Mohamed Khider University
of Biskra Faculty of Science
and Technology Department
of Electrical Engineering

MASTER THESIS

Electrical Engineering
Telecommunication
Networks and Telecommunication

Réf. : Entrez la référence du document

Submitted and Defended by:

Miloudi Soundous & Rizoug soulaf

On: June 2025

Automatic System for Stuttering Detection and Classification Using Deep Learning

---

Board of Examiners:

Ms.	FEDIAS MERIEM	**MCB**	University of Biskra	**President**
Ms.	AICHA BARKAT	**MAA**	University of Biskra	**Examiner**
Ms.	MEDOUAKH SAADIA	**MCB**	University of Biskra	**Supervisor**

University year: 2024 - 2025

Mohamed Khider University
of Biskra Faculty of Science
and Technology Department
of Electrical Engineering

MASTER THESIS

Electrical Engineering
Telecommunication
Networks and Telecommunication

Réf. : Entrez la référence du document

Submitted and Defended by:

Miloudi Soundous & Rizoug soulaf

On: June 2025

Automatic System for Stuttering Detection and Classification Using Deep Learning

Presented by:

Miloudi Soundous
Rizoug soulaf

Favorable opinion of the supervisor:

Dr. MEDOUAKH Saadia

Favorable opinion of the jury president

Stamp and signature

Acknowledgment

Praise be to God, by whose grace good deeds are accomplished. We praise and thank Him for His grace and favor. We have reached this level primarily through His grace and then through our own efforts. To You be praise, O Lord, until You are pleased, to You be praise when You are pleased, and to You be praise after You are pleased.

We must also extend our sincere thanks and gratitude to the distinguished professor, Medouakh Saadia, who graciously accepted to supervise this work and has been generous with her sound guidance and continuous support since the beginning of this project.

With pride and gratitude, we extend our sincere thanks and appreciation to all the distinguished professors who have accompanied us throughout our university journey, especially Dr. Ben Issa Elias and Zakaria Salem, who generously offered us their knowledge and valuable guidance during the preparation of this thesis. We thank them for their efforts, patience, and continuous support. Thanks to their guidance, we were able to overcome many difficulties and achieve this work.

May God reward them on our behalf with the best reward, and may He place what they have provided us in the balance of His good deeds.

Her constant guidance and endless patience with us played a fundamental role in preparing this thesis.

We also thank ourselves for avoiding obstacles and continuing to challenge ourselves to the end in completing this work, and for encouraging each other. Finally, we thank every invisible hand that has helped us, even with moral support.

Dedications

No matter how many words I write, I will never find anything more truthful than the Almighty's words: "Allah will raise those who have believed among you and those who have been given knowledge, by degrees." All praise is due to Allah, abundant, good, and blessed. A page of life has now turned, in which we worked hard and diligently, in which we planted our studies and toil to reap excellence and success.

We dedicate the fruits of our success to those about whom Allah Almighty said: "And your Lord has decreed that you worship none but Him. And to parents, good treatment."

To my first role model, the embodiment of love and compassion, to the one whose prayers were the secret to my success and whose compassion was the healing agent for my wounds, to the one who guided me and accompanied me on all my life's journeys, my mother.

To my dear father, you are the pride of my life and the role model I am proud of. What I have achieved today is thanks to Allah, then to your efforts, your toil, and your constant vigilance with me. May you continue to be my support and help throughout my life.

To my dear brothers, companions of the first and last steps, whether through moral or material support.

Praise be to God, who facilitated my beginnings, completed my endings, and brought me to my goals. No path ends, and no effort or endeavor is concluded except by the grace of God. Praise be to God at the beginning and at every conclusion.

Abstract

Stuttering is a common speech disorder during which the flow of speech is interrupted by involuntary pauses, prolongations and repetition of sounds, syllables, or words. Stuttering can affect communication and the person's self-efficacy and can have social and emotional ramifications. Therefore, it is important that we can accurately identify and describe simply stuttered speech to support accurate diagnosis, early treatment, and monitor therapy outcomes. Identification stuttering is considered one of the difficult and complex challenge in the field of speech processing due to its complex nature. Recent advancements in artificial intelligence (AI) has introduced new possibilities for augmenting stuttering detection and classification procedures for treatment. In this work, we developed and implemented an artificial intelligence model based on deep learning to automatically detect and classify stuttering patterns by analyzing vocal characteristics through spectrograms. These spectrograms were extracted from a carefully selected subset of the SEP-28k dataset. We proposed and evaluated three convolutional neural network (CNN) architectures, DenseNet121, MobileNetV2, and ResNet34, with the aim of identifying the most effective model for automatic stuttering detection and classification system. Experimental results showed that DenseNet121 delivered the best performance among the three models, achieving an accuracy of 67.23%, a precision of 59.33%, a recall of 56.87%, and an F1-score of 58.97%. Furthermore, a simple web application was developed, that allows users to upload audio files and receive instant stuttering classification results through the trained model.

Keywords: Stuttering detection, speech disorder, deep learning, CNN, DenseNet-121, MobileNetV2, ResNet34.

Résumé

Le bégaiement est un trouble courant de la parole, caractérisé par des interruptions involontaires, des prolongations et des répétitions de sons, de syllabes ou de mots. Il peut affecter la communication et l'auto-efficacité de la personne, et avoir des répercussions sociales et émotionnelles. Il est donc important de pouvoir identifier et décrire avec précision le bégaiement afin de permettre un diagnostic précis, un traitement précoce et un suivi des résultats thérapeutiques. L'identification du bégaiement est considérée comme l'un des défis les plus complexes du traitement de la parole en raison de sa complexité. Les progrès récents de l'intelligence artificielle (IA) ont ouvert de nouvelles perspectives pour améliorer la détection et la classification du bégaiement en vue de son traitement. Dans ce travail, nous avons développé et implémenté un modèle d'intelligence artificielle basé sur l'apprentissage profond pour détecter et classer automatiquement les schémas de bégaiement en analysant les caractéristiques vocales au moyen de spectrogrammes. Ces spectrogrammes ont été extraits d'un sous-ensemble soigneusement sélectionné de la base de données SEP-28k. Nous avons proposé et évalué trois architectures de réseaux de neurones convolutifs (CNN), DenseNet121, MobileNetV2 et ResNet34, afin d'identifier le modèle le plus performant pour la détection et la classification automatiques du bégaiement. Les résultats expérimentaux ont montré que DenseNet121 offrait les meilleures performances parmi les trois modèles, avec une précision de 67,23 %, une précision de 59,33 %, un rappel de 56,87 % et un score F1 de 58,97 %. De plus, une application web simple a été développée, permettant aux utilisateurs de télécharger des fichiers audio et de recevoir instantanément les résultats de classification du bégaiement via le modèle entraîné.

Mots-clés : Détection du bégaiement, troubles de la parole, apprentissage profond, CNN, DenseNet-121, MobileNetV2, ResNet34

الملخص

التأتأة هي اضطراب شائع في الكلام يتميز بالمقاطعات اللاإرادية، والإطالة، وتكرار الأصوات، أو المقاطع، أو الكلمات. ويمكن أن يؤثر ذلك على قدرة الشخص على التواصل وكفاءته الذاتية، كما يمكن أن يكون له عواقب اجتماعية وعاطفية. لذلك من المهم أن نكون قادرين على تحديد ووصف التأتأة بدقة من أجل تمكين التشخيص الدقيق والعلاج المبكر ومراقبة النتائج العلاجية. يعتبر التعرف على التأتأة أحد التحديات الأكثر تحديًا في معالجة الكلام بسبب تعقيدها. لقد فتحت التطورات الأخيرة في مجال الذكاء الاصطناعي آفاقًا جديدة لتحسين الكشف عن التأتأة وتصنيفها للعلاج. في هذا العمل، قمنا بتطوير وتنفيذ نموذج ذكاء اصطناعي يعتمد على التعلم العميق للكشف عن أنماط التأتأة وتصنيفها تلقائيًا من خلال تحليل سمات الكلام باستخدام المخططات الطيفية. تم استخراج هذه المخططات الطيفية من مجموعة فرعية مختارة بعناية من قاعدة بيانات SEP-28k. لقد اقترحنا وقمنا بتقييم ثلاث هياكل للشبكات العصبية التلافيفية ((CNN، وهي DenseNet121، وMobileNetV2، وResNet34، لتحديد النموذج الأفضل أداءً للكشف التلقائي عن التلعثم وتصنيفه. أظهرت النتائج التجريبية أن DenseNet121 حقق أفضل أداء بين النماذج الثلاثة، بدقة 67.23%، ودقة 59.33%، واسترجاع 56.87%، ونتيجة F1 58.97%. بالإضافة إلى ذلك، تم تطوير تطبيق ويب بسيط يسمح للمستخدمين بتحميل الملفات الصوتية وتلقي نتائج تصنيف التلعثم على الفور عبر النموذج المدرب.

الكلمات المفتاحية: اكتشاف التأتأة، اضطرابات الكلام، التعلم العميق، CNN، DenseNet-121، MobileNetV2، ResNet34

Table of Content

Chapter 3 : Implementation and Experimental Results

General Conclusion

References

List of figures

Chapter 1: Overview of Stuttering Detection and Classification

Chapter 2: Automatic Stutter Detection and Classification System

Chapter 3: Implementation and Experimental Results

List of table

Chapter 3: Implementation and Experimental Results

List of Abbreviations

AI: Artificial Intelligence
ML: Machine Learning
DL: Deep Learning
ANNs: Artificial Neural Networks
CNN: Convolutional Neural Network
RNN: Recurrent Neural Network
LSTMs: Long Short-Term Memory Networks
MMSD-Net: Multi-Modal Stuttering Detection Network
SVMs: Support Vector Machines
HMMs: Hidden Markov Models
ASR: Automatic Speech Recognition
NLP: Natural Language Processing
Cl: Convolution Layer
Pl: Pooling Layer
ReLU: Rectified Linear Unit
MFCC: Mel-Frequency Cepstral Coefficients
SEP-28K: Stuttering Events and Patterns Dataset (28,000 clips)
F1-Score: Harmonic Mean of Precision and Recall
CPU: Central Processing Unit
GPU: Graphics Processing Unit
VRAM: Video RAM
RAM: Random Access Memory
ROM: Read-Only Memory
SSD: Solid-State Drive
cuDNN: CUDA Deep Neural Network
CUDA: Compute Unified Device Architecture
NVIDIA: Company name – not an abbreviation
OS: Operating System
CMD: Command Prompt
HTML: HyperText Markup Language
PyTorch: Python-based scientific computing package
ONNX: Open Neural Network Exchange
PWS: People Who Stutter
SPLs: Sound Pressure Levels
IPA: International Phonetic Alphabet
URL: Uniform Resource Locator
GB: Gigabyte

General Introduction

General Introduction

Digital technology has opened many aspects of life for us, becoming a fundamental element that has revolutionized many vital sectors, including education and communication. With this rapid development in the fields of artificial intelligence (AI) and deep learning, these technologies have enabled us to develop intelligent systems capable of analyzing audio, text, and visual data with high accuracy, contributing to the emergence of innovative applications that help improve quality of life. Among the fields that have benefited from these transformations is the field of speech and language processing, where models capable of recognizing voices and analyzing speech have been developed to provide assistance to those with speech disorders. One of the most prominent of these disorders is stuttering, which is one of the most common problems and clearly impacts an individual's interaction in many areas of life. It manifests itself in the form of repetitions, prolongations, or sudden pauses during speech.

Traditionally, stuttering is diagnosed by speech therapists, a process that can be tiring and time-consuming, and results may vary depending on the specialist's experience. Here, the importance of employing artificial intelligence and deep learning techniques to develop systems capable of automatically and more accurately detecting and classifying stuttering emerges. Deep learning has enabled significant progress in automatic stutter detection, with models such as StutterNet, a prominent deep learning model, that leverage a time-delay neural network (TDNN) architecture designed to capture contextual aspects of disfluent utterances solely from acoustic signals, without relying on automatic speech recognition (ASR) or language models. Other models have used convolutional neural networks (CNNs) and recurrent neural networks (RNNs), including Long Short-Term Memory (LSTM) networks, to learn stutter-related features from spectrograms or Mel-frequency cepstral coefficients (MFCCs).

This memorandum aims to design and implement an artificial intelligence model using deep learning capable of detection and classifying stuttering patterns (such as repetition, prolongation, and pauses), by analyzing voice characteristics using visual representations such as spectrograms. Three different convolutional neural network-based models were used to design and compare models. The first model is MobileNet, designed for efficiency on mobile and embedded devices, using depthwise separable convolutions to reduce parameters and computation. The second model, DenseNet, focuses on dense connectivity between layers to improve gradient flow and feature reuse, enhancing learning in very deep networks. Finally, ResNet34 is a deep neural network model that uses residual connections to ease training. It consists of 34 layers and is known

for its efficiency in image classification and visual data processing by overcoming the vanishing gradient problem. We have chosen to organize our study around three main chapters as follows:

The first chapter presents the theoretical background on speech disorders in general, focusing on stuttering and the importance of early detection and its role in improving treatment. It also presents the most prominent challenges and reviews the traditional and modern methods currently used in this field.

Chapter Two covers the basics of machine learning and then delves into deep learning techniques, focusing on convolutional neural networks (CNNs) due to their high efficiency in processing audio data. This is achieved by converting it into representational images to accurately identify stuttering patterns.

Chapter Three presents the practical aspects of the project, starting with database preparation, audio data processing, model training and evaluation by the results obtained.

We will end this dissertation with a general conclusion and the perspectives.

Chapter 1:

Overview of Stuttering Detection and Classification

1.1. Introduction

Speech is the most common form of communication because it allows a person to express thoughts and feelings, and it typically begins to develop in early childhood. At this stage, any flaw could result in speech disorders like cluttering or stuttering. Stuttering is a neurodevelopmental speech disorder that affects 1% of the global population. Early detection and intervention have been shown to significantly increase the chances of recovery and improve the child's communicative confidence and social integration. Therefore, implementing reliable early detection systems including those based on automated speech analysis, machine learning and deep learning plays a crucial role in improving clinical outcomes and reducing the long-term impact of stuttering.

In this chapter, we provide an overview of the literature in speech disorder. Then, we present the importance of early Stuttering detection, its types and its challenges. Finally, we review several approaches to Stuttering detection and classification. We focus specifically on the deep learning-based stuttering detection and classification methods, whose application in stuttering detection is still relatively limited.

1.2. Overview of speech disorders

Speech disorder is a delay in producing language and developing the speech and it reduces the voice quality which involves the sound, volume, and the disturbance of sound rhythm and it is difficult to be understood. In speech, Feldman [1] categorizes the speech into several categorizes; first, speech sound consisted of articulation, coordinating breath and movements, motor planning, execution; second, voice and resonance, third, fluency. The way people speech has become an indicator to interact to the society. Suffering bad capability in speech can cause the sufferer feels isolated. Mostly, people who suffer from speech disorder usually get isolated in society. For example, children who suffer from speech disorder, mostly get bully at school. It is one of the evidences that having a speech disorder can cause the sufferer becomes isolated. According to Lanier (2010) [2], the speech disorders are divided into five types; those are apraxia, aphasia, stuttering, cluttering and dysarthria. Mostly apraxia, aphasia and dysarthria are caused by the damage of brain such as the sensory motor, and the neurological, while the stuttering and cluttering can be caused by the neurogenic or psychogenic [3].

In the wider literature, the term “speech and language disorders” is classified under communication impairments, along with hearing disorders, deafness, and physical disabilities that affect speech, as shown in figure 1.1.

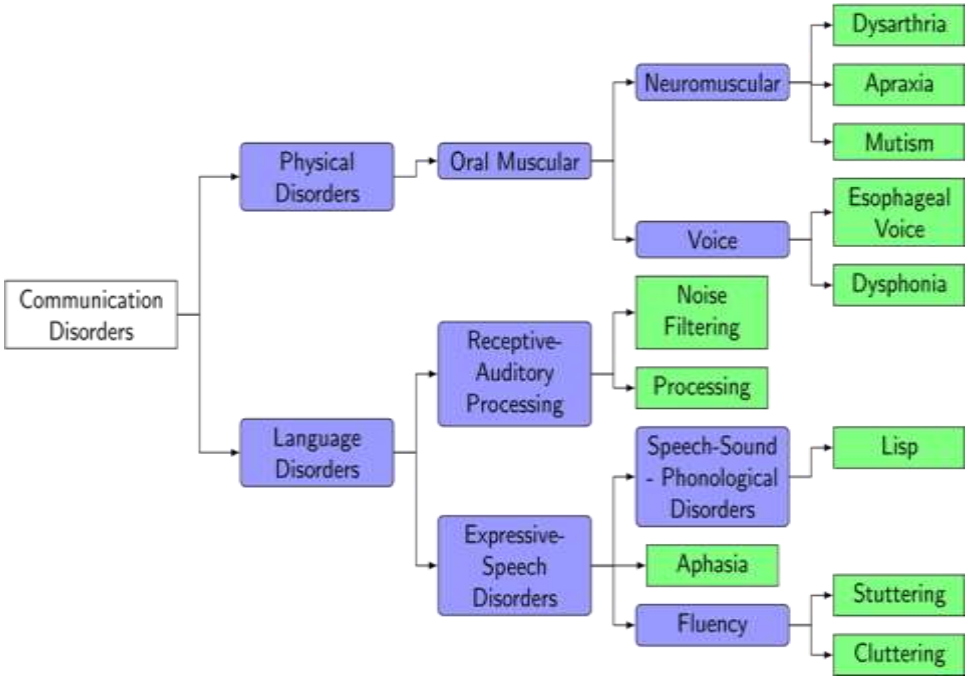

Figure 1.1: Speech disorder taxonomy [4].

1.3. Background on stuttering

1.3.1. Definition of Stuttering

Stuttering is a disorder that appears as an interruption in the smooth flow or "fluency" of speech. Breaks or disruptions that occur in the flow of speech are labelled "disfluencies." All speakers may experience disfluent events, especially under certain conditions, such as nervousness, stress, fatigue or complexity of language. Stuttering, on the other hand, is a different type of disfluency. People who stutter generally tend to have more disfluencies than other speakers overall. They may develop negative perceptions and thoughts about their speech and themselves as a result of their speaking difficulties [5].

1.3.2. Types of stuttering (Disfluencies)

Stuttering is a speech disorder characterized by the inability to pronounce words smoothly; it can be classified into several types based on the nature of the disfluencies. Understanding these types is important for specialists in order to track patient cases and summarize them accurately. The main types of stuttering are as follows [6]:

1.3.1.1 Developmental stuttering

This is the most common type of stuttering in children. It usually happens when a child is between ages two and five. It may happen when a child's speech and language development lags behind what he or she needs or wants to say.

1.3.1.2 Neurogenic stuttering

Neurogenic stuttering may happen after a stroke or brain injury. It happens when there are signal problems between the brain and nerves and muscles involved in speech.

1.3.1.3 Psychogenic stuttering

Psychogenic stuttering is not common. It may happen after emotional trauma. Or it can happen along with problems thinking or reasoning.

1.3.3. Stuttering Patterns

Among the speech, patterns commonly associated with all forms of stuttering, three types of stuttering are most common. These are also referred to as disfluencies.

- Repetition of sounds (e.g. I want want a turn)
- Prolongations; stretching of sounds (e.g. I wwwwwwant a turn)
- Blockages; silent pauses where speech is physically blocked (eg. I ……. want a turn)

1.3.3.1. Repetitions

Repetition, as the name suggests, is the type of disfluency that results due to the repetition of a part of an utterance, disrupting the speech flow; it can be at different levels, i.e. syllabic, word level (see figure 1.2), or phrase level [7].

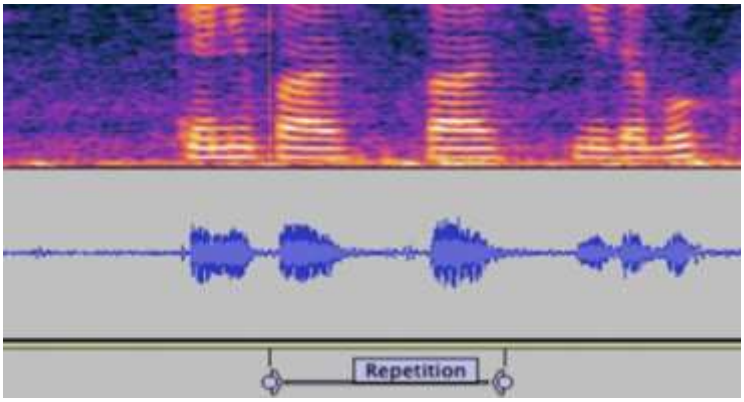

Figure 1.2: Waveform and Spectrogram plot for sentence “Can you pass pass me the book? [7]

1.3.3.2. Prolongations :

Prolongations can be referred to as lengthening of a particular sound or syllable, which tends to make the word longer than a fluent word. Prolongation can be of vocalized as well as nonvocalized sounds. One of the differential properties of prolongations that we can see from

figure 1.5 is that throughout the prolonged segment, it exhibits a consistent spectral structure, which is due to the continuation of the same sound through the segment [7].

