الجمهورية الجزائرية الديمقراطية الشعبية

Republique Algérienne Démocratique Populaire

وزارة التعليم العالي والبحث العلمي

Ministère de l'Enseignement Supérieur de la Recherche Scientifique

Université 20 août 1955 Skikda

جامعة 20 أوت 1955 – سكيكدة

Faculte des Sciences
Departement d'informatique

كلية العلوم
قسم الإعلام الآلي

الرقم: .... / ق/ إ/ أ/ ك.ع2026/

Autorisation de Dépôt de Mémoire de Master

Je soussigné : ...Dr. Samira HAZMOUNE.....

Certifie que l'étudiant(e) : ...Taghane Nedjoua.....

Spécialité : ..Artificial Intelligence.....

Ayant soutenu le projet intitulé : An explainable SSL-Based Ensemble Framework for audio deepfake Detection.

A apporté les corrections nécessaires sur son manuscrit de Master.

Signature de l'encadreur

Scanné avec CamScanner

الجمهورية الجزائرية الديمقراطية الشعبية

Republique Algérienne Démocratique Populaire

وزارة التعليم العالي والبحث العلمي

Ministère de l'Enseignement Supérieur de la Recherche Scientifique

Université 20 août 1955 Skikda

جامعة 20 أوت 1955 – سكيكدة

Faculte des Sciences
Departement d'informatique

كلية العلوم
قسم الإعلام الآلي
الرقم: .... / ق/ !/ أ/ ك.ع2026/

Autorisation de Dépôt de Mémoire de Master

Je soussigné : ..... Dr. Samira HAZMOUNE .....

Certifie que l'étudiant(e) : Rahmani Aya .....

Spécialité : Artificial Intelligence .....

Ayant soutenu le projet intitulé : An explainable SSL-Based Ensemble Framework for audio deepfake Detection.

A apporté les corrections nécessaires sur son manuscrit de Master.

Signature de l'encadreur

Scanné avec CamScanner

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of 20 Aout 1955 – Skikda



Faculty of Sciences
Department of Computer Science

MASTER'S THESIS

Submitted in partial fulfillment of the requirements for the degree of
Master in Artificial Intelligence

An Explainable Self-Supervised Learning-Based Ensemble Framework for Audio Deepfake Detection

Presented by
RAHMANI Aya
TAGHANE Nedjoua

Supervised by
Dr. HAZMOUNE Samira

Presented and defended on June 22, 2026, before the following examination committee:

President: Dr.HAMROUCHE Houda
Examiner: Dr.BOURRICHE Hanene

Academic Year 2025–2026

Acknowledgements

First and foremost, we thank God Almighty for granting us the strength, courage, perseverance, and guidance needed to complete this thesis. Without His blessings and mercy, this achievement would not have been possible.

We would like to express our sincere gratitude to our supervisor, ***Dr. Samira HAZMOUNE****, for her invaluable guidance, continuous support, constructive feedback, and encouragement throughout the realization of this thesis. Her expertise and dedication greatly contributed to the successful completion of this work.*

We would also like to extend our appreciation to the members of the jury for accepting to evaluate this dissertation and for the time and effort devoted to reviewing our work. Their valuable comments and suggestions are highly appreciated.

Our sincere thanks go to the faculty members and staff of ***University of 20 August 1955 – Skikda*** *for the knowledge, guidance, and academic support provided throughout our studies.*

We are also grateful to our friends and colleagues for their encouragement, motivation, and support during this academic journey.

Finally, and most importantly, we would like to express our deepest gratitude to our families for their unconditional love, patience, sacrifices, and unwavering support. Their confidence in us has been a constant source of strength and inspiration throughout our studies.

Aya and Nedjoua

Dedication

I dedicate this work, the fruit of years of learning, perseverance, and determination, to the people whose love, support, and prayers have been my greatest source of strength.

*To my beloved **parents**, whose sacrifices, guidance, and unconditional love have shaped the person I am today. Your faith in me has always been my motivation to move forward, and this achievement is as much yours as it is mine.*

*To my dear brothers and sisters, **Nour El Din, Achrafe, Souad, and Karima**, thank you for your constant encouragement, understanding, and unwavering support. Your presence in my life has been a source of comfort, inspiration, and strength throughout this journey.*

*To my precious little ones, **Kossi, Assil, Ranim, and Iskander**, whose innocence, laughter, and joy bring light to every moment. May you grow with confidence, pursue your dreams fearlessly, and always believe in your potential.*

*To my dear colleague and partner in this dissertation journey,**Aya**, with whom I shared every step of this work. Thank you for your collaboration, dedication, and constant support throughout the development of this project. Your hard work, patience, and teamwork made this experience more meaningful and less challenging, and I am truly grateful for everything we accomplished together.*

*To our supervisor, Professor **Dr. Samira Hazmoune**, thank you for your guidance, valuable advice, and trust throughout this project. The strong foundation in Artificial Intelligence that you provided through your teaching enabled us to undertake and successfully complete this work. Your support and expertise were greatly appreciated.*

To my friends and everyone who stood by me during this academic journey, thank you for your trust, advice, and unforgettable memories.

*I also dedicate this work to the resilient people of **Palestine**, whose courage, patience, and perseverance in the face of adversity continue to inspire the world. May their suffering come to an end, may justice and peace prevail, and may hope remain alive in the hearts of future generations.*

Finally, I dedicate this work to all those who strive for knowledge, believing that education is a path to growth, progress, and positive change.

With gratitude and love,

Nedjoua

Dedication

*I dedicate this work to the soul of my beloved **father**, may Allah have mercy upon him and grant him the highest place in Paradise.*

*to my beloved **mother**, the strongest and wisest woman I have ever known. Every achievement in my life is a reflection of your endless support, guidance, and sincere duaa. You will always be the woman I aspire to become and the person I strive to make proud.*

*To my beautiful sisters: the brilliant doctor **Ibtihel**; the kind-hearted future doctor **Dikra**, and my precious little **Nourane**. Thank you for standing by my side through both difficult and joyful moments. Your love, encouragement, and presence have made every obstacle easier to overcome.*

*To **Nedjoua**, who was not only my partner in this work but also my closest friend throughout our university years. Having you by my side made this journey less difficult and far more enjoyable. I am proud not only of our friendship but also of the work we were able to accomplish together.*

*To our supervisor, Professor **Dr. Samira Hazmoune**, thank you for your guidance, valuable advice, and trust throughout this project. The strong foundation in Artificial Intelligence that you provided through your teaching enabled us to undertake and successfully complete this work. Your support and expertise were greatly appreciated.*

To my friends, university professors, and everyone who supported me, believed in me, and encouraged me to move forward, I extend my deepest gratitude.

*Finally, I dedicate this work to our steadfast brothers and sisters in **Gaza**. Your resilience and patience in the face of the harsh realities of occupation have always been a great source of inspiration to every Arab and Muslim. Your unwavering determination and strength remind us that hope can endure even in the darkest times.*
From the river to the sea, Palestine will be free.

With gratitude and love,

Aya

Abstract

Recent advances in artificial intelligence have enabled the generation of highly realistic audio deepfakes through speech synthesis and voice conversion technologies. Although these technologies offer numerous beneficial applications, they also pose serious security, privacy, and misinformation risks, making reliable audio deepfake detection an increasingly important research challenge.

To address this issue, this work proposes an audio deepfake detection framework that combines self-supervised learning (SSL) and ensemble learning. Three state-of-the-art SSL models: Wav2Vec 2.0, HuBERT, and WavLM, were fine-tuned to distinguish real from fake speech. Several ensemble strategies were investigated to exploit the complementary strengths of the individual models. Furthermore, the Gradient Saliency explainable AI technique was employed to enhance model interpretability and provide insights into the features influencing the detection decisions.

The proposed framework was evaluated on the ASVspoof 2019 Logical Access dataset and further assessed on the DeepVoice dataset to examine its generalization capability. Experimental results demonstrated the effectiveness of SSL-based speech representations for audio deepfake detection. Among all evaluated methods, the stacking ensemble achieved the best performance, reaching an accuracy of 99.15%, an F1-score of 99.14%, and an EER of 0.38% on the ASVspoof 2019 LA dataset. Cross-dataset experiments on the DeepVoice dataset yielded an accuracy of 81.25% and an F1-score of 88.89%, confirming the robustness and generalization capability of the proposed approach. These findings show that combining self-supervised speech representations with ensemble learning provides an effective, robust, and interpretable solution for audio deepfake detection.

Keywords: Audio deepfake detection, Self-supervised learning, Wav2Vec 2.0, HuBERT, WavLM, Explainable AI, Gradient saliency, ASVspoof 2019, DeepVoice, Ensemble learning.

Résumé

Les récentes avancées en intelligence artificielle ont permis la génération de deepfakes audio hautement réalistes grâce aux technologies de synthèse vocale et de conversion de voix. Bien que ces technologies offrent de nombreuses applications bénéfiques, elles présentent également d'importants risques en matière de sécurité, de confidentialité et de désinformation, faisant de la détection fiable des deepfakes audio un défi de recherche de plus en plus crucial.

Pour répondre à cette problématique, ce travail propose un cadre de détection des deepfakes audio combinant l'apprentissage auto-supervisé (Self-Supervised Learning, SSL) et l'apprentissage par ensemble (Ensemble Learning). Trois modèles SSL de pointe, à savoir Wav2Vec 2.0, HuBERT et WavLM, ont été affinés afin de distinguer les discours authentiques des discours falsifiés. Plusieurs stratégies d'ensemble ont été étudiées afin d'exploiter les forces complémentaires des différents modèles. En outre, la technique d'intelligence artificielle explicable Gradient Saliency a été utilisée pour améliorer l'interprétabilité des modèles et fournir des informations sur les caractéristiques influençant les décisions de détection.

Le cadre proposé a été évalué sur le jeu de données ASVspoof 2019 Logical Access, puis testé sur le jeu de données DeepVoice afin d'examiner sa capacité de généralisation. Les résultats expérimentaux ont démontré l'efficacité des représentations vocales basées sur l'apprentissage auto-supervisé pour la détection des deepfakes audio. Parmi toutes les méthodes évaluées, l'ensemble par empilement (stacking ensemble) a obtenu les meilleures performances, atteignant une exactitude (accuracy) de 99,15 %, un score F1 de 99,14 % et un taux d'erreur égal (EER) de 0,38 % sur le jeu de données ASVspoof 2019 LA. Les expériences de généralisation inter-jeux de données menées sur DeepVoice ont obtenu une exactitude de 81,25 % et un score F1 de 88,89 %, confirmant la robustesse et la capacité de généralisation de l'approche proposée. Ces résultats montrent que la combinaison des représentations vocales auto-supervisées et de l'apprentissage par ensemble constitue une solution efficace, robuste et interprétable pour la détection des deepfakes audio.

Mots-clés : Détection des deepfakes audio, apprentissage auto-supervisé, Wav2Vec 2.0, HuBERT, WavLM, IA explicable, saillance par gradient, ASVspoof 2019, DeepVoice, apprentissage par ensemble.

ملخص

أتاحت التطورات الحديثة في مجال الذكاء الاصطناعي إنتاج تزييفات صوتية عميقة (Audio Deepfakes)شديدة الواقعية من خلال تقنيات تركيب الكلام وتحويل الأصوات. وعلى الرغم من أن هذه التقنيات توفر العديد من التطبيقات المفيدة، فإنها تطرح أيضًا مخاطر كبيرة تتعلق بالأمن والخصوصية ونشر المعلومات المضللة، مما يجعل الكشف الموثوق عن التزييفات الصوتية العميقة تحديًا بحثيًا متزايد الأهمية.

للتصدي لهذه الإشكالية، يقترح هذا العمل إطارًا للكشف عن التزييفات الصوتية العميقة يجمع بين التعلم الذاتي الإشراف (Self-Supervised Learning - SSL) والتعلم التجميعي (Ensemble Learning).وقد تم ضبط ثلاثة نماذج متقدمة من نماذج التعلم الذاتي الإشراف، وهي Wav2Vec 2.0و HuBERT و WavLM، لتمييز الكلام الحقيقي من الكلام المزيف. كما تمت دراسة عدة استراتيجيات تجميعية للاستفادة من نقاط القوة التكاملية للنماذج الفردية. بالإضافة إلى ذلك، استُخدمت تقنية الذكاء الاصطناعي القابل للتفسير Gradient Saliency لتعزيز قابلية تفسير النماذج وتوفير فهم أعمق للخصائص التي تؤثر في قرارات الكشف.

تم تقييم الإطار المقترح باستخدام مجموعة بياناتASVspoof 2019 Logical Access ، كما تم اختباره على مجموعة بيانات DeepVoice لدراسة قدرته على التعميم. وقد أظهرت النتائج التجريبية فعالية التمثيلات الصوتية المعتمدة على التعلم الذاتي الإشراف في كشف التزييفات الصوتية العميقة. ومن بين جميع الأساليب التي تم تقييمها، حققت طريقة التجميع بالتكديس (Stacking Ensemble)أفضل أداء، حيث بلغت الدقة 99.15% (Accuracy) ، ودرجة F1 مقدارها 99.14%، ومعدل الخطأ المتساوي (EER) بنسبة 0.38% على مجموعة بيانات ASVspoof 2019 LA.كما حققت تجارب التعميم بين مجموعات البيانات على DeepVoice دقة بلغت 81.25% ودرجة F1 بلغت 88.89%، مما يؤكد متانة النهج المقترح وقدرته على التعميم. وتُظهر هذه النتائج أن الجمع بين التمثيلات الصوتية القائمة على التعلم الذاتي الإشراف والتعلم التجميعي يوفر حلاً فعالاً ومتيناً وقابلاً للتفسير للكشف عن التزييفات الصوتية العميقة.

الكلمات المفتاحية:

كشف التزييفات الصوتية العميقة، التعلم الذاتي الإشراف، Wav2Vec 2.0، HuBERT، WavLM، الذكاء الاصطناعي القابل للتفسير، Gradient Saliency، ASVspoof 2019، DeepVoice، التعلم التجميعي.

Contents

Acknowledgements i

Dedication -Nedjoua ii

Dedication -Aya iii

Abstract iv

Résumé v

Contents vi

List of Figures x

List of Tables xii

General Introduction 1

1 Fundamentals of Speech Processing 4
- 1.1 Introduction . . . 4
- 1.2 Speech production . . . 4
  - 1.2.1 Human Speech Production and Perception . . . 4
  - 1.2.2 The speech-production process . . . 6
- 1.3 Properties of speech signals . . . 7
  - 1.3.1 Physical properties . . . 8
    - 1.3.1.1 Frequency (the pitch of a sound) . . . 8
    - 1.3.1.2 Amplitude (the loudness of a sound) . . . 8
  - 1.3.2 Linguistic properties . . . 9
- 1.4 Time-domain and frequency-domain representation . . . 10
  - 1.4.1 Time-domain representation . . . 10
  - 1.4.2 Frequency-domain representation . . . 11
  - 1.4.3 Conversion between time domain and frequency domain . . . 12
- 1.5 Speech signal preprocessing . . . 12
  - 1.5.1 Pre-emphasis . . . 12
  - 1.5.2 Framing and windowing . . . 13
  - 1.5.3 Noise removal . . . 14

1.5.4 Normalization . . . 14
1.5.4.1 Cepstral mean normalization (CMN) . . . 14
1.5.4.2 Cepstral variance normalization (CVN) . . . 15
1.5.4.3 Cepstral mean and variance normalization (CMVN) . . . 15
1.6 Feature extraction techniques . . . 15
1.6.1 Time-frequency representations . . . 16
1.6.1.1 Time–frequency representation using the Short-Time Fourier (STFT) 16
1.6.1.2 Spectrograms . . . 16
1.6.2 Mel frequency cepstral coefficients (MFCCs) . . . 17
1.6.3 Linear prediction cepstral coefficients (LPCCs) . . . 18
1.6.4 Perceptual linear prediction (PLP) cepstral coefficients . . . 19
1.6.5 Relative-spectral PLP (RASTA-PLP) feature . . . 20
1.7 Conclusion . . . 23

2 State of the Art in Audio Deepfake Detection 24
2.1 Introduction . . . 24
2.2 Audio deepfake . . . 24
2.3 Audio deepfake generation methodologies . . . 26
2.3.1 Text-to-speech (TTS) . . . 26
2.3.2 Voice conversion (VC) . . . 26
2.3.3 Comparison between text-to-speech and voice conversion . . . 28
2.4 Datasets for audio deepfake detection . . . 28
2.4.1 Public benchmark datasets . . . 28
2.4.1.1 ASVspoof challenge . . . 28
2.4.1.2 DEEP-VOICE: deepFake voice recognition . . . 29
2.4.2 Dataset limitations . . . 29
2.5 Audio deepfake detection approaches . . . 30
2.5.1 Traditional machine learning approaches . . . 30
2.5.1.1 Handcrafted feature extraction . . . 30
2.5.1.2 Classical machine learning classifiers . . . 32
2.5.2 Deep learning-based approaches . . . 34
2.5.2.1 Convolutional neural network (CNN)–based methods . . . 34
2.5.2.2 Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) methods . . . 35
2.5.2.3 Transformer-based architectures . . . 37
2.6 Self-supervised learning for audio deepfake detection . . . 41
2.6.1 Concept of self-supervised learning for speech . . . 41
2.6.2 Self-supervised speech representation models . . . 43
2.6.2.1 Wav2Vec 2.0 . . . 43
2.6.2.2 HuBERT . . . 46
2.6.2.3 WavLM . . . 49
2.6.3 SSL-based deepfake detection frameworks . . . 52
2.7 Transfer learning with SSL models . . . 54
2.8 Evaluation metrics . . . 54
2.8.1 Confusion matrix . . . 55

