sequence-to-sequence learning paradigm in conjunction with CTC loss to reveal the disfluent type at each frame-level in the stuttered speech utterance. Similar to part of speech tagging, language models such as bidirectional encoder representations from transformers models can also be adapted for frame-level disfluency detection, which later on can be auto-corrected on the fly and can be tuned towards ASR task.

Stuttering can also be characterized as an audio-visual problem. Cues are present both in the visual (e.g., head nodding, lip tremors, quick eye blinks, and unusual lip shapes)

as well as in the audio modality. This multimodal learning paradigm could be helpful in learning robust stutter-specific hidden representations across the cross-modality platform, and could also help in building robust automatic stuttering detection systems. Self- supervised learning such as AV-HuBERT, Whisper models can be exploited to capture acoustic stutter-specific representations based on guided video frames. This framework could be helpful in learning stutter-specific features from audio signals guided by visual frames or vice versa (Shukla et al., 2020). Textual and magnetic resonance imaging modalities can also be added on top to enrich the stuttering latent representations. In addition, functional magnetic resonance imaging modality of stuttering can be used to explore and explain the stutter latent embedding space, which the model is trying to learn from acoustic input feature space.

For privacy preserving, adversarial and federated learning are the tools that can be employed with the models proposed in this thesis in developing *privacy preserving stuttering detection* systems. In federated learning paradigm, the SD model can be downloaded from the cloud onto the edge devices such as mobile phones and trains the model on each mobile device separately using their local *disfluent* data only. These locally trained models will then be transmitted from the devices to the main server in encrypted form, where they will be integrated, by weight averaging, etc,. and a single consolidated and enhanced global model will be delivered back to the devices without compromising their privacy.

Different studies have used different evaluation metrics to test the performance of their SD systems. Accuracy, F1-score, UAR, etc., have been used widely in SD. However, there is no single consistent metric that is being used across the SD community which makes it hard to compare the newly developed SD systems to the existing ones. Moreover the unbiased nature of these metrics doesn't reflect the true evaluation performance of SD methods. There are no consistent data selection (train/val/test) rules and no clear evaluation protocol for evaluating various SD methods which makes it extremely difficult to compare the proposed SD methods with the state-of-the-art SD techniques. Therefore, it is essential to have a unified data selection procedure for comparing the performance of different SD systems, as well as a single consistent evaluation metric, such as macro F1-score, which considers the class count when evaluating the performance of the SD methods.

One more potential work could be the synthesis of stuttered speech using generative

techniques such as generative adversarial networks and diffusion models that can act as a *stutter-specific* data augmentation for the models proposed in this thesis. This can be further studied by generating a cross lingual/domain stuttered speech samples that aims to generate the stuttered speech from the same speaker but in another language, or from a different speaker in the same language, or from a different speaker in a different language.

Last but not least, it is really interesting to evaluate and analyse how the methodology presented in this thesis performs in other speech disorders such as apraxia, cluttering, lisping and dysarthria.

A. Appendix: Stuttering datasets

Stuttering datasets

Data is an indispensable component of any deep learning model. Deep learning saves feature engineering costs by automatically generating relevant features, however, requires substantial amounts of annotated data. Most stuttering identification studies so far are based on in-house datasets (Howell et al., 2009; Kourkounakis et al., 2020; Lea et al., 2021; Ratner and MacWhinney, 2018; Bayerl et al., 2022a) with limited speakers. In the stuttering domain, there is a lack of datasets and several stuttering datasets that have been collected so far are discussed below:

UCLASS

The most common concern in stuttering research is the lack of training data. UCLASS English public dataset (although very small and unlabelled) is the most commonly used amongst the stuttering research community (Howell et al., 2009). The UCLASS comes in two releases from the University College London's department of psychology and language sciences. This contains monologues, conversations, and readings with a total audio recording of 457. Some of these recordings contain transcriptions like orthographic, phonetic, and standard ones. Of these, orthographic are the ones that are best suitable for stutter labelling. The UCLASS[1] release 1 contains 139 monologue samples from 81 PWS, aged from 5 to 47 years. The male samples are 121 and the female samples are 18.

LibriStutter

The availability of a small amount of labelled stuttered speech led to synthetic LibriStutter's creation (Kourkounakis et al., 2020). The LibriStutter (English) consists of 50

[1] http://www.uclass.psychol.ucl.ac.uk/uclass1.htm

Dataset	Language	Speech Type	Data (hrs)	Ac
UCLASS	Eng (110 M, 18 F)	Monologue	≈ 5.19 (4.7K)	Pb (Un)
LibriStutter	Eng (23 M, 27 F)	Synthetic	≈ 20.00 (15K)	Pb
FluencyBank	Eng (32 PWS)	Interview	≈ 03.50 (4K)	Pb
SEP-28k	Eng (385 Podcasts)	CS	≈ 23.00 (28K)	Pb
KSoF	Gr (28 M, 9 F)	CS, RS, SS	≈ 04.64 (5.6K)	Pr

Table A.1.: Stuttering datasets (audio) (C: Conversational speech, RS: Read speech, SS, Spontaneous speech, Ac: Accessibility, Eng: English, Gr: German, Pb: Public, Pr: Private, Un: Unannotated, M: Male, F: Female)

speakers (23 males and 27 females), is approximately 20 hours and includes synthetic stutters for repetitions, prolongations, and interjections. For each spoken word, Kourkounakis et al. (2020) used Google cloud speech-to-text application programming interface to generate timestamp correspondingly. Random stuttering was inserted within the four-second window of each speech signal. The details about the 10-fold datasplit are shown in Table A.2.

SetID	Train	Val	Test	$\sum$
1	143875	19579	17301	180755
2	143777	18314	18664	180755
3	147560	15181	18014	180755
4	146403	19003	15349	180755
5	146867	16101	17787	180755
6	143002	18956	18797	180755
7	144858	17588	18309	180755
8	146228	17502	17025	180755
9	145761	17994	17000	180755
10	146841	16651	17263	180755

Table A.2.: LibriStutter dataset split with number of train speakers = 40, validation speakers = 5, testing speakers = 5. We randomly selected 80% (40) speakers for training, 10% (5) speakers for validation , and remaining 10% (5) speakers for testing. The numbers describe the count of samples in 10-fold cross validation scheme.

SEP-28k

The public unlabelled stuttering datasets are too small to build well generalizable SD systems. To address this, Lea et al. (2021) recently curated a public version of stuttering events in podcasts (SEP-28k) dataset. This dataset contains total three-second samples of 28,177 extracted from 385 podcasts which contain utterances from PWS conversing with other PWS. The SEP-28k dataset is the first publicly available annotated dataset. The original SEP-28k dataset contains two types of labels: stuttering and non-stuttering labels. The stuttering labels include blocks, prolongations, repetitions, interjections, and fluent segments, whereas the non-stuttering labels include unintelligible, unsure, no speech, poor audio quality, and music which are not relevant to our study.