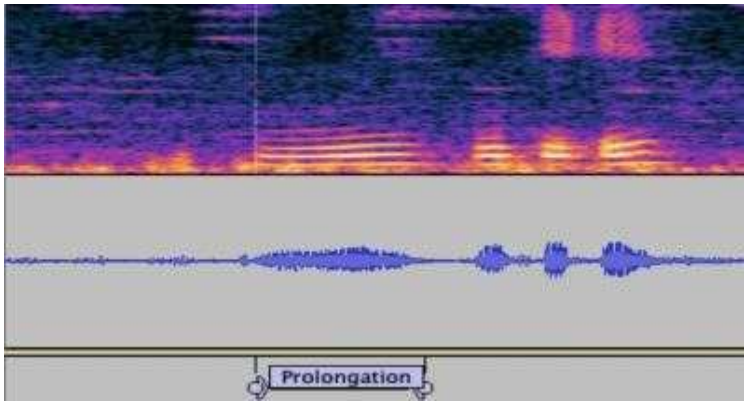

Figure 1.3: Waveform and Spectogram plot for sentence “whooose book it is? [7]

1.3.3.3. Blockages

At the heart of chronic stuttering, specifically the kind of dysfluency that ties you up so you momentarily cannot utter a word, is something called a “speech block.” We have traditionally seen speech blocks as having a life of their own, mysterious and unexplainable. Speech blocks seem to “strike” us at odd moments, usually without our knowing why [8].

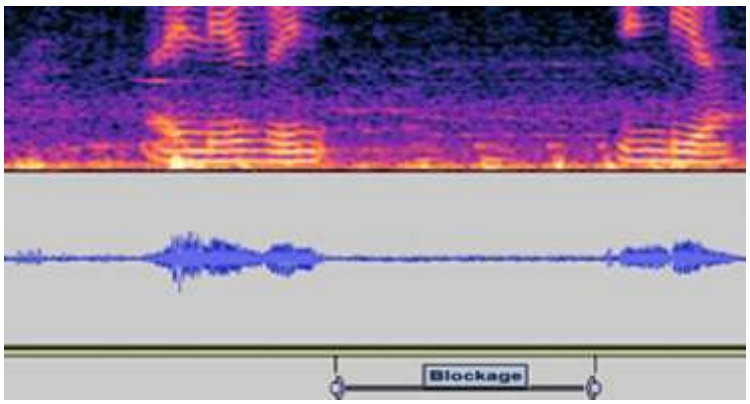

Figure 1.4: Waveform and Spectogram plot for sentence “I live in …….... India.” [7]

1.3.4. Importance of Early Stuttering detection

Early identification of stuttering is important so that therapy can begin while compensatory changes to the brain can still occur. Treatment is most effective when a child is younger than 6 years of age, as younger children have greater neural plasticity, which facilitates better recovery outcomes. Early detection helps minimize the chances of the patient developing social anxiety, impaired social skills, maladaptive compensatory behaviours, and negative attitudes toward communication.

Identifying stuttering at an early stage enables speech-language pathologists to differentiate between transient disfluencies and persistent stuttering, allowing timely and appropriate therapeutic intervention. Delayed diagnosis, on the other hand, may lead to the development of secondary behaviors such as avoidance, anxiety, and reduced self-esteem, which can persist into adolescence and adulthood [9].

Importance of early detection and intervention have been shown to significantly increase the chances of recovery and improve the child's communicative confidence and social integration. Therefore, implementing reliable early detection systems including those based on automated speech analysis, machine learning and deep learning plays a crucial role in improving clinical outcomes and reducing the long-term impact of stuttering [10].

1.3.5. Role of Technology in Diagnosing Speech Disorders

Based on the causes of underlying speech disorders, some studies have provided treatment or assistance interventions for individuals with speech impairments. While others apply machine learning and deep learning methods to detect, classify, predict, and assess speech disorders. Machine learning showed notable impacts on improving communication tools for individuals with speech impairments as they enhance the accuracy and accessibility of speech recognition and word predictability, such as AI-driven speech-to-text and text-to-speech applications. Moreover, ML provides a host of powerful, automated algorithms designed to handle vast amounts of data across various disciplines like speech recognition. Recent research demonstrated that deep signal analysis of voice using ML techniques to recognize speech with disorders showed promising results by extracting significant features from these signals [11].

Modern diagnostic tools employ machine learning (ML) and AI to analyse speech signals with high precision. Techniques such as deep learning, convolutional neural networks (CNNs), and signal processing extract critical speech features, such as Mel-frequency Cepstral Coefficients (MFCCs) and spectrogram to identify speech disorders. These models can differentiate between healthy and disordered speech patterns, enabling early and accurate diagnosis of various speech disorders, including stuttering.

Technology plays an increasingly pivotal role in the diagnosis, objective assessment, and management of stuttering, a neurodevelopmental speech fluency disorder. Advances in digital tools, machine learning, and acoustic analysis have enhanced the precision and reliability of stuttering diagnosis beyond traditional perceptual clinical evaluations.

1.4. Stuttering Detection Application Areas

Detecting stuttering is a vital field that is receiving increasing attention in many sectors because it has a great impact on improving the quality of life and human communication. It is used in various fields, including:

1.4.1. Speech Therapy and Clinical Assessment

AI-driven stuttering detection aids speech-language pathologists (SLPs) in diagnosing and monitoring stuttering. By analysing speech patterns, these tools can identify disfluencies such as repetitions, prolongations, and blocks. This facilitates objective assessments and personalized therapy plans. For instance, computational intelligence-based systems have been developed to classify stuttering events, enhancing the accuracy of evaluation [12].

Figure 1.5: Image showing a clinical assessment session for stuttering.

1.4.2. Educationnel Settings

In schools, early detection of stuttering is crucial for timely intervention. Automated tools assist educators and SLPs in identifying students who may require support, enabling the implementation of appropriate strategies to improve communication skills and academic performance [12].

Figure 1.6: Classroom Setting for Observing and Detection stuttering

1.4.3. Customer Service and Voice-Activated Technologies

Integrating stuttering detection into voice recognition systems enhances accessibility for individuals who stutter. Companies like Apple have focused on improving automatic speech recognition (ASR) models to better accommodate atypical speech patterns, ensuring more inclusive user experiences [13].

Figure 1.7: Image showing Using Voice Technology to Support Individuals Who Stutter

1.4.4. Telehealth and Remote Therapy

Stuttering detection technology supports teletherapy by providing real-time analysis of speech during virtual sessions. This enables therapists to monitor progress and adjust treatment plans accordingly. Telerehabilitation has proven effective in delivering speech therapy services, especially in remote or underserved areas [14].

Figure 1.8: Teletherapy Session with Integrated Stuttering Detection

1.5. Challenges in Stuttering Detection Systems

Stuttering detection systems, especially those based on automated speech processing and machine learning, several challenges that limit their performance and real-world applicability. These challenges emerge from the variability of speech patterns, limited data availability, and the complexity of distinguishing stuttering-like disfluencies from normal disfluencies.

1.5.1. Problem of Data

One of the major challenges in stuttering detection research is the limited availability of relevant data, particularly datasets containing natural speech with disfluencies. Several studies [15,16,17,18] have highlighted that the scarcity of data significantly impacts the research results, often leading to outcomes that are less accurate than expected. The lack of extensive and diverse datasets greatly affects the performance and robustness of the proposed methods. Therefore, addressing this issue is a fundamental step toward improving the accuracy and reliability of research findings.

Medical data collection in general, including stuttering-related data, is a costly and resource-intensive process [19]. Additionally, a wide variety of speakers and sentences are needed for a comprehensive analysis. One of the main obstacles contributing to data scarcity is the difficulty of data collection itself. It requires organizing meetings and recording sessions with people who stutter (PWS) as they engage in spontaneous speech. Asking PWS to read from a predetermined list can often reduce the frequency of stuttering occurrences [20]. Sheikh et al. [21] proposed using data augmentation as a solution to address the limited availability of datasets.

1.6. Overview of Stuttering detection and classification Approaches.

1.6.1. Traditional Approaches

The traditional approach to evaluating stuttering is to manually tally the instances of different stuttering types and express them as a ratio that is relative to the total words in a speech segment. Nevertheless, owing to its time-intensive and subjective nature, this method is not without limitations that lead to inconsistencies and potential errors when different evaluators are involved [22]. Manually detecting stuttering exhibits several challenges. First, distinguishing stuttering from other speech disfluencies can be difficult, considering that subtle instances may resemble hesitations or pauses. Furthermore, consistent detection of stuttering becomes a complex task because the severity and frequency of stuttering can vary widely among people and across different contexts. Moreover, factors such as the speaker’s age, gender, and language as well as the speaking task and the context in which the speech is produced can further complicate the identification of stuttering [23]. There are several traditional approaches:

1.6.1.1. Clinical Evaluation by Speech-Language Pathologists

In traditional stuttering treatment method, a Speech Language Pathologist (SLP) counts the dysfluencies, classify the abnormality and estimate the severity manually, to keep track further improvement in the quality of treatment. This type of stuttering treatment method is biased and subjective [24] and purely depends on the practice and experience of the SLPs may be causes the human error. Hence, to make it feasible and reliable SLPs need to facilitate with speech processing technology and artificial intelligence [25].

1.6.1.2. Auditory and Phonetic Analysis

Traditionally, stuttering has been assessed and classified using auditory-perceptual and phonetic analysis techniques. These methods rely on human expertise to identify and categorize disfluencies in speech. Key components include [26]:

- Auditory Perceptual Evaluation.
- Phonetic Transcription.
- Temporal Analysis.
- Visual Inspection of Spectrograms.

1.6.1.3. Self-Assessment Questionnaires and Reports

Assessment plays a crucial role in stuttering treatment, whether self-assessment by a speech-language pathologist (SLP) or self-assessment by people who stutter (PWS). In this case, performance analysis is conducted before, during, and after treatment to determine stuttering severity using a rating scale [27]. To measure stuttering severity in adults and children, the Stuttering Severity Measurement Tool (SSI) is used, and for young children, the Stuttering Predictor Tool (SPT) is used. These tools provide information about the type, frequency, and duration of stuttering and evaluate the entire utterance to determine any secondary characteristics, enabling appropriate treatment to be determined [28].

1.6.2. Automatic for Stuttering Detection and Classification Approaches

The increasing need for improved detection and management of stuttering, there is a noticeable trend in adopting innovative technologies, particularly artificial intelligence (AI) [29]. The application of AI in identifying and classifying stuttering indicates an essential development in the study of speech-related issues. AI has a special ability to understand complex speech patterns that might not be easy for humans to notice, and this capability can help in the early detection of stuttering [30].

1.6.2.1. Machine Learning-Based Approaches

Recent developments in machine learning have shown significant promise for enhancing stuttering analysis and identification. Machine learning models can precisely identify stuttering events and offer insights into their characteristics and underlying causes by employing algorithms that can learn from vast volumes of data. We can increase the precision and usefulness of voice assistants for those who stutter by using machine learning for stuttering detection. The subjectivity and variability of human judgment, one of the major problems in the detection and categorization of stuttering, may be resolved by using machine learning to detect stuttering [30]. Among the approaches and methods of statistical machine learning techniques used to detect and classify stuttering that have proven effective are: support vector machines (SVMs) [31], artificial neural networks (ANNs) [32], and hidden Markov models (HMMs) [33]. SVMs have emerged as the most popular classification tools, demonstrating high accuracy across different types of stuttering [31, 34, 35].

The ML technical approach is based on stages of digital processing of the audio signal, including feature extraction and analysis, in preparation for classifying different stuttering patterns using the following machine learning techniques, as shown in the following figure:

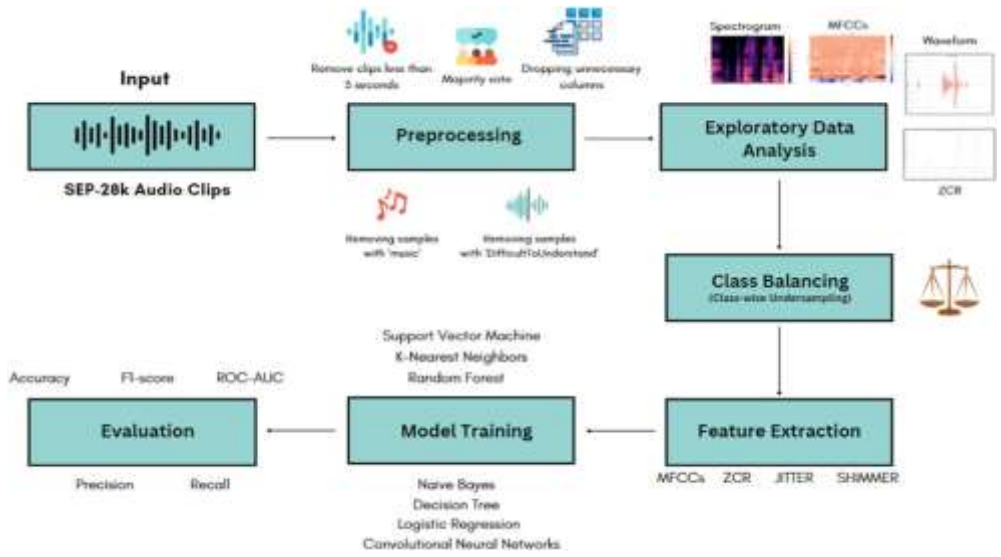

Figure 1.9: Diagram of Stuttering Detection Using Machine Learning [36].

a) Support Vector Machines (SVMs)

Support vector machines (SVMs) are among the most popular methods for classifying speech disorders, due to their ability to handle multidimensional audio data. In a 2013 study, Mahesha and Vinod [37] used a multiclass SVM model to classify three categories of stuttering: prolongation, word repetition, and syllable repetition. The researchers relied on three types of

acoustic features: linear predictive syllable coefficients (LPCCs), MFCC coefficients, and linear predictive coding (LPC). The model achieved accuracy rates of 92%, 88%, and 75%, depending on the type of feature used. In another study, Ravi Kumar [38] and colleagues time-matching (DTW) algorithm, achieving an accuracy of 94.35%. Pálfy and Pospíchal [39] also compared two types of SVM kernels (linear and RBF), with the linear kernel achieving 98% accuracy, while the RBF kernel achieved 96.4% accuracy, also using MFCC features [40].

b) Hidden Markov Models (HMMs)

HMMs lie at the heart of all contemporary speech recognition systems and have been successfully extended to disfluency classification systems. A simple and effective framework is provided by HMMs for modeling temporal sequences. Wisniewski et al. [41] used Euclidean distance as a codebook based on 20 MFCCs with HMMs. They reported an average recognition rate of 70% for two stuttering classes including blocks and prolongation with deleted silence and 60 frames of window length. Tan et al. [42] used 12 MFCC features with HMMs. The average recognition rate is 93% . This tool recognizes only normal and stutter utterances and is not classifying different types of disfluencies [43].

c) Artificial Neural Networks (ANNs)

Artificial neural networks (ANNs) are essential tools in speech recognition and speaker recognition, and their use has recently expanded to include the classification and identification of stuttered speech. Howell et al. [44] used two neural networks to identify two types of stuttering: repetition and prolongation. The network was trained using 20 eigenvectors (ACFs), 19 vocoder coefficients with a frame length of 10 ms, and 20 frames of envelope coefficients. The results showed that the best accuracy achieved when using envelope coefficients was 82% for prolongation and 77% for repetition [44]. Using ACF-SC coefficients resulted in an accuracy of 79% for prolongation and 71% for repetition. In a subsequent study, Howell and colleagues [45, 46] designed a two-stage system to detect two types of fluency mismatches (repetition and prolongation). Speech is divided into linguistic units and then classified into the appropriate category using inputs including duration and energy peaks. On a sample of 12 speakers, the average recognition accuracy was 78.01%. [46].

1.6.2.2. Deep Learning-Based Approaches

In addition to machine learning, deep learning approaches such as convolution recurrent neural networks, and sequence to sequence methods are commonly used in stuttering detection models.the past research also underscores a paradigm shift towards deep learning techniques, particularly CNNs [47] and recurrent neural networks (RNNs), in stuttering identification. These

approaches offer automatic feature extraction capabilities and have exhibited superior performance compared to ML methods. The outputs changes based on what we want to classify

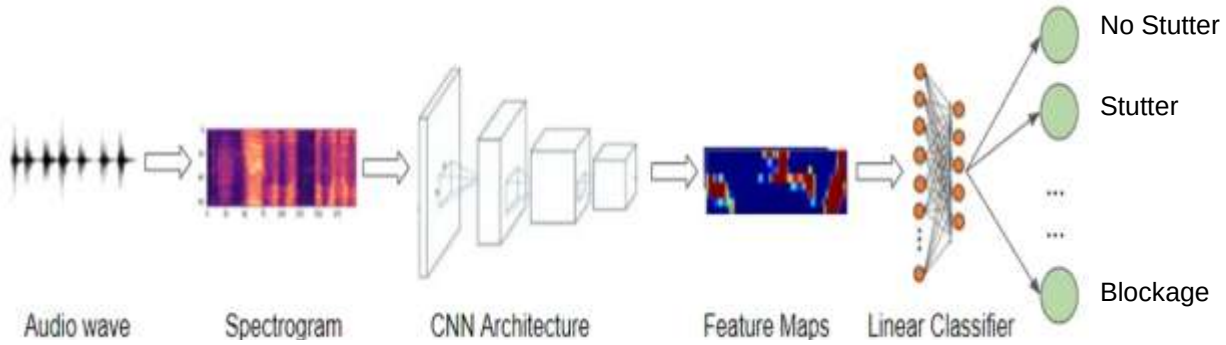

Figure 1.10: Diagram of Stuttering Detection Using Deep Learning [48].

a) Convolutional Neural Networks (CNNs)

Despite the encouraging results achieved by ML models, they suffer from limitations in extracting complex visual features from audio signals. This has led some researchers, to use convolutional neural networks (CNNs) due to their ability to automatically learn temporal and spatial patterns from speech spectral representations such as spectrograms. Convolutional Neural Networks (CNNs) are a powerful class of deep learning models widely applied in various tasks, including object detection, speech recognition, computer vision, image classification, and bioinformatics [49]. They have also demonstrated success in time series prediction tasks [50]. CNNs are feedforward neural networks that leverage convolutional structures to extract features from data [51]. CNN has a two-stage architecture that combines a classifier and a feature extractor to provide automatic feature extraction and end-to-end training with the least amount of pre-processing necessary [52]. Unlike traditional methods, CNNs automatically learn and recognize features from the data without the need for manual feature extraction by humans [53]. The design of CNNs is inspired by visual perception [51].

The major components of CNNs include the convolutional layer, pooling layer, fully connected layer, and activation function [54, 55]. CNN models have shown that displayed average accuracies of 91.15 % to 91.75 % on different datasets [56]. While challenges persist in generalizing models to larger datasets and capturing diverse stuttering patterns effectively, the advancements in machine learning and deep learning offer renewed hope for the automated identification of stuttering [57].

b. StutterNet,

StutterNet [58], 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 automatic speech recognition (ASR) combined with language models for stuttering detection. Compared to the existing work, StutterNet method relies solely on the acoustic signal, and uses a time-delay neural network (TDNN) suitable for capturing contextual aspects of the disfluent utterances. This system was evaluating on the UCLASS stuttering dataset consisting of more than 100 speakers. StutterNet achieves promising results and outperforms the state-of-the-art residual neural network based method. The number of trainable parameters of the proposed method is also substantially less due to the parameter sharing scheme of TDNN. This system reported an overall average accuracy of 50.79% and Mathews correlation coefficient (MCC) of 0.23, in comparison to the ResNet+BiLSTM-based system comprising 46.10% overall average accuracy and 0.21 MC.

c. Recurrent Neural Networks (RNNs)

Recurrent neural networks (RNNs) are a widely used and familiar algorithm in the field of deep learning [59–60]. RNNs are mainly applied in speech processing and natural language processing contexts [60, 61]. Unlike traditional networks, RNNs use sequential data in the network. Because the structure embedded in the data sequence provides valuable information, this feature is essential for a range of different applications, A specific type of recurrent neural network (RNN) called a long short-term memory (LSTM) was created to model long-term dependencies in sequential data. When processing sequential data, such as time series, natural language text, or speech, recurrent neural networks (RNNs), a type of neural network, use loops to pass information from one step to the next in the sequence. Traditional RNNs tend to "forget" information from an earlier stage in the sequence as they analyze subsequent steps, making it difficult to model long-term relationships in the data.