2.8.2 Accuracy . . . 55
2.8.3 Precision . . . 55
2.8.4 Recall . . . 56
2.8.5 F1-score . . . 56
2.8.6 Equal Error Rate (EER) . . . 56
2.9 Comparative analysis and limitations of existing approaches . . . 57
2.10 Explainable artificial intelligence for audio deepfake detection . . . 58
2.10.1 Introduction to explainable AI . . . 58
2.10.2 Local interpretable model-agnostic explanations (LIME) . . . 59
2.10.3 Shapley additive explanations (SHAP) . . . 59
2.10.4 Gradient-Based Saliency Methods . . . 60
2.10.5 Comparative Analysis . . . 60
2.11 Conclusion . . . 61

3 Design of an Explainable Self-Supervised Ensemble Learning Framework for Audio Deepfake Detection 62
3.1 Introduction . . . 62
3.2 General presentation of the proposed framework . . . 63
3.3 Base models . . . 65
3.4 Ensemble methods . . . 68
3.4.1 Mean Soft Voting . . . 68
3.4.2 Weighted Soft Voting . . . 69
3.4.3 Logit Averaging . . . 70
3.4.4 Hard Majority Voting . . . 72
3.4.5 Max Confidence . . . 73
3.4.6 Bagging Soft Voting . . . 74
3.4.7 Stacking with Logistic Regression . . . 77
3.5 Single-Audio Inference Pipeline . . . 78
3.6 Conclusion . . . 79

4 Results and Discussion 81
4.1 Introduction . . . 81
4.2 Software and tools . . . 81
4.2.1 Google Colab . . . 81
4.2.2 Kaggle . . . 82
4.2.3 Hugging Face Transformers . . . 82
4.2.4 Python . . . 82
4.2.5 Google drive . . . 82
4.3 Dataset used . . . 83
4.4 Hyperparameters tuning . . . 84
4.5 Experimental results and discussion . . . 86
4.5.1 Wav2Vec2 . . . 87
4.5.1.1 Overall Evaluation Performance . . . 87
4.5.1.2 Confusion Matrix Analysis . . . 87
4.5.1.3 Generalization Analysis . . . 88

4.5.1.4 Attack-wise Analysis . . . 90
4.5.1.5 Discussion . . . 91
4.5.2 Hubert . . . 91
4.5.2.1 Overall Evaluation Performance . . . 91
4.5.2.2 Confusion Matrix Analysis . . . 91
4.5.2.3 Generalization Analysis . . . 92
4.5.2.4 Attack-wise Analysis . . . 93
4.5.2.5 Discussion . . . 95
4.5.3 WavLm . . . 95
4.5.3.1 Overall Evaluation Performance . . . 95
4.5.3.2 Confusion Matrix Analysis . . . 96
4.5.3.3 Generalization Analysis . . . 97
4.5.3.4 Attack-wise Analysis . . . 98
4.5.3.5 Discussion . . . 99
4.5.4 Evaluation of ensemble methods . . . 99
4.5.5 Ablation study . . . 102
4.5.6 Ensemble Methods versus Base Models . . . 104
4.5.7 Cross-Dataset Generalization . . . 105
4.5.8 Explainable AI Analysis . . . 107
4.5.8.1 Global Analysis of Saliency Maps . . . 107
4.5.8.2 Representative Example . . . 107
4.5.8.3 Discussion . . . 110
4.6 Comparison with State-of-the-Art Methods . . . 111
4.7 Conclusion . . . 112

General Conclusion **114**

Bibliography **116**

List of Figures

1.1 Schematic diagram of speech production[2]. . . . 5
1.2 Alternative view of speech production[2]. . . . 6
1.3 Schematic view of the human vocal mechanism[2]. . . . 7
1.4 High and low frequency sound waves[3]. . . . 8
1.5 Amplitude representation of soft and loud sounds[3]. . . . 9
1.6 A speech waveform with phonetic labels for the text message “Should we chase”[5]. 10
1.7 Time-domain representation[6]. . . . 11
1.8 Frequency-domain representation[6]. . . . 12
1.9 Diagram of framing[7]. . . . 13
1.10 Applying Hann windowing function on a signal sample[16]. . . . 14
1.11 Basic operation of feature extraction[8] 15
1.12 Illustration of the STFT calculation [9]. . . . 16
1.13 Spectrogram for speech waveform[16]. . . . 17
1.14 MFCC feature extraction process[8]. . . . 18
1.15 Linear predictive coding (LPC) feature extraction process[8]. . . . 19
1.16 PLP feature extraction process[8]. . . . 20
1.17 RASTA-PLP feature extraction process[8]. . . . 21
1.18 MFCC, LPCC and PLP feature extraction steps[11]. . . . 22

2.1 Audio deepfake generation frameworks used in text-to speech [15]. . . . 26
2.2 Audio deepfake generation frameworks used in voice conversion[15]. . . . 27
2.3 Support vector machine algorithm [34]. . . . 33
2.4 Hierarchical hidden Markov model[37]. . . . 34
2.5 The pipeline of a Convolutional Neural Network.[39]. . . . 35
2.6 Simple RNN internal operation[38]. . . . 36
2.7 The inner structure of LSTM unit[38]. . . . 37
2.8 The architecture of the Transformer model[38]. . . . 38
2.9 (a) Scaled dot-product Attention, (b) Multi-head attention[38]. . . . 39
2.10 A schematic view of APC model for self-supervised learning [42]. . . . 42
2.11 Architecture of Wav2Vec 2.0 model. . . . 43
2.12 Self-supervised training process in Wav2Vec 2.0. . . . 45
2.13 Architecture of HuBERT model[45]. . . . 47
2.14 Self-Supervised Training Process in HuBERT 48
2.15 WavLM model architecture.[45] 50
2.16 SSL feature extraction. . . . 53

2.17 SSL embedding generation. . . . 53
2.18 SSL classification. . . . 54
2.19 Example of Confusion Matrix [56]. . . . 55
2.20 Explainable Artificial intelligence (XAI) Approach [62]. . . . 59

3.1 General schema of the proposed audio deepfake detection framework 64
3.2 Schemas of base models used for the ensemble learning: (a) Wav2Vec 2.0 model, (b) HuBERT model, and (c) WavLM model. . . . 66
3.3 Proposed soft voting ensemble method 68
3.4 Proposed Weighted Soft Voting ensemble method 70
3.5 Proposed Logit average ensemble method 71
3.6 Proposed Hard Voting Majority ensemble method 72
3.7 Proposed max confidence ensemble method 74
3.8 Proposed Bagging Soft Voting ensemble method 76
3.9 Proposed Stacking Logistic Regression ensemble method 78

4.1 Confusion matrix of Wav2Vec 2.0 model 88
4.2 Training and development loss curves of the Wav2Vec 2.0 model. . . . 89
4.3 Confusion matrix of HuBERT model. . . . 92
4.4 Training and development loss curves of the Hubert model. . . . 93
4.5 Confusion matrix of WavLm model. . . . 96
4.6 Training and development loss curves of the WavLM model. . . . 97
4.7 Confusion matrix for the stacking ensemble model using Logistic Regression as meta-learner. . . . 102
4.8 Example of the Inference Process and Final Prediction for an Input Audio Sample. 106
4.9 Gradient Saliency visualization for the bonafide(real) utterance LA_E_1005795. The red curve represents the attribution score, highlighting the temporal regions that contributed most strongly to the ensemble prediction. . . . 108
4.10 Gradient Saliency visualization for the spoof utterance LA_E_1000147. The red curve represents the attribution score, highlighting the temporal regions that contributed most strongly to the ensemble prediction. . . . 108
4.11 Gradient Saliency visualization for the misclassified spoof utterance LA_E_1026061. The true label is spoof(fake), while the ensemble prediction is bonafide(real) with a confidence score of 98.65%. The red curve represents the attribution score highlighting the temporal regions that contributed most strongly to the prediction. . . . 110

List of Tables

2.1 Comparison of speech synthesis and voice conversion techniques [12]. 28
2.2 Comparison of audio deepfake detection approaches. 57
2.3 Comparison of XAI Methods. 60

4.1 Number of samples after the balanced sampling strategy. 83
4.2 ASVspoof 2019 Logical Access (LA) Evaluation Set Attack Types [70]. 84
4.3 Description of the hyperparameters used in the proposed models. 85
4.4 Hyperparameters of Wav2Vec2, HuBERT, and WavLM. 86
4.5 Performance of the fine-tuned Wav2Vec2 model. 87
4.6 Generalization performance from the development set to the evaluation set of the Wav2Vec 2.0 model. 89
4.7 Best detected spoofing attacks. 90
4.8 Most difficult spoofing attacks. 90
4.9 Performance of the fine-tuned Hubert model. 91
4.10 Generalization performance from the development set to the evaluation set of the Hubert model. 93
4.11 Best detected spoofing attacks. 94
4.12 Most difficult spoofing attacks. 94
4.13 Performance of the fine-tuned WavLM model. 95
4.14 Generalization performance from the development set to the evaluation set of the WavLM model. 97
4.15 Well-detected spoofing attacks. 98
4.16 Challenging spoofing attacks. 98
4.17 Ensemble learning results on ASVspoof 2019 evaluation set. 100
4.18 Ablation study of the proposed ensemble using stacking logistic regression on the ASVspoof 2019 LA evaluation set. 103
4.19 Performance degradation relative to the full ensemble. 103
4.20 Performance comparison of base models and ensemble methods on the ASVspoof 2019 LA evaluation set. 104
4.21 Cross-dataset evaluation setup. 105
4.22 Cross-Dataset Performance Evaluation of the Proposed Stacking Ensemble (Logistic Regression) Across ASVspoof and DeepVoice Datasets. 105
4.23 Comparison of the proposed ensemble model with selected methods on the ASVspoof 2019 LA evaluation set. 112

General introduction

Speech has become one of the most fundamental forms of human communication, serving as a primary medium for exchanging information, expressing emotions, and interacting with digital systems. In recent years, the rapid advancement of artificial intelligence (AI), particularly deep learning (DL), has significantly transformed speech technologies. Modern speech synthesis systems are now capable of generating highly natural and human-like speech, enabling a wide range of applications including virtual assistants, accessibility tools, customer service automation, and human-computer interaction.

Despite these remarkable achievements, the same technologies have facilitated the emergence of audio deepfakes, which are artificially generated or manipulated speech recordings designed to imitate genuine human voices with a high degree of realism. Advanced speech generation techniques, such as Text-to-Speech (TTS) and Voice Conversion (VC), can produce synthetic speech that is often indistinguishable from authentic recordings, creating serious challenges for security, privacy, and trust in digital communications.

The increasing accessibility of audio deepfake generation tools has raised significant concerns across multiple domains. Synthetic speech can be exploited for identity impersonation, financial fraud, social engineering attacks, misinformation campaigns, and unauthorized access to voice-based authentication systems. As voice technologies become increasingly integrated into modern applications, the ability to reliably distinguish genuine speech from manipulated audio has become a critical research problem.

Traditional audio deepfake detection methods primarily rely on handcrafted acoustic features, such as Mel-Frequency Cepstral Coefficients (MFCCs), Linear Prediction Cepstral Coefficients (LPCCs), and Perceptual Linear Prediction (PLP) features, combined with conventional machine learning algorithms. Although these approaches have demonstrated effectiveness under specific conditions, their performance often deteriorates when confronted with unseen spoofing attacks, diverse recording environments, and variations in data distributions.

The emergence of deep learning has considerably improved audio deepfake detection by enabling automatic feature extraction directly from raw speech signals. In particular, transformer-based architectures have demonstrated exceptional capabilities in modeling long-range dependencies and contextual information within speech. More recently, Self-Supervised Learning (SSL) has revolutionized speech representation learning by allowing models to learn meaningful and robust representations from large amounts of unlabeled audio data. State-of-the-art SSL models, including Wav2Vec 2.0, HuBERT, and WavLM, have achieved remarkable success across numerous speech processing tasks and have shown great potential for audio deepfake detection. Nevertheless, each SSL model exhibits distinct strengths and limitations depending on the characteristics of the spoofing attack, acoustic conditions, and dataset properties.

Consequently, relying on a single model may not always provide optimal performance. Ensemble Learning offers a promising solution by combining the predictions of multiple models to produce more accurate, stable, and robust decisions. By leveraging the complementary capabilities of different SSL models, ensemble methods can improve generalization, reduce prediction variance, and enhance resistance to previously unseen attacks.

In this work, we propose an audio deepfake detection framework based on Self-Supervised Learning and Ensemble Learning techniques. Three state-of-the-art SSL models, namely Wav2Vec 2.0, HuBERT, and WavLM, are employed as base classifiers. Several ensemble strategies are investigated, including Mean Soft Voting, Weighted Soft Voting, Logit Average, Hard Majority Voting, Max Confidence, Bagging Soft Voting, and Stacking with Logistic Regression. The proposed framework is evaluated using audio deepfake datasets to assess its effectiveness in improving detection performance compared to individual SSL models. Furthermore, an ablation study is conducted to analyze the contribution of each base model to the overall ensemble performance. In addition, explainable artificial intelligence (XAI) techniques are utilized to enhance the interpretability and transparency of the detection process.

By integrating powerful self-supervised speech representations with advanced ensemble learning strategies, this work aims to contribute to the development of accurate, robust, and trustworthy audio deepfake detection systems capable of addressing the growing challenges posed by synthetic speech technologies.

The main contributions of this work can be summarized as follows:

- Development of an audio deepfake detection framework that combines Self-Supervised Learning (SSL) and Ensemble Learning to improve detection robustness and reliability.
- Fine-tuning and evaluation of three state-of-the-art SSL speech representation models: Wav2Vec 2.0, HuBERT, and WavLM.
- Design and implementation of multiple ensemble learning strategies.
- Comprehensive experimental evaluation and comparison of individual SSL models and ensemble methods on the ASVspoof 2019 Logical Access dataset.
- Performance analysis through an ablation study to quantify the contribution of each base model to the overall ensemble performance.
- Investigation of cross-dataset generalization by evaluating the proposed approach on the DeepVoice dataset to assess its robustness against unseen data distributions.
- Integration of Explainable Artificial Intelligence (XAI) through Gradient Saliency analysis to improve model interpretability and provide insights into the decision-making process.

The rest of this master's thesis is organized as follows:

- **Chapter 1. Fundamentals of Speech Processing:** This chapter introduces the fundamental concepts of speech processing and feature extraction methods commonly employed in speech analysis.

- **Chapter 2. State of the Art in Audio Deepfake Detection:** This chapter reviews audio deepfake generation techniques and benchmark datasets as well as traditional and deep learning based detection approaches. It also presents self supervised learning models, evaluation metrics, and explainable artificial intelligence methods.
- **Chapter 3. Design of an Explainable Self-Supervised-Based Ensemble Learning Framework for Audio Deepfake Detection:** This chapter presents the proposed framework, and the architecture developed for audio deepfake detection.
- **Chapter 4. Results and Discussion:** This chapter describes the experimental setup and implementation details. It presents the performance evaluation of the individual and ensemble models as well as the ablation studies, cross-dataset generalization experiments, explainability analysis, and discussion of the obtained results.

Chapter 1

Fundamentals of Speech Processing

1.1 Introduction

Speech is one of the most important forms of human communication. It enables the transmission of information, emotions, and intentions through acoustic signals generated by the human vocal system. Speech processing focuses on analyzing, representing, and interpreting these signals for various applications such as speech recognition, speaker identification, and speech synthesis[1].

In this chapter, we present the fundamental concepts of speech processing, starting with an overview of speech production mechanisms and perception in human beings. We then examine the main properties of speech signals, including both physical properties such as frequency and amplitude, and linguistic properties such as phonemes, which represent the fundamental units of speech. Next, we explore different representations of speech signals in both time and frequency domains, as well as the conversion between them. Furthermore, we discuss essential speech signal preprocessing techniques, including pre-emphasis, framing, windowing, noise removal, and normalization methods. Finally, we introduce key feature extraction techniques widely used in speech analysis, such as Short-Time Fourier Transform (STFT), spectrograms, Mel Frequency Cepstral Coefficients (MFCCs), Linear Prediction Cepstral Coefficients (LPCCs), and Perceptual Linear Prediction (PLP).

1.2 Speech production

This section introduces the fundamental principles of speech production and perception, followed by a detailed description of the speech-production process.

1.2.1 Human Speech Production and Perception

The production (speech-generation) process begins when a speaker formulates a message (in his mind) that he wants to transmit to the listener via speech. The machine counterpart to the process of message formulation is the creation of printed text expressing the words of the message. The next step in the process is the conversion of the message into a language code. This roughly corresponds to converting the printed text of the message into a set of phoneme sequences corresponding to the sounds that make up the words, along with prosody markers denoting duration of

sounds, loudness of sounds, and pitch accent associated with the sounds. Once the language code is chosen, the talker must execute a series of neuromuscular commands to cause the vocal cords to vibrate when appropriate and to shape the vocal tract such that the proper sequence of speech sounds is created and spoken by the talker, thereby producing an acoustic signal as the final output. The neuromuscular commands must simultaneously control all aspects of articulatory motion including control of the lips, jaw, tongue, and velum (a "trapdoor" controlling the acoustic flow to the nasal mechanism)[2].

Once the speech signal is generated and propagated to the listener, the speech-perception (or speech-recognition) process begins. First the listener processes the acoustic signal along the basilar membrane in the inner ear, which provides a running spectrum analysis of the incoming signal. A neural transduction process converts the spectral signal at the output of the basilar membrane into activity signals on the auditory nerve, corresponding roughly to a feature extraction process. In a manner that is not well understood, the neural activity along the auditory nerve is converted into a language code at the higher centers. of processing within the brain, and finally message comprehension (understanding of meaning) is achieved[2].