FluencyBank

This is a shared database for the study of fluency development which has been developed by Nan Bernstein Ratner (University of Maryland) and Brian MacWhinney (Carnegie Mellon University) (Ratner and MacWhinney, 2018). The platform proposes stutter files with transcriptions of adults and children who stutter. The FluencyBank is an interview data of 32 PWS. In this thesis, we use the annotations developed similar to SEP-28k paper from (Lea et al., 2021). The total samples are 4,144 including stuttering and non-stutering labels.

KSoF

The KSoF is a German stuttered dataset collected recently and is comprised of 5597 three-second stuttered speech clips extracted from 214 recordings of 37 PWS with 28 males and 9 females (Bayerl et al., 2022a). This dataset contains a recording from three types of environments including telephonic conversation, read speech, and spontaneous speech. After collecting the stuttered recordings, the samples were downsampled to 16 kHz. This dataset also provides meta information in terms of gender, microphone type, and speaker identity. Like SEP-28k, the KSoF dataset also contains two types of labels: stuttering and non-stuttering labels. The stuttering labels contain core behaviors with blocks, prolongations, repetitions, interjections, and fluent segments, while non-stuttering labels include natural pause, unintelligible, unsure, no speech, poor audio quality, and music.

In this manuscript, we used mainly SEP-28k dataset for experimental purposes and FluencyBank for cross corpora evaluation purposes. For *StutterNet*, we tested it on both UCLASS and SEP-28k datasets.

B. Résumé étendu

B.1. La motivation

Au cours des dernières décennies, l'intérêt pour la création de systèmes imitant la façon dont les humains comprennent et appréhendent la parole en termes de contenu linguistique, de personnalité, de sexe, d'âge, de santé, d'émotion et d'autres signes paralinguistiques tels que *le bégaiement ou les disfluences* (Huang et al., 2001; Guitar, 2013). Les méthodes les plus pratiques et les plus expressives de la communication humaine sont la parole et l'écoute. Pendant la parole, le cerveau exécute une série de processus cognitifs en termes de sélection, d'activation et d'organisation d'unités lexicales et sensori motrices qui sont nécessaires aux articulateurs de la parole pour délivrer l'énoncé vocal prévu (Harrington and Tabain, 2013). Normalement, l'énoncé de la parole est régi par un certain nombre de processus non linéaires où les locuteurs remodèlent continuellement la direction de leur discours. Ils peuvent faire une pause pour réfléchir à ce qu'ils vont dire ensuite, reprendre une phrase déjà prononcée, ajouter quelque chose de nouveau ou abandonner l'énoncé en cours (Kosmala, 2021; Hardcastle et al., 2012). Par conséquent, les énoncés oraux sont souvent régulés par des cycles de parole *fluente* (lexicalement et grammaticalement exact) contre *disfluente* (éléments non-lexicaux qui perturbent le flux du discours) (Kosmala, 2021).

Notez que les disfluences normales sont utiles dans la production du discours puisqu'elles peuvent être prises en compte à un moment où le locuteur peut corriger ou planifier le discours à venir. Cependant, dans les cas comme les *troubles de la parole ou les déficiences*, les disfluences normales n'aident pas le locuteur à organiser son discours.

Contrairement aux personnes ne présentant aucun trouble de la fluidité, les PWS savent ce qu'elles veulent prononcer mais sont momentanément incapables de le faire en raison du dysfonctionnement de leur système nerveux central dans la génération des schémas de commandes motrices nécessaires à la production d'un discours fluide (ASHA, 2020;

Smith and Weber, 2017).

Les troubles de la parole font qu'il est difficile pour les PWS de former des paroles correctes et nécessaires et altèrent la capacité d'une personne à produire ces paroles correctes qui leurs permettent de communiquer avec les autres (Guitar, 2013; Ward, 2017).

Ces troubles peuvent toucher tous les groupes d'âge et prennent généralement la forme d'un *bégaiement*, d'une apraxie (le cerveau est incapable d'envoyer un message exact aux muscles de la parole), d'une dysarthrie (causée par une faiblesse musculaire), ou encore d'un encombrement (parole trop saccadée ou trop rapide) (Guitar, 2013; Ward, 2017; Duffy, 2019). Parmi ces troubles, le *bégaiement* ou *balbutiement* est le plus fréquemment rencontré (Minnis, 2020). Le dysfonctionnement des régions sensorimotrices responsables de la production de la parole entraîne le *bégaiement* sous la forme de *comportements principaux* comprenant les *répétitions, les blocages et les prolongations* et son évolution est dirigée par des aspects lexicaux et psychologiques (Smith and Weber, 2017; Guitar, 2013). Contrairement à d'autres troubles de la parole, le bégaiement peut être traité si une intervention précoce est mise en place pour aider les PWS à atteindre une fluidité normale (Guitar, 2013). Avec les progrès des technologies vocales et l'utilisation généralisée des interfaces vocales, allant des applications mobiles aux assistants numériques, tels que Cortana, Siri, etc, la communication entre les utilisateurs finaux a été facilitée. Piloter par les appareils domestiques intelligents, on estime à 8 milliards le nombre d'assistants vocaux numériques utilisés d'ici 2030. D'une part, une technologie vocale comme celle-ci améliore l'utilisation des gadgets au quotidien et facilite la vie de certains groupes minoritaires tels que les personnes en situation de déficience visuelle ou physique qui sont incapables d'accéder au mode textuel des appareils. D'autre pour les personnes souffrant de problèmes d'élocution, une technologie aussi influente reste inaccessible.

B.2. Objectifs

Lors des séances de traitement et de thérapie conventionnelles, les orthophonistes ou les audiologues analysent manuellement la parole du patient (Nöth et al., 2000). Les orthophonistes observent et surveillent les schémas d'élocution des PWS pour les rectifier (Guitar, 2013). Cette convention de SD est très laborieuse, prend beaucoup de temps et est également influencée par la croyance idiosyncrasique des orthophonistes. Par ailleurs,

ils peuvent ne pas être disponibles en permanence. De plus, les ASR fonctionnent bien pour une parole normale et fluide, mais ils ne parviennent pas à reconnaître les bégaiements (Mitra et al., 2021), ce qui ne permet pas aux PWS d'accéder facilement aux assistants virtuels, comme indiqué précédemment. Par conséquent, les systèmes SD automatiques interactifs qui fournissent une évaluation impartiale et cohérente de la parole bégayée sont fortement encouragés. Nous pensons que ces systèmes de détection automatique du bégaiement fourniront également une évaluation juste, cohérente et impartiale de la parole bégayée qui, plus tard, pourra également être adaptée aux assistants numériques vocaux pour les PWS.