By introducing a new type of unit called a memory cell, which can store and modify information over a long period of time, recurrent neural networks (LSTMs) were created to solve this problem. An input, output, and forget gate controls the flow of information into, from, and within each memory cell. Each memory cell is connected to one of these gates. The input gate determines the currently entered information that should be stored in the cell, the output gate determines the information that should be output from the cell, and the forget gate determines the information that should be forgotten from the cell. The weights acquired through training are used to open and close these gates [62]. In addition to their use in processing sequential data such as speech and text, RNNs, especially LSTMs, have been effectively employed to detect and classify stuttering

patterns, such as repetition, prolongation, and blocks. These models perform well due to their ability to capture long-term temporal relationships in audio signals, making them suitable for automated diagnosis of speech disorders. For example, a study by [Alharbi et al., 2021] demonstrated that an LSTM-based model was able to accurately classify stuttering types when trained on audio data from speakers who stutter. [63]

ConvLSTM networks are an evolution that combines convolutional networks (CNNs) and long-term memory (LSTM), allowing them to simultaneously extract spatial and temporal features from audio signals. ConvLSTM has been successfully used to detect stuttering with high accuracy, especially when dealing with spectral representations such as mel-spectrograms. In a recent study, a ConvLSTM model was trained on audio data from speakers who stutter and demonstrated its effectiveness in classifying speech segments as normal or stuttered, with improved performance compared to traditional models [64]

1.8 Conclusion:

In summary, this chapter has explored the phenomenon of stuttering in its general context, reviewed its primary forms, and emphasized the significance of early detection in improving treatment outcomes. Furthermore, it has addressed the shortcomings of conventional methods in handling speech disorders. By introducing artificial intelligence—especially deep learning and machine learning techniques—a new path emerges, offering the potential for faster, more precise, and automated diagnostic tools that may surpass traditional approaches in both efficiency and reliability.

Chapter 2:

Automatic Stutter Detection and Classification System

2.1 Introduction

Deep learning plays a critical role in structuring the diagnosis and analysis of speech disorders by leveraging advanced neural network architectures to detect, classify, and even reconstruct pathological speech. Audio classification is the process of analyzing audio recordings and categorizing them. Recently, many algorithms have been proposed for strutting detection and classification using deep neural networks. In this chapter, we focus on deep learning approaches using Convolutional Neural Networks (CNNs) for detect and classify strutting. Specifically, we explore and evaluate different CNN architectures, including MobileNetV2, DenseNet121, and ResNet34, to analyze their performance and effectiveness in addressing the challenges of strutting classification problems**.**

2.2 Machine learning

Machine learning is a subfield of artificial intelligence that teaches machines how to learn, whereas artificial intelligence (AI) is a broader science that seeks to replicate human abilities. Machine learning is an artificial intelligence method that teaches computers to learn from their previous experiences. Machine learning algorithms do not use a predetermined equation as a model, but rather "learn" information directly from data through computational techniques. As the number of learning examples grows, the algorithms adapt and improve their performance. [65]

Machine learning is categorized into three main types: supervised learning, unsupervised learning, and reinforcement learning.

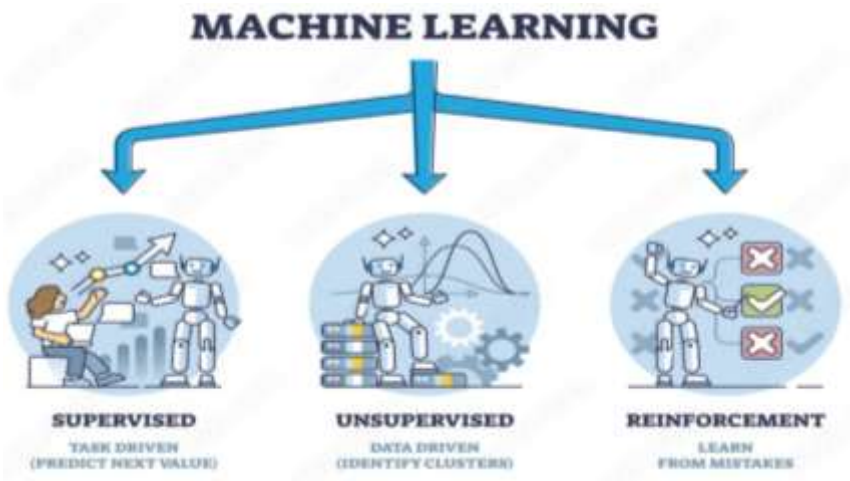

Figure 2.1: Types of machine learning [66].

2.2.1. Supervised learning

Supervised learning is a type of machine learning technique in which we train the machine learning system with sample labeled data and then observe how it predicts the outcome. To predict future events, supervised machine learning algorithms apply what they have learned in the past to new data. The learning method examines a known training dataset to generate an inferred function for forecasting output values [67].

2.2.2. Unsupervised learning

Unsupervised learning is a type of learning in which a computer acquires information without human intervention. The machine is trained on a set of unlabeled, unclassified, or uncategorized data, and the algorithm must respond independently to that data [68].

2.2.3. Reinforcement learning

A learning agent in a reinforcement learning system is rewarded for correct actions and penalized for incorrect ones. This feedback helps the agent automatically learn and perform better [69].

2.3 Deep Learning

Deep learning (DL) has significantly transformed the field of artificial intelligence (AI), achieving excellent performance in a wide range of applications, and demonstrating robust capabilities in managing massive amounts of data and complex computations [70-72]. Deep learning is a subset of machine learning. It employs artificial neural networks to work with datasets and complete tasks. These neural networks' unit cells are neurons, which function similarly to the human brain [73] .

Deep learning employs architectures that include numerous layers of nodes or neurons, with each layer designed to model increasingly complex patterns in data [74]. Initially based on simple models such as perceptron's (see figure 2.2), deep learning has evolved to include sophisticated neural networks capable of performing a wide range of tasks, including image recognition and natural language processing.

One of the most widely used applications in Deep Learning is Audio classification, in which the model learns to classify sounds based on audio features. When given the input, it will predict the label for that audio feature. These can be used in different industrial applications like classifying short utterances of the speakers, music genre classification, etc. [75]

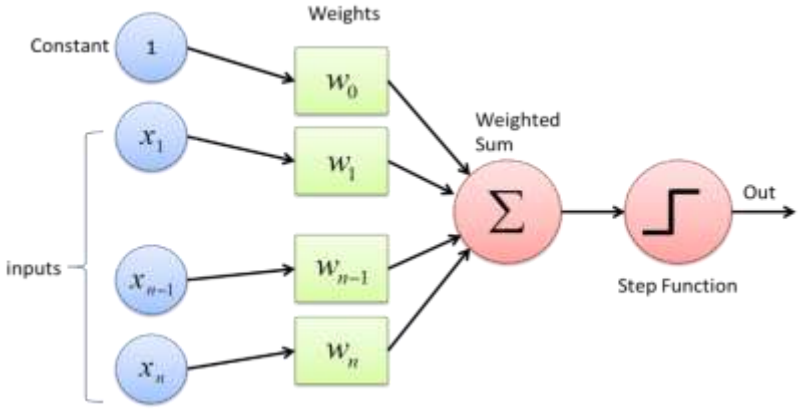

Figure 2.2: Architecture of Perceptron Model [76].

2.3.1 Evolution of Neural Architectures

The deep learning journey began with the Perceptron, a single-layer neural network introduced in the 1950s. While innovative, Perceptrons could only solve linearly separable problems, failing at more complex tasks like the XOR problem. This limitation led to the development of Multi-Layer Perceptrons (MLPs) . It introduced hidden layers and non-linear activation functions, as shown in figure 2.2. MLPs, trained using backpropagation, could model complex, non-linear relationships, marking a significant leap in neural network capabilities. This evolution from perceptrons to MLPs laid the groundwork for advanced architectures like CNNs and RNNs, showcasing the power of layered structures in solving real-world problems [77].

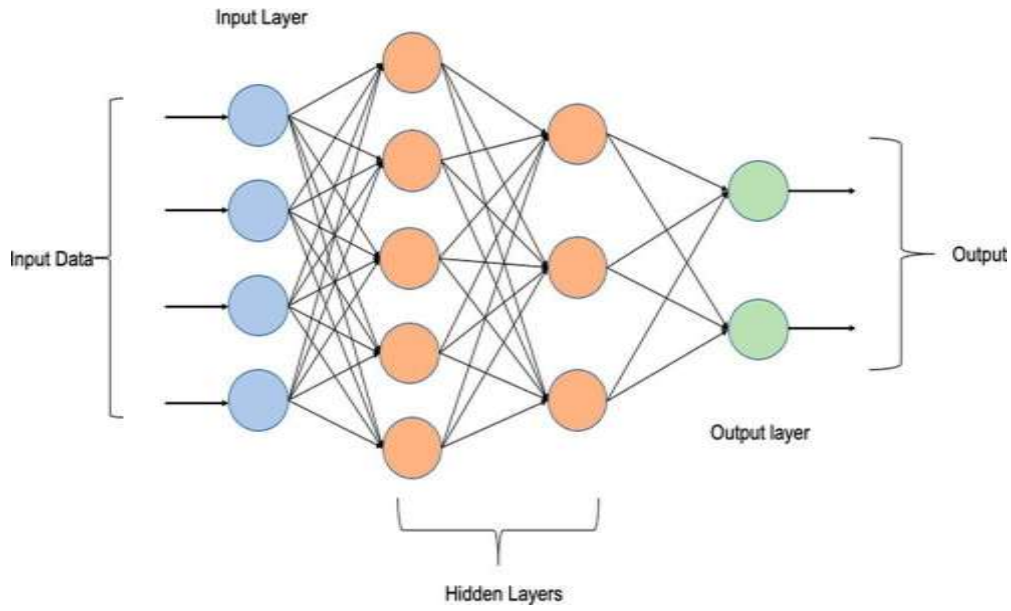

Figure 2.3: Architecture of Multi-Layer Perceptron (MLP) [78].

2.3.2. Works of Deep Learning

Neural networks, also known as artificial neural networks, attempt to mimic the human brain by combining data inputs, weights, and biases that act as silicon neurons. These components work together to accurately identify, classify, and describe objects in data. Deep neural networks are made up of multiple layers of interconnected nodes (see figure 2.2), with each layer refining and optimizing the previous layer's prediction or categorization. Forward propagation refers to the progression of computations through a network. A deep neural network's input and output layers are referred to as "visible layers." The input layer is where the deep learning model receives data for processing, while the output layer is where the final prediction or classification is made. Back propagation is a process that uses algorithms like gradient descent to calculate errors in predictions before adjusting the weights and biases of the function by moving backwards through the layers to train the model. Forward propagation and backpropagation allow a neural network to make predictions while correcting for errors. Over time, the algorithm improves in accuracy [79].

2.3.2 Deep learning models

Deep learning models are complex networks that learn independently without human intervention, composed of either a single or multiple models, which is able to learn from large amounts of data to do specific tasks. Typically, the models have three or more layers of neural networks to help process data. Deep learning systems use a variety of constructions and frameworks to achieve specific tasks and goals [80]. Some common types of deep learning models include:

Convolutional neural networks (CNNs) are a deep learning algorithm that processes structured grid data like images. They have succeeded in image classification, object detection, and face recognition tasks. Convolutional neural networks are a popular neural networking model that uses a multilayer perceptron and one or more convolutional layers. These layers can be pooled or fully connected. [81]

Recurrent neural networks (RNN) [82] are networks where previous outputs are used as inputs while having hidden states. Due to this, they have memory and are able to model sequential data. Consequently, RNN have potential to be able to learn from longitudinal data. They have succeeded in speech recognition, stuttering classification and natural language processing.

CNNs (Convolutional Neural Networks) are generally better suited for classification tasks, especially in domains like image processing, because they are designed to exploit spatial hierarchies and local patterns in data. Unlike RNNs (Recurrent Neural Networks), which process

sequential data by iterating through time steps, CNNs use convolutional layers to scan grids (e.g., pixels in images) and detect features hierarchically. For example, in image classification, a CNN's early layers might recognize edges or textures, middle layers combine these into shapes, and deeper layers identify complex objects like faces or vehicles. This spatial processing aligns with how many classification tasks are structured, where local relationships (e.g., adjacent pixels) matter more than sequential dependencies. [83]

CNN-based models have evolved significantly over the past decade, resulting in various architectural variations designed to improve upon the limitations of earlier CNNs in terms of accuracy, efficiency, or training behavior. Due to this, we choose ResNet34, MobileNetV2, and DenseNet-121 models for automatic stutter detection and classification, because these models are all architectural variants of Convolutional Neural Networks (CNNs).

2.3.3. Convolutional Neural Network model (CNNs)

CNN is widely used in deep learning (84, 85-90). Convolutional neural network (CNN) is one of the most popular and used of DL networks. Because of CNN, DL is very popular nowadays [91]. CNN outperforms previous methods by automatically identifying relevant features without human intervention. CNNs have been widely used in various fields, such as computer vision, speech processing and facial recognition CNNs, like traditional neural networks, are inspired by neurons in human and animal brains. The visual cortex in a cat's brain is made up of a complex sequence of cells, which is simulated by CNNs.

CNN is a biologically inspired model and firstly proposed by LeCun et al. (1998) [92]. Shown in Figure 2.4 is a general structure of a CNN. In this structure, the input layer receives normalized images with identical size. A set of units in a small neighborhood (local receptive field) in the input layer will be processed by a convolution kernel to form the unit in a feature map of the subsequent convolutional layer. [93]

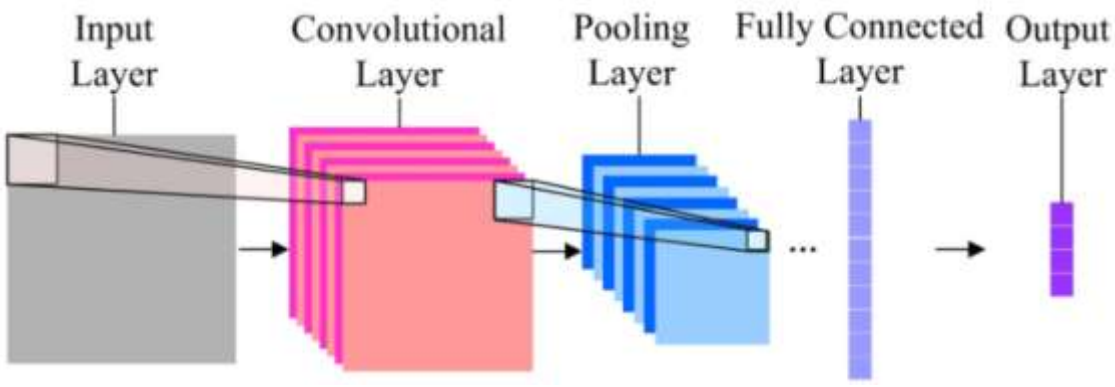

Figure 2.4: The General structure of a CNN [94]

2.3.3.1 Convolutional Neural Network Layer and Architecture

A convolution neural network has multiple hidden layers that help in extracting information from an image. The four important layers in CNN are [96]:

a. Convolutional Layer

The most important component in CNN architecture is the convolutional layer. A convolutional layer. It consists of convolutional filters, often known as kernels. The input image, represented in 2D dimensions, is convolved with these.

- Kernel definition: A grid of discrete numbers or values that describe the kernel. Each value is referred to as the kernel weight. Random numbers are assigned to serve as kernel weights at the start of CNN training. In addition, many approaches are utilized to initialize weights. Next, these Weights are modified at each training epoch, allowing the kernel to extract significant features. [95].
- Convolutional operation: The convolutional layer employs a kernel filter to calculate the convolution of input images, extracting the fundamental features. The filter kernel has the same dimension size but a smaller constant parameter value than the input image. For instance, the acceptable length of a kernel filter for a 2D spectrogram with a size of However, the filter size has to be smaller than the size of the input image. The filter mask slides across the input image step by step and estimates the product point between the kernel filter weight and the pixel value of the input image. This process results in a 2D activation map. CNN will then learn the visual feature of the image. Figure 2.5 shows a simple illustration of the computational process in CNN that results in the activation map [96].

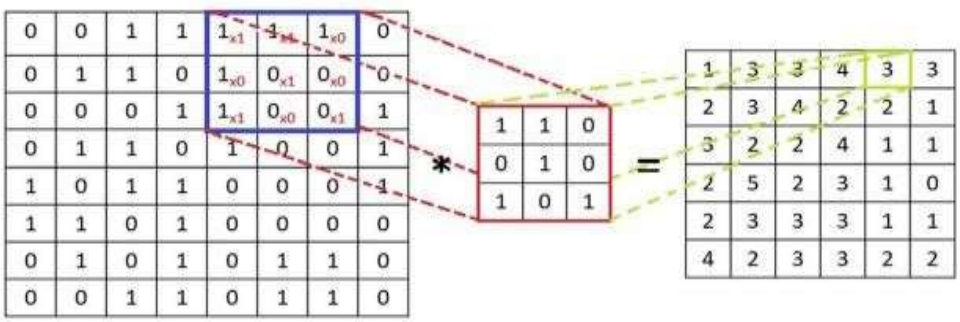

Figure 2.5: Convolutional operation [96]

b. ReLU layer (Activation function)

Activation functions are crucial components of deep learning models as they introduce non-linearity into the network, enabling it to learn complex patterns and make sophisticated predictions. Without activation functions, a neural network would simply perform linear transformations, limiting its ability to model complex relationships in the data [97]. Among activation functions, the Rectified Linear Unit (ReLU), is widely used. It stands for the rectified linear unit. Once the feature maps are extracted, the next step is to move them to a ReLU layer. ReLU layer sets negative values to zero and keeping positive values unchanged. It introduces non-linearity to the network, and the generated output is a rectified feature map [98]. It promotes sparsity in activations, allowing the network to focus on relevant features. Below is the graph of a ReLU function.

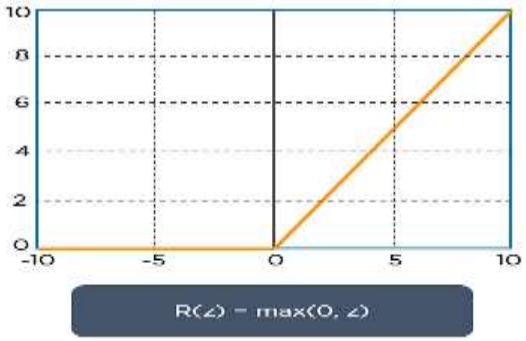

Figure 2.6: ReLU layer [98]

c. Pooling Layer

The pooling layer will combine two consecutive convolutional layers. It reduces the number of parameters and computation loads by making down-sampling representations. The function in the pooling layer can result in a maximized or averaged value. A maximizing combination is often used for an optimal function [99]. The pooling layer is also helpful in reducing overfitting or computation weights. Figure. 2.7 represents a simple operation in dimension reduction of an activation map using the max-pooling function [99].

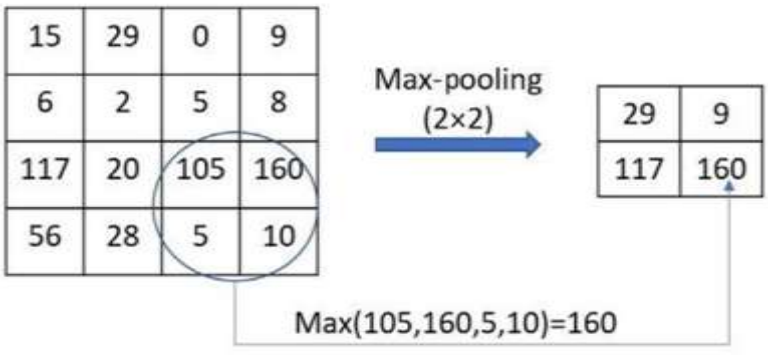

Figure 2.7: Pooling operation [99]

d. Fully Connected Layer

The third layer is the fully connected layer, commonly called the convolutional output layer. The fully connected layer is similar to a feedforward neural network, as shown in Figure. 2.8. Inside this layer, each neuron is connected to all neurons of the previous layer, the so-called Fully Connected (FC) approach. It is utilized as the CNN classifier. The layer is commonly found in the bottom layer of the network. It receives input from the final pooling or the convolutional output layer, flattened before being sent to the subsequent layer. Even distribution of the output means unrolling all the values of the result obtained after the last pooling or convolutional layer into a vector (3D matrix). This method is a simple technique for studying high-level non-linear combinations of a feature represented by the output convolutional layer [99].