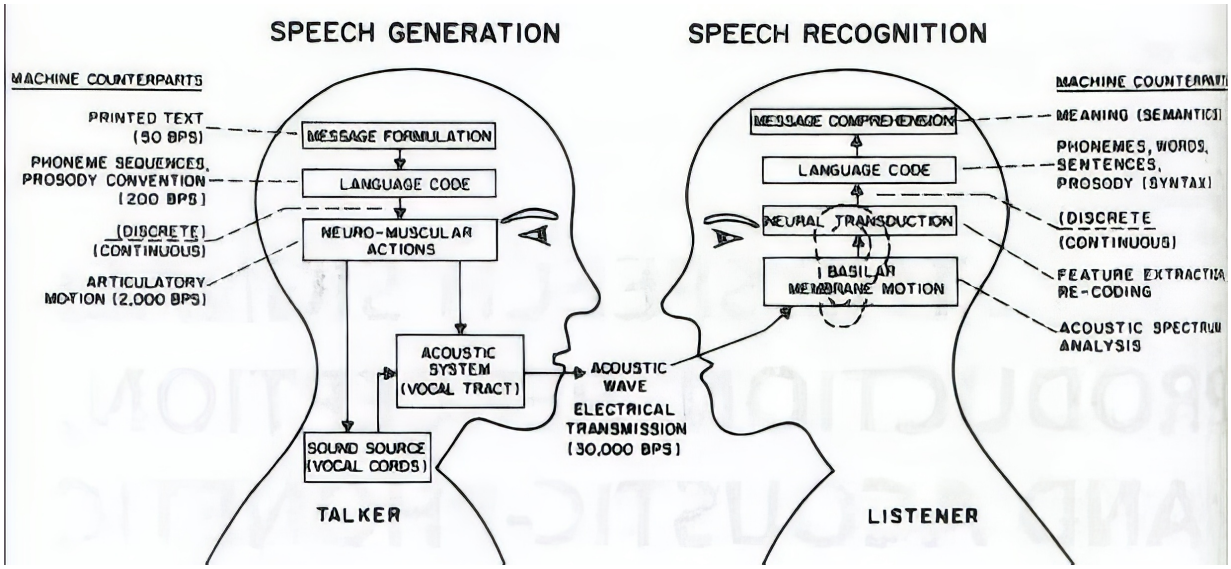


Figure 1.1: Schematic diagram of speech production[2].

A slightly different view of the speech-production/speech-perception process is shown in Figure 1.2. Here we see the steps in the process laid out along a line corre-sponding to the basic information rate of the signal (or control) at various stages of the process. The discrete symbol information rate in the raw message text is rather low (about 50 bps [bits per second]) corresponding to about 8 sounds per second, where each sound is one of about 50 distinct symbols). After the language code conversion, with the inclusion of prosody information, the information rate rises to about 200 bps. Somewhere in the next stage the representation of the information in the signal (or the control) becomes continuous with an equivalent rate of about 2000 bps at the neuromuscular control level, and about 30,000-50,000 bps at the acoustic signal level. A transmission channel is shown in Figure 1.2, indicating that any of several well-known coding techniques could be used to

transmit the acoustic waveform from the talker to the listener. The steps in the speech-perception mechanism can also be interpreted in terms of information rate in the signal or its control and follows the inverse pattern of the production process. Thus the continuous information rate at the basilar membrane is in the range of 30,000-50,000 bps, while at the neural transduction stage it is about 2000 bps. The higher-level processing within the brain converts the neural signals to a discrete representation, which ultimately is decoded into a low-bit-rate message[2].

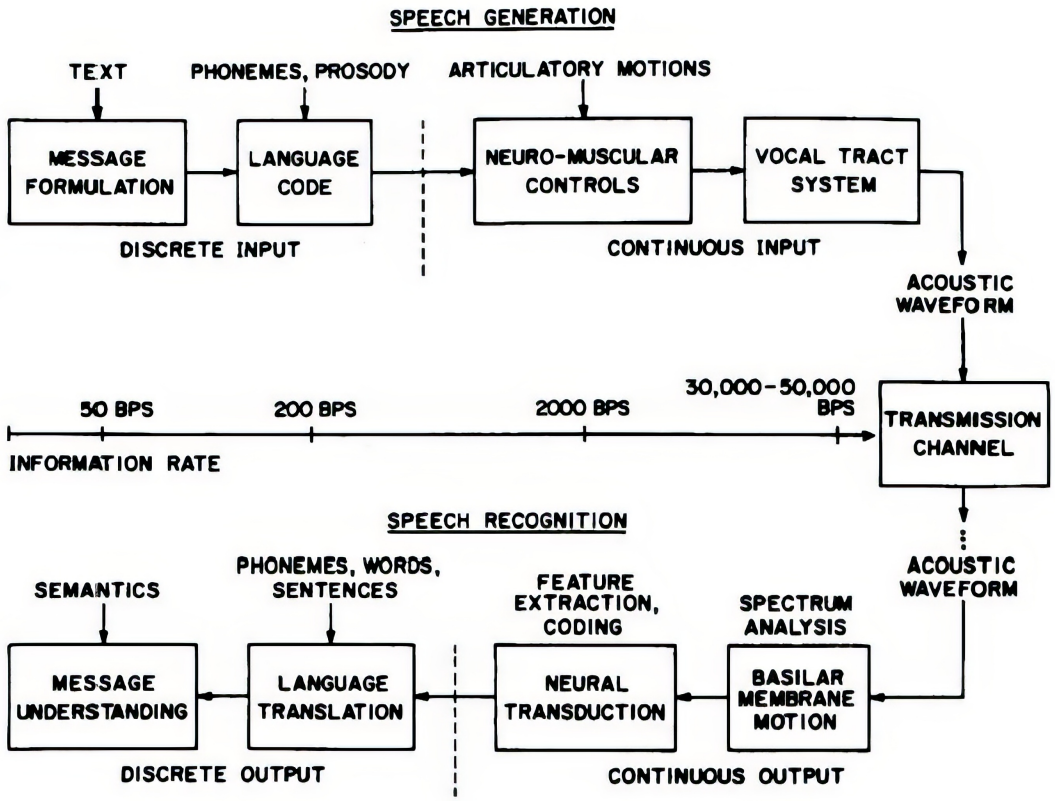


Figure 1.2: Alternative view of speech production[2].

1.2.2 The speech-production process

The vocal tract begins at the opening of the vocal cords, or glottis, and ends at the lips. The vocal tract consists of the pharynx (the connection from the esophagus to the mouth) and the mouth, or oral cavity. In the average male, the total length of the vocal tract is about 17 cm. The cross-sectional area of the vocal tract, determined by the positions of the tongue, lips, jaw, and velum, varies from zero (complete closure) to about 20 cm^2. The nasal tract begins at the velum and ends at the nostrils. When the velum, (a trapdoor-like mechanism at the back of the mouth cavity) is lowered, the nasal tract is acoustically coupled to the vocal tract to produce the nasal sounds of speech[2].

A schematic diagram of the human vocal mechanism is shown in Figure 1.3. Air enters the lungs via the normal breathing mechanism. As air is expelled from the lungs through the trachea (or windpipe), the tensed vocal cords within the larynx are caused to vibrate (in the mode of a relaxation oscillator) by the air flow. The air flow is chopped into quasi-periodic pulses which are then modulated in frequency in passing through the pharynx (the throat cavity), the mouth cavity, and possibly the nasal cavity. Depending on the positions of the various articulators (i.e., jaw, tongue, velum, lips, mouth), different sounds are produced[2].

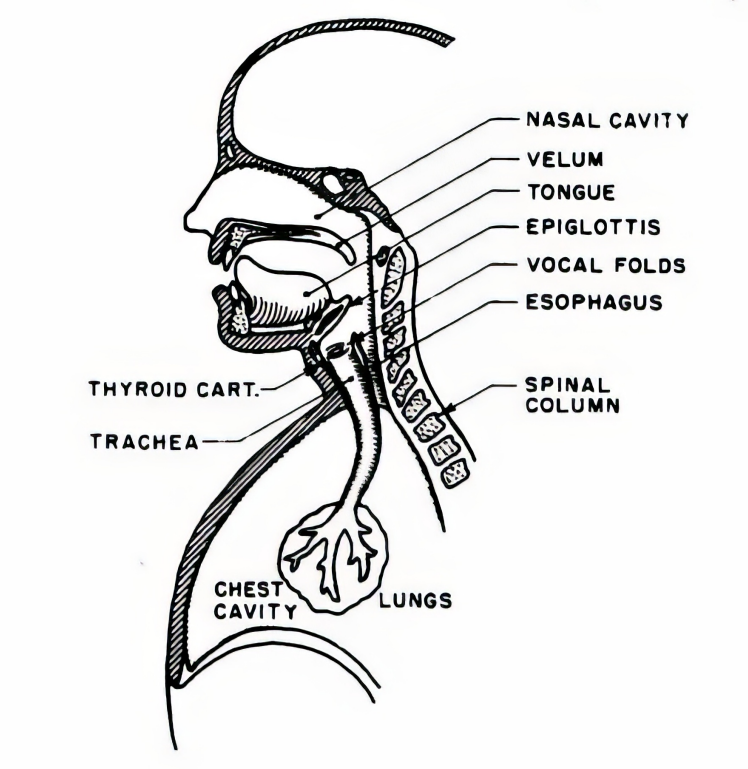


Figure 1.3: Schematic view of the human vocal mechanism[2].

The lungs and the associated muscles act as the source of air for exciting the vocal mechanism. The muscle force pushes air out of the lungs and through the bronchi and trachea. When the vocal cords are tensed, the air flow causes them to vibrate, producing sound[2].

1.3 Properties of speech signals

The speech signal is a complex sound wave produced when a person speaks. This wave changes with time; that is, its amplitude and frequency change over time. Frequency and amplitude are two essential properties of the speech signal and play a crucial role in human communication[3].

1.3.1 Physical properties

Speech signals possess many physical properties:

1.3.1.1 Frequency (the pitch of a sound)

The fundamental frequency of a speech signal, often denoted by F0, refers to the approximate frequency of the (quasi-)periodic structure of voiced speech signals. The oscillation originates from the vocal folds, which oscillate in the airflow when appropriately tensed. The fundamental frequency is defined as the average number of oscillations per second and expressed in Hertz. Since the oscillation originates from an organic structure, it is not exactly periodic but contains significant fluctuations. In particular, amount of variation in period length and amplitude are known respectively as jitter and shimmer. Moreover, the F0 is typically not stationary, but changes constantly within a sentence. In fact, the F0 can be used for expressive purposes to signify, for example, emphasis and questions[4].

Typically fundamental frequencies lie roughly in the range 80 to 450 Hz, where males have lower voices than females and children. The F0 of an individual speaker depends primarily on the length of the vocal folds, which is in turn correlated with overall body size. Cultural and stylistic aspects of speech naturally have also a large impact[4].

The fundamental frequency is closely related to pitch, which is defined as our perception of fundamental frequency. That is, the F0 describes the actual physical phenomenon, whereas pitch describes how our ears and brains interpret the signal, in terms of periodicity. The lowest remaining periodic component would be 500 Hz, which corresponds to the fifth harmonic of the original F0. However, a human listener would then typically still perceive a pitch of 100 Hz, even if it does not exist anymore. The brain somehow reconstructs the fundamental from the upper harmonics[4].

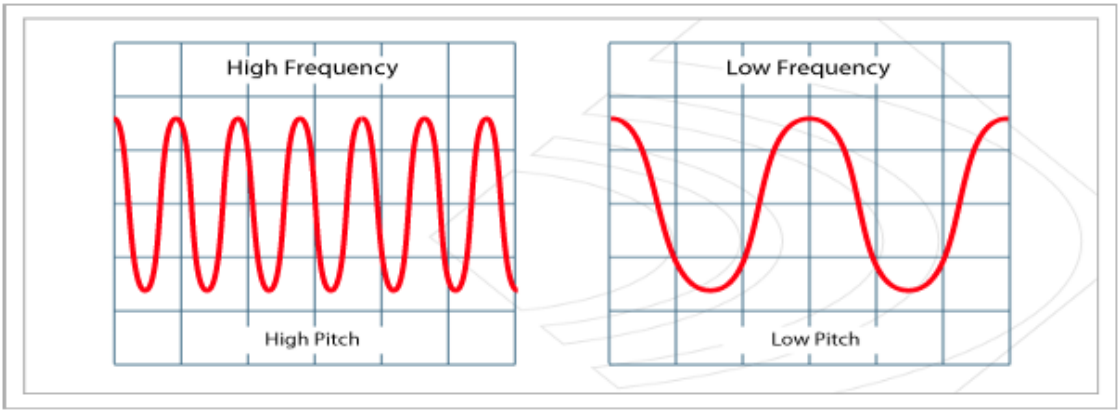


Figure 1.4: High and low frequency sound waves[3].

1.3.1.2 Amplitude (the loudness of a sound)

Amplitude in sound represents the intensity of the sound wave, as loud sounds have high amplitude while quiet sounds have low amplitude. Several factors affect amplitude, including the

distance between the speaker and the listener, the recording device used, and the acoustic potential exchanged[3].

Frequency and amplitude are independent properties of a sound wave. This means that a sound can have a high frequency (high pitch) and high amplitude (loud volume), a high frequency and low amplitude, and so on[3].

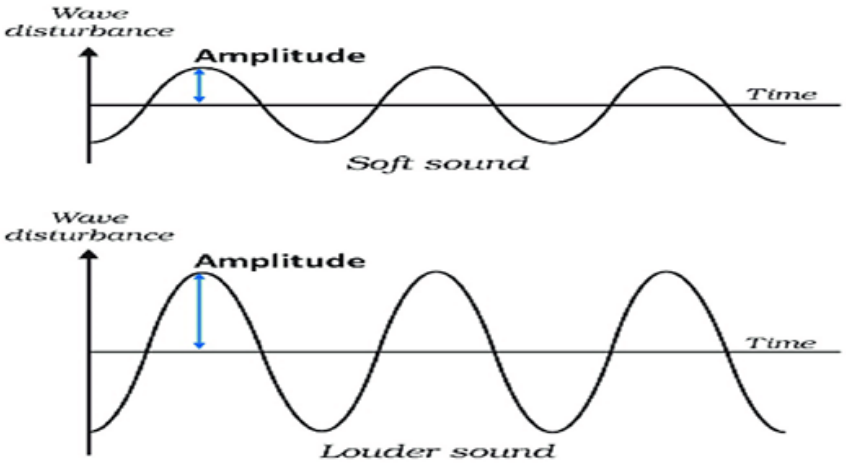


Figure 1.5: Amplitude representation of soft and loud sounds[3].

1.3.2 Linguistic properties

A phoneme is the smallest sound units in a language that distinguish one word from another. Speech can be represented phonetically by a finite set of symbols called the phonemes of the language, the number of which depends upon the language and the refinement of the analysis. For most languages the number of phonemes is between 32 and 64[5].

Figure 1.6 shows how the sounds corresponding to the text "should we chase" are encoded into a speech waveform. We see that, for the most part, phonemes have a distinctive appearance in the speech waveform. Thus, sounds like /SH/ and /S/ look like spectrally shaped random noise, while the vowel sounds /UH/, /IY/, and /EY/ are highly structured and quasi-periodic. These differences result from the distinctively different ways that these sounds are produced[5].

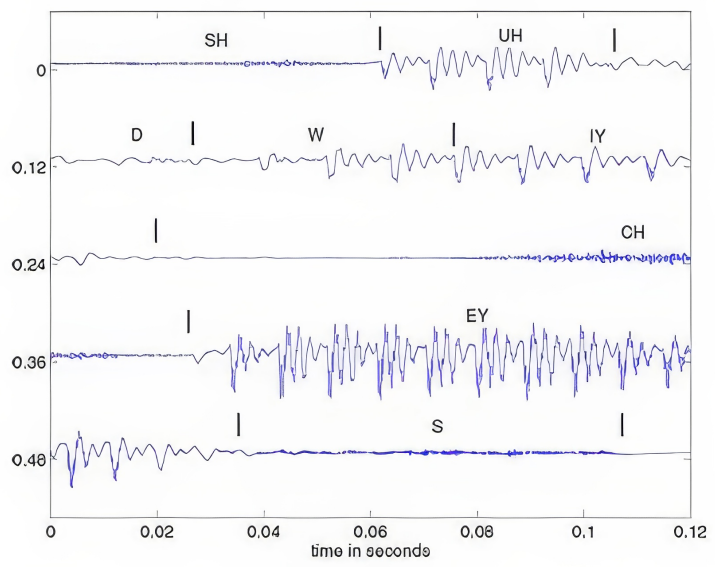


Figure 1.6: A speech waveform with phonetic labels for the text message "Should we chase"[5].

1.4 Time-domain and frequency-domain representation

When processing audio signals, there are two important things that must be understood: the time domain and the frequency domain. This is in order to analyze the signal, as they represent it in different ways: the time domain focuses on how the signal changes over time, while the frequency domain shows the components of the signal's frequency[6].

1.4.1 Time-domain representation

It is the domain in which all the signals are represented. Time domain signals can be tested or verified with the use of an oscilloscope. Signals in time domain are plotted with amplitude on the vertical axis and time on the horizontal axis [6].