Malgré le fait qu'il ait de nombreuses applications potentielles, très peu d'attention a été accordée à la recherche dans le domaine du SD et de la mesure, ce qui est l'objectif principal de cette thèse. La détection et l'identification des épisodes de bégaiement peuvent être un problème difficile et complexe en raison de plusieurs facteurs variables, notamment la langue, le sexe, l'âge, l'accent, le débit de parole, etc. La plupart des travaux antérieurs sur les méthodes de SD sont basés soit sur des systèmes de ASR (Alharbi et al., 2018, 2020) ou des modèles de langage (Zayats et al., 2016a; Chen et al., 2020). Ces méthodes sont des approches en deux étapes qui convertissent d'abord les signaux acoustiques de la parole en leur modalité textuelle correspondante, puis détectent le bégaiement par l'application de modèles de langage. Bien que cette approche en deux étapes basée sur le ASR pour identifier le bégaiement ait donné des résultats prometteurs, la dépendance à l'égard de l'unité ASR la rend coûteuse en calcul et sujette aux erreurs. De plus, l'adaptation à la tâche ASR entraîne la perte possible d'informations relatives au bégaiement telles que les informations prosodiques, émotionnelles et autres informations paralinguistiques. En résumé,

Nous essayons de développer des systèmes de détection de bégaiement basés sur l'audio qui capturent avec succès les différentes variabilités des énoncés audio de bégaiement et qui permettent de distinguer entre la parole fluente et diverses catégories de parole disfluente telles que les blocages, les prolongements, les répétitions et les interjections.

B.3. Résumé de la contribution

L'objectif principal de cette thèse est de développer des systèmes de classification du bégaiement basés sur l'apprentissage profond. Ils pourront être utilisés ultérieurement dans des environnements cliniques pour aider les orthophonistes à analyser les différents

types de bégaiement des PWS. Cependant, comme énoncé exprimé dans la Section 1.1, le bégaiement peut être exprimé dans de multiples modalités, y compris le texte, l'audio, le visuel et les biosignaux. Nous nous concentrons principalement sur le développement de systèmes de classification du bégaiement basés sur la modalité *acoustique* en abordant certains des défis auxquels le domaine du bégaiement est confronté en termes de variabilité des locuteurs, de disponibilité des données, de déséquilibre des données, etc. lorsqu'il est abordé à partir du paradigme de l'apprentissage profond. Ce manuscrit de thèse présente ma contribution en proposant *StutterNet*, ses variantes et leur évaluation extensive sur le jeu de donnée SEP-28k de bégaiement SEP-28k dataset récemment créé. Cette thèse présente également ma contribution à l'adoption de l'architecture SSL dans le domaine de la détection du bégaiement.

***StutterNet*:** La première contribution de cette thèse est le développement de *StutterNet*. Ce système de détection du bégaiement basé sur un réseau neuronal à retard temporal est un classificateur SD multi-classes capable de détecter plusieurs types de bégaiement à l'aide d'un seul système sur un grand ensemble de locuteurs. Ce contrairement aux systèmes SD de l'état de l'art qui sont pour la plupart des systèmes de classification SD binaires, qui ne prennent en compte que quelques locuteurs bègues. Le réseau proposé *StutterNet* calcule d'abord les caractéristiques MFCC à partir d'échantillons audio bruts, qui sont ensuite transmises au TDNN pour apprendre et capturer le contexte temporel de divers types de classes de disfluences et bégaiements. *StutterNet* a été évalué sur les jeux de données UCLASS et SEP-28k. De plus, nous prenons également en compte les échantillons de parole fluente des PWS qui influencent considérablement les systèmes de classification du bégaiement, contrairement aux systèmes SD de l'état de l'art, qui se concentrent principalement sur les classes de bégaiement.

MB *StutterNet*: En raison de la nature inhérente du bégaiement, l'obtention d'ensembles de données équilibrées par classe est à la fois extrêmement difficile et coûteuse. Les classificateurs de réseaux neuronaux profonds sont généralement biaisés en faveur de la classe majoritaire lorsqu'ils sont entraînés sur des ensembles de données fortement déséquilibrées en termes de classe, ce qui entraîne une modélisation médiocre des classes minoritaires. De plus, les données sur le bégaiement sont très limitées et sont généralement recueillies

dans des environnements cliniques contrôlés, ce qui limite l'adoption de méthodes avancées d'apprentissage profond pour la détection du bégaiement et peut être grandement affecté par les environnements bruyants. Pour contribuer aux problèmes mentionnés précédemment, nous avons proposé MB *StutterNet* (traitant *le problème des équilibres de classes*), qui comprend un encodeur de base ($\mathcal{E}$) suivi de deux branches parallèles appelées *DisfluentBranch* ($\mathcal{D}$) et *FluentBranch* ($\mathcal{F}$). Les representations générées par ($\mathcal{E}$) sont simultanément transmises au *FluentBranch* et au *DisfluentBranch*, où le *FluentBranch* est entraîné à distinguer les échantillons fluents et disfluents et le *DisfluentBranch* est entraîné à différencier et à classer les sous-catégories disfluentes. De plus, nous avons également contribué en fournissant une analyse approfondie sur l'utilisation d'entraînement à classes équilibrées en appliquant plus de poids aux classes de bégaiement minoritaires dans la fonction de perte, ce qui améliore considérablement leur performance de détection. Pour tester d'avantage nos propositions de *StutterNet* et de MB *StutterNet* dans des environnements réels, nous les évaluons de manière approfondie dans des conditions d'environnements bruyants simulées en utilisant des techniques d'augmentation des données telles que le bruit, le babillage, la musique et la réverbération sur les échantillons de parole bégayée. L'avantage d'appliquer l'augmentation des données à la parole bégayée est double. Premièrement, elle augmente la quantité de données dans les bégaiements, ce qui permet d'utiliser des méthodes avancées d'apprentissage profond. Deuxièmement, elle permet de simuler des conditions environnementales bruyantes en temps réel, ce qui rend les systèmes SD proposés robustes à ces facteurs.

MC *StutterNet*: En outre, nous avons également constaté précédemment que le contexte optimisé pour une catégorie n'est pas bon pour une autre classe de bégaiement. En exploitant ce fait, nous proposons un MC StutterNet, qui est une architecture basée sur un réseau neuronal à TDNN multi-contexte de multi-branché SD. L'architecture multi-contexte est basée sur la façon dont les humains perçoivent la parole. Dans la cochlée, le signal acoustique d'entrée de la parole est divisé en plusieurs bandes de fréquences afin que les informations de chaque bande puissent être filtrées indépendamment et traitées en parallèle par le cerveau humain.

Multi-task and adversarial learning in stuttering detection: Durant ces dernières années, le domaine de la détection du bégaiement basé sur l'apprentissage profond a connu une évolution continue. Pourtant, tous les systèmes actuels se concentrent uniquement sur la méthode d'apprentissage du bégaiement sans tenir compte des tâches auxiliaires. Malheureusement, ce n'est pas de cette manière que nous percevons la parole. Au contraire, nous décodons simultanément la tâche principale avec divers méta-contenus tels que les caractéristiques du locuteur, le contenu linguistique, l'émotion, etc. de manière multitâche. Dans cette optique, nous avons exploité les caractéristiques du locuteur dans la détection du bégaiement. La troisième contribution de cette thèse est d'apprendre et d'exploiter ces caractéristiques du locuteur en combinaison avec la tâche primaire de détection du bégaiement dans un paradigme d'apprentissage multi-tâches. Comme la tâche de détection du bégaiement ne devrait pas dépendre d'un ensemble spécifique de locuteurs, une approche alternative consiste à supposer qu'une représentation acoustique robuste du bégaiement doit être invariante par rapport aux tâches secondaires, en particulier la reconnaissance du locuteur. Nous avons également contribué à cette approche en proposant une méthode d'apprentissage contradictoire MTL, qui apprend des représentations latentes et qui est invariante par rapport au locuteur, mais qui est en même temps discriminantes pour le bégaiement.