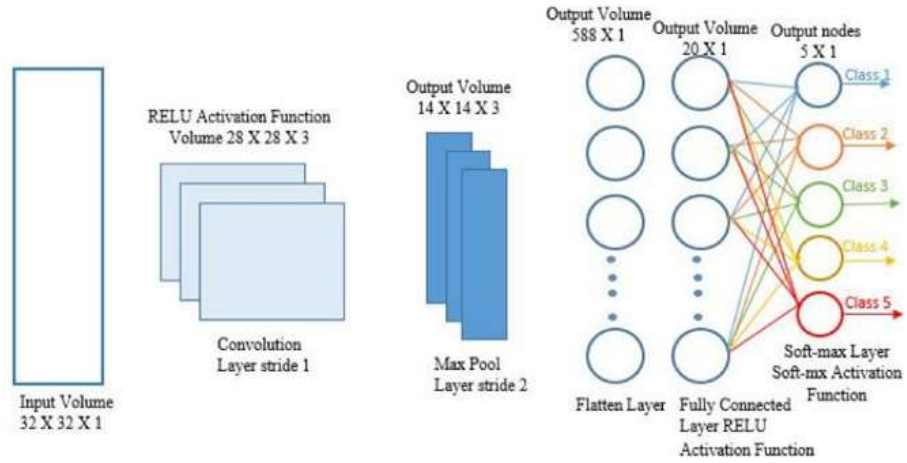

Figure.2.8: Fully Connected Layer [99]

e. Output Layer

In the output layer, the final result from the fully connected layers is processed through a logistic function, such as sigmoid or SoftMax. These functions convert the raw scores into probability distributions, enabling the model to predict the most likely class label[99] .

*2.3.3.*2 Loss Function.

Loss functions, also known as cost functions or objective functions, are fundamental components in training deep learning models as they quantify the difference between the model's predictions and the actual target values. Loss functions guide the optimization process by indicating how the model parameters should be adjusted to minimize errors and improve predictive accuracy. The choice of loss function depends on the specific task and data characteristics, as it directly influences how the model learns during training [93]. One of the most common loss functions is

the Mean Squared Error (MSE), primarily used in regression tasks. MSE calculates the average squared difference between the actual target values y_i and the model's predictions $\hat{y}_i$:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad \text{.....(2.1)}$$

Where n is the number of samples, y_i represents the true value, and $\hat{y}_i$ denotes the predicted value.

2.3.3.3 Optimizer selection

Deep learning relies heavily on optimization algorithms. We use them whenever we train a neural network and adjust the model parameters to reduce loss. The optimizer is like a coach that adjusts the network's weights to help it do better. It tweaks the model's parameters to minimize the loss function, ultimately leading to more accurate predictions over time. Adaptive Moment Estimation (Adam) is optimization technique or learning algorithm that is widely used. Adam represents the latest trends in deep learning optimization. This is represented by the Hessian matrix, which employs a second-order derivative. Adam is a learning strategy that has been designed specifically for training deep neural networks. More memory efficient and less computational power are two advantages of Adam [100].

2.4. Automatic Stutter Detection and Classification by deep learning

With the rise of Deep Learning (DL) techniques, researchers have increasingly turned to DL models to classify and analyze stuttered speech. The current research in this field mainly uses convolutional neural network to transcribe audio signals into spectrograms and then applies language models to identify and detect stutters [101,102,103]. While this method has been successful and produced positive outcomes, relying on CNNs can introduce errors and additional computational steps that may not be necessary. The identification and detection of stuttering incidences are usually a difficult and bothersome problem due to several variable factors including language, gender, age, accent, speech rate.

2.4.1. System flow chart

The system takes an audio input that is preprocessed with MP3-to-WAV conversion, noise reduction, and spectrogram generation. This data is fed to a CNN model for feature extraction and then a classifier gives one of four outputs: no stutter, block, repetition, or prolongation.

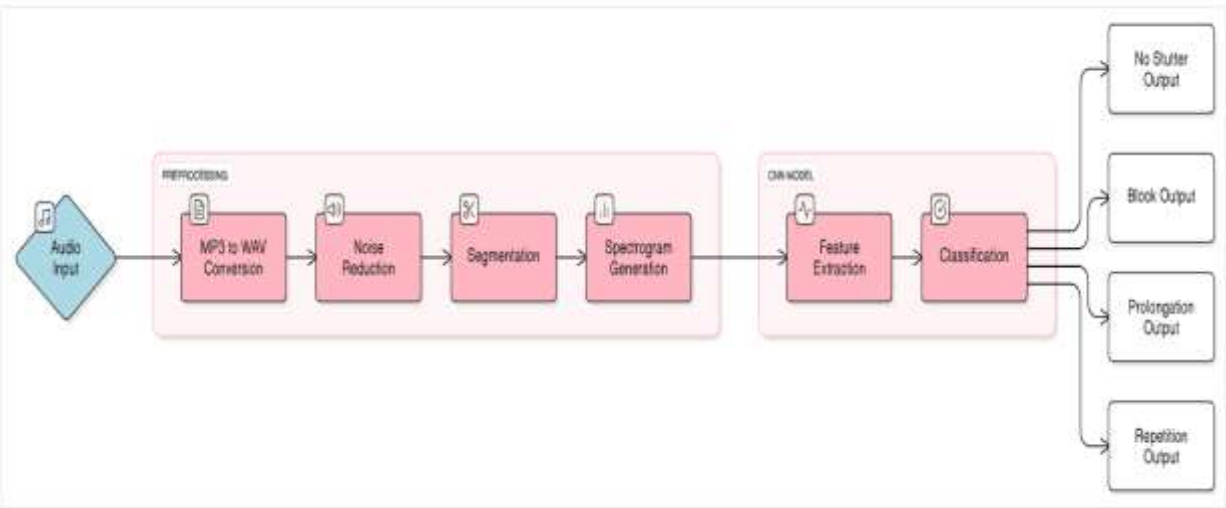

Figure 2.9: Stutter Classification Flow Diagram

2.4.1.1. Audio Input

The input to the system is a single recorded MP3 audio file containing a sample of stuttered speech. This audio is uploaded by the user and becomes the basis for further analysis and classification.

2.4.1.2. Preprocessing:

a. MP3 to WAV conversion

MP3, or MPEG audio layer 3, is notable for its compression technology, which balances space conservation with sound quality. Because of its tiny size and high quality, this format, popularized by devices such as the Apple iPod. A WAV file, which stands for Waveform Audio File, stores audio waveform data in specified regions of the file, reflecting volume and sound strength. It can be compressed, which is uncommon and mainly utilized on Windows computers [104].

In audio classification we convert MP3 to WAV files because the latter type of storing sound keeps all quality, as it is an uncompressed format while MP3 is lossy compressed. It is easier to build machine learning models on consistent high quality data. Most audio processing tools operate with greater precision on WAV files and so do many feature extraction algorithms. Further, converting to WAV format greatly enhances the classification system's efficiency. You can easily use Media Encoder to convert MP3s to WAV file format for all your sound engineering and audio editing needs. [105,106]

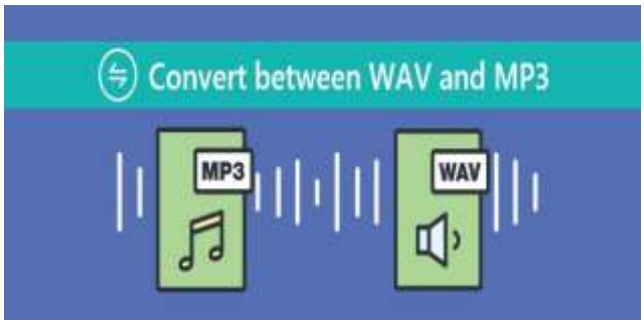

Figure 2.10: conversion between MP3 and WAV [107]

b. Noise reduction

One of the most important tasks of modern science is the development of software tools for human communication with devices (for example, a computer) in natural language, where speech input and output of information is carried out in the most user-friendly way. To create such tools, it is required to solve speech recognition problems. On the basis of many experimental studies, it can be concluded that the quality of speech classification depends on the results of preliminary signal processing. Improving the quality of speech classification requires new efficient and high-speed signal preprocessing methods and algorithms [108]

- **Noise reduce Algorithm**

Noisereduce is a form of spectral gating, or noise gating algorithm. Noise gates attenuate or suppress signals deemed noise, allowing the desired signal to pass through unaffected. The spectral gate performs this gating in the spectro-temporal domain. Noisereduce accepts two inputs: (1) X, the time-domain recording to be denoised and (2, optionally) Xnoise, a time-domain recording containing only noise, used to calculate noise statistics. Noisereduce operates through the following steps:

1) **Estimate noise :**

- Compute a Short-Time Fourier Transform (STFT; Sn) on each channel of the noise recording (Xnoise).
- For each frequency channel, compute spectral statistics (μn, σn) over the noise STFT (Sn).
- Compute a noise threshold based upon the statistics of the noise and the desired sensitivity.

2) Mask noise:

- Compute a STFT (SX) over each channel of the recording (X).
- Compute a mask (M) over the signal STFT (SX), based on the thresholds for each frequency channel.
- (optional) smooth the mask (Msmooth) with a filter over frequency and time
- Apply the mask (Msmooth) to the STFT of the signal (SX) to produce the masked STFT (Sm).
- Invert the masked STFT (Sm) back into the time-domain (Xdenoised).

Figure 2.5 represents Basic outline of Noisereduce algorithm. (A) A block diagram of the steps of Noisereduce. The stationary version of the time-frequency mask is depicted. (B) An example waveform (U.S. President George W Bush stating" I know that human beings and fish can coexist peacefully") passing through the Noisereduce pipeline [108] .

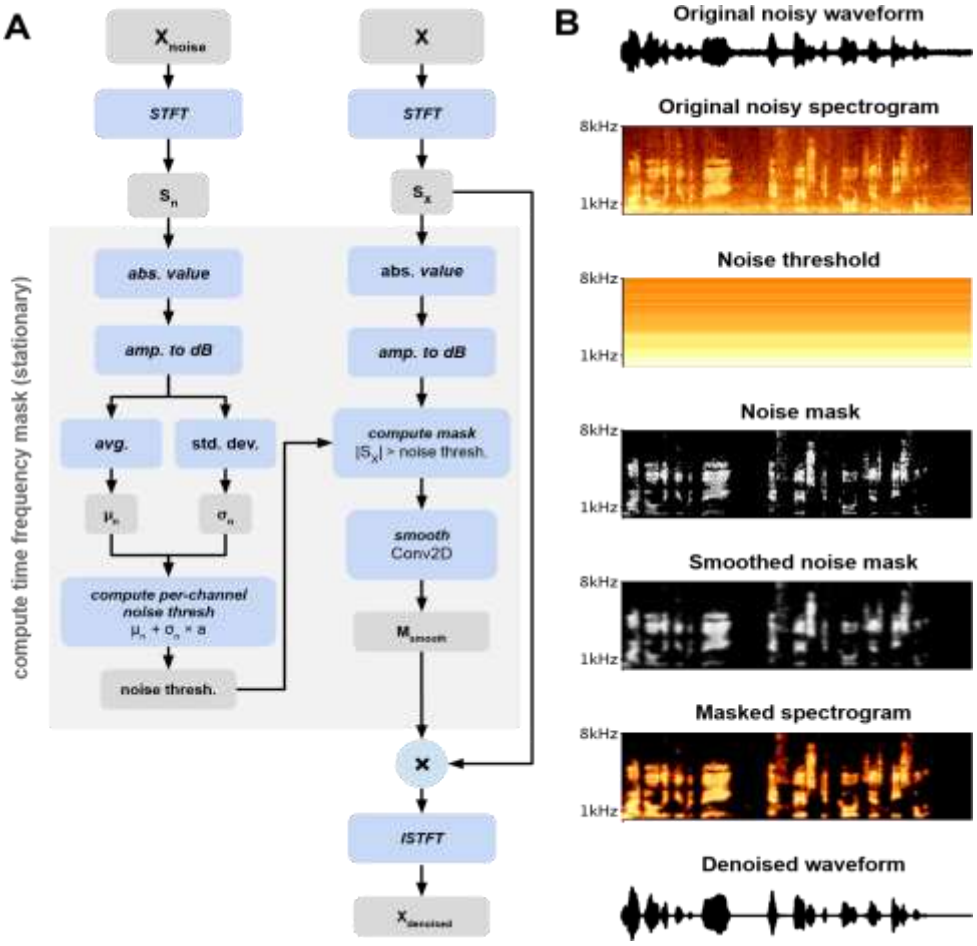

Figure 2.11: Basic outline of Noisereduce algorithm [108].

c. Segmentation :

As speech technologies applications are progressing, audio segmentation techniques significance is also increasing [109]. The huge surge in the number of research articles on deep learning-based audio segmentation indicates the paramount importance of these techniques. The fundamental goal of audio segmentation is to divide an audio signal into small segments so that the entities may be easily identified. Segmentation plays an important role in audio signal processing. The most important aspect is to secure a large amount of high-quality data when training a deep learning network [110].

The input speech is divided into 3-second segments, and each segment is examined for stuttering evidence. This enables more precise and reproducible classification of stutter events across the audio.

d. Spectrogram Generation :

A spectrogram is a visual 2D representation of audio signals in the frequency domain that displays how the frequencies within a sound evolve over time by breaking down an audio signal into small segments and computing the intensity of different frequency components within each segment. The spectrogram, or time-frequency representation of an audio signal, helps us to understand valuable insights about the audio content, like distinguishing between various sounds, patterns, or characteristics. The efficient creation of spectrograms is a key step in audio classification using spectrograms. [111]

To visualize the audio files in the form of the frequency-time domain, the audio files are converted into spectrograms, as shown in Figure 2.5. To achieve this task, fast Fourier transform (FFT) is used to transform the time-domain signal into frequency-domain. We then extract Mel-Spectrogram images from the input audio files using Python library, Librosa, where the Log-Mel spectrogram is considered the best feature representation in this method. We reform the spectrograms into the three-channel input according to the standard CNN models. After that, the three-channel spectrograms are computed using different window sizes and hop lengths. These are {100 ms, 50 ms}, {50 ms, 25 ms}, and {25 ms, 10 ms}, respectively. The different window sizes with hop lengths obtain the details of frequency and time of network levels for every channel. The spectrograms are resized into a unique shape through different windows. The entire Mel spectrum is fixed to 128 bands equal space frequencies [112]. The spectrogram size is empirically set to (128, 128) to fit the SEP-28k dataset used in this work.

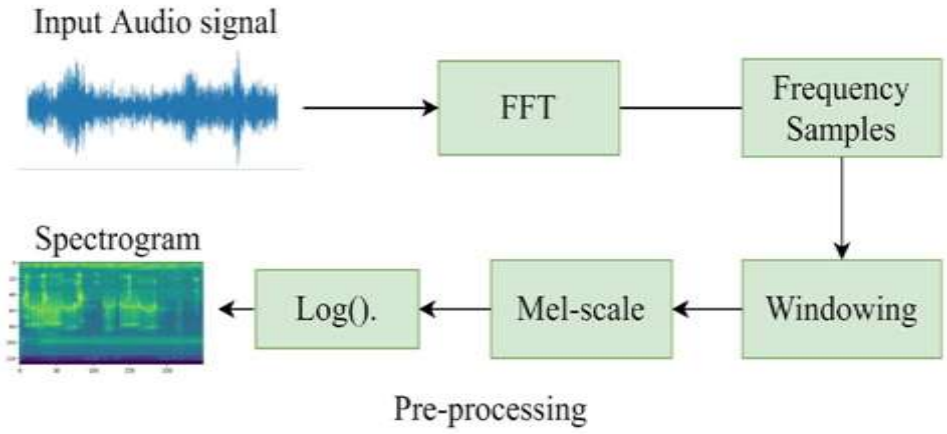

Figure 2.12: Conversion process of an input audio signal into spectrogram [111].

2.4.1.3 CNN-Based Model Architectures

In this step, the sound is converted to a spectrogram, which is an image that displays the evolution of the sound over time. This spectrogram is fed through a CNN-based model, which allows the system to learn the patterns of the speech and determine whether there is a stutter

Feature extraction is a critical step in deep learning because it transforms raw data into meaningful representations that make it easier for models to learn patterns and relationships. Raw data, such as images, text, or sensor readings, often contains noise or irrelevant details that can confuse a model. Feature extraction simplifies this data by identifying key characteristics—like edges in an image or word frequencies in a document—that are most relevant to the task. This process reduces computational complexity and helps models focus on the most informative aspects of the data, improving both training efficiency and model performance. [113]

In this work, we use three CNN-Based models to detect and classify of stuttered speech: MobileNet V2, DenseNet-121 and ResNet34. Each of these models has unique characteristics and design principles, which affect its performance in the classification task.

a. MobileNet V2

MobileNet V2 is a depth-wise separable convolution network that reduces complexity, cost, and size for mobile devices with low computational power (Howard et al., 2017). MobileNet V2 introduces an inverted residual structure, which helps in Detecting objects and segmenting them semantically. It can be performed using MobileNet V2. (Sandler et al. 2018). The architecture is very intuitively designed. The system consists of two blocks: the residual block with a stride of 1 and the downsizing block with a stride of 2. Each block has three layers, with the first being a 1 ×

1 convolution with RelU6. Figure 2.12 shows the process of depth-wise convolution, followed by 1 × 1 convolution without non-linearity. The input image size must be 224 x 224 [114].

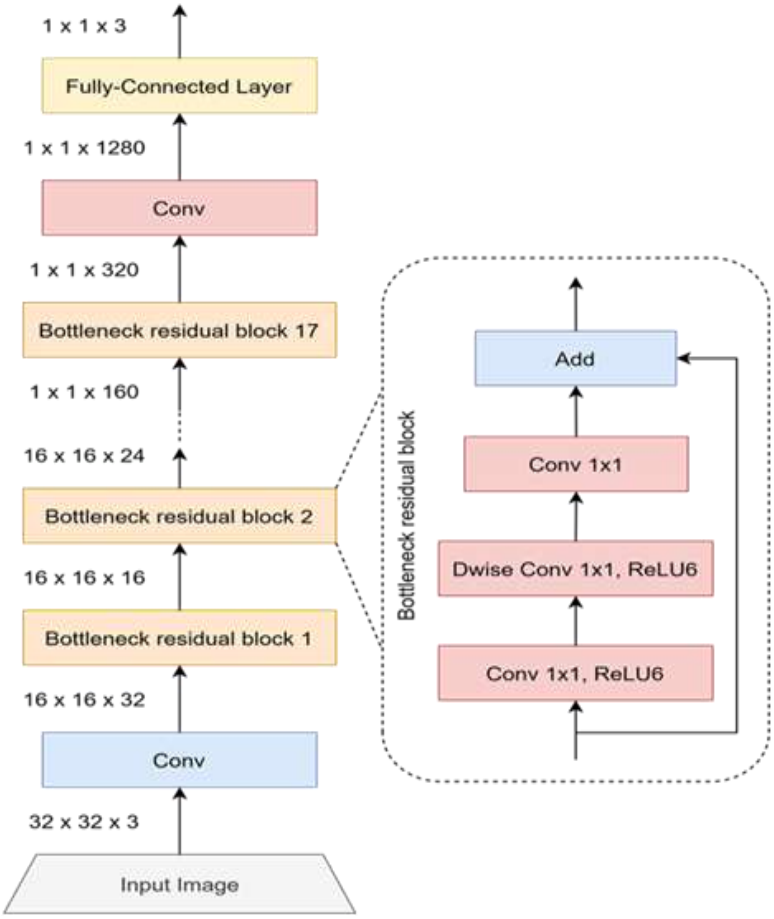

Figure 2.13: MobileNet V2 Architecture[114]

b. ResNet 34

Deeper neural networks are more difficult to train, and the deeper the network model is, the more information can be obtained, and the richer the characteristics are. However, the main problem its faces is the vanishing gradients problem. With the deepening of the deep learning network, the model optimization effect becomes worse, and the accuracy of test data and training data decreases accordingly. When deeper networks are able to start converging, a degradation problem has been exposed: with the network depth increasing, accuracy is saturated and then degrades rapidly. He et al [115] present a Deep Residual Learning framework (ResNet-34) to facilitate the training of networks that are substantially deeper than those used previously. ResNet (residual network) is a type of neural network that alleviates this problem of training deep learning networks by using skip-connections to “skip” a number of convolutional layers in every basic block in the network (figure 2.13), a thing that provides alternative paths for original and derived data, rendering training faster and more possible. ResNet-34 is a 34-layer convolutional neural

network (CNN) architecture that belongs to the ResNet (Residual Network). It was designed to address the vanishing gradient problem, enabling the training of deeper networks by using residual connections or skip connections that allow gradients to flow more effectively during back propagation. It has a good effect in image classification and target recognition[116] .