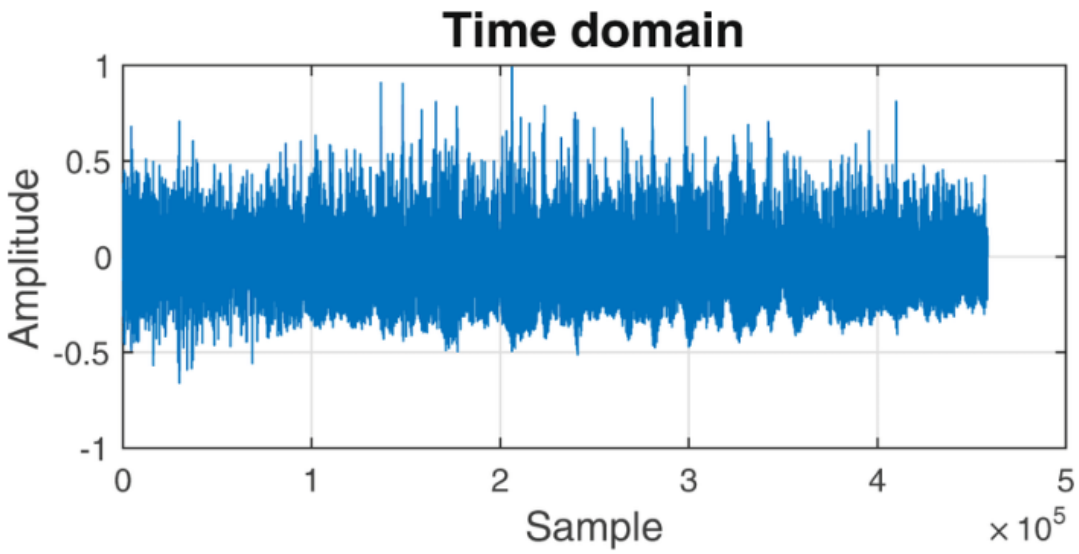


Figure 1.7: Time-domain representation[6].

The signal is displayed as a waveform showing its changes over time. The time domain is used to analyze signals and study their time-related behavior. Signals are represented by time functions f(t)[6].

1.4.2 Frequency-domain representation

It is useful to do more in-depth analysis of the time domain signal. Frequency domain helps study the frequency contents of the discrete time domain signals as well as the continuous time domain signal. The frequency domain signal can be analyzed with the use of a spectrum analyzer. This domain represents a signal by decomposing it into sinusoidal components of varying frequencies using techniques like the Fourier Transform. In the frequency domain, signals are represented by power (amplitude squared) on the Y-axis and frequency on the X-axis [6].

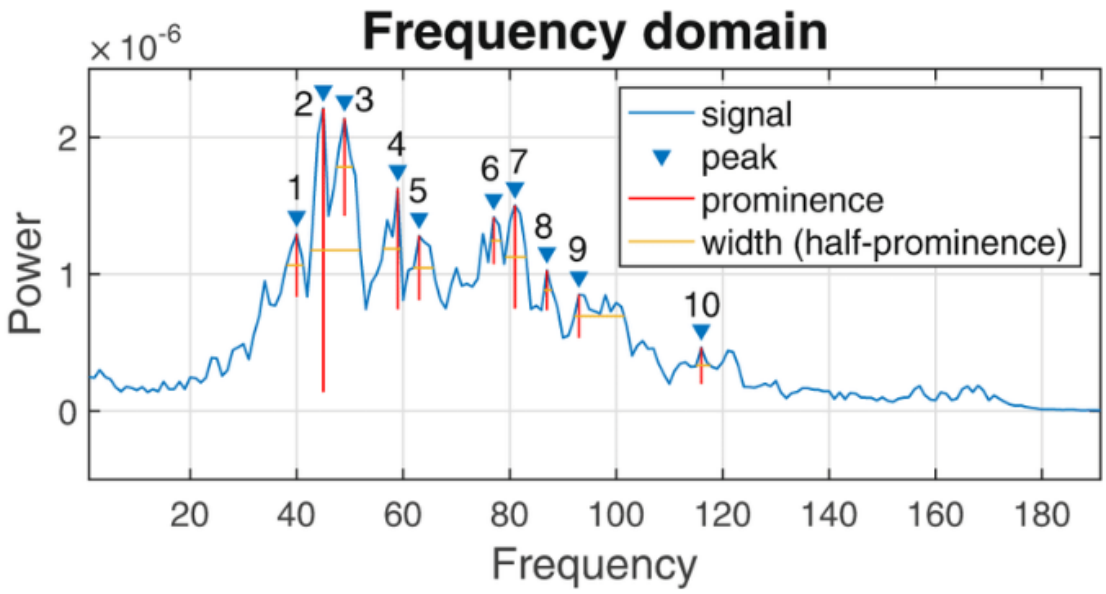


Figure 1.8: Frequency-domain representation[6].

In the frequency domain, a signal is displayed as a spectrum showing its amplitude versus frequency, and this is used in modulating and analyzing periodic signals. Signals are represented as functions of frequency, such as F(f). The frequency domain provides valuable insights into a signal's spectral characteristics, making it crucial for applications of audio processing. A time domain signal can be converted to a frequency domain signal with the use of Discrete Fourier Transform or Fast Fourier Transform (FFT)[6].

1.4.3 Conversion between time domain and frequency domain

The following two methods are very popular for said conversion[6]:

- **IFFT** - Converts Frequency Domain Vector to Time Domain Vector
- **DFT or FFT** - Converts Time Domain Vector to Frequency Domain Vector.

1.5 Speech signal preprocessing

Before speech processing for analysis or synthesis, there are some essential preprocessing steps that must be applied to the raw speech signal to enhance its quality and prepare it for feature extraction and further processing.

1.5.1 Pre-emphasis

The frequency components of a speech signal fall in the high frequencies, the pre-emphasis is applied to flatten the magnitude spectrum and balance the high and low-frequency components of the speech signal. On the other hand, pre emphasis filtering is used for reducing the high dynamic

range of speech waveform and enhancing the signal-to-noise ratio. The pre-emphasis preprocessing method is less used in speech recognition processing. However, this preprocessing technique increases the energy of the speech signal at high frequencies, which affects the consistency of the resulting spectrum among frames[7].

1.5.2 Framing and windowing

Since the speech signal has a non-stationary nature, the properties of a signal are not static over time, however, in short time intervals, the speech signal is considered as a stationary signal and easy to process. The Framing step fills this gap by splitting the continuous speech signal into a series of blocks called frames of 20-40ms length to allow block processing of the speech signal. The obtained frames have an equal length and are stationary over time, which makes extracting useful properties of the speech signal much easier[7].

Schematic diagram of framing is shown in the figure1.9 (Where N is the frame length and M is the frame shift)

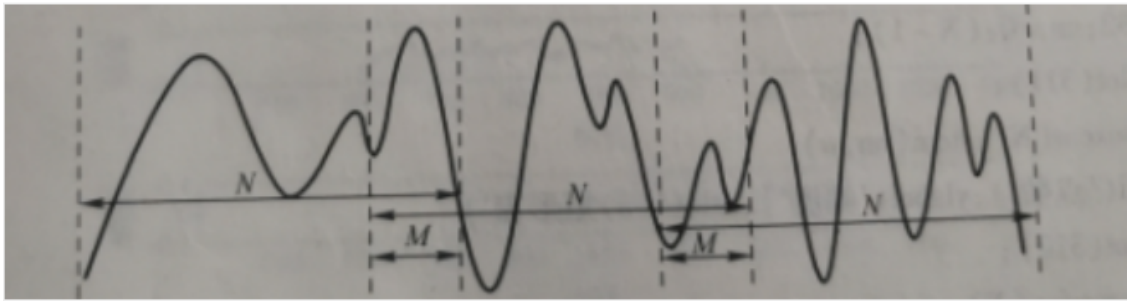


Figure 1.9: Diagram of framing[7].

Windowing is the analysis process of speech signals in each frame, which consists of multiplying a speech signal waveform by a time window function, to emphasize predefined properties of the speech signal. The concept of windowing is primarily used to help smooth the signal and avoid signal discontinuity produced by the spectral distortion during the framing stage, by reducing the signal at the beginning and end of each frame to zero. However, the use of windowing techniques sometimes has an impact on changing the signal at the beginning and end of the frame. Multiplying the speech wave by the window function has two main impacts. Firstly, to avoid a sharp change at the endpoints of a frame it gradually reduces the amplitude at both ends of the time interval. The second effect is to convolve the speech spectrum and the Fourier transform of the window function. Thus, the choice of the window function to be applied to reduce spectral distortion must satisfy two conditions: high-frequency resolution and low spectral leakage[7].

Windowing functions can be divided into two groups[7]:

- **Fixed window functions:** rectangular window, Hann window, Hamming window, Blackman window...

- **Adjustable window functions:** Kaiser window, Saramäki window, ultraspherical window...

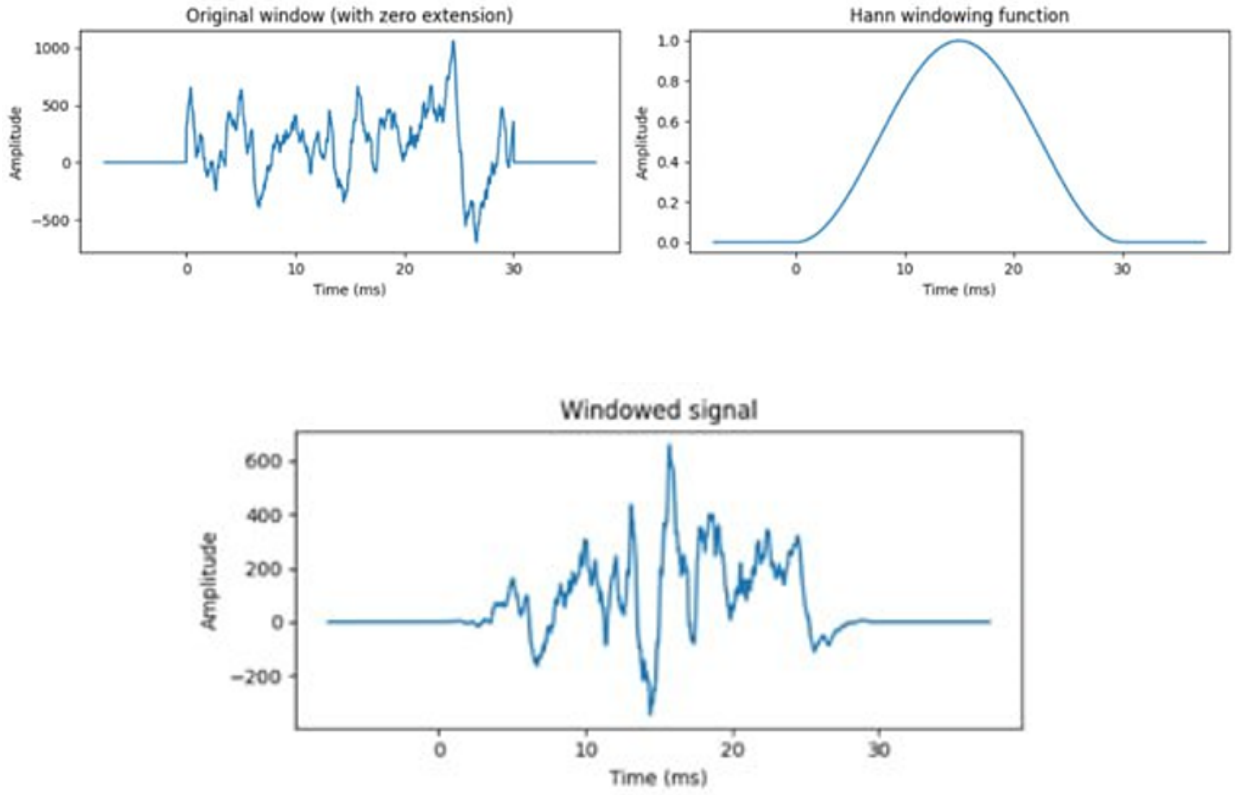


Figure 1.10: Applying Hann windowing function on a signal sample[16].

1.5.3 Noise removal

Environment noise plays an important role in speech preprocessing, denoising a speech signal improves the quality of speech and enhances the robustness of speech recognition systems. Noise removal is the process of cleaning the noise from a mixed sound of speech and noise to keep only the clean speech[7].

1.5.4 Normalization

The normalization of a speech signal is the final step in speech signal preprocessing, which involves balancing the signal spectrum and transforming the signal data into a normalized form based on a threshold. Normalization is mainly used for reducing the noise impact, speech signal distortion, and channel distortion. In the normalization phase, SNR is improved and the signal spectral variation is normalized or eliminated There exist multiple normalization functions [7]:

1.5.4.1 Cepstral mean normalization (CMN)

CMN is the widely used technique for compensating the variability of the speech signal in the cepstral domain. This technique is considered the simplest to implement and the most used normalization technique for large vocabulary ASR applications. The main role of CMN is to remove

the convolutional distortions caused by the environmental characteristics of speech(recording device, noises ...), at the same time it reduces the effect of speech style. However, the CMN fails to remove convolutional distortions when their time constant approaches the length of the analysis window[7].

1.5.4.2 Cepstral variance normalization (CVN)

CVN is a similar technique to CMN, which consists of estimating the variance of each cepstral dimension and then normalizing it into a unity. CVN is not associated with any particular type of distortion. However, it does offer some robustness against speaker variability, additive noise, and acoustic channels[7].

1.5.4.3 Cepstral mean and variance normalization (CMVN)

CVN and CMN are often paired together as the cepstral mean and variance normalization. Therefore, CMVN benefits from both the sample mean and standard deviation, to normalize the cepstral sequence. The use of CMVN positively affects the accuracy and reduces the error rate when using a clean or multi-style acoustic model. Unlike CMN, CMVN shifts and scales the energy term[7].

1.6 Feature extraction techniques

In speech processing, the process of extracting important information from a speech signal and reducing noise and unwanted information is called feature extraction. The basic operations of feature extraction involve spectral analysis, parametric transformation, and statistical modeling. The output is a parameter vector. However, it is normal to lose useful information while removing unnecessary information. Feature extraction also involves converting the speech signal into digital form. This basic step of feature extraction is shown in Figure 1.11 [8].

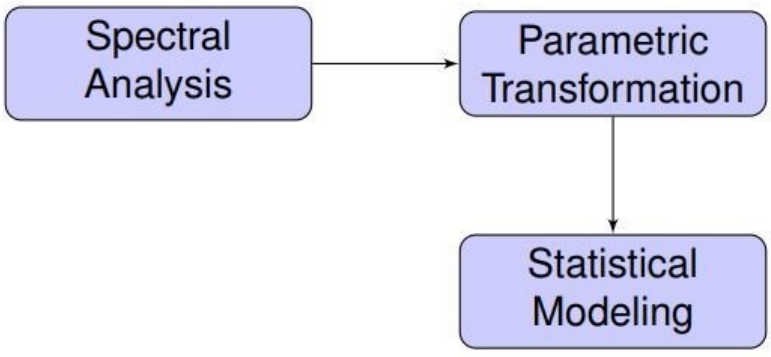


Figure 1.11: Basic operation of feature extraction[8]

.

Spectral Analysis is the first stage of speech analysis and it includes spectro temporal analysis of signal. In Parametric Transforms, two fundamental operations that differentiation and concatenation stage are applied to create signal parameters from signal measurements. Signal parameters were generated from few underlying multivariate random processes and this happen in Statistical Modeling stage[8].

1.6.1 Time-frequency representations

1.6.1.1 Time–frequency representation using the Short-Time Fourier (STFT)

The Fourier transform provides spectral content of a signal. It has been a valuable tool in various applications. However, for non-stationary signals the Fourier transform can not give satisfactory results since the information about frequency components variations in time is required[9].

An illustration of the STFT calculation is shown in Figure 1.18. Note that the spectral content is calculated for each windowed part of the signal. The central point of the sliding window is the time instant for which the spectrum is calculated. The energetic version of the STFT is known as the spectrogram. An important property of the STFT is its linearity. Namely, the STFT of a multicomponent signal

Consequently, if the signal components do not intersect in the time-frequency plane the spectrogram will be equal to the sum of spectrograms of each of the signal components[9].

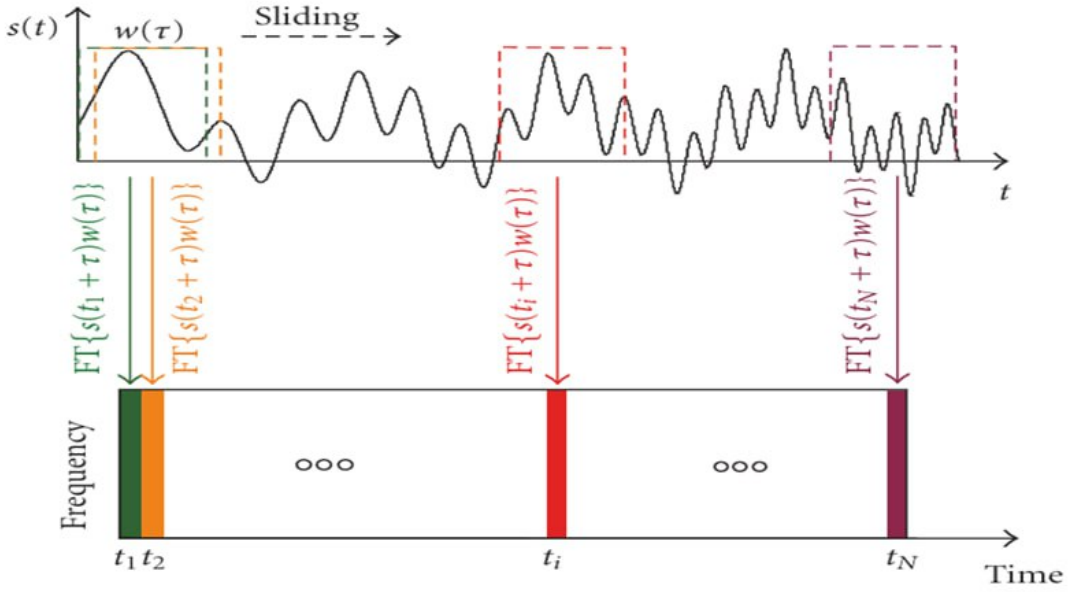


Figure 1.12: Illustration of the STFT calculation [9].