Introducing self-supervised learning to stuttering detection: En raison de la rareté des données *étiquetées*, la tâche de détection automatique du bégaiement est limitée dans son utilisation des modèles d'apprentissages profonds de grand taille. Comme indiqué dans Appendix A, plusieurs ensembles de données sur le bégaiement, tels que SEP-28k (Lea et al., 2021), UCLASS (Howell and Sackin, 1995), LibriStutter (Kourkounakis et al., 2020) et FluencyBank (Lea et al., 2021), ont été créés au fil du temps pour examiner divers modèles SD. Cependant, l'amélioration est limitée, très probablement en raison de la petite taille et du nombre de données non étiquetées des ensembles sur le bégaiement. Ces ensembles sont incapables d'encapsuler plusieurs facteurs variables tels que le contenu linguistique, le style de parole, le sexe, l'âge, l'accent, le tempo de la parole, les conditions d'enregistrement, etc. Notre contribution à ce défi est que nous avons introduit le tout premier cadre d'apprentissage auto-supervisé au domaine de la détection du

bégaiement. Le cadre d'apprentissage auto-supervisé entraîne d'abord un extracteur de caractéristiques pour une tâche de pré-texte en utilisant une grande quantité de données audio non bégayantes *non étiquetées* pour capturer différents styles de parole, d'accents, de groupes d'âge, de contenu linguistique, de contenu paralinguistique, de prosodie, conditions environnementales, etc. Puis, il applique l'extracteur de caractéristiques appris à une tâche de détection du bégaiement en aval en utilisant des données audio de bégaiement *étiquetées* limitées.

Bibliography

Adams, M. R. (1974). A physiologic and aerodynamic interpretation of fluent and stuttered speech. *Journal of Fluency Disorders*, 1(1):35–47.

Adams, M. R. (1987). Voice onsets and segment durations of normal speakers and beginning stutterers. *Journal of Fluency Disorders*, 12(2):133–139.

Ahmed, T., Suffian, M., Khan, M. Y., and Bogliolo, A. (2022). Discovering lexical similarity using articulatory feature-based phonetic edit distance. *IEEE Access*, 10:1533–1544.

Akçay, M. B. and Oğuz, K. (2020). Speech emotion recognition: Emotional models, databases, features, preprocessing methods, supporting modalities, and classifiers. *Speech Communication*, 116:56–76.

Alharbi, S., Hasan, M., J H Simons, A., Brumfitt, S., and Green, P. (2018). A lightly supervised approach to detect stuttering in children's speech. In *Proc. Interspeech 2018*, pages 3433–3437.

Alharbi, S., Hasan, M., Simons, A. J. H., Brumfitt, S., and Green, P. (2020). Sequence labeling to detect stuttering events in read speech. *Computer Speech & Language*, 62:101052.

Allen, J. B. (1995). How do humans process and recognize speech? In *Modern methods of speech processing*, pages 251–275. Springer.

Almeida, J. S., Rebouças Filho, P. P., Carneiro, T., Wei, W., Damaševičius, R., Maskeliūnas, R., and de Albuquerque, V. H. C. (2019). Detecting Parkinson's disease with sustained phonation and speech signals using machine learning techniques. *Pattern Recognition Letters*, 125:55–62.

Antipova, E. A., Purdy, S. C., Blakeley, M., and Williams, S. (2008). Effects of altered auditory feedback (AAF) on stuttering frequency during monologue speech production. *Journal of Fluency Disorders*, 33(4):274–290.

ASHA (2020). Stuttering.

Bader-El-Den, M., Teitei, E., and Adda, M. (2016). Hierarchical classification for dealing with the class imbalance problem. In *Proc. 2016 International Joint Conference on Neural Networks (IJCNN)*, pages 3584–3591.

Baevski, A. et al. (2020). Wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Proc. NeurIPS*, volume 33, pages 12449–12460. Curran Associates, Inc.

Baevski, A., Hsu, W.-N., Conneau, A., and Auli, M. (2021). Unsupervised speech recognition. *Advances in Neural Information Processing Systems*, 34:27826–27839.

Bayerl, S. P., von Gudenberg, A. W., Hönig, F., Nöth, E., and Riedhammer, K. (2022a). KSoF: The Kassel state of fluency dataset–A therapy centered dataset of stuttering. In *Proc. 13th Conference on Language Resources and Evaluation (LREC 2022)*, pages 1780–1787.

Bayerl, S. P., Wagner, D., Nöth, E., and Riedhammer, K. (2022b). Detecting dysfluencies in stuttering therapy using wav2vec 2.0. In *Proc. Interspeech 2022*.

Bell, P. J., Gales, M. J. F., Lanchantin, P., Liu, X., Long, Y., Renals, S., Swietojanski, P., and Woodland, P. C. (2012). Transcription of multi-genre media archives using out-of-domain data. In *Proc. 2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 324–329.

Belyk, M., Kraft, S. J., and Brown, S. (2015). Stuttering as a trait or state–an ale meta-analysis of neuroimaging studies. *European Journal of Neuroscience*, 41(2):275–284.

Blomgren, M., Alqhazo, M., and Metzger, E. (2012). Do speech sound characteristics really influence stuttering frequency. In *Proc. of the 7th World Congress of Fluency Disorders*.

Blomgren, M., Robb, M., and Chen, Y. (1998). A note on vowel centralization in stuttering and nonstuttering individuals. *Journal of Speech, Language, and Hearing Research*, 41(5):1042–1051.

Blood, G. W. and Blood, I. M. (2016). Long-term consequences of childhood bullying in adults who stutter: Social anxiety, fear of negative evaluation, self-esteem, and satisfaction with life. *Journal of Fluency Disorders*, 50:72–84.

Borden, G. J., Baer, T., and Kenney, M. K. (1985). Onset of voicing in stuttered and fluent utterances. *Journal of Speech, Language, and Hearing Research*, 28(3):363–372.

Bourlard, H. and Dupont, S. (1996). A new ASR approach based on independent processing and recombination of partial frequency bands. In *Proc. Fourth International Conference on Spoken Language Processing. ICSLP '96*, volume 1, pages 426–429 vol.1.

Bourlard, H., Dupont, S., Hermansky, H., and Morgan, N. (1996). Towards subband-based speech recognition. In *Proc. 1996 Eighth European Signal Processing Conference (EUSIPCO)*, pages 1–4. IEEE.