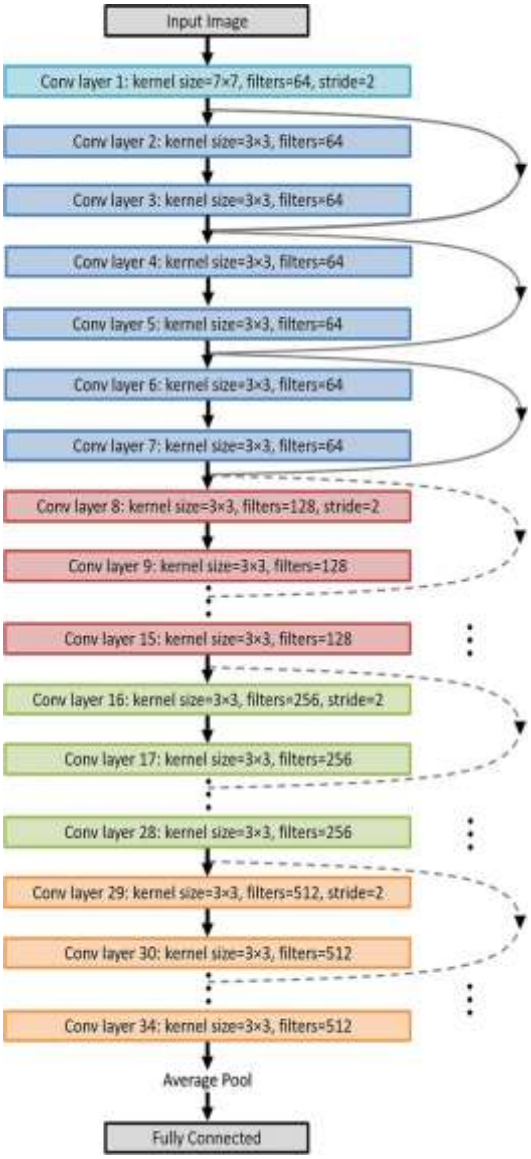

Figure 2.14: ResNet34 architecture [115]

c. DenseNet-121

Similar to ResNets, Densely Connected Convolutional Networks, or DenseNets [116], propose a connectivity pattern to address the vanishing gradient problem that occurs when training deeper architectures while ensuring maximum information and gradient flow throughout the network. In the DenseNet architecture (Figure 2.14), each layer is connected to every other layer in a feed forward fashion, so that the input for each layer is the concatenated feature maps of all previous

layers, and the output is used as input for all subsequent layers. This has the advantage of reducing the number of parameters used because it encourages feature reuse, resulting in fewer redundant feature maps. DenseNets are made up of alternating dense and transition blocks. Within a dense block, the feature maps' dimensions remain constant to allow for concatenation, but their volume varies. Transition blocks downsample between dense blocks using 1 × 1 convolution and 2 × 2 pooling layers. The network architecture has one hyperparameter: growth rate, which controls the number of feature maps added by each layer, thereby regulating how much information each layer contributes to the global state. DenseNets have achieved state-of-the-art performance on object recognition benchmarks, while using fewer parameters and requiring less computation than other cutting-edge architectures.

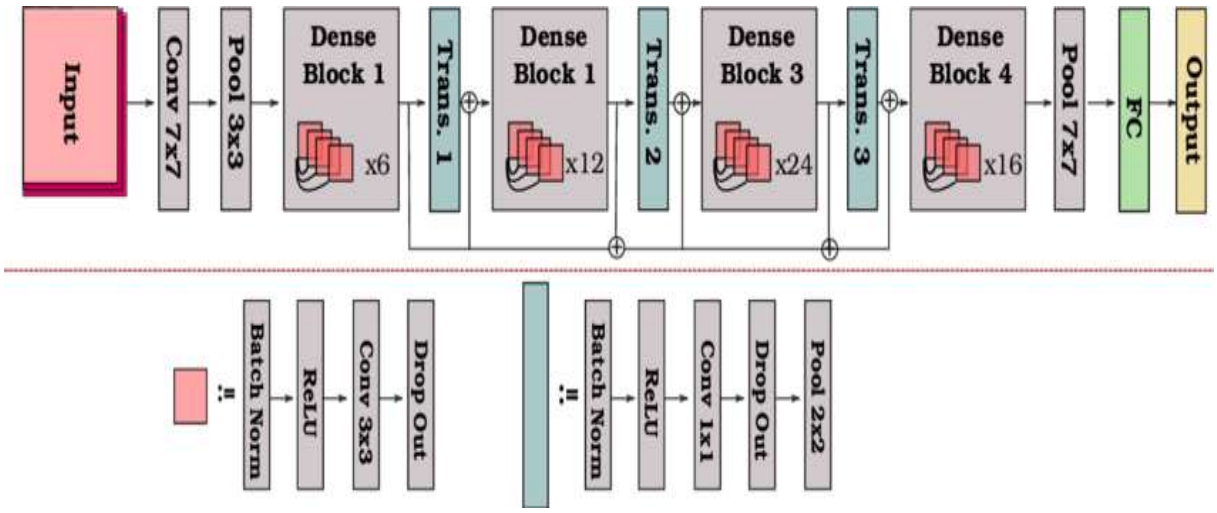

Figure 2.15: DenseNet-121 Architecture [116].

2.4.1.4 Output

The stutter speech classification model yields four distinct outputs: No Stutter, Block, Prolongation, and Repetition. These categories stem from the classification made by the deep learning model based on fed speech signal's temporal and spectral features, which are often illustrated in spectrograms. The No Stutter outcome denotes the presence of coherent speech without any disengagement markers, whereas Block indicates complete halting of speech, which is the temporary inability of the speaker to produce sound. Prolongation output refers to disrhythmic prolongations of single sounds or syllables which often show stuttering tendencies. Lastly, the Repetition category tracks the occurrence of recurring sounds, syllables, or words, which represent a characteristic of stuttered speech. This approach to multi-class classification facilitates a detailed examination of the various levels of fluency in speech and allows for particular patterns of stuttering to be detected in a given audio signal.

2.5 Conclusion

In conclusion, this study demonstrates the effectiveness of deep learning, particularly CNNs, in stuttering classification. By adopting spectrogram-based feature extraction and segment-level classification, the system effectively classifies different types of stuttering. The approach demonstrates the potential of deep learning in enhancing automatic speech disorder detection and facilitating clinical diagnoses.

Chapter 3

Implementation and Experimental Results

3.1. Introduction

This chapter presents and analyses the results obtained after applying our proposed models for stuttering detection and classification. It also aims to evaluate the performance of the adopted tools and algorithms deep learning-based using a set of well-known technical standards in the field. Furthermore, the obtained results will be compared with previous work to highlight areas for improvement. The chapter concludes with a general discussion of the future scope.

3.2. Dataset Used

There are few publicly available datasets on this topic, as stuttering affects approximately 1% of the global population. This limited availability of data makes it challenging for researchers to develop effective treatments and therapeutic approaches, as well as to gain a comprehensive understanding of the nature and causes of stuttering. Moreover, relying on the same datasets -even when using slightly different algorithms- often leads to similar results, despite variations in methodology,

3.2.1. Choose target varieties

In this work, we used SEP-28k fluency bank stuttering dataset [118], which consists of 28,177 speech samples from 385 podcasts. Four main categories of audio data available in this database were selected: *repetition*, *prolongation*, *block*, and *no stuttered* speech. This classification was adopted based on the priority of analysing the basic behaviours of stuttering, while providing a reference category that enables clear distinction between stuttered and normal speech.

The first three types of stuttering are the most common, representing the core behaviours used in clinical assessment of a speaker's condition. Their auditory and visual characteristics enable them to be accurately captured in spectrograms, making them suitable for computer vision-based models. The normal speech category was included to enable the model to distinguish between disordered and normal speech, a key component of any diagnostic system based on binary or multiple classification. The inclusion of this category also enhances the model's accuracy and effectiveness in practical use, especially when later employed in a real-world setting such as a self-diagnostic web application.

On the other hand, some other categories in the base, such as interjections and unintelligible syllables, were excluded due to their ambiguity or infrequency, which may negatively affect sample balance and training accuracy.

3.2.2. Data pre-processing (labelling)

We reclassified the SEP-28k Fluency Bank data into four main categories: "Not Stutter," "Block," "Prolongation," and "Repetition." This categorization aims to facilitate the process of distinguishing between normal speech and situations in which stuttering occurs. The first category represents audio clips that do not contain any form of stuttering and is considered a reference to normal speech. The other three categories reflect the most common stuttering manifestations. "Block" refers to a temporary break in the flow of speech, "Prolongation" refers to an abnormal prolongation of a sound, and "Repetition" refers to the repetition of a part of a word or sound.

To facilitate data processing and improve model performance, we divided all audio clips into short segments, each 3 seconds long. This segmentation allows for the extraction of accurate acoustic features and improves training quality, especially in deep learning-based models, where clip length is an important factor in the effectiveness of prediction and classification. After the segmentation and classification process, we now have (1377) voice, which provides a rich and organized database that can be used to train AI models more efficiently.

3.2.3. Division of data(Validation /Train /Test)

Due to the lack of precise information about the identity of the speakers in the SEP-28K database, a podcast identity-based partitioning was adopted, where each podcast was considered an independent source (assumed to be from a different speaker), to avoid overlap between datasets and ensure independent evaluation.

To ensure effective training of the model and accurate evaluation of its performance, the audio data was divided into three main sets:

- **Training set:** 70% of the data was allocated to the training set, which was used to teach the model and extract patterns that characterize each stuttering class.
- **Validation set:** The validation set (10%) was used during the training process to monitor model performance and adjust parameters (such as the learning rate and number of cycles) using early stopping techniques to mitigate overfitting.
- **Test set:** Finally, the test set (20%) was allocated to final evaluation of the model on previously unseen data, allowing for an objective measure of the model's performance in real-world conditions.

3.3. Preparation of data

Before starting to train the model, a series of pre-processing steps were implemented to prepare the raw audio data in a format suitable for the machine learning phase. This process included four main steps:

- **First step,** noise reduction algorithms were applied to improve the quality of the audio signal and ensure the clarity of stuttering segments, thereby reducing noise that could negatively impact model performance.
- **Second step,** the processed audio signals were converted into spectrograms, two-dimensional images that show the frequency distribution over time. This visual representation is effective in capturing the acoustic features of stuttering and is used as a starting point for feature extraction in computer vision models.
- **Third step,** spectrogram enhancement techniques such as normalization and dimensionality adjustment were applied to standardize the data and facilitate learning, especially when using convolutional neural networks (CNNs), which are sensitive to the size and range of values in the data.
- **Finally step,** the audio clips were trimmed to a fixed length of three (03) seconds, to standardize the input length and ensure consistent temporal structure across the different samples. This length was carefully chosen to be sufficient to contain the targeted stuttering behaviours without causing loss of information or unnecessary volume.

3.4. The general architecture of the proposed system

Figure 3.1 illustrates the general architecture of the proposed automatic stuttering detection and classification system. We implement three deep learning models based on CNNs: MobileNetV2, ResNet34, and DenseNet121, in order to compare their performance and select the better model for our system.

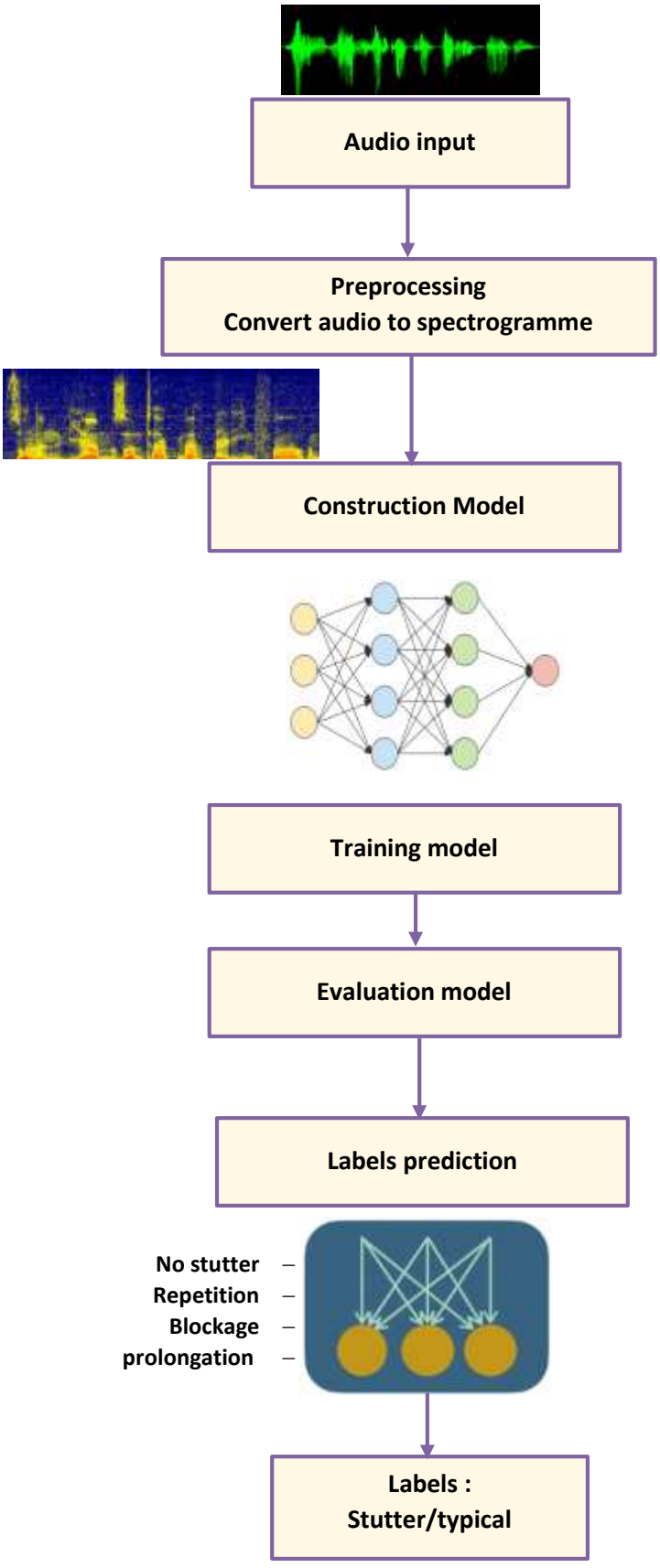

Figure 3.1: Aarchitecture of the automatic stuttering detection and classification system.

3.5. Hardware and software tools

3.5.1. Hardware

In order to carry out this project, the following materials were made available to us:

- CPU: Intel Core i5-10300H @ 2.50GHz
- RAM: 16 GB
- GPU: NVIDIA GeForce RTX 2060 (6 GB VRAM)
- ROM: 477 GB SSD (Kingston)
- System type: 64-bit operating system, x64-based processor
- Windows edition: Windows 10

The training of the model was performed using the dedicated NVIDIA GPU to accelerate computation and reduce training time.

3.5.2. Software

The model was developed using Python (version 3.10), high-level programming language, due to its powerful and flexible tools for data processing and developing AI models. The PyTorch framework (version 2.1.2) was adopted as the primary tool for building and training neural networks, due to its ease of use and broad support for deep learning research.

In addition to PyTorch, a set of auxiliary libraries were used to contribute to data processing and analysis, including (table 3.1):

- NumPy (version 1.26): For matrix and numerical operations.
- Matplotlib: For displaying graphs and analysing training results.
- Skit-learn: For calculating performance indicators (such as precision, specificity, and recall) and performing some classification tasks.
- Albumentations: For applying data augmentation techniques, particularly on spectral images.
- Pillow (version 3.1): For image processing and formatting spectral representations into model-appropriate formats.

This integrated software environment represents the technical foundation upon which the project relied to implement various processing stages, from data preparation to model performance evaluation.

Table 3.1: Overview of Programming libraries and Reasons for their use

Library	Purpose	Common Use Cases	Key Features	Reference
NumPy	Numerical computing with arrays	Matrix operations, scientific	Multidimensional arrays, broadcasting, integration with other libraries	https://numpy.org/doc/stable/
Matplotlib	Data visualization	Plotting graphs and charts	Line plots, histograms, scatter plots, customizable graphics	https://matplotlib.org/stable/users/index.html
Scikit-learn	Machine learning library	Classification, regression, clustering	Simple and efficient tools for data mining and analysis	https://scikit-learn.org/stable/index.html
Albumenttions	Image augmentation for machine learning	Preprocessing in computer vision tasks	Fast and flexible augmentations (flip, rotate, blur, etc.), integration with PyTorch/TensorFlow	https://albumentations.ai/docs/
Pillow	Image processing	Opening, editing, and saving images	Supports many formats (JPEG, PNG, etc.), cropping, resizing, filtering	https://pillow.readthedocs.io/en/stable/

3.5.2.1. Software and hardware layers in implementing AI models using graphics processing units (GPUs):

The figure below shows the layers of software and hardware required to run deep learning models on graphics processors (GPUs). They start with the highest layer (software libraries) and end with the actual hardware (GPU and computer).

- **Python Frameworks:** (TensorFlow, PyTorch, ONNX Runtime) These are popular deep learning libraries written in Python. Developers use them to build, train, and run models.
- **Acceleration libraries:** (cuDNN and CUDA Toolkit): cuDNN A library provided by NVIDIA containing optimized functions for performing common operations in neural networks. CUDA Toolkit: A toolkit that allows developers to execute code directly on the GPU.
- **Operating System:** The operating system is responsible for coordinating the operation of software and linking it with the hardware.
- **Drivers:** These are the drivers for the graphics card.

• **Hardware:** GPU Card that performs computations, speeding up model performance. Computer is the host environment that includes the CPU, GPU, memory, and disk, which acts as the operating center for all layers.

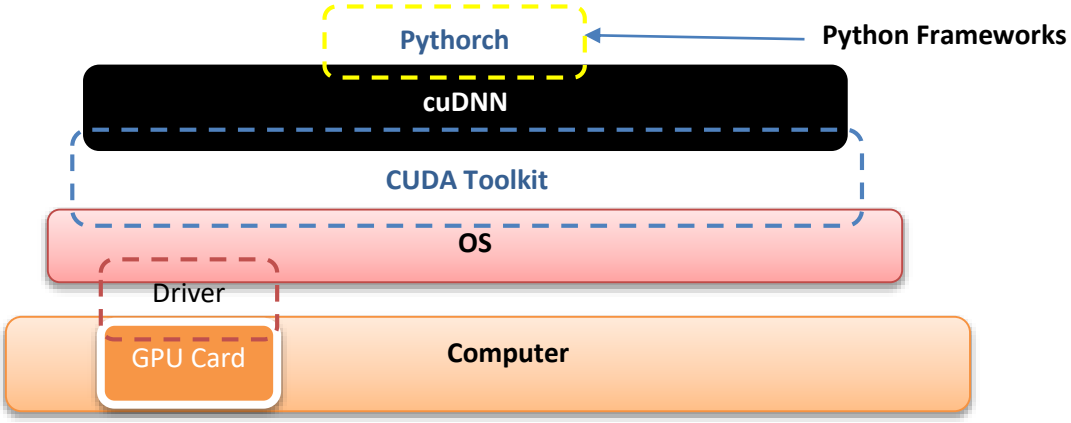

Figure 3.2: Software and hardware layers in implementing deep learning models using graphics cards (GPUs)

3.5.3. CUDA Toolkit and cuDNN

3.5.3.1 CUDA Toolkit

Compute Unified Device Architecture, a.k.a. CUDA is a parallel computing platform developed by NVIDIA with an initial release date of 23 June 2007. It allows developers to use GPUs' power for general-purpose tasks, not just graphics rendering. NVIDIA CUDA made it possible to use GPUs for various applications, including scientific research, engineering simulations, and, eventually, AI and deep learning. By around 2015, the development of CUDA's focus shifted towards neural networks and AI. We used CUDA to speed up the training process of the EDSR model using a graphics processing unit (GPU). CUDA allowed us to perform calculations faster on special NVIDIA hardware, which contributed to reducing the training time and improving the overall performance of the model. We used PyTorch because it is a library that supports CUDA to achieve the greatest benefit and speed up the training time of our model, as training large models on the CPU consumes a large amount of time, which may waste days and days of waiting [119].