1.6.1.2 Spectrograms

Since the 1940s, the sound spectrogram has been a basic tool for gaining understanding of how the sounds of speech are produced and how phonetic information is encoded in the speech signal.

Today, spectrograms are made by DSP techniques and displayed as either pseudo-color or gray-scale images on computer screens. Sound spectrograms are simply a display of the magnitude of the STFT. The spectrogram represents a function of time and frequency and can be plotted as either a gray-scale or a color-mapped image[5]. A careful interpretation of the spectrogram images leads to valuable insight into the nature of the speech signal. First note that the window sequence w[m] is nonzero only over the interval $0 \leq m \leq L-1$. The length of the window has a major effect on the spectrogram image. If the analysis window is short, each individual pitch period is resolved in the time dimension, but the resolution in the frequency dimension is poor. For this reason, if the analysis window is short, the spectrogram is called a wide-band spectrogram. This is consistent with the linear filtering interpretation of the STFT, since a short analysis filter has a wide passband[5].

Conversely, when the window length is long, the spectrogram is a narrow-band spectrogram, which is characterized by good frequency resolution and poor time resolution. In this case, the fundamental frequency and its harmonics are resolved [5].

Periodic signals have Fourier spectra that are composed of impulses at the fundamental frequency f_0, and at integer multiples (harmonics) of the fundamental frequency. Multiplying by the analysis window in the time domain results in convolution of the Fourier transform of the window with the impulses in the spectrum of the periodic signal[5].

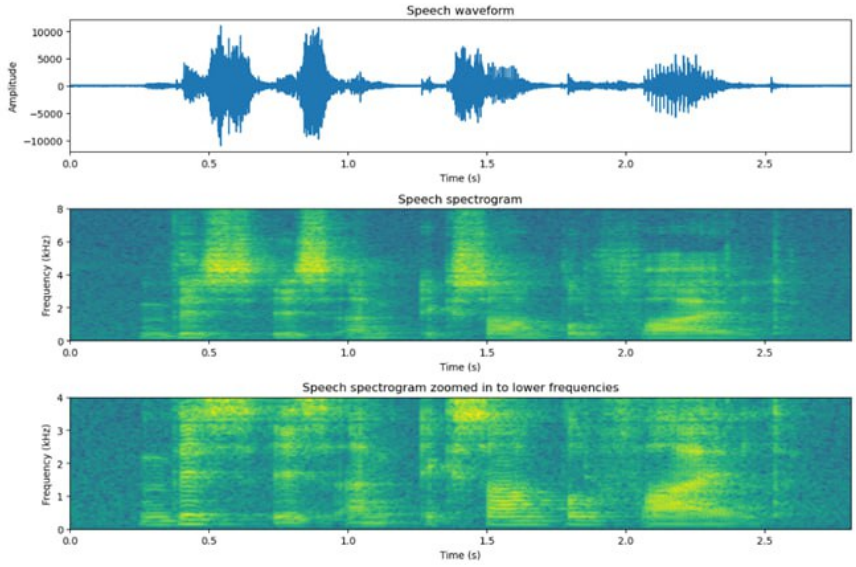


Figure 1.13: Spectrogram for speech waveform[16].

1.6.2 Mel frequency cepstral coefficients (MFCCs)

MFCCs are derived from the cepstral representation of an audio clip. MFCCs represent the short-time power spectrum of an audio clip based on the discrete cosine transform of the log power spectrum on a non-linear Mel scale. In MFCCs, the frequency bands are equally spaced on the mel scale, which mimics the human auditory system very closely, making MFCCs a key feature in various audio signal processing applications. MFCCs have been widely used in speech recognition,

speech enhancement, speaker recognition, music genre classification, music information retrieval, audio similarity measurement, vowel detection, etc [10].

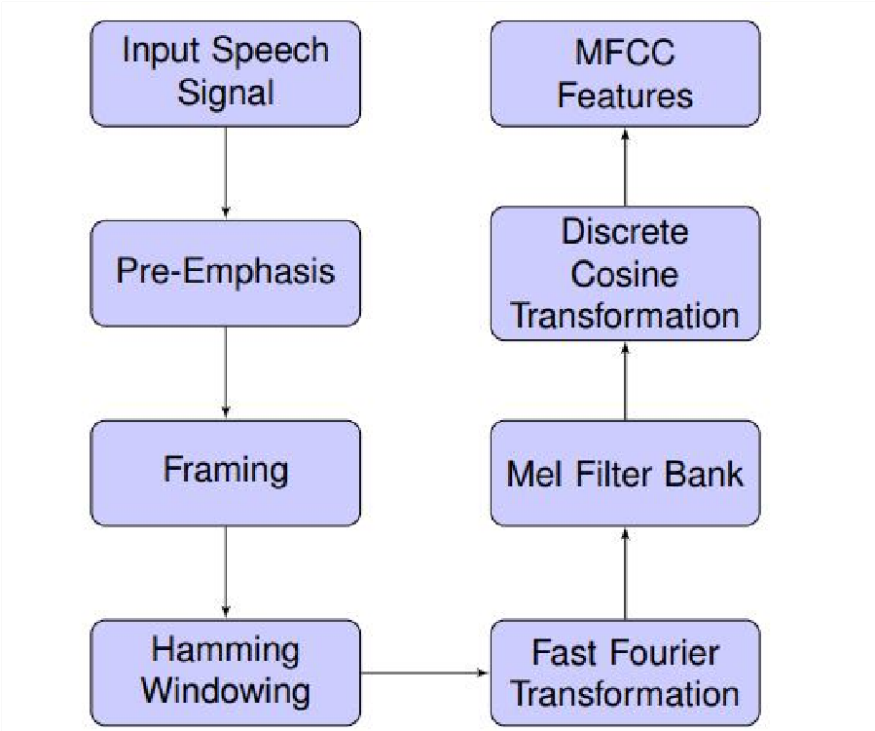


Figure 1.14: MFCC feature extraction process[8].

First input speech signal is divided into overlapping frames. Windowing is applied and then it is subjected to fast Fourier Transform. In the next step the frequency domain signal is converted to Mel frequency scale. Then the log Mel scale spectrum is converted to time domain using Discrete Cosine Transform (DCT) . The result of the conversion is called Mel Frequency Cepstrum Coefficient. MFCC mainly concentrates on the static characteristics of a signal[8].

1.6.3 Linear prediction cepstral coefficients (LPCCs)

The cepstrum has a number of advantages like source-filter separation, orthogonality, compactness etc. These properties make cepstrum coefficients robust and suitable for machine learning. On the other hand, linear prediction coefficients (LPC) are too sensitive to numerical precision, hence it is desirable to transform the LPC to the cepstral domain. The resultant transformed coefficients are called as LPCCs. We can say that LPCCs are derived from LPCs. LPCCs have been used in various application areas like speech recognition, speech analysis, noise removal, music genre classification etc [10].

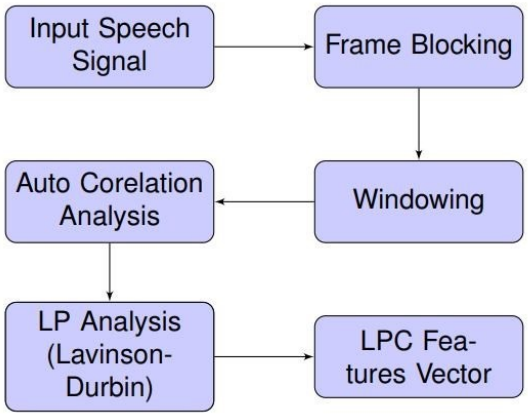


Figure 1.15: Linear predictive coding (LPC) feature extraction process[8].

LPC feature extraction process are shown in Figure 1.15 Adjacent frames in input speech signal are separated and is framed blocked into frames of samples. In order to minimize the signal discontinuities each individual frame is windowed. This is followed by auto correlating each frame of windowed signal and then it converts each frame of autocorrelations into LPC parameter set by using Durbins method. The LPC features vector was then created[8].

1.6.4 Perceptual linear prediction (PLP) cepstral coefficients

The PLP coefficients are based on the concepts: critical band spectral resolution, equal-loudness curve and intensity loudness power law. The PLP coefficients are derived from the linear prediction coefficients (LPC) by performing perceptual processing before auto-regressive modelling. After this processing, the linear coefficients are converted into cepstral coefficients. It has been widely employed in emotion recognition, speech recognition, baby crying sound analysis, animal sound vocalization analysis[10].

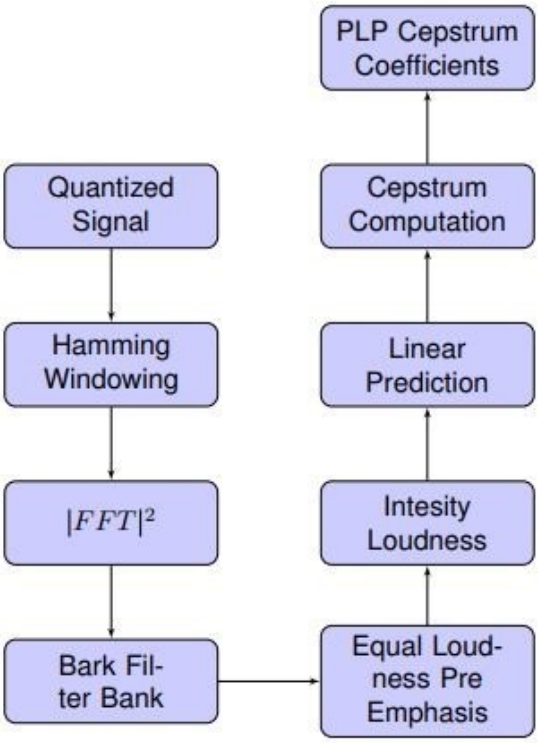


Figure 1.16: PLP feature extraction process[8].

First, the quantized signal is windowed. This is to limit the signal discontinuities. Then, hamming Window is utilized and the power range of the windowed signal is computed using the FFT. The three stages of recurrence distorting, smoothing and examining are incorporated into a solitary channel bank called Bark Filter Bank. To enhance the human auditory sensitivity an equal-loudness pre-emphasis is used to loads the channel bank yields. The yield that is the sound-related twisted line range is then prepared by the Linear Prediction to organize. The last advance is the calculation of the Cepstral Coefficients[8].

1.6.5 Relative-spectral PLP (RASTA-PLP) feature

RASTA is a technique that applies a band-pass filter to the energy in each frequency sub-band in order to smooth over short-term noise variations and to remove any constant offset resulting from static spectral coloration in the speech signal. The RASTA-PLP features are robust to noise version of PLP features. The advantage of the RASTA-PLP feature is that it tries to incorporate noise cancellation feature of human auditory system. This hybrid feature is widely employed in speech recognition, gender classification, speaker verification etc [10].

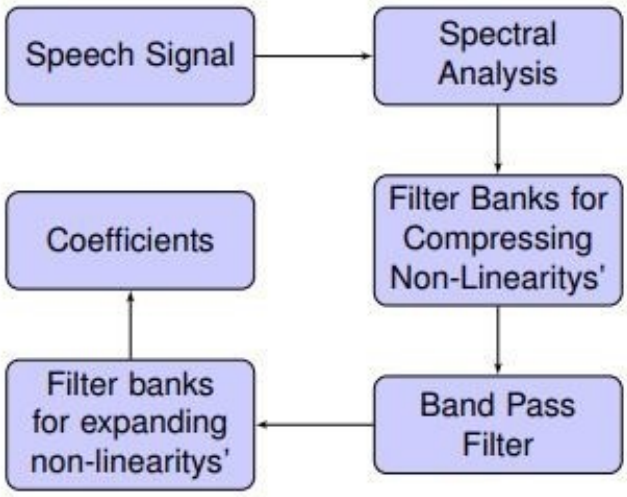


Figure 1.17: RASTA-PLP feature extraction process[8].

The input speech signal goes through the process of spectral analysis by using MFCC or PLP. This is then modified by the compressing the static non-linearity and will be filtered by band pass filter. Then another filter bank is used to expand the non-linearity and coefficients are produced[8].

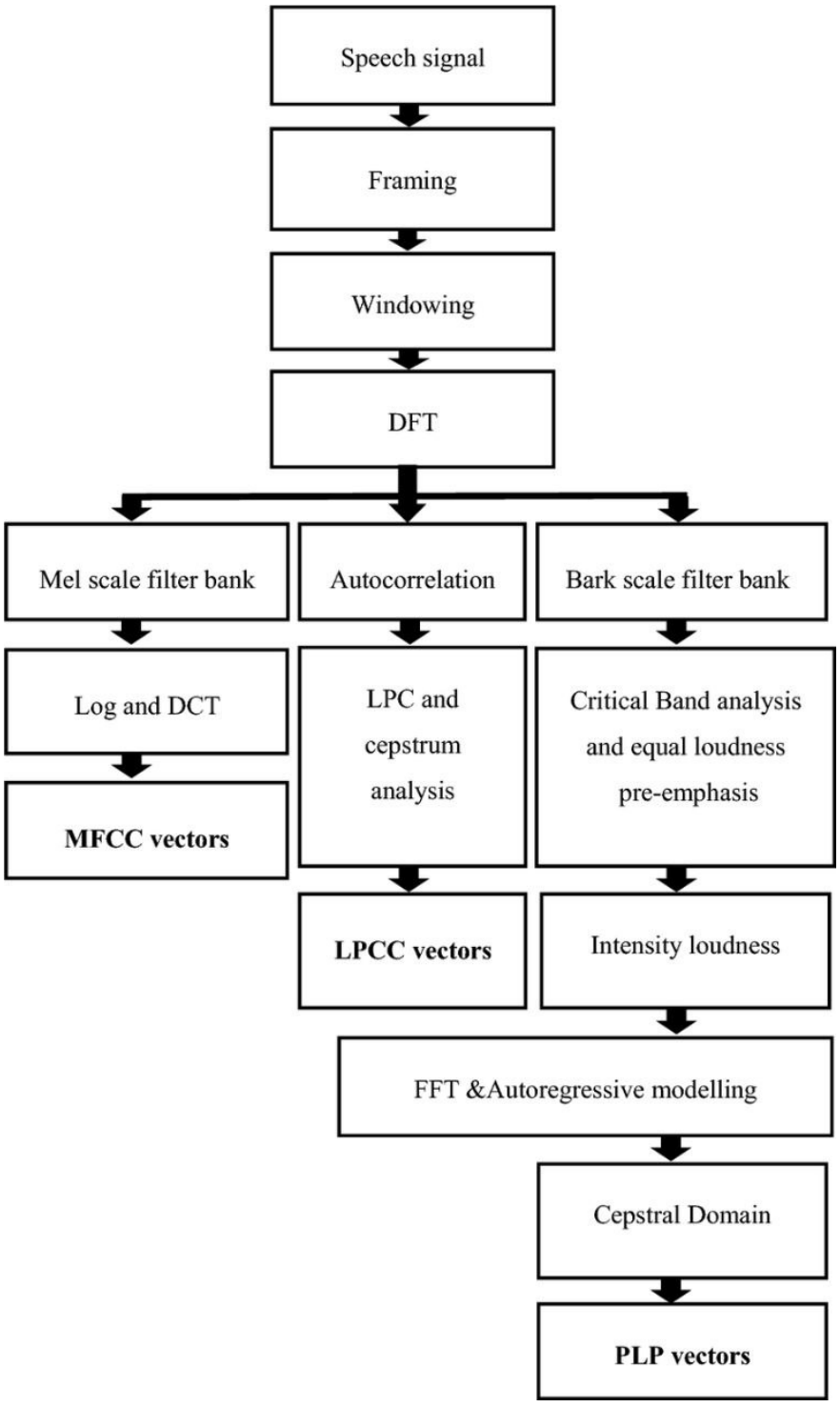


Figure 1.18: MFCC, LPCC and PLP feature extraction steps[11].

1.7 Conclusion

In this chapter, we discussed the fundamental concepts of speech processing. This includes the process of production of the speech signal and the representation of the same in both time and frequency domains. Since the process of speech can be considered a non-stationary process, appropriate preprocessing techniques should be applied to obtain the desired time-frequency representation. Subsequently, spectrograms are utilized to provide a visual representation.

In addition to that, we introduced feature extraction techniques such as MFCC, LPCC, PLP and RASTA-PLP as compact and discriminative representations of speech signals. These features are the foundation of modern speech analysis systems, which include speaker recognition, speech synthesis, and classification tasks.

The concepts discussed in this chapter provide the necessary foundation for understanding how audio deepfakes are both generated and detected. Many deepfake systems are built on advanced speech synthesis techniques that manipulate acoustic features, whereas detection approaches aim to identify subtle inconsistencies within those same features. Based on that, the next chapter will explore audio deepfakes in more details, including their methodologies, potential consequences, and the current state of the art in detection techniques.