Brosch, S., Häge, A., and Johannsen, H. S. (2002). Prognostic indicators for stuttering: The value of computer-based speech analysis. *Brain and Language*, 82(1):75–86.

Can, D., Martinez, V. R., Papadopoulos, P., and Narayanan, S. S. (2018). PyKaldi: A Python wrapper for Kaldi. In *Proc. ICASSP*. IEEE.

Caruana, R. (1997). Multitask learning. *Machine learning*, 28(1):41–75.

Celeste, L. C. and de Oliveira Martins-Reis, V. (2015). The impact of a dysfluency environment on the temporal organization of consonants in stuttering. *Audiology-Communication Research*, 20(1):10–17.

Chang, S.-E., Ohde, R. N., and Conture, E. G. (2002). Coarticulation and formant transition rate in young children who stutter. *Journal of Speech, Language, and Hearing Research*.

Chee, L. S., Ai, O. C., and Yaacob, S. (2009). Overview of automatic stuttering recognition system. In *Proc. International Conference on Man-Machine Systems 2009, October, Batu Ferringhi, Penang Malaysia*, pages 1–6.

Chen, Q., Chen, M., Li, B., and Wang, W. (2020). Controllable time-delay transformer for real-time punctuation prediction and disfluency detection. In *Proc. ICASSP 2020*, pages 8069–8073.

Chiba, Y., Nose, T., and Ito, A. (2020). Multi-stream attention-based BLSTM with feature segmentation for speech emotion recognition. In *Proc. Interspeech 2020*.

Christensen, H., Aniol, M., Bell, P., Green, P. D., Hain, T., King, S., and Swietojanski, P. (2013). Combining in-domain and out-of-domain speech data for automatic recognition of disordered speech. In *Proc. Interspeech 2013*, pages 3642–3645.

Conture, E. G., McCall, G. N., and Brewer, D. W. (1977). Laryngeal behavior during stuttering. *Journal of Speech and Hearing Research*, 20(4):661–668.

Conture, E. G., Schwartz, H. D., and Brewer, D. W. (1985). Laryngeal behavior during stuttering: A further study. *Journal of Speech, Language, and Hearing Research*, 28(2):233–240.

Crawshaw, M. (2020). Multi-task learning with deep neural networks: A survey. *arXiv preprint arXiv:2009.09796*.

Cui, Y., Jia, M., Lin, T.-Y., Song, Y., and Belongie, S. (2019). Class-balanced loss based on effective number of samples. In *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9268–9277.

Czyzewski, A., Kaczmarek, A., and Kostek, B. (2003). Intelligent processing of stuttered speech. *Journal of Intelligent Information Systems*, 21(2):143–171.

Dawalatabad, N. et al. (2021). ECAPA-TDNN embeddings for speaker diarization. In *Proc. Interspeech 2021*, pages 3560–3564.

De Nil, L. F. and Brutten, G. (1991). Voice onset times of stuttering and nonstuttering children: The influence of externally and linguistically imposed time pressure. *Journal of Fluency Disorders*, 16(2-3):143–158.

Dehqan, A., Yadegari, F., Blomgren, M., and Scherer, R. C. (2016). Formant transitions in the fluent speech of Farsi-speaking people who stutter. *Journal of Fluency Disorders*, 48:1–15.

Desplanques, B. et al. (2020). ECAPA-TDNN: Emphasized channel attention, propagation and aggregation in TDNN based speaker verification. In *Proc. Interspeech 2020*.

Didirkova, I. (2016). *Parole, langues et disfluences: Une étude linguistique et phonétique du bégaiement*. PhD thesis, Université Paul Valéry-Montpellier III.

Didirková, I. and Hirsch, F. (2020). A two-case study of coarticulation in stuttered speech. an articulatory approach. *Clinical Linguistics & Phonetics*, 34(6):517–535.

Didirkova, I., Le Maguer, S., and Hirsch, F. (2020). An articulatory study of differences and similarities between stuttered disfluencies and non-pathological disfluencies. *Clinical Linguistics & Phonetics*, pages 1–21.

Drayna, D. and Kang, C. (2011). Genetic approaches to understanding the causes of stuttering. *Journal of Neurodevelopmental Disorders*, 3(4):374–380.

Duffy, J. R. (2019). *Motor Speech Disorders e-Book: Substrates, Differential Diagnosis, and Management*. Elsevier Health Sciences.

Esmaili, I., Dabanloo, N. J., and Vali, M. (2017). An automatic prolongation detection approach in continuous speech with robustness against speaking rate variations. *Journal of Medical Signals and Sensors*, 7:1.

Etchell, A. C., Civier, O., Ballard, K. J., and Sowman, P. F. (2018). A systematic literature review of neuroimaging research on developmental stuttering between 1995 and 2016. *Journal of Fluency Disorders*, 55:6–45.

Fer, R., Matejka, P., Grezl, F., Plchot, O., Vesely, K., and Cernocky, J. H. (2017). Multilingually trained bottleneck features in spoken language recognition. *Computer Speech & Language*, 46(Supplement C):252 – 267.

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., and Herrera, F. (2018). *Learning from Imbalanced Data Sets*, volume 10. Springer.

Fosnot, S. M. and Jun, S. (1999). Prosodic characteristics in children with stuttering or autism during reading and imitation. In *Proc. of the 14th International Congress of Phonetic Sciences*, pages 1925–1928.

Foundas, A., Lane, A., Corey, D., Hurley, M., and Heilman, K. (2001). Anomalous anatomy in adults with persistent developmental stuttering: A volumetric MRI study of cortical speech and language areas. *Neurology*, 56(8):A157–A158.

Ganin, Y. and Lempitsky, V. (2015a). Unsupervised domain adaptation by backpropagation. In *Proc. of the 32nd ICML - Volume 37*, ICML'15, page 1180–1189. JMLR.org.

Ganin, Y. and Lempitsky, V. (2015b). Unsupervised domain adaptation by backpropagation. In *Proc. of the 32nd ICML*, volume 37, pages 1180–1189, Lille, France. PMLR.

Ghonem, S. A., Abdou, S., Esmael, M. A., and Ghamry, N. (2017). Classification of stuttering events using i-vector. *The Egyptian Journal of Language Engineering*, 4(1):11–19.

Goodfellow, I., Bengio, Y., Courville, A., and Bengio, Y. (2016). *Deep Learning*, volume 1. MIT press Cambridge.

Guitar, B. (2013). *Stuttering: An Integrated Approach to its Nature and Treatment*. Lippincott Williams & Wilkins.

Han, K. J., Pan, J., Tadala, V. K. N., Ma, T., and Povey, D. (2021). Multistream CNN for robust acoustic modeling. In *Proc. ICASSP 2021*, pages 6873–6877.

Hardcastle, W. J., Laver, J., and Gibbon, F. E. (2012). *The Handbook of Phonetic Sciences*. John Wiley & Sons.