3.5.3.2. cuDNN

cuDNN (CUDA Deep Neural Network library) is a specialized, GPU-accelerated library that provides essential building blocks for deep neural networks. It's designed to deliver high-performance components for convolutional neural networks and other complex deep learning algorithms to speed up the execution of repetitive mathematical operations like Convolution,

Pooling Normalization. By implementing cuDNN, frameworks such as TensorFlow and PyTorch can take advantage of optimized GPU performance. [120].

NVIDIA's CUDA installation lays the groundwork for GPU computing, whereas cuDNN provides targeted resources for deep learning. This combination enables remarkable GPU acceleration for tasks that a traditional CPU could otherwise require days or weeks to complete.

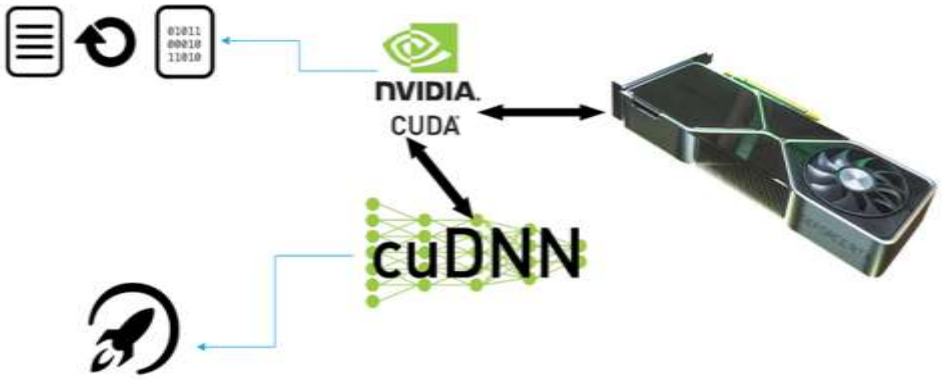

Figure 3.3: Accelerating Neural Networks Using CUDA and cuDNN on NVIDIA GPUs

3.5.3.3 System Requirements for installation CUDA and cuDNN:

Before you start the NVIDIA CUDA installation or cuDNN installation steps, your system fulfils the following requirements:

1) Hardware Requirements:

- NVIDIA graphics card (GPU): The card must be CUDA supported. Most recent NVIDIA GPUs support CUDA. In our case: RTX 2060 6GB VRAM.
- At least 8GB of RAM (16GB or more recommended). In our case: 16GB RAM.
- Storage: 477 GB SSD available, with at least 10 GB free for installation and configuration.
- NVIDIA graphics card (GPU) The card must be CUDA supported. While most recent NVIDIA GPUs support CUDA.
- Setting up CUDA, cuDNN, and the necessary drivers may require several gigabytes of storage. You must have a minimum of 5–10 GB of free disk space available.

2) Software Requirements:

- Windows 10/11 The system must be (64-bit) or Linux (Ubuntu ،CentOS ،RHEL)
- NVIDIA Driver: It must match the CUDA version you want to install.
- CUDA Toolkit Release: Choose the version that suits your graphics card and system.
- cuDNN: must be compatible with the installed CUDA version.
- Visual Studio (Windows only): Visual Studio 2019 and 2022 are not supported. Install "Desktop development with C++" during installation.

3.5.3.4: Installing CUDA and cuDNN on Windows:

This section provides a detailed guide on installing CUDA and cuDNN on a Windows system.

Step1: Verify GPU Compatibility:

To determine your GPU model and check if it is compatible with CUDA, right-click on the Start Menu, choose Device Manager, and then expand the Display Adapters section to locate your NVIDIA GPU. After finding it, head over to the NVIDIA CUDA-Enabled GPU List to verify whether the specific GPU model supports CUDA for GPU acceleration**.**

Step 2: **Install NVIDIA GPU Drivers:**

If u don't have a NVIDIA drivers go to download and set up the latest NVIDIA drivers, go to the official NVIDIA website and download it from there. ' https://www.nvidia.com/en-us/drivers/' and choose the correct driver for your GPU and Windows version. In my case: RTX 2060 16G VRAM

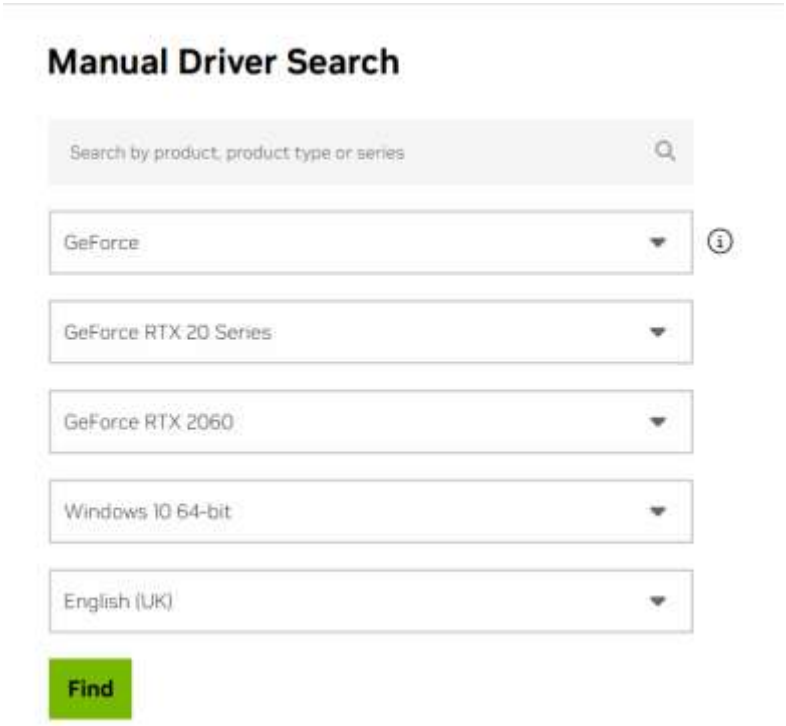

Figure 3.2. Manual driver search

Click on FIND and you will get the download interface for the version.

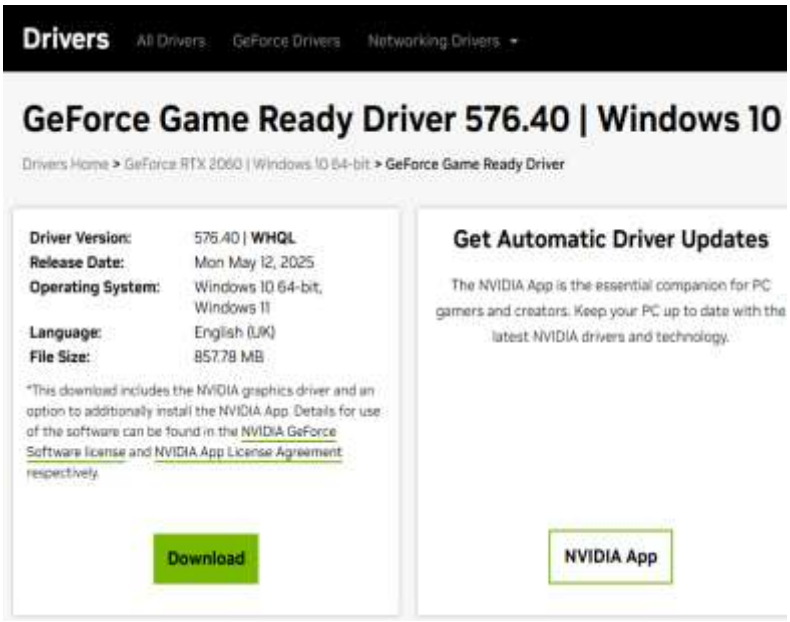

Figure 3.3. Select and download the driver

- Download the driver.
- Follow the installation instructions.
- Reboot your computer.
- Open CMD and run 'nvidia-smi' command:

The command will give you information and tables about the version you installed like this:

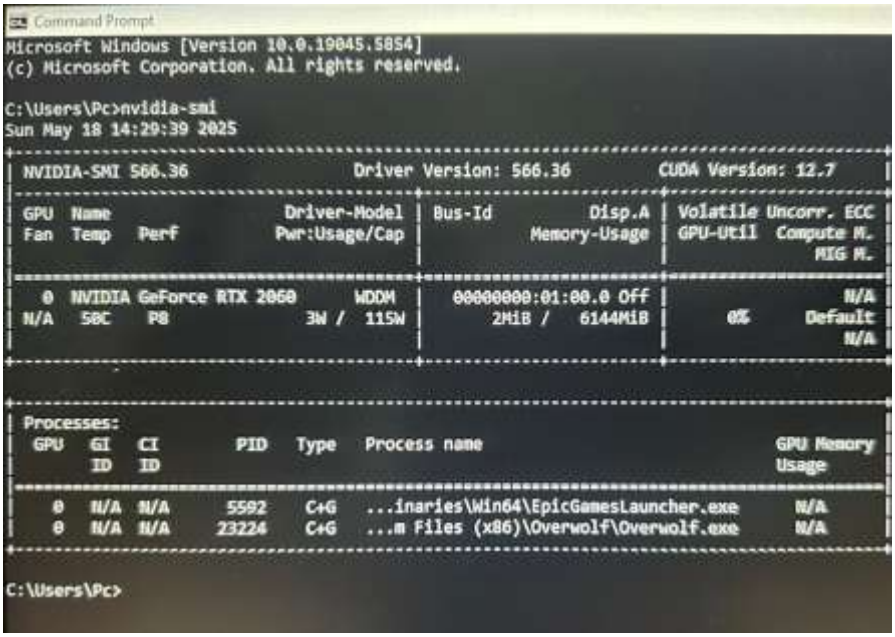

Figure 3.4.Open CMD interface

It means that the version NVIDIA Driver was installed successfully.

Note: CUDA version number in the table represents the latest CUDA Toolkit version your current NVIDIA Driver supports. It does not represent your currently installed CUDA Toolkit version, or even if you have it installed.

Step 3: Installing CUDA Toolkit:

Open following link in your browser: ' https://developer.nvidia.com/cuda-toolkit-archive

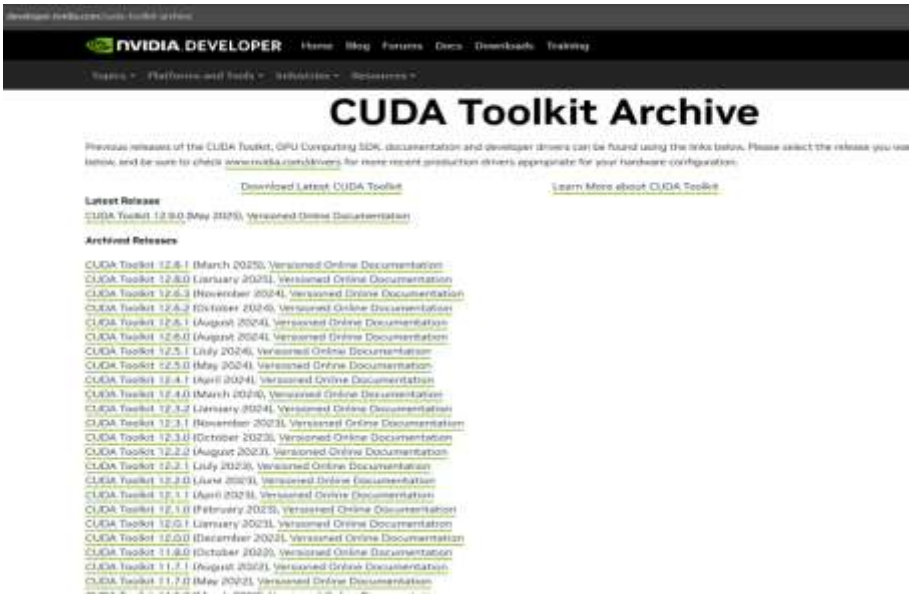

Figure 3.5. Choosing the version of CUDA

Choose the version number you want to download from CUDA. It is preferable to install older versions because they are more stable and without problems. In my case I chose 12.6.0

Figure 3.6. Choosing configuration of your PC

After downloading the version of CUDA, you must install it on the PC. These are simple steps, just follow the instructions.

Step 4: Installing cuDNN library:

Open following link in your browser:' https://developer.nvidia.com/rdp/cudnn-archive

It’s important to ensure the cuDNN version aligns with your installed CUDA version. Download the latest version of cuDNN compatible with the CUDA you downloaded 12.x or 11. x. in my case I chose cuDNN version v9.5.0 that compatible with CUDA 12.6

Figure 3.7. Select and load a library cuDNN 9.5.0

After downloading cuDNN from NVIDIA's website, you need to copy its files to the CUDA folder you have on your device, so that PyTorch or TensorFlow can use it.

- Extract the cuDNN file you downloaded (usually in ZIP format).
- Contains 3 main folders: bin / include / lib
- Go to the CUDA installation folder: Copy the contents of the folders from cuDNN to the CUDA folder

Step 5: Check installation:

Open CMD, and type: nvcc –version

Figure 3.8. Check installation via CMD

Now it means CUDA and cuDNN are installed correctly.

3.6. Model preparation and training

The models ResNet34, MobileNetV2 and DenseNet121, all CNN-based architecture, were trained and tested to compare their performance in classifying stuttering types. Each model was adapted to process audio spectrograms, which are converted into two-dimensional images used as input for the network. Training was performed using the PyTorch library, with technical settings standardized across models to ensure fairness in evaluation. These settings include:

During the training process, a set of technical settings were adopted, the most important of which were:

- Number of Epochs: A specific number of learning iterations was set to ensure stable performance Its 100 epochs (Although the training was initially set for 100 epochs, the model's learning process effectively stopped at epoch 24, either due to early convergence or manual interruption)
- Batch Size: This was chosen to balance memory usage and learning speed 32 batch size.
- Learning Rate: This was manually adjusted and then optimized using the validation results.
- Loss Function: The Cross Entropy function was used to evaluate the differences between predictions and actual results.
- Optimizer Algorithm: The Adam algorithm was adopted due to its efficiency and speed of convergence.
- Early stopping was also used to monitor the model's performance on the validation set and mitigate the overfitting problem. The best version of the model was saved based on its performance on the validation set.

3.7. Evaluation Metrics

metrics adopted within DL tasks play a crucial role in achieving the optimized classifier [121]. They are utilized within a usual data classification procedure through two main stages: training and testing. It is utilized to optimize the classification algorithm during the training stage. This means that the evaluation metric is utilized to discriminate and select the optimized solution, e.g., as a discriminator, which can generate an extra-accurate forecast of upcoming evaluations related to a specific classifier [122].

The Accuracy, Precision, Recall, and F1 Score metrics were adopted for their ability to provide a comprehensive analysis of model performance, especially under data imbalance. The F1 Score, in particular, is a precise measure for assessing performance in underrepresented classes, ensuring the reliability of the model in actual classification and diagnosis. In the following are the evaluation metrics used:

3.7.1. Accuracy score

Accuracy is a metric that measures how often a deep learning model correctly predicts the outcome. Calculates the ratio of correct predicted classes to the total number of samples evaluated. The accuracy score is calculated as follows (Eq. 3.1):

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.1)$$

Where:

- **TP** (True Positives): Correctly predicted positive cases
- **TN** (True Negatives): Correctly predicted negative cases
- **FP** (False Positives): Incorrectly predicted positive cases
- **FN** (False Negatives): Incorrectly predicted negative cases

3.7.2 Precision

Precision is a metric that gives you the proportion of true positives to the number of total positives that the model predicts. It is utilized to calculate the positive patterns that are correctly predicted by all predicted patterns in a positive class (Eq. 3.2).

$$\text{Precision} = \frac{TP}{TP+FP} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.2)$$

3.7.3. Recall

Recall focuses on how good the model is at finding all the positives. It is Utilized to calculate the fraction of positive patterns that are correctly classified (Eq. 3.3).

$$\text{Recall} = \frac{TP}{TP+FN} \quad \text{(3.3)}$$

3.7.4. F1-Score

F1-Score is a measure that combines recall and precision. There is a trade-off between precision and recall, F1 can therefore be used to measure how effectively our models make that trade-off. F1-Score: Calculates the harmonic average between recall and precision rates (Eq. 3.4).

$$\text{F1}_{\text{Score}} = 2 \times \frac{precision \times recall}{precision + recall} \quad \text{(3.4)}$$

3.8. Results and discussions

The table 3.2 shows the performance of the models used for stuttering classification: DenseNet121, MobileNetV2 and ResNet34. Comparing the performance of these models used, we observe that they are largely similar in terms of accuracy and evaluation metrics, with DenseNet121 slightly outperforming the others by achieving the highest F1 score of 58.97%. This demonstrates a good balance between the model's ability to correctly identify stuttering (Precision) and capture the maximum possible number of actual stuttering instances (Recall). On the other hand, the MobileNetV2 model achieved the highest overall accuracy of 67.50%, reflecting its strength in general voice classification. Finally, ResNet34, on the other hand, achieved the weakest results among the group, scoring the lowest percentages in all metrics, indicating its limited performance on this task.

Table 3.2: Performance Comparison of Deep Learning Models in Stuttering Classification (DenseNet121, MobileNetV2, ResNet34)

Model	Accuracy%	Precision%	Recall%	F1-Score%
DenseNet 121	67.23	59.33	56.87	58.97
MobileNet V2	67.50	59.98	57.21	58.44
ResNet 34	66.68	58.49	56.02	57.77

To enhance the visual understanding of the results, Figure (3.10) shows a graphical comparison between the three models according to the four evaluation indicators, where it can be noted that DenseNet121 outperforms in F1-score, demonstrating a good balance between correct predictions and actual case recall, compared to a slight advance for MobileNetV2 in Accuracy and Recall.

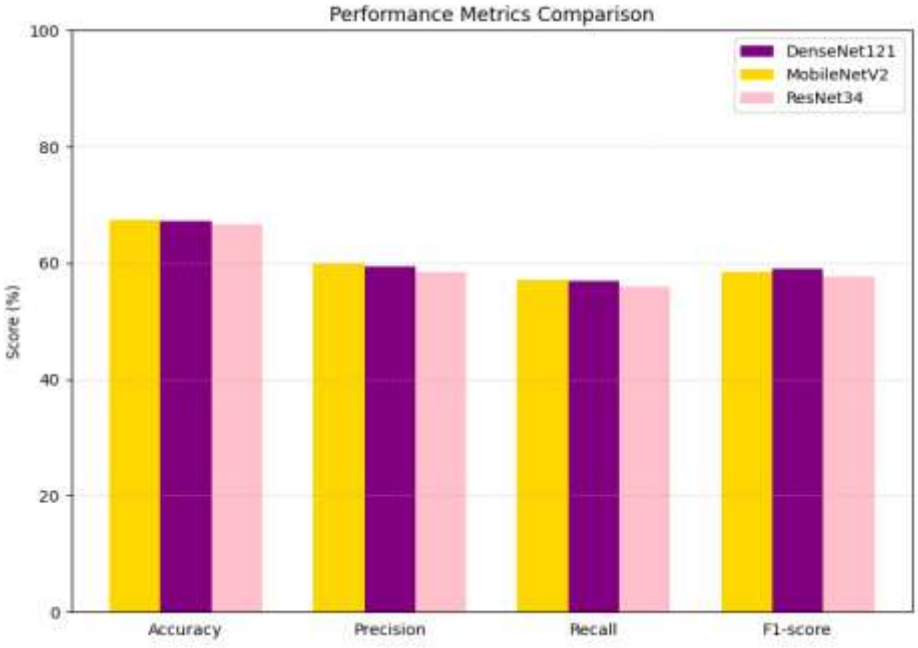

Figure 3.9: Comparison of performance indicators between the three classification model

The results shown in Figure (3.11) indicate that the DenseNet121 model performed highly on the training data, while its performance on the validation set was more variable, particularly in terms of precision and recall, which may indicate some overfitting. However, the stability of the F1 curve on the validation set demonstrates the model's ability to generalize within acceptable limits.