Chapter 2

State of the Art in Audio Deepfake Detection

2.1 Introduction

Recently, advances in speech synthesis technologies have enabled the generation of audio deepfakes that closely resemble real human voices. These recordings can imitate specific individuals, including public figures, and may be exploited for misinformation, impersonation, or cybercrime. Audio deepfakes are typically generated using techniques such as text-to-speech (TTS) and voice conversion (VC). Although these technologies have significantly improved speech-based applications, they have also introduced serious security, ethical, and social challenges. Compared to image and video deepfake detection, audio deepfake detection has received relatively less research attention, although interest in this field has been growing in recent years[12].

This chapter begins by introducing the concept of deepfake technology and providing a clear definition of audio deepfakes, followed by a discussion of their risks and real-world implications. It then examines the main generation methodologies with a comparative analysis between them.

Furthermore, widely used datasets for training and evaluating detection systems are presented, highlighting their strengths and limitations. The chapter also reviews the principal audio deepfake detection approaches, starting with traditional machine learning methods based on handcrafted acoustic features and classical classifiers. Subsequently, modern deep learning approaches are explored, including transformer-based architectures. Particular attention is given to self-supervised learning (SSL) models, such as Wav2Vec 2.0, HuBERT, and WavLM, as well as transfer learning strategies that enhance detection performance.

In addition, the chapter introduces Explainable Artificial Intelligence (XAI) techniques, including Gradient-Based Saliency methods, which aim to improve the transparency and interpretability of deepfake detection systems.

Finally, the evaluation metrics used in this domain are presented, followed by a comparative analysis of existing methods.

2.2 Audio deepfake

Deepfake is content or material that is synthetically generated or manipulated using artificial intelligence (AI) methods, to be passed off as real. The term “deepfake” gained widespread attention in 2017 when a Reddit user under the same moniker shared a video featuring celebrity faces

generated using AI, which quickly went viral within the Reddit community [12]. While originally associated with visual media, the concept has since expanded to include other forms of synthetic content, including audio, text, and multimodal deepfakes.

Audio deepfakes are AI generated or edited/synthesized to create fake audio of a specific individual that seems real. It can be used for legitimate purposes, such as entertainment or virtual assistants, but can also be misused for misinformation, impersonation, fraud, or other malicious activities [12].

This dual-use nature makes audio deepfakes particularly vulnerable to misuse. One of the most significant risks associated with this technology is its potential for misinformation and impersonation. By replicating the voices of well-known personalities, including political figures, attackers can create convincing fake recordings that may influence public opinion or manipulate political processes [12]. For example, recent reports indicate that synthetic audio mimicking the voice of the U.S. president Joe Biden was used to discourage voters from participating in the 2024 New Hampshire primary election [13]. Similarly, audio deepfakes have been used in cybercrime scenarios where fraudsters impersonate company executives to deceive employees into transferring large sums of money [14]. These incidents illustrate the growing threat posed by synthetic speech in political, financial, and social contexts.

Beyond misinformation and fraud, the realism of audio deepfakes also introduces new attack vectors in modern technological systems. Many emerging applications rely on voice-based interfaces, particularly in wireless sensor networks, edge computing systems, and smart environments. In such systems, voice commands are often used to facilitate interaction between humans and devices. However, adversarial deepfake audio could potentially deceive these interfaces, resulting in unauthorized access or disruption of operations. Furthermore, systems that rely on embedded networked sensors—such as those used in industrial Internet of Things (IIoT) environments or healthcare monitoring—must be robust against manipulated audio signals to maintain accurate decision-making and anomaly detection processes [12].

2.3 Audio deepfake generation methodologies

Detecting audio deepfakes is really important since there have been some criminal activities using audio deepfakes in recent years. To achieve audio deepfake detection, one firstly needs to know the generation methods:

2.3.1 Text-to-speech (TTS)

Audio deepfake includes text-to-speech (TTS), which analyzes the text and makes the speech sound in line with text inputted using the rules of linguistic description of the text. An advantage of using text-to-speech is that it makes human-like speech from scratch and can be used for purposes like reading text and being a personal AI assistant, like Siri. Another benefit is that text-to-speech can offer different accents and voices instead of pre-recorded human voices. Text-to-speech takes text as an input; however, when a synthesis request is sent to TTS, a voice should be specified to speak the words; therefore, we can say that the SS-TTS models have been trained using audio samples of actual speeches. Besides the voice, some of the other aspects of the data output created by speech synthesis can be configured. TTS supports configuring the speaking rate, pitch, volume, and sample rate hertz[15].

There are various generative networks that can be used to perform TTS-based audio deepake generation.Figure 2.1 indicates the frameworks that are often used in text-to speech systems:

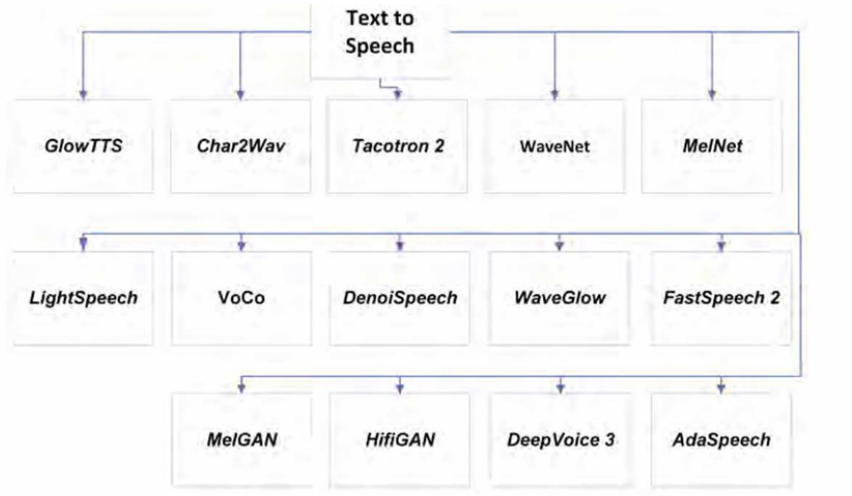


Figure 2.1: Audio deepfake generation frameworks used in text-to speech [15].

2.3.2 Voice conversion (VC)

Voice conversion takes the speech signal of a source speaker and modifies it to sound like that of a target speaker. It can provide flexible control of speaker identity in synthetic speech for text-to-speech (TTS) systems. Voice conversion can also help individuals with speech disorders in rehabilitation medicine. For example, speakers with dysarthria can transform their vowels into the vowel space of a speaker without dysarthria, improving speech intelligibility[15].

It can be used to personalize speaking and hearing aid devices, as well as in speech-to-speech translation. In education, voice conversion can generate prosodically correct utterances for pronunciation training in computer-assisted language learning. It is also used in entertainment to transform emotional or neutral speech and to create multi-singer voices. Impersonation, which can be considered a type of voice conversion, involves pretending to be another person for fraud or entertainment. It can reduce the need for voice actors in movies or audiobooks and enables rapid voice imitation using modern technology[15].

Some drawbacks of voice conversion include phonetic issues, prosody, quality, similarity, overfitting, and threats to speaker verification systems; therefore, anti-spoofing capabilities need to improve in normal speaker verification systems. Voice conversion can be used for fraud, such as the impersonation of the German CEO, which was the first known instance of a deepfake voice scam[15].

There are various generative networks that can be used to perform voice conversion-based audio deepake generation.Figure 2.2 indicates the frameworks that are often used in voice conversion systems:

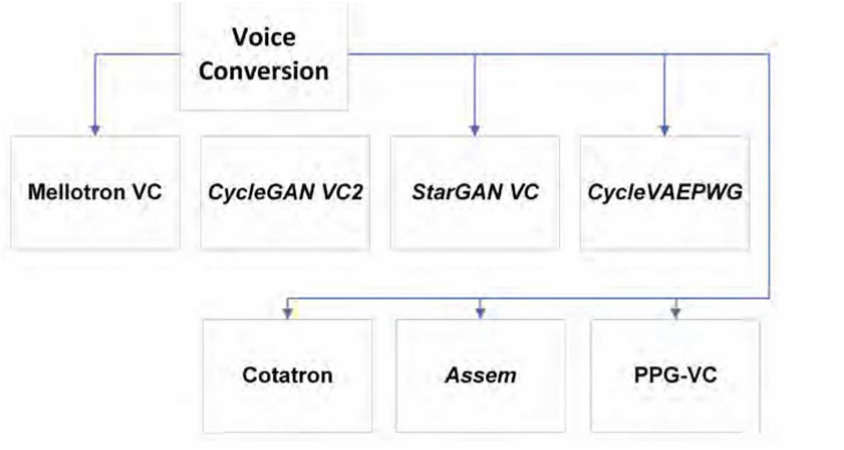


Figure 2.2: Audio deepfake generation frameworks used in voice conversion[15].

Voice conversion can be performed using both traditional and deep learning-based approaches. Traditional methods rely on statistical techniques to transform the characteristics of a source speaker's voice into those of a target speaker. More recently, deep learning has significantly improved voice conversion performance by enabling the generation of more natural and realistic speech[15].

When voice conversion is based on deep learning techniques, it is often considered a form of audio deepfake generation. Modern approaches frequently employ generative models to learn speaker-independent representations and produce high-quality converted speech, even when parallel training data are unavailable[15].

In addition, text-to-speech architectures can be incorporated into voice conversion frameworks. In such systems, the quality of the generated speech depends on factors such as the effectiveness of feature extraction, alignment accuracy, pitch modeling, and vocoder performance[15].

2.3.3 Comparison between text-to-speech and voice conversion

The table below summarizes the main differences between speech synthesis and voice conversion techniques, including their objectives, input and output types, system complexity, representative models, and key challenges[12].

Table 2.1: Comparison of speech synthesis and voice conversion techniques [12].

Attribute	Speech Synthesis	Voice Conversion
Objective	Generate speech from non-speech input	Transform input speech into target speech
Input	Text, phonemes, etc.	Source speech signal
Output	Speech signal	Speech signal with consistent content but altered style or attributes
Complexity	Language understanding and speech generation	Feature transformation and target reconstruction
Representative models	DeepVoice, Tacotron, FastSpeech	CycleGAN-VC, AutoVC, MulliVC, VQVC, FreeVC
Main challenges	Enhancing speech naturalness and fluency	Maintaining content consistency while changing target attributes

2.4 Datasets for audio deepfake detection

Several benchmark datasets have been developed to support the training, validation, and evaluation of audio deepfake detection systems. These datasets provide diverse spoofing attacks and realistic speech samples, enabling researchers to assess the robustness and generalization capabilities of detection models.

2.4.1 Public benchmark datasets

There are several widely used datasets for training and evaluating audio deepfake detection systems:

2.4.1.1 ASVspoof challenge

Automatic Speaker Verification Spoofing and Countermeasures Challenge (ASVspoof official website) stands as the largest and most comprehensive international competition in the field of spoofed speech detection. Its primary objective is to enhance the resilience of ASV systems against spoofing attacks. Since its beginning in 2015, ASVspoof has conducted multiple editions in 2015, 2017, 2019, 2021, and 2024 each contributing significantly to the advancement of spoofed speech detection technologies[12].

Previous research on spoofing attacks and countermeasures has relied on full knowledge of specific ASV systems for vulnerability assessments. Similarly, countermeasures were designed

with prior knowledge of the spoofing attacks they aimed to detect, including the type of attack and even the specific algorithms used. This approach, however, is inconsistent with real-world scenarios, where neither the exact nature of the attack nor its underlying algorithms are known in advance[12].

a) **ASVspoof 2019** introduced for the first time a distinction between two types of spoofing attacks: Logical Access (LA) and Physical Access (PA). LA involves directly injecting spoofed audio into the ASV system, utilising techniques such as TTS and VC. In contrast, PA deceives the ASV system by replaying spoofed audio through devices like loudspeakers. This dataset was constructed using the VCTK corpus, featuring recordings from 107 speakers. It incorporated state-of-the-art neural acoustic and wave form models at the time to generate spoofed audio and employed more controlled replay attack methods[12].

b) **ASVspoof 2021** built upon the ASVspoof 2019 framework by introducing, for the first time, a Speech Deepfake (DF) detection task aimed at identifying spoofed speech outside ASV scenarios. With no new training or development partitions provided, the challenge included three new progress and evaluation partitions for each task. The LA task incorporated the effects of telephony encoding and transmission, simulating more realistic communication scenarios. The PA task introduced reverberation and additive noise, challenging participants to address attacks under varying physical conditions. The DF task utilised additional undisclosed corpora and various lossy codecs to better simulate real-world media storage and transmission[12].

2.4.1.2 DEEP-VOICE: deepFake voice recognition

The DEEP-VOICE: DeepFake Voice Recognition dataset, available on Kaggle, is designed to support research in the detection of synthetic speech. It comprises both genuine human voice recordings and artificially generated counterparts produced using retrieval-based voice conversion (RVC) techniques. The dataset includes speech samples from multiple speakers, where original recordings are converted to mimic the vocal characteristics of different target speakers, thereby simulating realistic voice cloning scenarios. The data are organized into two primary classes, "REAL" and "FAKE", facilitating supervised learning tasks. In addition to raw audio files, the dataset provides a structured feature set extracted from one-second audio segments, with class balancing applied to ensure uniform representation. To enhance realism, background noise is reintroduced after the conversion process, reflecting real-world acoustic conditions. This makes the dataset particularly suitable for training and evaluating machine learning and deep learning models aimed at distinguishing authentic speech from deepfake audio in applications such as security, speaker verification, and misinformation detection[17].

2.4.2 Dataset limitations

A major limitation in audio deepfake detection is that existing benchmark datasets are often narrow in spoofing methods, recording conditions, and speaker diversity, which leads to poor generalization when models face new synthesis techniques or real-world environments,. Because labeled deepfake data are scarce and artificially constrained, purely supervised models tend to overfit to dataset-specific artifacts rather than learning robust, generalizable cues of manipulation.

Large pretrained models help mitigate these data limitations by providing rich, self-supervised representations learned from vast amounts of unlabeled speech, these representations capture broad acoustic and linguistic regularities, enabling downstream detectors to achieve strong performance even with limited or non-matching training data and to generalize better to unseen spoofing systems and “in-the-wild” scenarios[18].

2.5 Audio deepfake detection approaches

Various approaches have been proposed to distinguish genuine speech from deepfake audio. These methods range from traditional machine learning techniques based on handcrafted features to advanced deep learning models capable of learning discriminative representations directly from raw audio signals.

2.5.1 Traditional machine learning approaches

Early audio deepfake detection systems relied on manually designed acoustic features combined with conventional machine learning algorithms. The performance of these systems depends largely on the quality of the extracted features and the effectiveness of the classifier used.

2.5.1.1 Handcrafted feature extraction

This section is adapted from the paper "Audio deepfake detection: What has been achieved and what lies ahead" [12].

Handcrafted features rely on domain expertise in audio signal processing to manually design fixed feature representations derived from time, frequency, or cepstral domains. They have been widely used in traditional audio analysis tasks and remain relevant in audio deepfake detection due to their interpretability and relatively low computational requirements. In recent years, commonly used handcrafted features can be generally categorised into time–frequency domain features and cepstral domain features.

a) **Log power spectrum (LPS):** is derived from the STFT of an audio signal, followed by logarithmic transformation and power calculation. This feature effectively captures the time–frequency dynamics of audio signals while enhancing weak signals and compressing strong ones. This process results in a more intuitive and manageable spectral representation of audio data. Zhang et al.[19] employed LPS as a primary frontend feature and introduced a method involving per-band processing and multiband fusion. This approach significantly improved the robustness and generalisation ability of the detection system. Experimental results demonstrated that low-frequency features performed better in detect ing known attack types. These findings underscore the strong potential of LPS, particularly when combined with multiband fusion techniques.

b) **Linear filter bank (LFB):** a simplified version of STFT, offers reduced computational complexity and lower risks of overfitting during training. Chen et al. [20] employed augmented LFB features to train a deep residual network (ResNet), resulting in a model with strong

generalisation capabilities, underscoring the balance between computational efficiency and spectral representation provided by LFB. Building on the effectiveness of LFB, Tak et al. [21] used log-scale LFB with a ResNet for embedding extraction and employed a graph attention network (GAT) to model temporal and spectral relationships. This approach achieved SOTA performance in the ASVspoof 2019 LA dataset. In a comparative study, Wang and Yamagishi [22] evaluated several frontend features, including LFB. While LFB demonstrated strong performance in ResNet-based models, it was not the optimal choice for LCNN-based architectures.

c) **Mel spectrogram:** is a time–frequency domain feature representation based on the Mel frequency scale, which emulates the human auditory system's frequency perception. By mapping the spectrum onto the Mel scale, it captures the frequency characteristics of a signal while reducing high-frequency resolution to focus on frequency ranges sen sitive to human hearing. This property has made the Mel Spectrogram a mainstream feature in speech processing and audio analysis. Fathan et al. [23] explored the use of Mel Spectrograms combined withVGG16andWaveletCNNarchitectures, significantly out performing traditional cepstral features in detecting LA attacks. Similarly, Wani et al. [24] utilised an Attention-Based Cascaded Capsule Network (ABC-CapsNet) as a backend model, achieving exceptional results that surpassed other SOTA methods. These findings underscore the Mel Spectrogram's ability to retain critical frequency and temporal patterns, enhancing its utility in audio deepfake detection.