Hariharan, M., Chee, L. S., Ai, O. C., and Yaacob, S. (2012). Classification of speech dysfluencies using LPC based parameterization techniques. *Journal of Medical Systems*, 36(3):1821–1830.

Harrington, J. and Tabain, M. (2013). *Speech Production: Models, Phonetic Processes, and Techniques*. Psychology Press.

Healey, E. C. and Ramig, P. R. (1986). Acoustic measures of stutterers' and nonstutterers' fluency in two speech contexts. *Journal of Speech, Language, and Hearing Research*, 29(3):325–331.

Hennansky, H., Tibrewala, S., and Pavel, M. (1996). Towards ASR on partially corrupted speech. In *Proc. Fourth International Conference on Spoken Language Processing. ICSLP '96*, volume 1, pages 462–465 vol.1.

Hillman, R. E. and Gilbert, H. R. (1977). Voice onset time for voiceless stop consonants in the fluent reading of stutterers and nonstutterers. *The Journal of the Acoustical Society of America*, 61(2):610–611.

Hirsch, F., Bouarourou, F., Vaxelaire, B., Monfrais-Pfauwadel, M.-C., Bechet, M., Sturm, J., and Sock, R. (2008). Formant structures of vowels produced by stutterers in normal and fast speech rates. In *Proc. 8th International Seminar On Speech Production*, page NC, France.

Howell, P., Davis, S., and Bartrip, J. (2009). The university college London archive of stuttered speech (UCLASS). *Journal of Speech, Language, and Hearing Research*.

Howell, P. and Sackin, S. (1995). Automatic recognition of repetitions and prolongations in stuttered speech. In *Proc. of the first World Congress on Fluency Disorders*, volume 2, pages 372–374. University Press Nijmegen Nijmegen, The Netherlands.

Howell, P., Sackin, S., and Glenn, K. (1997a). Development of a two-stage procedure for the automatic recognition of dysfluencies in the speech of children who stutter: I. psychometric procedures appropriate for selection of training material for lexical dysfluency classifiers. *Journal of Speech, Language, and Hearing Research*, 40(5):1073–1084.

Howell, P., Sackin, S., and Glenn, K. (1997b). Development of a two-stage procedure for the automatic recognition of dysfluencies in the speech of children who stutter: Ii. ann recognition of repetitions and prolongations with supplied word segment markers. *Journal of Speech, Language, and Hearing Research*, 40(5):1085–1096.

Huang, X., Acero, A., Hon, H.-W., and Reddy, R. (2001). *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development*. Prentice hall PTR.

Ingham, R. J., Fox, P. T., Ingham, J. C., Zamarripa, F., Martin, C., Jerabek, P., and Cotton, J. (1996). Functional-lesion investigation of developmental stuttering with positron emission tomography. *Journal of Speech, Language, and Hearing Research*, 39(6):1208–1227.

Iverach, L., Jones, M., McLellan, L. F., Lyneham, H. J., Menzies, R. G., Onslow, M., and Rapee, R. M. (2016). Prevalence of anxiety disorders among children who stutter. *Journal of Fluency Disorders*, 49:13–28.

Jäncke, L. (1994). Variability and duration of voice onset time and phonation in stuttering and nonstuttering adults. *Journal of Fluency Disorders*, 19(1):21–37.

Jayaram, M. (1983). Phonetic influences on stuttering in monolingual and bilingual stutterers. *Journal of Communication Disorders*, 16(4):287–297.

Jiao, Y., Tu, M., Berisha, V., and Liss, J. (2018). Simulating dysarthric speech for training data augmentation in clinical speech applications. In *Proc. ICASSP 2018*, pages 6009–6013.

Jouaiti, M. and Dautenhahn, K. (2022). Dysfluency classification in stuttered speech using deep learning for real-time applications. In *Proc. ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6482–6486.

Jurafsky, D. and Martin, J. H. (2018). Speech and Language Processing (draft). *Preparation [cited 2022 August 18] Available from: https://web. stanford. edu/~ jurafsky/slp3*.

Kanda, N., Takeda, R., and Obuchi, Y. (2013). Elastic spectral distortion for low resource speech recognition with deep neural networks. In *Proc. 2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 309–314.

Kehoe, T. D. and Contributors, W. (2006). *Speech Language Pathology-Stuttering*. Kiambo Ridge.

Khara, S., Singh, S., and Vir, D. (2018). A comparative study of the techniques for feature extraction and classification in stuttering. In *Proc. 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 887–893. IEEE.

Klein, J. F. and Hood, S. B. (2004). The impact of stuttering on employment opportunities and job performance. *Journal of Fluency Disorders*, 29(4):255–273.

Ko, T., Peddinti, V., Povey, D., and Khudanpur, S. (2015). Audio augmentation for speech recognition. In *Proc. 16th Annual Conference of the International Speech Communication Association*.

Ko, T., Peddinti, V., Povey, D., Seltzer, M. L., and Khudanpur, S. (2017). A study on data augmentation of reverberant speech for robust speech recognition. In *Proc. ICASSP 2017*, pages 5220–5224.

Kodrasi, I. (2021). Temporal envelope and fine structure cues for Dysarthric speech detection using CNNs. *IEEE Signal Processing Letters*, 28:1853–1857.

Kosmala, L. (2021). *A multimodal contrastive study of (dis)fluency across languages and settings: Towards a multidimensional scale of inter-(dis)fluency*. Theses, Sorbonne Nouvelle.

Kourkounakis, T., Hajavi, A., and Etemad, A. (2020). Detecting multiple speech disfluencies using a deep residual network with bidirectional long short-term memory. In *Proc. ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6089–6093. IEEE.

Kourkounakis, T., Hajavi, A., and Etemad, A. (2021). FluentNet: End-to-end detection of stuttered speech disfluencies with deep learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:2986–2999.

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232.

Latif, S. et al. (2020a). Deep representation learning in speech processing: Challenges, recent advances, and future trends. *arXiv preprint arXiv:2001.00378*.

Latif, S., Rana, R., Khalifa, S., Jurdak, R., Qadir, J., and Schuller, B. W. (2020b). Deep representation learning in speech processing: Challenges, recent advances, and future trends. *arXiv preprint arXiv:2001.00378*.

Lea, C., Mitra, V., Joshi, A., Kajarekar, S., and Bigham, J. P. (2021). Sep-28k: A dataset for stuttering event detection from podcasts with people who stutter. In *Proc. ICASSP 2021*, pages 6798–6802.

Li, H. et al. (2020). Speaker-invariant affective representation learning via adversarial training. In *Proc. ICASSP 2020*, pages 7144–7148.

Li, Y., Zhang, X., Jin, H., Li, X., Wang, Q., He, Q., and Huang, Q. (2018). Using multi-stream hierarchical deep neural network to extract deep audio feature for acoustic event detection. *Multimedia Tools and Applications*, 77(1):897–916.