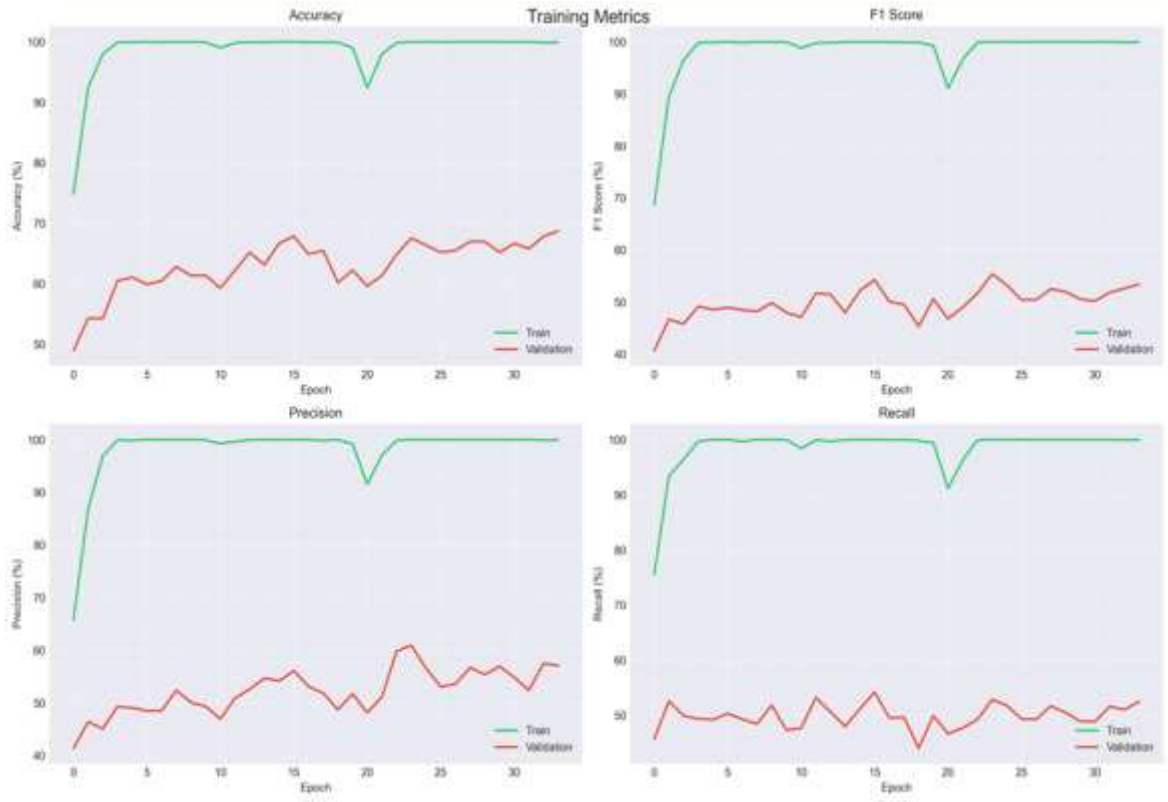

Figure 3:10: Accuracy, Precision, Recall, and F1-score curves during the training and validation of the DenseNet121 model over 30 cycles (Epoch)

Table 3.3 shows the classification report of the DenseNet 121 model for each stuttering type. Looking at the detailed classification report, we notice a discrepancy in performance as follows:

- The NoStuttered Words class showed the highest accuracy (0.82) and recall (0.66), demonstrating the DenseNet121 model's ability to recognize smooth speech and reduce false alarms.
- The Block class achieved the lowest performance, with an accuracy of 0.25 and a recall of 0.18, reflecting the difficulty of accurately identifying this phenomenon due to its rarity and complexity.
- The Prolongation and Rep classes achieved average performance, with F1 values ranging between 0.42 and 0.48, indicating that the model needs further refinement to address these specific stuttering patterns.

This discrepancy in performance underscores the challenges associated with classifying different stuttering patterns and calls for the development of specialized strategies to comprehensively improve model accuracy.

Table 3.3: Classification Report for the model DenseNet 121.

Stuttering Types	Precision%	Recall%	F1 -score%	Support
Block	0.25	0.18	0.21	107
No Stutter Words	0.82	0.66	0.73	585
Prolongation	0.45	0.38	0.42	196
Repetition	0.52	0.45	0.48	489

Based on the results obtained, DenseNet121 was chosen for this project due to its compelling combination of technical prowess and operational efficiency. Its core mechanism, characterized by direct connections between each layer and all preceding layers, facilitates deeper and more effective learning by preserving crucial information throughout the training process. Moreover, DenseNet121 is notably parameter-efficient compared to other models, making it particularly well-suited for analyzing stuttered speech data. This allows us to maintain high accuracy and computational efficiency, thereby reliably achieving the project's objectives.

3.9. Self-diagnostic web App application

This project aims to create an easy-to-use web application that enables users to assess their speech patterns by uploading audio clips. The primary functionality is an analysis model that instantly processes the uploaded audio clip to determine the type of stuttering present. This tool serves as an initial diagnostic assessment, providing individuals with speech disorders, or their parents, with an initial understanding of their stuttering pattern. This early insight enables immediate action toward therapeutic intervention or specialized consultation.

3.9.1. Technical Implementation

To achieve this, the development process will include the following key phases:

- **Project Structure:** We will create a robust project structure, including dedicated folders for HTML, CSS, and JavaScript files, along with directories for storing the analysis model and user-uploaded audio files.
- **User Interface (UI) Design:** The UI will be built using HTML to display essential elements such as audio upload fields and submit buttons. CSS will then be applied to style these elements, ensuring an engaging and user-friendly experience.

- **Front-end Logic**: JavaScript will be used to handle the front-end logic, including collecting the audio file from the user and securely sending it to the back-end server via an API. JavaScript will also be responsible for displaying the analysis results to the user on the web page.
- **Back-end Development:** For the application's overall functionality, a back-end server is essential. We recommend building this using a language like Python and leveraging a framework like Flask. The back-end server will be responsible for receiving the audio file, processing it through the trained stutter detection model, and then returning the identified stutter type to the front-end.

3.9.2. How to integrate the form into the web App

To integrate a pre-trained AI model into a web application, the Flask framework in Python offers a simple and efficient solution for building lightweight web interfaces. The process begins by setting up the environment and installing essential libraries such as flask and torch. The trained model is then loaded using torch.load() within the Flask server and set to evaluation mode using eval ().

Flask receives the file through request. files, and preprocessing is performed on the server side, this could involve converting the image into a tensor using torchvision. transforms, or converting audio into a spectrogram using libraries like librosa, if necessary. Once the input is prepared, it is passed to the model, which returns a prediction (e.g., the type of stuttering), and the result is displayed back to the user either via an HTML page or as a JSON response.

This approach effectively turns the AI model into a full web service, accessible through a browser by any user, enhancing the model’s usability and real-world applicability—especially for building supportive tools for people who stutter. The application can later be deployed on platforms like Render or Heroku to provide persistent online access.

3.9.3. How to use by the visitor

When a user visits the Stutter classify website for the first time, they are greeted with a clean and user-friendly interface that welcomes them and prompts them to enter some basic information such as name, age, and gender. This step aims to collect initial data that can help personalize the experience and improve the model’s performance in later stages. Once the fields are filled out, the user can simply click the "Next" button to proceed. The interface is designed to be simple and clear, making the platform accessible even to users with no technical background. As shown in the image below

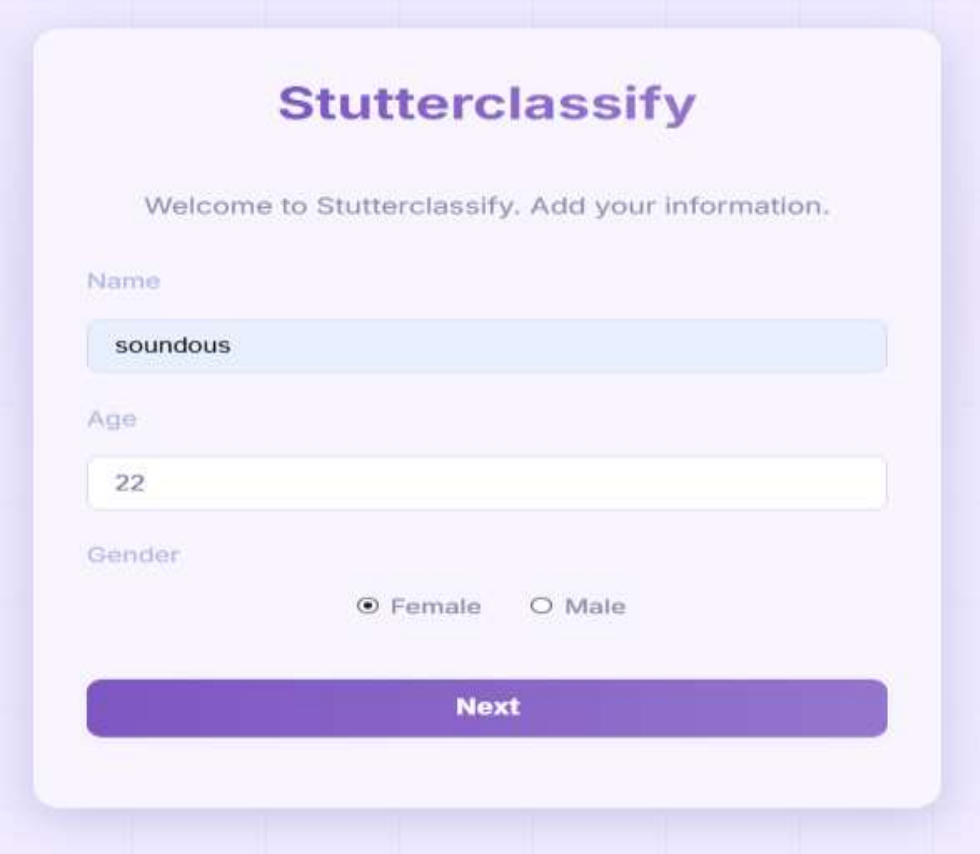

Figure 3.11: User information input interface before audio analysis

Following the initial demographic input, users are enabled to upload an audio file directly from their device. This submitted audio then undergoes analysis by a trained computational model, as depicted within the "Audio Analysis" interface. Upon completion of this processing, the platform presents the classification results. This informs the user of the presence of stuttering and, if identified, specifies its phenomenological type (e.g., repetitions, prolongations, or blocks).

Figure 3.12: Illustrative images of the steps for using the website after entering personal information in the interactive interface to analyze the voice and display the results.

3.10. Conclusion

In this work, we attempted to develop an intelligent model capable of classifying stuttering types using computer vision and deep learning techniques. We converted audio recordings into spectrograms and trained them on three different architectures: DenseNet121, MobileNetV1, and ResNet34. The results showed that all models performed well, but DenseNet121 outperformed both ResNet34 and MobileNetV1, offering a strong balance between accuracy and the ability to distinguish between different stuttering types. MobileNetV1 was lightweight and efficient for training, making it suitable for low-resource environments. ResNet34 showed solid performance but struggled slightly in capturing subtle distinctions between similar stuttering patterns. We faced some challenges, particularly in distinguishing visually similar types such as repetition and prolongation, and dealing with class imbalance in the dataset. Finally, we developed a web application to simplify the operation of stuttering detection and classification for end users.

General conclusion

General Conclusion

Development of a stutter detection and classification system has been a very rewarding experience. Speech is at the heart of human communication but is also very rich and variable between individuals. Detection of stuttered speech is especially challenging since disfluencies are highly subtle, highly variable, and even troublesome for humans to detect unambiguously. This makes automatic stutter detection difficult in speech processing and deep learning as well.

In this work, we sought to create a system that would be able to classify speech into no stutter, block, prolongation, and repetition. From labeled speech data, we created an end-to-end pipeline to analyze and process audio inputs. One of the key aspects of this project was creating a web application through which the user would be able to record audio or upload audio files in order to receive real-time feedback on the kind of stutter present. This real-time interface brings the technology to the threshold of utilization. The system utilizes a chain of carefully designed preprocessing steps. Sound is first translated into WAV to render it compatible, and noise reduction is then performed to improve clarity. Speech is segmented into short snippets to enable finer detection granularity. Each portion is translated to a spectrogram — a graphic representation of sound frequency over time — and passed through deep learning models. This process allows the neural networks to "see" patterns in speech, making it easier for them to detect subtle stuttering features.

We experimented with a number of widely used convolutional neural network models like DenseNet121, MobileNetV2, and ResNet34. The results showed, when applied to the SEP-28k FluencyBank dataset, all models demonstrated strong performance. However, DenseNet121 performed best in terms of the highest accuracy and reliability in distinguishing different types of stuttering. MobileNetV2 was less precise but immensely faster and smaller in size, making it a suitable choice for real-time or low-resource setups. Despite these positive findings, the project also indicated some problems. Stuttering is highly variable and highly individual, and repeated categorization is difficult even for experts. Environmental conditions such as background noise, varying recording quality, and varying speakers impacted model performance. Nevertheless, this study illustrates that with suitable preprocessing and labeled training data, deep learning models can be excellent aids to learn about and assist people who stutter. Aside from the technical output, this project was also eye-opener for us on a personal level. It deepened my appreciation for the intricacies of human speech and the promise of artificial intelligence to solve genuine human needs. I saw how infusing technology with compassion could produce solutions that not only work but also have meaning.

In short, this work represents an important bridge between artificial intelligence and human communication. It reminds us that each dataset and algorithm has behind it a set of people with distinct voices and stories — and that technology's greatest potential is to build understanding and connection.

There remains significant room for future improvement. Model robustness can be improved, particularly in noisy or dynamic environments. Additional features may be integrated, and the web application can be further developed to improve user experience. Most importantly, I hope this research inspires continued innovation in speech technology to create more inclusive and empowering solutions for individuals with stuttering and other speech disorders.

References

References

[1]Feit, D., & Feldman, H. M. (2007). The parents' guide to speech and language problems. New York, NY: McGraw-Hill

[2] Lanier, W. H. (2010). Speech Disorders. Lucent Books.

[3] Yonnie, K., Yusrita, Y., & Diana, C. H. (2017). An analysis of speech disorder produced by the character of The King's Speech movie (Undergraduate thesis, Universitas Bung Hatta). http://repo.bunghatta.ac.id/14385

4]] Adapted from Defining Speech and Language Disorders; 2023. Available from: https://speechandlanguagedisabilities.weebly.com/. Accessed December 11, 2023.

[5] Children's Hospital. (n.d.). Conditions we treat. Retrieved October 1, 2023, from https://www.childrenshospital.com/conditions-and-treatments

[6] Johns Hopkins Medicine, nd, Stuttering in Children, Retrieved, (21/04/2025) https://www.hopkinsmedicine.org/health/conditions-and-diseases/stuttering

[7] Sparsh Garg, 20161025, Deep Learning based Speech Disfluency Detection, International Institute of Information Technology, p5-6

[8] Harrison, J. C. (n.d.). Understanding the speech block. Stuttering Specialist. Retrieved May 16, 2025, from https://www.stuttering-specialist.com/post/understanding-the-speech-block

[9] Robert W. Sander, MD, Charles A. Osborne, MA, CCC-SLP, 2019, Stuttering: Understanding and Treating a Common Disability, University of Wisconsin, Stevens Point, Wisconsin, Medical College of Wisconsin-Central Wisconsin, Wausau, Wisconsin, p1

[10] Frontiers in Neurology. (2023). Acoustic analysis in stuttering: a machine-learning study

https://www.frontiersin.org/journals/neurology/articles/10.3389/fneur.2023.1169707/full

[11] Brahmi, Z., Mahyoob, M., Al-Sarem, M., Algaraady, J., Bousselmi, K., & Alblwi, A. (2024). Exploring the role of machine learning in diagnosing and treating speech disorders: A systematic literature review. Psychology Research and Behavior Management, 17, 373–390. https://doi.org/10.2147/PRBM.S451778

[12] SLP NOW. (n.d) stuttering-assessment: A guide for school-based speech- language pathologists. Retrieved April 27,2025, from https://slpnow.com/blog/stuttering-assessment/?utm_source=chatgpt.com

[13] Apple Machine Learning Research. (2023, May 18). Reconnaissance vocale améliorée pour les personnes qui bégaient. Retrieved April 27, 2025, from https://machinelearning.apple.com/research/speech-recognition?utm_source=chatgpt.com

[14] Wikipedia contributors. (n.d.). Téléréadaptation. Wikipedia. Retrieved April 27, 2025, from https://en.wikipedia.org/wiki/Telerehabilitation

[15] Alharbi, S.; Hasan, M.; Simons, A.J.H.; Brumfitt, S.; Green, P. Sequence Labeling

To Detect Stuttering Events in Read Speech. Comput. Speech Lang. 2020, 62, 101052.

[16] Kourkounakis, T.; Hajavi, A.; Etemad, A. Detecting Multiple Speech Disfluencies Using a Deep Residual Network with Bidirectional Long Short-Term Memory; IEEE:

Barcelona, Spain, 2020; p. 6093. [Google Scholar]

[17] Jouaiti, M.; Dautenhahn, K. Dysfluency Classification in Stuttered Speech Using

Deep Learning for Real-Time Applications. In Proceedings of the ICASSP 2022—2022 IEEE

International Conference on Acoustics, Speech and Signal Processing (ICASSP),

Singapore, 23–27 May 2022; pp. 6482–6486.

[18] Sheikh, S.A.; Sahidullah, M.; Hirsch, F.; Ouni, S. Introducing ECAPA-TDNN and

Wav2Vec2.0 Embeddings to Stuttering Detection. arXiv 2022, arXiv:2204.01564.

[19] Sheikh, S.A.; Sahidullah, M.; Hirsch, F.; Ouni, S. Introducing ECAPA-TDNN and

Wav2Vec2.0 Embeddings to Stuttering Detection. arXiv 2022, arXiv:2204.01564.

[20] Constantino, C.D.; Leslie, P.; Quesal, R.W.; Yaruss, J.S. A Preliminary Investigation

Of Daily Variability of Stuttering in Adults. J. Commun. Disord. 2016, 60, 39–50. [Google

Scholar] [CrossRef] [PubMed]

[21] Sheikh, S.A.; Sahidullah, M.; Hirsch, F.; Ouni, S. Advancing Stuttering Detection

Via Data Augmentation, Class-Balanced Loss and Multi-Contextual Deep Learning. IEEE

J. Biomed. Health Inform. 2023, 27, 2553–2564. [Google Scholar] [CrossRef]

[22] Guitar, B. Stuttering: An Integrated Approach to Its Nature and Treatment;

Lippincott Williams & Wilkins: Philadelphia, PA, USA, 2013; ISBN 978-1-49634612-4.

[23] Sheikh, S.A.; Sahidullah, M.; Hirsch, F.; Ouni, S. Advancing Stuttering Detection

Via Data Augmentation, Class-Balanced Loss and Multi-Contextual Deep Learning. IEEE

J. Biomed. Health Inform. 2023, 27, 2553–2564. [Google Scholar] [CrossRef]

[24] P. Mahesha, D. S. Vinod, “LP-Hillbert Transform Based MFCC for Effective Discrimination of Stuttering Dysfluencies,” This full-text paper was peer-reviewed and accepted to be presented at the IEEE WiSPNET 2017 conference

[25] P. Mahesha, D. S. Vinoda, “Combining Cepstral and Prosody Features for

Classification of Disfluencies in Stutter Speech,” 1 Springer India 2015 L.C. Jain et al.

(eds.), Intelligent Computing, Communication and Devices, Advances in Intelligent Systems and Computing 308, DOI 10.1007/978-81-322-2012-1_67

[26] Yairi, E., & Seery, C. H. (2015). Stuttering: Foundations and clinical applications (2nd ed.). Pearson.

Guitar, B. (2013). Stuttering: An integrated approach to its nature and treatment (4th ed.). Lippincott Williams & Wilkins.

References

Ambrose, N. G., & Yairi, E. (1999). Normative disfluency data for early childhood stuttering. Journal of Speech, Language, and Hearing Research, 42(4), 895–909.

https://doi.org/10.1044/jslhr.4204.895

Conture, E. G. (2001). Stuttering: Its nature, diagnosis, and treatment. Allyn & Bacon.

Howell, P. (2007). Signs of developmental stuttering up to age eight and at 12 plus.