d) **Log power magnitude spectrogram (Spec):** is a logarithmic representation of the spectrogram obtained through STFT. By squaring the magnitude spectrum and apply ing a logarithmic transformation, Spec mimics the nonlinear perception of human hear ing, enhancing low-frequency signals while compressing high-frequency components. In the study conducted by Li et al. [25], Spec, along with several other acoustic features, was evaluated using the SE-Res2Net50 architecture. Experimental results indicated that Spec struggled with generalisation when facing synthetic speech. This limitation was at tributed to the absence of phase information in Spec features, as many spoofing cues for LA attacks are embedded in the phase domain. To address this issue, Ling et al. [26] proposed an attention-based convolutional neural network (CNN) to enhance Spec repre sentations. The attention mechanism ensured that the system emphasised critical regions of the spectrogram, leading to improved deepfake detection performance. The proposed model outperformed all single systems in the ASVspoof 2019 LA dataset, highlighting the effectiveness of Spec when combined with attention-based enhancements for capturing subtle spectral anomalies.

e) **Cepstral domain features:** extract key spectral characteristics of audio signals by analysing the logarithmic energy distribution of their spectra. Typically, the computation involves applying a logarithmic transformation to the spectrum, followed by an inverse Fourier transform or discrete cosine transform (DCT). This process effectively separates spectral envelope information from fine spectral details, making cepstral features particularly valuable in speech signal processing [27]. These features not only capture hidden artefacts in deeply forged signals but also offer a compact and meaningful representation.

f) **Mel-frequency cepstral coefficients (MFCC):** are a cepstral feature based on the Mel

frequency scale. By transforming spectral information into a compact cepstral representa tion, MFCC effectively capture the core spectral characteristics of speech. Zhang et al.[28] utilised MFCC as a frontend feature in their TE-ResNet framework, achieving near optimal detection performance. The robustness of MFCC-derived spectral features con tributed to the model's stable performance, making it competitive with SOTA methods. Hamza et al. [29] further investigated MFCC in various machine learning and deep learning frameworks, demonstrating its effectiveness in distinguishing genuine and fake audio. Their experiments revealed that while traditional machine learning models per formed well on clean datasets, deep learning models exhibited superior robustness in more complex data scenarios. Firc et al. [30] expanded the application of MFCC by in troducing the MFCC-Cepstrogram, a visualised cepstral feature representation used as input to Temporal Convolutional Networks (TCNs). This approach examined the impact of different spectrogram features on audio deepfake detection. Experimental results on the ASVspoof 2019 LA and FoRdatasets indicated that MFCC-Cepstrogram achieved excep tional performance and exhibited better stability on unseen data compared to other spectral features. This highlights MFCC-Cepstrogram's computational efficiency and adaptability, especially in resource-constrained scenarios. Moreover, the study demonstrated that en hancing MFCC features through visualisation can further improve detection performance.

g) **Linear frequency cepstral coefficients (LFCC):** similar to MFCC, LFCC utilise cepstral anal ysis but replace the Mel frequency scale with a linear frequency scale, enabling them to retain more detailed information in the high-frequency regions. This makes LFCC par ticularly suitable for detecting spectral distortions in deepfake audio. Zhang et al. [31] leveraged LFCC within a one-class learning framework combined with a ResNet for fea ture processing. Their experiments significantly outperformed traditional LFCC-GMM approaches, revealing LFCC's superior ability to capture high-frequency artefacts. Ad ditionally, the one-class learning framework demonstrated LFCC's potential in handling unknown attack types. Luo et al. [32] adopted LFCC as a frontend feature and proposed a backend framework based on capsule networks. The combination of capsule networks and LFCCsignificantly improved the generalisation ability to unknown attacks, highlighting its strong potential in deepfake audio detection. Furthermore, Wang et al. [33] introduced Multiscale Permutation Entropy (MPE) to complement LFCC by capturing the dynamic structural information lost during extraction. By directly integrating MPE with LFCC, they developed a more expressive joint feature representation. This study highlights the endur ing value of handcrafted features when integrated with novel methodologies, particularly in resource-constrained environments where large-scale pretraining is infeasible.

2.5.1.2 Classical machine learning classifiers

a) **Gaussian mixture model (GMM):** is a probabilistic model and an extension of a single Gaussian probability density function designed to represent a mixture of multiple Gaussian distributions. By treating data as a weighted combination of several Gaussian distributions, GMM can effectively capture complex data distributions. It has been widely used in audio deepfake detection, speaker verification, and other pattern recognition tasks. Although deep learning models have become the mainstream in recent years, GMM remains valuable in

specific tasks due to its efficiency and interpretability, and it is used as a standard baseline model in the ASVspoof Challenge and ADD Challenge[12].

b) **Support vector machine (SVM):** is a supervised machine learning algorithm capable of performing linear or nonlinear classification, regression, and outlier detection. The goal of the SVM algorithm is to learn how important each training data point is to represent the decision boundary (hyperplane) between the two classes and produces the largest minimum distance (margin) between the objects that belong to different classes. SVM can work on both classification tasks and regression tasks. In the classification task, SVM is trying to fit the largest possible street between two classes while limiting margin violations. Whereas in the regression task, SVM tries to fit as many instances as possible on the street while limiting margin violations and the width of the street is controlled by a hyperparameter[34].

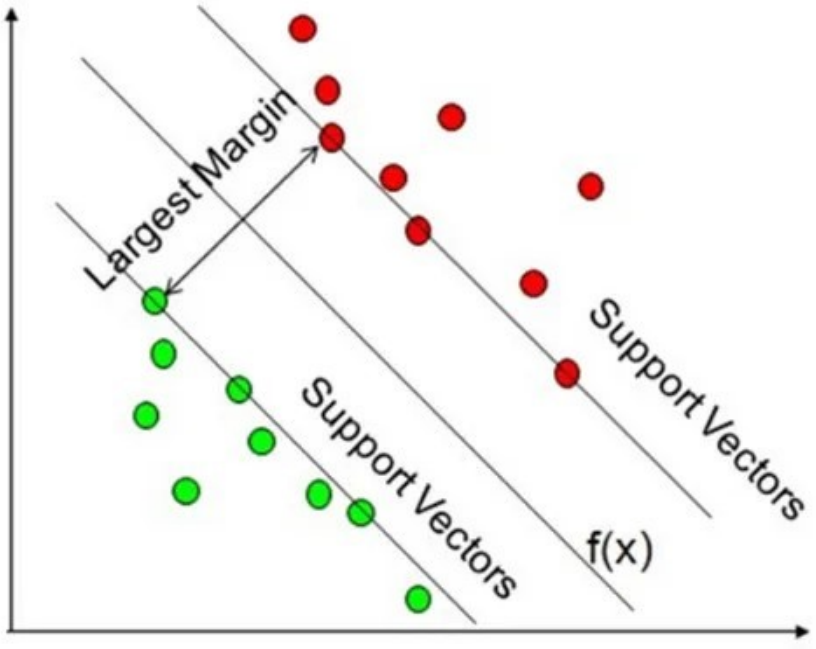


Figure 2.3: Support vector machine algorithm [34].

Support Vector Machines (SVMs) have become quite popular for many applications of pattern classification, mainly due to their great ability to generalize, often resulting in better performance than traditional techniques. such as artificial neural networks. They suffer however from the effort of computer time for training which scales at least quadratic with respect to the number of training vectors[35].

c) **Hidden Markov models (HMM):** are statistical models that work as a sequence of labeling problems. These are the types of problems that describe the evolution of observable events, which themselves, are dependent on internal factors that can't be directly observed they are hidden. An Hidden Markov Model is made of two distinct stochastic processes, meaning

those are processes that can be defined as sequences of random variables that depend on random events.There's an invisible process and an observable process. The invisible process is a Markov Chain, like chaining together multiple hidden states that are traversed over time in order to reach an outcome. This is a probabilistic process because all the parameters of the Markov Chain, as well as the score of each sequence, are in fact probabilities. Hidden Markov Models describe the evolution of observable events, which themselves, are dependent on internal factors that can't be directly observed they are hidden[36].

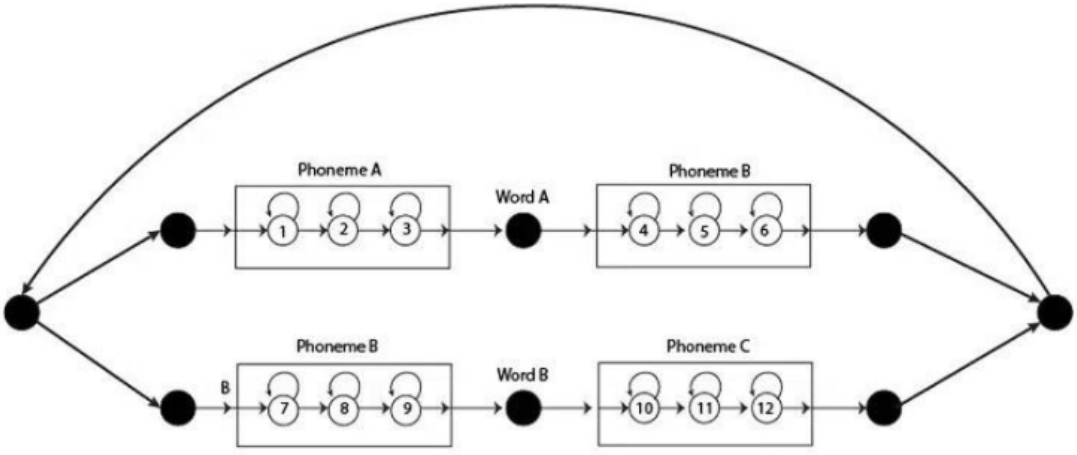


Figure 2.4: Hierarchical hidden Markov model[37].

2.5.2 Deep learning-based approaches

Deep learning has emerged as a powerful paradigm for audio deepfake detection by enabling the automatic extraction of relevant features directly from speech signals. This capability has led to substantial improvements in detection accuracy and generalization performance.

2.5.2.1 Convolutional neural network (CNN)–based methods

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 bio informatics. They have also demonstrated success in time series prediction tasks. CNNs are feed forward neural networks that leverage convolutional structures to extract features from data. 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. Unlike traditional methods, CNNs automatically learn and recognize features from the data without the need for manual feature extraction by humans. The design of CNNs is inspired by visual perception. The major components of CNNs include the convolutional layer, pooling layer, fully connected layer, and activation function. Figure 2.5, presents the pipeline of the convolutional neural network, highlighting how each layer contributes to the efficient processing and successful progression of input data through the network [38].

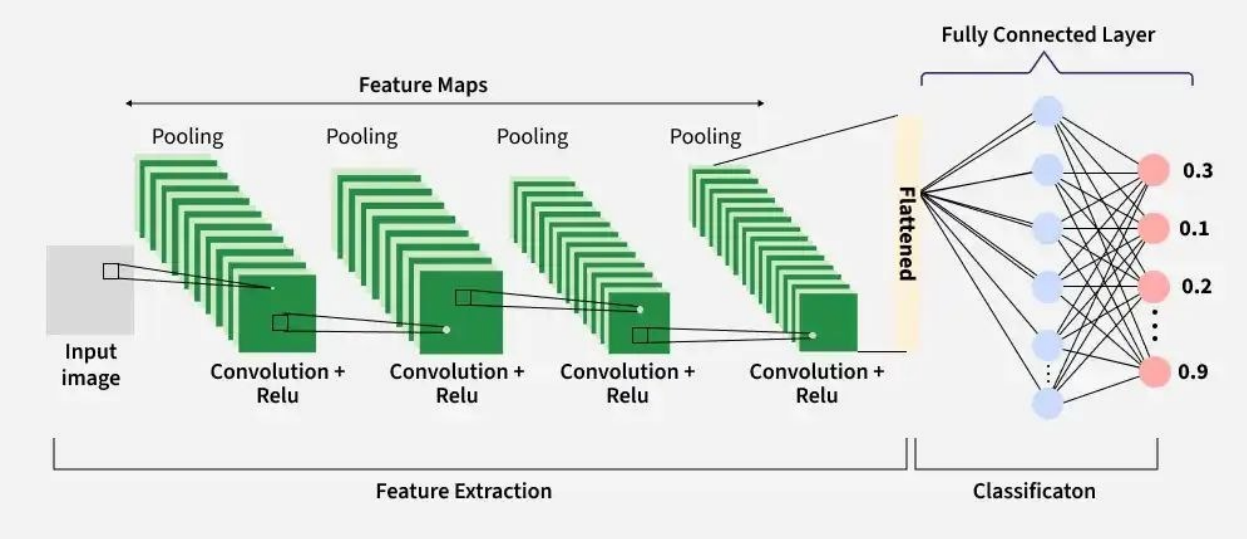


Figure 2.5: The pipeline of a Convolutional Neural Network.[39].

In the field of audio deepfake detection, CNNs are particularly adept at capturing localised artefacts with inspectro-temporal decompositions, such as spectrograms. Light CNN (LCNN) build supon traditional CNN architectures by introducing the Max-Feature-Map (MFM) activation function and smaller convolutional kernels, significantly reducing the number of parameters while maintaining a balance between utility and performance[12]. LCNN has been extensively used as a baseline model in challenges like ASVspoof and ADD.For instance,Wanget al. [33] employed LCNN as a classifier to process fused LFCC and MPE feature vectors for deepfake audio detection. Their results demonstrated that LCNN effectively captures temporal and frequency patterns with fused features, significantly enhancing detection accuracy and robustness. ResNet, with its innovative Residual Learning Framework, addresses challenges such as gradient vanishing and degradation issues in deep networks. Initially achieving significant breakthroughs in image classification tasks on ImageNet, ResNet has also proven effective in audio applications [12]. Zhang et al. [31] utilised a ResNet-18 architecture, replacing the standard global average pooling with attentive temporal pooling to better focus on time-dimensional features critical for deepfake detection.Combined with the newly proposed One-Class (OC) softmax loss function, this model further improved the differentiation between real and fake audio. Experimental results showed outstanding performance on unseen attacks, underscoring its strong generalisation capabilities[12].

2.5.2.2 Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) methods

Neural networks have long struggled with understanding sequences, whether it's a sentence in natural language or a time-series dataset. Before transformers became the go-to architecture, Recurrent Neural Networks(RNNs) and their improved versions, Long Short-Term Memory networks(LSTMs), were the dominant choices for handling sequential data[40].

a) **Recurrent neural networks (RNNs):** Recurrent Neural Networks (RNNs) are a class of deep learning models that possess internal memory, enabling them to capture sequential dependencies. Unlike traditional neural networks that treat inputs as independent entities,

RNNs consider the temporal order of inputs, making them suitable for tasks involving sequential information. By employing a loop, RNNs apply the same operation to each element in a series, with the current computation depending on both the current input and the previous computations. The ability of RNNs to utilize contextual information is particularly valuable in tasks such as natural language processing, video classification, and speech recognition. For example, in language modeling, understanding the preceding words in a sentence is crucial for predicting the next word. RNNs excel at capturing such dependencies due to their recurrent nature. However, a limitation of simple RNN is their short-term memory, which restricts their ability to retain information over long sequences.To overcome this, more advanced RNN variants have been developed, including Long Short-Term Memory (LSTM) , bidirectional LSTM , Gated Recurrent Unit (GRU) , bidirectional GRU , Bayesian RNN,and others[38].

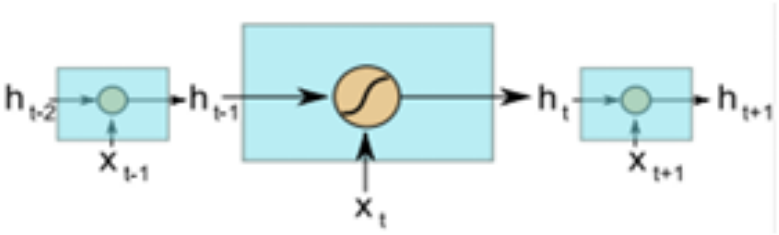


Figure 2.6: Simple RNN internal operation[38].

Limitations of RNNs: RNNs introduced a major improvement in sequential processing but had serious drawbacks[40]:

- **Vanishing gradient problem:** As sequences grow longer, early inputs lose influence because gradients shrink exponentially during backpropagation.
- **Difficulty capturing long-term dependencies:** Since past information is passed step by step, it fades over time.
- **Slow training:** Since RNNs process data sequentially, they cannot take advantage of parallel computation.

b) Long short-term memory (LSTM):

Long short-term memory (LSTM) is an advanced variant of Recurrent Neural Networks (RNN) that addresses the issue of capturing long-term dependencies. LSTM was initially introduced by in 1997 and further improved by in 2013, gaining significant popularity in the deep learning community. Compared to standard RNN, LSTM models have proven to be more effective at retaining and utilizing information over longer sequences. In an LSTM network, the current input at a specific time step and the output from the previous time step are fed into the LSTM unit, which then generates an output that is passed to the next time step. The final hidden layer of the last time step, sometimes along with all hidden layers, is commonly employed for classification purposes. LSTM consists of three gates: input

gate, forget gate, and output gate. Each gate performs a specific function in controlling the flow of information. The input gate decides how to update the internal state based on the current input and the previous internal state. The forget gate determines how much of the previous internal state should be forgotten. Finally, the output gate regulates the influence of the internal state on the system[38].

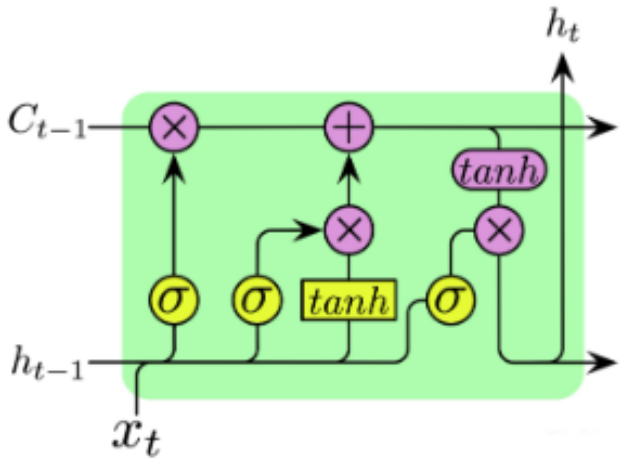


Figure 2.7: The inner structure of LSTM unit[38].