López-de Ipiña, K., Martinez-de Lizarduy, U., Calvo, P., Beitia, B., García-Melero, J., Fernández, E., Ecay-Torres, M., Faundez-Zanuy, M., and Sanz, P. (2018). On the analysis of speech and disfluencies for automatic detection of mild cognitive impairment. *Neural Computing and Applications*, pages 1–9.

Meng, L., Xu, J., Tan, X., Wang, J., Qin, T., and Xu, B. (2021). Mixspeech: Data augmentation for low-resource automatic speech recognition. In *Proc. ICASSP 2021*, pages 7008–7012.

Meng, Z. et al. (2018a). Speaker-invariant training via adversarial learning. In *Proc. ICASSP 2018*, pages 5969–5973.

Meng, Z., Li, J., Chen, Z., Zhao, Y., Mazalov, V., Gong, Y., and Juang, B.-H. (2018b). Speaker-invariant training via adversarial learning. In *Proc. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5969–5973.

Minnis, S. (2020). Speech disorders. [Online; accessed 10-November-2022].

Mitra, V., Huang, Z., Lea, C., Tooley, L., Wu, S., Botten, D., Palekar, A., Thelapurath, S., Georgiou, P., Kajarekar, S., and Bigham, J. (2021). Analysis and tuning of a voice assistant system for dysfluent speech. In *Proc. Interspeech 2021*, pages 4848–4852.

Mohamed, A., Lee, H.-y., Borgholt, L., Havtorn, J. D., Edin, J., Igel, C., Kirchhoff, K., Li, S.-W., Livescu, K., Maaløe, L., Sainath, T. N., and Watanabe, S. (2022). Self-supervised speech representation learning: A review. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1179–1210.

Murphy, K. P. (2012). *Machine Learning: a Probabilistic Perspective*. MIT press.

Naderi, N. and Nasersharif, B. (2017). Multiresolution convolutional neural network for robust speech recognition. In *Proc. 2017 Iranian Conference on Electrical Engineering (ICEE)*, pages 1459–1464.

Narendra, N. and Alku, P. (2019). Dysarthric speech classification from coded telephone speech using glottal features. *Speech Communication*, 110:47–55.

Nassif, A. B. and *et al.* (2019). Speech recognition using deep neural networks: A systematic review. *IEEE Access*, 7:19143–19165.

Neef, N. E., Hoang, T. L., Neef, A., Paulus, W., and Sommer, M. (2015). Speech dynamics are coded in the left motor cortex in fluent speakers but not in adults who stutter. *Brain*, 138(3):712–725.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., and Ng, A. Y. (2011). Multimodal deep learning. In *Proc. ICML*.

NIDCD (2015). Stuttering. *National Institute on Deafness and Other Communication Disorders (NIDCD)*.

Ning, Y., He, S., Wu, Z., Xing, C., and Zhang, L.-J. (2019a). A review of deep learning based speech synthesis. *Applied Sciences*, 9(19):4050.

Ning, Y., He, S., Wu, Z., Xing, C., and Zhang, L.-J. (2019b). A review of deep learning based speech synthesis. *Applied Sciences*, 9(19):4050.

Nöth, E., Niemann, H., Haderlein, T., Decher, M., Eysholdt, U., Rosanowski, F., and Wittenberg, T. (2000). Automatic stuttering recognition using hidden Markov models. In *Proc. Sixth International Conference on Spoken Language Processing*.

NSA, N. S. A. (2009). The experience of people who stutter: A survey by the national stuttering association. *New York, NY: Author*.

Opitz, J. and Burst, S. (2019). Macro F1 and macro F1 a note. *arXiv preprint arXiv:1911.03347*.

Organization, W. H. et al. (1977). *Manual of the international statistical classification of diseases, injuries, and causes of death: based on the recommendations of the ninth revision conference, 1975, and adopted by the Twenty-ninth World Health Assembly*. World Health Organization.

Panayotov, V., Chen, G., Povey, D., and Khudanpur, S. (2015). Librispeech: An ASR corpus based on public domain audio books. In *Proc. 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Park, D. S., Chan, W., Zhang, Y., Chiu, C.-C., Zoph, B., Cubuk, E. D., and Le, Q. V. (2019). SpecAugment: A simple data augmentation method for automatic speech recognition. In *Proc. Interspeech 2019*, pages 2613–2617.

Pepino, L. et al. (2021). Emotion recognition from speech using wav2vec 2.0 embeddings. In *Proc. Interspeech 2021*, pages 3400–3404.

Pironkov, G., Dupont, S., and Dutoit, T. (2016). Multi-task learning for speech recognition: an overview. In *Proc. European Symposium on Artificial Neural Networks*.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., and Vesely, K. (2011). The Kaldi speech recognition toolkit. In *Proc. IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*.

Ramig, P. R. and Adams, M. R. (1981). Vocal changes in stutterers and nonstutterers during high-and low-pitched speech. *Journal of Fluency Disorders*, 6(1):15–33.

Ratner, N. B. and MacWhinney, B. (2018). Fluency bank: A new resource for fluency research and practice. *Journal of Fluency Disorders*, 56:69.

Ravanelli, M., Parcollet, T., Plantinga, P., Rouhe, A., Cornell, S., Lugosch, L., Subakan, C., Dawalatabad, N., Heba, A., Zhong, J., et al. (2021). Speechbrain: A general-purpose speech toolkit. *arXiv preprint arXiv:2106.04624*.

Ravikumar, K., Rajagopal, R., and Nagaraj, H. (2009). An approach for objective assessment of stuttered speech using MFCC. In *Proc. International Congress for Global Science and Technology*, page 19.

Riaz, N., Steinberg, S., Ahmad, J., Pluzhnikov, A., Riazuddin, S., Cox, N. J., and Drayna, D. (2005). Genomewide significant linkage to stuttering on chromosome 12. *The American Journal of Human Genetics*, 76(4):647–651.

Riva-Posse, P., Busto-Marolt, L., Schteinschnaider, Á., Martinez-Echenique, L., Cammarota, Á., and Merello, M. (2008). Phenomenology of abnormal movements in stuttering. *Parkinsonism & Related Disorders*, 14(5):415–419.

Robb, M., Blomgren, M., and Chen, Y. (1998). Formant frequency fluctuation in stuttering and nonstuttering adults. *Journal of Fluency Disorders*, 23(1):73–84.

Roberts, M. Y. (2011). *Using emprical benchmarks to assess the effects of a parentimplemented language intervention for children with language impairments*. Vanderbilt University.

Roberts, P. M., Meltzer, A., and Wilding, J. (2009). Disfluencies in non-stuttering adults across sample lengths and topics. *Journal of Communication Disorders*, 42(6):414–427.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Saltuklaroglu, T. and Kalinowski, J. (2005). How effective is therapy for childhood stuttering? dissecting and reinterpreting the evidence in light of spontaneous recovery rates. *International Journal of Language & Communication Disorders*, 40(3):359–374.

Sawyer, J. (2010). By the numbers: Disfluency analysis for preschool children who stutter. In *Proc. International Stuttering Awareness Day Online Conference*.