Clinical Psychology Review, 27(3), 287–306. https://doi.org/10.1016/j.cpr.2006.10.001

[27] Jeanna Rileya, Glyndon Rileyb, Gerald Maguirec, "Subjective Screening of

Stuttering severity, locus of control and avoidance: research edition," Journal of

Fluency Disorders 29 (2004) 51–62

[28] Kusuma. H.R1 and G. Seshikala2, 2022, An Overview of Subjective and Objective

Assessment of Stuttering, 1Department of Electronics and communication, PES College of Engineering Mandya, India,p593.

[29] Korinek A., Schindler M., Stiglitz J. Technological Progress, Artificial Intelligence,

And Inclusive Growth. International Monetary Fund; Washington, DC, USA: 2021. IMF Working

Paper no. 2021/166.

[30] Raghad Alnashwan 1 ,2023, Noura Alhakbani 1 , Abeer Al-Nafjan 2 , Abdulaziz

Almudhi 3 and Waleed Al-Nuwaiser 2, Computational Intelligence-Based Stuttering Detection: A Systematic Review, Information Technology Department, College of Computer and

Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia;

444203361@student.ksu.edu.sa (R.A.); nhakbani@ksu.edu.sa (N.A.),P2

[31] S.P. Bayerl, D. Wagner, E. Nöth, T. Bocklet, K. Riedhammer, The influence of dataset partitioning on dysfluency detection systems, in: International Conference on Text, Speech, and Dialogue, 2022, pp. 423–436. Cham: Springer International Publishing.

[32] P. Howell, S. Sackin, Automatic recognition of repetitions and prolongations in stuttered speech, in: Proc. of the first World Congress on Fluency Disorders, Vol. 2, University Press Nijmegen Nijmegen, The Netherlands, 1995, pp. 372–374.

[33] M. Wi´sniewski, W. Kuniszyk-J´o´zkowiak, E. Smo lka, W. Suszy´nski, Automatic detection of disorders in a continuous speech with the hidden markov models approach, in: Computer Recognition Systems 2, Springer, 2007, pp. 445–453 .

[34] P. Mahesha, D. Vinod, Classification of speech dysfluencies using speech parameterization techniques and multiclass svm, in: Proc. International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, Springer, 2013, pp. 298–308.

[35] S.P. Bayerl, D. Wagner, E. N¨oth, K. Riedhammer, Detecting dysfluencies in stuttering therapy using wav2vec 2.0, arXiv preprint arXiv:2204.03417.

[36] Ramitha, V., Chainani, R., Mehrotra, S., Sah, S., & Mahajan, S. (2024). Evaluative comparison of machine learning algorithms for stutter detection and classification. MethodsX, 13, 103050.

[37] Mahesha, P., & Vinod, D. S. (2013). Classification of speech dysfluencies using speech parameterization techniques and multiclass SVM. In N. Meghanathan, D. Nagamalai, & N. Chaki (Eds.), Advances in Computing and Information Technology (Vol. 2, pp. 615–623). Springer. https://doi.org/10.1007/978-3-642-37949-9_26

[38] Ravikumar, K. M., Rajagopal, R., & Nagaraj, H. C. (2008). An approach for objective assessment of stuttered speech using MFCC features. DSP Journal, 9(1), 22–27. Retrieved from https://www.itie.in/Ravi_Paper_itie_ICGST

[39] Pálfy, J., & Pospíchal, J. (2011). Recognition of repetitions using support vector machines. In Signal Processing Algorithms, Architectures, Arrangements, and Applications (SPA) (pp. 1–6). IEEE

[40] Korvel, G.; Kostek, B. Comparison of Lithuanian and Polish Consonant Phonemes Based on Acoustic Analysis—Preliminary Results. Arch. Acoust. 2019, 44, 693–707. [Google Scholar] [CrossRef],

Mahesha, P.; Vinod, D. Classification of speech disfluencies using speech parameterization techniques and multiclass svm. In Proceedings of the International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness, Greader Noida, India, 11–12 January 2013; Springer: Berlin/Heidelberg, Germany, 2013; pp. 298–308. [Google Scholar] [CrossRef],

Ravikumar, K.; Rajagopal, R.; Nagaraj, H. An approach for objective assessment of stuttered speech using MFCC. ICGST Int. J. Digit. Signal Process. 2009, 9, 19–24.

Ravikumar, K.; Rajagopal, R.; Nagaraj, H. An approach for objective assessment of stuttered speech using MFCC. ICGST Int. J. Digit. Signal Process. 2009, 9, 19–24.

Ravikumar, K.; Rajagopal, R.; Nagaraj, H. An approach for objective assessment of stuttered speech using MFCC. ICGST Int. J. Digit. Signal Process. 2009, 9, 19–24.

[41] M. Wi´sniewski, W. Kuniszyk-J´o´zkowiak, E. Smo lka, W. Suszy´nski, Automatic Detection of Disorders in a Continuous Speech with the Hidden Markov Models Approach, Springer, 2007, pp. 445–453.

[42] T.-S. Tan, A. Ariff, C.-M. Ting, S.-H. Salleh, et al., Application of malay speech technology in malay speech therapy assistance tools, in: Proc. 2007 International Conference on Intelligent and Advanced Systems, IEEE, 2007, pp. 330–334

[43] E. N¨oth, H. Niemann, T. Haderlein, M. Decher, U. Eysholdt, F. Rosanowski, T. Wittenberg, Auto-matic stuttering recognition using hidden Markov models, in: Proc. Sixth International Conference on Spoken Language Processing, 2000.

[44] P. Howell, S. Sackin, Automatic recognition of repetitions and prolongations in stuttered speech,

in: Proc. of the first World Congress on Fluency Disorders, Vol. 2, University Press Nijmegen

Nijmegen, The Netherlands, 1995, pp. 372{374.

[45] P. Howell, S. Sackin, K. Glenn, 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) (1997) 1073{1084.

[46] P. Howell, S. Sackin, K. Glenn, 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) (1997) 1085{1096.

[47] Y. Prabhu, N. Seliya, A CNN-based automated stuttering identification system, in: 2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA), Nassau, Bahamas, 2022, pp. 1601–1605.

[48]ence/audio-deep-learning-made-simple-sound-classification-step-by-step-cebc936bbe5

[49] Tasdelen and B. Sen, "A hybrid CNN-LSTM model for pre-miRNA classification," Sci. Rep., vol.

11, no. 1, pp. 1-9, 2021, doi: https://doi.org/10.1038/s41598-021-93656-0.

[50] L. Qin, N. Yu, and D. Zhao, "Applying the convolutional neural network deep learning technology to behavioural recognition in intelligent video," Tehnički vjesnik, vol. 25, no. 2, pp. 528-535, 2018, doi:https://doi.org/10.17559/TV-20171229024444.

[51] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," IEEE Trans. Neural Netw. Learn. Syst., vol. 33, no. 12, pp. 6999-7019,Dec 2022, doi: https://doi.org/10.1109/TNNLS.2021.3084827.

[52] B. P. Babu and S. J. Narayanan, "One-vs-All Convolutional Neural Networks for Synthetic Aperture Radar Target Recognition," Cybern. Inf. Technol, vol. 22, pp. 179-197, 2022, doi:

https://doi.org/10.2478/cait-2022-0035.

[53] S. Mekruksavanich and A. Jitpattanakul, "Deep convolutional neural network with rnns for complex activity recognition using wrist-worn wearable sensor data," Electro., vol. 10, no. 14, pp. 1685, 2021,doi: https://doi.org/10.3390/electronics10141685.

[54] W. Lu, J. Li, J. Wang, and L. Qin, "A CNN-BiLSTM-AM method for stock price prediction," Neural

Comput. Appl., vol. 33, pp. 4741-4753, 2021, doi: https://doi.org/10.1007/s00521-020-05532-z.

[55] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A

comprehensive review," Neural Comput., vol. 29, no. 9, pp. 2352-2449, 2017

[56] T. Kourkounakis, A. Hajavi, A. Etemad, Detecting Multiple Speech Disfluencies Using a Deep Residual Network with Bidirectional Long Short-Term Mem- ory, ICASSP 2020 - 2020

IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Barcelona, Spain, 2020, pp. 6089–6093,doi:10.1109/ICASSP40776.2020.9053893.

[57] B. Guitar, Stuttering: An Integrated Approach to Its Nature and Treatment, Lippincott Williams & Wilkins, 2013.

International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2020, pp.6089–6093.

[58] S. A. Sheikh, M. Sahidullah, F. Hirsch, S. Ouni, StutterNet: Stuttering Detection Using Time Delay Neural Network, in: Proc. EUSIPCO 2021 – 29th European Signal Processing Conference,Dublin, Ireland, 2021.

[59] Hewamalage H, Bergmeir C, Bandara K. Recurrent neural networks for time series forecasting: current status and future directions. Int J Forecast. 2020;37(1):388–427.

[60] John RA, Acharya J, Zhu C, Surendran A, Bose SK, Chaturvedi A, Tiwari N, Gao Y, He Y, Zhang KK, et al. Optogenetics inspired transition metal dichalcogenide neuristors for in-memory deep recurrent neural networks. Nat Commun.2020;11(1):1–9.

[61] Batur Dinler Ö, Aydin N. An optimal feature parameter set based on gated recurrent unit recurrent neural net works for speech segment detection. Appl Sci. 2020;10(4):1273.

[62] Jagannatha AN, Yu H. Structured prediction models for RNN based sequence labeling in clinical text. In: Proceed ings of the conference on empirical methods in natural language processing. conference on empirical methods in natural language processing, vol. 2016, NIH Public Access; 2016. p. 856.

[63] YAŞAR ANIL SANSAK,2023, AUTOMATIC STUTTERING DETECTION

AND CLASSIFICATION, A Thesis Submitted To The Graduate School of TED University,p26-27,https://avesis.tedu.edu.tr/dosya?id=06a20977-2356-4deb-8e67-73b4803a9ba2

[64] Alharbi, S., Alhindi, T., & Alkanhal, M. A. (2021). Automatic stuttering detection using deep learning techniques. IEEE Access, 9, 16142–16150.

https://doi.org/10.1109/ACCESS.2021.3052577

[65]. Assistant Professor, DCSA, Guru Nanak College, Ferozepur Cantt, Punjab, India.

International Journal of Science and Research Archive, 2023, 09(01), 281–285 0000

Publication history: Received on 18 April 2023; revised on 26 May 2023; accepted on 28 May 2023

[66]Concannon, M. (2024, August 29). Understanding machine Learning: A beginner's guide. Ntiva. https://www.ntiva.com/blog/what-is-machine-learning

[67]. Löning, Markus, et al. "sktime: A unified interface for machine learning with time series." arXiv preprint arXiv:1909.07872 (2019).

[68]. Alpaydin, E. (2010). Introduction to machine learning (2nd ed.). Cambridge, MA: MIT Press.

[69]. Bishop, C. M. (2006). Pattern recognition and machine learning. New York, NY: Springer.

[70]. Al-Jumaili, A.H.A.; Muniyandi, R.C.; Hasan, M.K.; Paw, J.K.S.; Singh, M.J. Big data analytics using cloud computing based frameworks for power management systems: Status, constraints, and future recommendations. Sensors 2023, 23, 2952. [CrossRef]

[71]. Gill, S.S.; Wu, H.; Patros, P.; Ottaviani, C.; Arora, P.; Pujol, V.C.; Haunschild, D.; Parlikad, A.K.; Cetinkaya, O.; Lutfiyya, H.; et al. Modern computing: Vision and challenges. Telemat. Inform. Rep. 2024, 13, 100116. [CrossRef]

[72] Mienye, I.D.; Obaido, G.; Emmanuel, I.D.; Ajani, A.A. A Survey of Bias and Fairness in Healthcare AI. In Proceedings of the 2024 IEEE 12th International Conference on Healthcare Informatics (ICHI), Orlando, FL, USA, 3–6 June 2024; pp. 642–650.

[73] Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems NIPS, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.

[74] Mienye, I. D., & Swart, T. G. (2024). A Comprehensive Review of Deep Learning: Architectures,

Recent Advances, and Applications. Information (Switzerland), 15(12), Article 755.

[75] https://www.analyticsvidhya.com/blog/2022/04/guide-to-audio-classification-using-deep-learning/

[76] Prasenjit, C. (2021, March 27). Perceptron – A simple yet mighty Machine Learning algorithm. Medium. https://medium.com/@cprasenjit32/perceptron-a-simple-yet-mighty-machine-learning-algorithm-9ff6b7d86a71

[77] https://www.geeksforgeeks.org/introduction-deep-learning/

[78] Khaki, M., & Asghari, K. (2021). Modeling the fluctuations of groundwater level by employing ensemble deep learning techniques. Engineering Applications of Computational Fluid Mechanics, 15(1), 1420–1439

[79] https://www.ibm.com/think/topics/deep-learning

[80] https://www.coursera.org/articles/deep-learning-models

[81] https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-a-neural-network/

[82] Gao C, Yan J, Zhou S, Varshney PK, Liu H. Long short-term memory-based deep recurrent neural networks for target tracking. Inf Sci. 2019;502:279–96.

[83] https://milvus.io/ai-quick-reference/what-are-spot-instances-in-cloud-computing

[84] . Krizhevsky A, Sutskever I, Hinton GE. Imagenet classifcation with deep convolutional neural networks. Commun ACM. 2017;60(6):84–90.

[85]. Zhou DX. Theory of deep convolutional neural networks: downsampling. Neural Netw.

2020;124:319–27.

[86] Zhou DX. Theory of deep convolutional neural networks: downsampling. Neural Netw. 2020;124:319–27.

[86]. Jhong SY, Tseng PY, Siriphockpirom N, Hsia CH, Huang MS, Hua KL, Chen YY. An automated biometric identifca tion system using CNN-based palm vein recognition. In: 2020 international conference on advanced robotics and intelligent systems (ARIS). IEEE; 2020. p. 1–6.

[87] Al-Azzawi A, Ouadou A, Max H, Duan Y, Tanner JJ, Cheng J. Deepcryopicker: fully automated deep neural network for single protein particle picking in cryo-EM. BMC Bioinform. 2020;21(1):1–38.

[88] Wang T, Lu C, Yang M, Hong F, Liu C. A hybrid method for heartbeat classifcation via convolutional neural net works, multilayer perceptrons and focal loss. PeerJ Comput Sci. 2020;6:324.

[89] Li G, Zhang M, Li J, Lv F, Tong G. Efcient densely connected convolutional neural networks. Pattern Recogn. 2021;109:107610.

[90] Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.;

Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: Concepts,

CNN architectures, challenges, applications, future directions. J. Big Data 2021, 8, 53.

[91] Gu J, Wang Z, Kuen J, Ma L, Shahroudy A, Shuai B, Liu T, Wang X, Wang G, Cai J, et al. Recent

advances in convolu tional neural networks. Pattern Recogn. 2018;77:354–77.

[92] Hubel DH, Wiesel TN. Receptive felds, binocular interaction and functional architecture in the cat's visual cortex. J Physiol. 1962;160(1):106.

[93] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-Based Learning

Applied to Document Recognition. Proceedings of the IEEE, 86(11), 2278–2324

[94] Peng, M., Wang, C., Chen, T., Xu, X., & Fu, X. (2017). Dual temporal scale convolutional neural network for micro-expression recognition. Frontiers in Psychology, 8, 1745. https://doi.org/10.3389/fpsyg.2017.01745

[95] M. Peng, C. Wang, T. Chen, G. Liu, and X. Fu, "Dual temporal scale convolutional neural network for micro-expression recognition," Front. Psychol., vol. 8, 2017, Art. no. 1745

[96] Review of deep learning: concepts, CNN architectures, challenges, applications, future

directions Laith Alzubaidi1,5* , Jinglan Zhang1, Amjad J. Humaidi2, Ayad Al-Dujaili3, Ye Duan4, Omran

Al-Shamma5, J. Santamaría6, Mohammed A. Fadhel7, Muthana Al-Amidie4 and Laith Farhan8

[97] Purwono a,1 , Alfian Ma'arif b,2,* , Wahyu Rahmaniar c,3 , Haris Imam Karim Fathurrahman b,4 , Aufaclav Zatu Kusuma Frisky d,e,5 , Qazi Mazhar ul Haq f,6 a Universitas Harapan Bangsa, Jl. Raden Patah No. 100 Kedunglongsir Ledug Kembaran, Banyumas.

[98] Mienye, I. D., & Swart, T. G. (2024). A Comprehensive Review of Deep Learning: Architectures, Recent Advances, and Applications. Information (Switzerland), 15(12), Article 755.

[99] https://www.simplilearn.com/tutorials/deep-learning-tutorial/convolutional-neural-network

[100] Purwono a,1 , Alfian Ma'arif b,2,* , Wahyu Rahmaniar c,3 , Haris Imam Karim Fathurrahman b,4 , Aufaclav Zatu Kusuma Frisky d,e,5 , Qazi Mazhar ul Haq f,6 a Universitas Harapan Bangsa, Jl. Raden Patah No. 100 Kedunglongsir Ledug Kembaran, Banyumas.

[101] https://builtin.com/machine-learning/adam-optimization

[102] Alharbi, S.; Hasan, M.; Simons, A.J.; Brumfitt, S.; Green, P. Detecting stuttering events n transcripts of children's speech. In Proceedings of the Statistical Language and Speech Processing: 5th International Conference, SLSP 2017, Le Mans, France, 23–25 October 2017; Springer: Berlin/Heidelberg, Germany; pp. 217–228. [Google Scholar]

[103] Heeman, P.A.; Lunsford, R.; McMillin, A.; Yaruss, J.S. Using Clinician Annotations to Improve Automatic Speech Recognition of Stuttered Speech. In Proceedings of the Interspeech, San Francisco, CA, USA, 8–12 September 2016; pp. 2651–2655. [Google Scholar]

[104] Advancing Stuttering Detection via Data Augmentation, Class-Balanced Loss andMulti-Contextual Deep Learning Shakeel A. Sheikh, Md Sahidullah, Fabrice Hirsch, Slim Ouni

[105] https://repairit.wondershare.com/audio-repair/how-to-convert-mp3-to-wav.html

[106] Q. Ke, J. Liu, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, "Computer vision for human-machine interaction," Computer vision for human–machine interaction, Computer Vision For Assistive Healthcare, pp. 127–145, 2018,

[107] Wondershare. (n.d.). How to convert MP3 to WAV with simple steps. Repairit. https://repairit.wondershare.com/audio-repair/how-to-convert-mp3-to-wav.html

[108] https://www.adobe.com/gr_en/creativecloud/video/discover/convert-mp3-to-wav.html

[109] S. Boll. Suppression of acoustic noise in speech using spectral subtraction. IEEE Transactions on acoustics, speech, and signal processing, 27(2):113–120, 1979

[110] audio Segmentation Techniques and Applications Based onDeep Learning Shruti Aggarwal , 1 Vasukidevi G,2 S. Selvakanmani,3 Bhaskar Pant,4 Kiranjeet Kaur,5Amit Verma,5 and Geleta Negasa Binegde

[111] https://onlinelibrary.wiley.com/doi/10.1155/2022/7994191

[112] https://www.geeksforgeeks.org/audio-classification-using-spectrograms/

[113] Said Karam 1 · ShanqJang Ruan1 · Qazi Mazhar ul Haq2 · Lieber PoHung Li3,4,5 Received: 20 May 2021 / Accepted: 8 February 2023 / Published online: 26 February 2023 © The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2023

[114] https://milvus.io/ai-quick-reference/what-is-the-importance-of-feature-extraction-in-deep-

Learning

[115] Bunny Saini[1], Divya Venkatesh1 , Nikita Chaudhari1 , Tanaya Shelake1 , Shilpa Gite1,2, Biswajeet Pradhan3

[116] He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778

[117] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," arXiv preprint arXiv:1608.06993, 2016.

[118] C. Lea et al., "Sep-28k: A dataset for stuttering event detection from podcasts with people who stutter," in Proc. ICASSP, 2021, pp. 6798–6802.

[119] Introduction to NVIDIA CUDA Achieving Peak Performance with H100 for AI and Deep

Learning. 30 nov 2024. <https://www.digitalocean.com/community/tutorials/intro-to-cuda>.

[120] Payong, Adrien. Step-by-Step Guide to Installing CUDA and cuDNN for GPU Acceleration. 10 feb 2025. <https://www.digitalocean.com/community/tutorials/install-cuda-cudnn-for-gpu>.

[121]Hossin M, Sulaiman M. A review on evaluation metrics for data classification evaluations. Int J Data Min Knowl Manag Process. 2015;5(2):1.

[122] Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. J. Big Data 2021, 8, 53.