Schuller, B. W., Batliner, A., Amiriparian, S., Bergler, C., Gerczuk, M., Holz, N., Larrouy-Maestri, P., Bayerl, S. P., Riedhammer, K., Mallol-Ragolta, A., Pateraki, M., Coppock, H., Kiskin, I., Sinka, M., and Roberts, S. (2022). The ACM multimedia 2022 computational paralinguistics challenge: Vocalisations, stuttering, activity, & mosquitos. In *Proc. ACM Multimedia 2022*, Lisbon, Portugal. ISCA. to appear.

Shah, J., Singla, Y. K., Chen, C., and Shah, R. R. (2021). What all do audio transformer models hear? probing acoustic representations for language delivery and its structure. *arXiv preprint arXiv:2101.00387*.

Sheikh, S. A., Sahidullah, M., Hirsch, F., and Ouni, S. (2021). StutterNet: Stuttering detection using time delay neural network. In *Proc. 2021 29th EUSIPCO*, pages 426–430.

Sheikh, S. A., Sahidullah, M., Hirsch, F., and Ouni, S. (2022). Machine learning for stuttering identification: Review, challenges and future directions. *Neurocomputing*, 514:385–402.

Shukla, A., Vougioukas, K., Ma, P., Petridis, S., and Pantic, M. (2020). Visually guided self supervised learning of speech representations. In *Proc. ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6299–6303.

Simonyan, K. and Zisserman, A. (2014). Two-stream convolutional networks for action recognition in videos. In *Proc. of the 27th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'14, page 568–576, Cambridge, MA, USA. MIT Press.

Sisman, B., Yamagishi, J., King, S., and Li, H. (2020). An overview of voice conversion and its challenges: From statistical modeling to deep learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Smith, A. and Weber, C. (2017). How stuttering develops: The multifactorial dynamic pathways theory. *Journal of Speech, Language, and Hearing Research*, 60(9):2483–2505.

Smith, S. (2022). Digital voice assistants in use to triple to 8 billion by 2023, driven by smart home devices. [Online; accessed 10-November-2022].

Snyder, D. et al. (2018). X-vectors: Robust DNN embeddings for speaker recognition. In *Proc. ICASSP*, pages 5329–5333. IEEE.

Starkweather, C. W. (1987). *Fluency and Stuttering*. Prentice-Hall, Inc.

Su, J., Jin, Z., and Finkelstein, A. (2020). Acoustic matching by embedding impulse responses. In *Proc. ICASSP, 2020*, pages 426–430.

Subramanian, A., Yairi, E., and Amir, O. (2003). Second formant transitions in fluent speech of persistent and recovered preschool children who stutter. *Journal of Communication Disorders*, 36(1):59–75.

Świetlicka, I., Kuniszyk-Jóźkowiak, W., and Smołka, E. (2009). Artificial neural networks in the disabled speech analysis. In *Computer Recognition Systems 3*, pages 347–354. Springer.

Szczurowska, I., Kuniszyk-Jóźkowiak, W., and Smołka, E. (2014). The application of kohonen and multilayer perceptron networks in the speech nonfluency analysis. *Archives of Acoustics*, 31(4 (S)):205–210.

Vachhani, B., Bhat, C., and Kopparapu, S. K. (2018). Data augmentation using healthy speech for dysarthric speech recognition. In *Proc. Interspeech 2018*, pages 471–475.

Vandenhende, S. et al. (2021). Multi-task learning for dense prediction tasks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1.

Vanhoutte, S., Cosyns, M., van Mierlo, P., Batens, K., Corthals, P., De Letter, M., Van Borsel, J., and Santens, P. (2016). When will a stuttering moment occur? the determining role of speech motor preparation. *Neuropsychologia*, 86:93–102.

Verde, L., De Pietro, G., and Sannino, G. (2018). Voice disorder identification by using machine learning techniques. *IEEE Access*, 6:16246–16255.

Villegas, B. et al. (2019a). A novel stuttering disfluency classification system based on respiratory biosignals. In *Proc. EMBC*, pages 4660–4663.

Villegas, B., Flores, K. M., Acuña, K. J., Pacheco-Barrios, K., and Elias, D. (2019b). A novel stuttering disfluency classification system based on respiratory biosignals. In *Proc. 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 4660–4663. IEEE.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. (1989). Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(3):328–339.

Ward, D. (2008). *Stuttering and Cluttering: Frameworks for Understanding and Treatment*. Psychology Press.

Ward, D. (2017). *Stuttering and Cluttering: Frameworks for Understanding and Treatment*. Psychology Press.

Watson, B. C. and Alfonso, P. J. (1982). A comparison of lrt and vot values between stutterers and nonstutterers. *Journal of Fluency Disorders*, 7(2):219–241.

Weiss, G. M. (2013). Foundations of Imbalanced Learning. *Imbalanced Learning: Foundations, Algorithms, and Applications*, pages 13–41.

Winata, G. I., Wang, G., Xiong, C., and Hoi, S. (2020). Adapt-and-adjust: Overcoming the long-tail problem of multilingual speech recognition. *arXiv preprint arXiv:2012.01687*.

Wingate, M. E. (1969). Stuttering as phonetic transition defect. *Journal of Speech and Hearing Disorders*, 34(1):107–108.

Woszczyk, D., Hedlikova, A., Akman, A., Demetriou, S., and Schuller, B. (2022). Data augmentation for Dementia detection in spoken language. In *Proc. Interspeech 2022*, pages 2858–2862.

Xiong, F., Barker, J., and Christensen, H. (2019). Phonetic analysis of dysarthric speech tempo and applications to robust personalised dysarthric speech recognition. In *Proc. ICASSP 2019*, pages 5836–5840.

Yairi, E. and Ambrose, N. (2013). Epidemiology of stuttering: 21st century advances. *Journal of Fluency Disorders*, 38(2):66–87.

Yaruss, J. S. and Conture, E. G. (1993). F2 transitions during sound/syllable repetitions of children who stutter and predictions of stuttering chronicity. *Journal of Speech, Language, and Hearing Research*, 36(5):883–896.

Zayats, V., Ostendorf, M., and Hajishirzi, H. (2016a). Disfluency detection using a bidirectional lstm. In *Proc. Interspeech 2016*, pages 2523–2527.

Zayats, V., Ostendorf, M., and Hajishirzi, H. (2016b). Disfluency detection using a bidirectional LSTM. In *Proc. Interspeech 2016*, pages 2523–2527.

Zebrowski, P. M., Conture, E. G., and Cudahy, E. A. (1985). Acoustic analysis of young stutterers' fluency: Preliminary observations. *Journal of Fluency Disorders*, 10(3):173–192.

Zeiler, M., Ranzato, M., Monga, R., Mao, M., Yang, K., Le, Q., Nguyen, P., Senior, A., Vanhoucke, V., Dean, J., and Hinton, G. (2013). On rectified linear units for speech processing. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3517–3521.

Zhang, X.-L. and Wang, D. (2016). A deep ensemble learning method for monaural speech separation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):967–977.