2.5.2.3 Transformer-based architectures

The Transformer architecture was originally introduced by Vaswani et al [41] in 2017 for machine translation and has since become a foundational model in deep learning, especially for natural language processing (NLP). The transformer functions as a self-attention encoder-decoder structure. The encoder consists of a stack of identical layers, and each layer consists of two sublayers. A multi-head self-attention mechanism is the first layer, while the other layer is a position-wise fully connected feed-forward network. Also, A normalizing layer and residual connections connect the multi-headed self-attention module's inputs and output. After that, a decoder uses the representation that the encoder produced to create an output sequence. A stack of identical layers makes up the decoder as well. The decoder adds a third sub-layer to each encoder layer in addition to the primary two, and this sub-layer handles multi-head attention over the encoder stack's output. Like the encoder, residual connections and a normalizing layer are used surrounding each of the sublayers. The encoder and decoder's overall Transformer design is depicted in Figure 2.8, left and right halves respectively.Traditional RNN-based Seq2Seq models could be replaced with attention layers. Using various projection matrices, the query, key, and value vectors in the self-attention layer are all produced from the same sequence. RNN training takes a very long period because it is sequential and iterative. Transformer training, on the other hand, is parallel and enables all features to be learned concurrently, significantly improving computational efficiency and cutting down on the amount of time needed for model training[38].

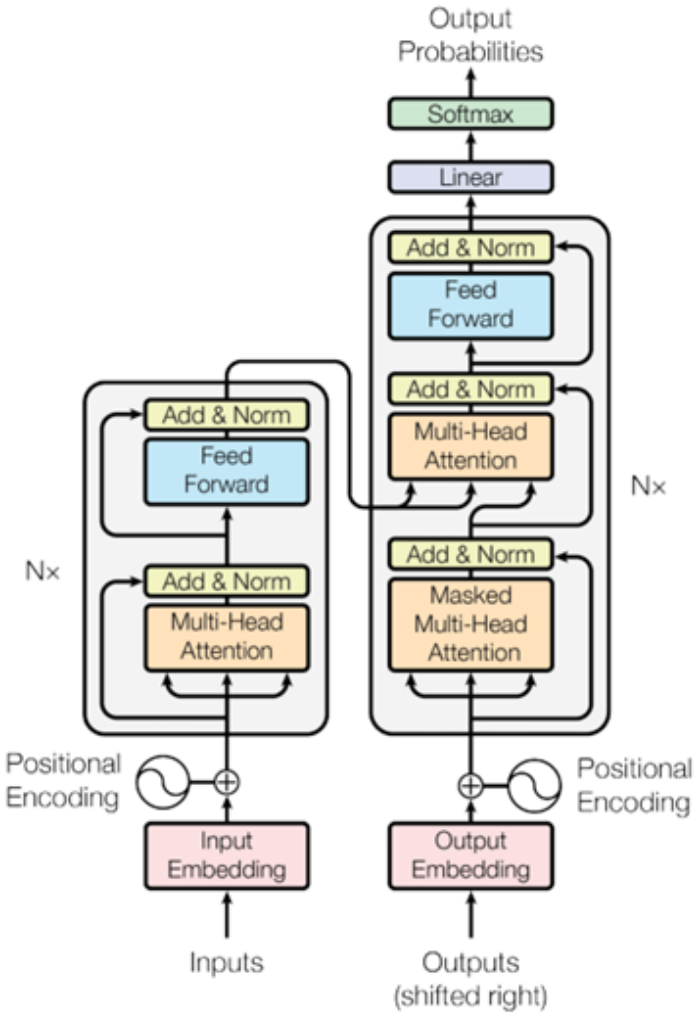


Figure 2.8: The architecture of the Transformer model[38].

Multi-head attention: multi-head attention extends the self-attention mechanism by allowing the model to attend to information from multiple representation subspaces simultaneously. Instead of relying on a single attention operation, several attention heads process the input in parallel, enabling the Transformer to capture different types of relationships and contextual dependencies within the sequence. The outputs of these attention heads are then combined to produce a richer and more informative representation. In practice, the input embeddings are transformed into query (Q), key (K), and value (V) vectors, which are used to determine the relevance of different elements in the sequence. The attention mechanism assigns higher importance to the most relevant features while suppressing less informative ones, allowing the model to focus on critical patterns in the data. This capability is particularly beneficial for complex tasks involving sequential information, such as speech and audio analysis, where long-range dependencies and contextual cues play a significant role in classification performance [38].

Position-wise feed-forward networks: each encoder and decoder layer have a fully connected

feed-forward network in addition to attention sub-layers. This feed-forward network is applied to each position independently and in the same way. This is made up of two linear transformations connected by a ReLU activation [38].

Positional encoding: since the Transformer model does not rely on recurrence or convolution, it requires a way to capture the relative or absolute positions of tokens within a sequence to effectively utilize the sequence's order. To address this, positional encoding is introduced at the input level of both the encoder and decoder stacks. These positional encodings are added to the input embeddings, as they share the same dimensionality,d_{model}.This combination enables the model to incorporate positional information, allowing it to better understand the sequential nature of the data [38].

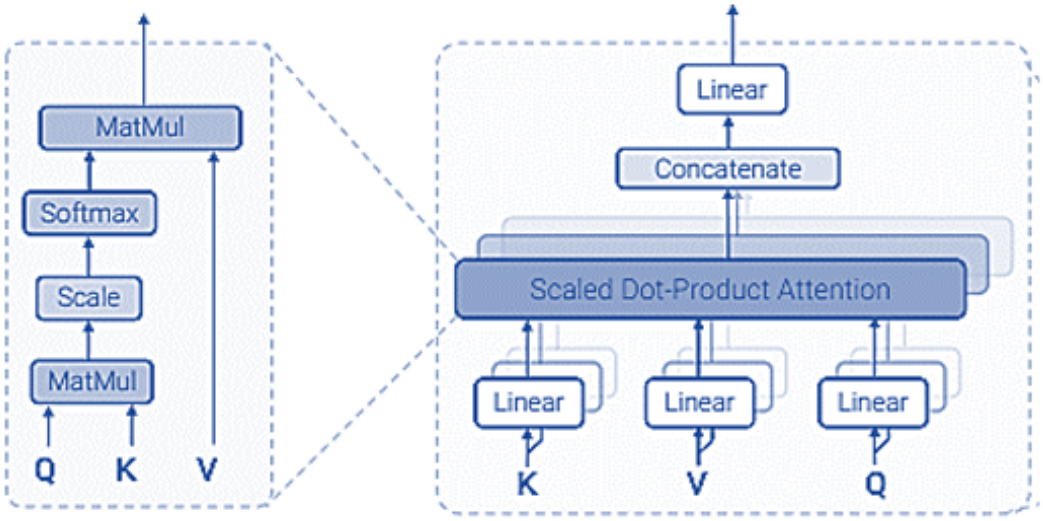


Figure 2.9: (a) Scaled dot-product Attention, (b) Multi-head attention[38].

Transformer variants: the transformer architecture has proven to be highly versatile, with numerous variants developed to address specific challenges across different domains. Typically, Transformers are pretrained on large datasets using unsupervised methods to learn general representations, which are then fine-tuned on specific tasks using supervised learning. This hybrid approach leverages the strengths of both learning paradigms. Some notable Transformer variants include[38]:

a) **Bidirectional encoder representations from transformers (BERT):** a multi-layer bidirectional Transformer encoder for unsupervised pre-training in natural language understanding (NLU) tasks.

b) **Generative pre-training transformer (GPT):** a type of Transformer model developed by OpenAI that excels in natural language processing (NLP) tasks through unsupervised pre-training followed by supervised fine-tuning.

c) **Transformer-XL:** it is proposed for language modeling to permit learning reliance beyond a set length without compromising temporal coherence. Transformer-XL (Transformer-Extra

Long) comprises a unique relative positional encoding method and a segment-level recurrence mechanism. This approach not only makes it possible to record longer-term dependencies, but also fixes the issue of context fragmentation.

d) XLNet: it is a generalized autoregressive (AR) pretraining technique that combines the benefits of autoencoding (AE) and autoregressive (AR) techniques with a permutation language modeling aim. XLNet's neural architecture, which integrates Transformer-XL and the two-stream attention mechanism, is built to function effortlessly with the autoregressive (AR) objective.

e) Fast transformer: it introduces multi-query attention as an alternative to multi-head attention. This approach reduces memory bandwidth requirements, leading to increased processing speed.

f) Multimodal transformer (MulT): it is designed for analyzing human multimodal language. At the heart of MulT is the crossmodal attention mechanism, which provides a latent crossmodal adaptation that fuses multimodal information by directly attending to low-level features in other modalities.

g) Vision transformer (ViT) : an innovative approach based on Transformer structure for visual tasks like image classification.

h) Pyramid vision transformer (PVT): a Transformer framework for complex prediction tasks like semantic segmentation and object recognition.

i) Swin transformer: an hierarchical transformer that uses shifted windows to construct its representation. A wide variety of vision tasks, including semantic segmentation, object detection, and image classification, may be performed with Swin Transformer.

j) Tokens-to-token vision transformer (T2T-ViT): a vision transformer that can be trained from scratch on ImageNet. T2T-ViT overcomes ViT's drawbacks by accurately modeling the structural information of images and enhancing feature richness.

k) Transformer in transformer (TNT) : a vision transformer for visual recognition. Both local and global representations are extracted by the TNT architecture through the use of an inner transformer and an outer transformer.

l) PyramidTNT: an improved TNT model which used pyramid architecture, and convolutional stem in order to greatly enhance the original TNT model.

m) Switch transformers : it is suggested as a straightforward and computationally effective method of increasing a Transformer model's parameter count.

n) ConvNeXt : a redesigned transformer architecture that makes use of the transformer attention mechanism and incorporates convolutional layers into the encoder and decoder modules to extract spatially localized data.

o) Evolutionary algorithm transformer (EATFormer): an improved vision transformer influenced by an evolutionary algorithm.

Despite the progress of deep learning approaches such as CNNs, RNNs, and transformers, audio deepfake detection still faces key limitations. Most supervised models require large labeled datasets, which are difficult to obtain and quickly become outdated due to the rapid evolution of speech synthesis techniques. Consequently, these models often show limited generalization to unseen attacks or different recording conditions. In addition, supervised approaches rely on task-specific feature learning, which may not capture sufficiently rich and transferable speech representations. Although deep models can learn discriminative features from spectrograms or raw waveforms, their performance remains constrained by data availability and diversity. To overcome these limitations, self-supervised learning (SSL) has emerged as a promising approach. SSL leverages large-scale unlabeled speech data to learn general-purpose representations through pretext tasks, without manual annotation. These pretrained models can then be fine-tuned for downstream tasks such as audio deepfake detection, often achieving improved robustness and generalization. Notable SSL models include Wav2Vec 2.0, HuBERT, WavLM, and XLS-R. Therefore, the following section presents self-supervised learning techniques for speech representation and their application to audio deepfake detection.

2.6 Self-supervised learning for audio deepfake detection

Self-supervised learning (SSL) has become a key paradigm in speech representation learning, especially when labeled data is limited. It leverages large-scale unlabeled audio to learn transferable representations through pretext tasks that capture phonetic content, speaker traits, and temporal structure without manual annotations. In audio deepfake detection, SSL is particularly important because it produces general-purpose embeddings that are less dependent on specific datasets and therefore improve generalization to unseen attacks. These representations better capture intrinsic acoustic patterns that are difficult for synthesis systems to reproduce. Modern SSL models such as Wav2Vec 2.0, HuBERT, WavLM, and XLS-R follow a pretraining–fine-tuning scheme and have become central to state-of-the-art speech and deepfake detection systems. The following section reviews SSL principles, key architectures, and their application to audio deepfake detection.

2.6.1 Concept of self-supervised learning for speech

Self-supervised learning (SSL) refers to a family of artificial neural network models that are used to learn useful signal representations from data without any supporting information, such as task-specific data labels. Instead of extracting manually specified signal features, such as MFCCs, SSL algorithms learn the features by taking statistical properties of the input data into acocunt. The concept of useful features refers to signal representations that can act as powerful features in a particular downstream task or a variety of tasks, for which labeled training data exists. Besides acting as a feature extractor, a pre-trained SSL model (neural network) can also be used as a model that is then trained for a downstream task. This is usually done by augmenting a trained SSL model with a small number of additional classification layers, and then training the new layers or the entire model to the target task using labeled data related to the task. This process is called model fine-tuning[42].

In general, SSL algorithms belong to the family of unsupervised learning algorithms and they are practically implemented as deep neural networks. The reason they are referred to as self-

supervised comes from the optimization criterion used to train the models. Classical unsupervised learning operates by performing unsupervised data clustering using a heuristic algorithm (as in k-means) or by modeling the data distribution directly with a generative model (as in Gaussian mixture models, hidden-Markov models, or autoencoders). SSL algorithms, on the other hand, can be viewed as regression models (or classifiers) that try to perform regression mapping from the input data to model's own internal representations derived from the same input data[42].

In case of speech data, one example of a self-supervised regression task is to predict the input speech signal over time. When a deep neural network is tasked with this prediction problem and optimized to solve it, the network has to learn higher-level properties of the data in order to solve the problem adequately. Note that the task has to be difficult enough, so that it cannot be solved by trivial means (e.g., linear interpolation from the observed values). Figure 2.10, illustrates a self-supervised speech prediction task, as it is implemented in Autoregressive Predictive Coding (APC) algorithm (Chung et al., 2019). In the APC model, the task of the model is to predict spectral envelope features (e.g., log-Mel spectra or MFCCs) approximately 50 ms in the future, given access to the current and past observations of the input features[42].

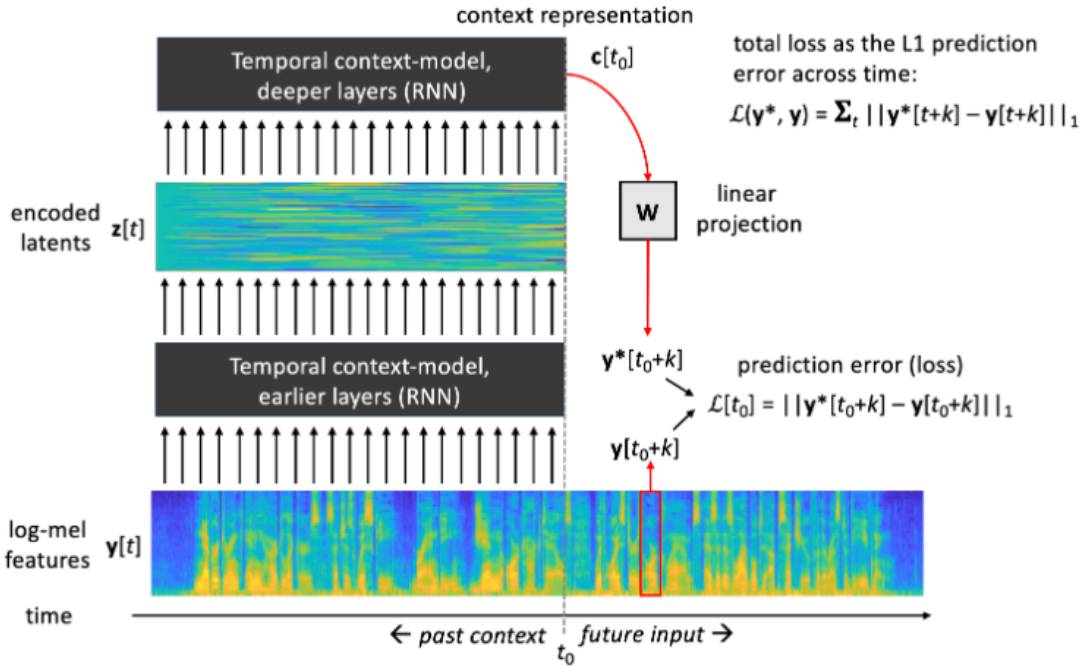


Figure 2.10: A schematic view of APC model for self-supervised learning [42].

Types of self-supervised learning for speech:

a) Generative methods: these methods aim to reconstruct or generate the input from a corrupted or partial version, such as autoencoders and masked reconstruction techniques[43].

b) Contrastive methods: these approaches learn representations by distinguishing between positive and negative samples, maximizing similarity between related inputs and minimizing similarity with unrelated ones [43].

c) **Predictive methods:** these methods predict target representations, often generated by another model, without relying on contrastive loss[43].

2.6.2 Self-supervised speech representation models

2.6.2.1 Wav2Vec 2.0

This section is adapted from the original Wav2Vec 2.0 paper proposed by Facebook AI, with a focus on the aspects relevant to this work[44].

Wav2Vec 2.0 is a self-supervised framework proposed to learn high-quality speech representations directly from raw audio signals. The model is pre-trained on large amounts of unlabeled speech and subsequently fine-tuned on a small amount of labeled data for downstream tasks such as automatic speech recognition (ASR). This approach has been shown to outperform previous semi-supervised methods while being conceptually simpler and more efficient.

- **Architecture:** Wav2Vec 2.0 follows a multi-stage architecture, as shown in Figure 2.11, that composed of three main components: a feature encoder, a context network, and a quantization module. The overall pipeline begins with raw waveform input, which is transformed into latent representations, contextualized through a Transformer network, and trained using a self-supervised objective. The key idea behind the model is to mask portions of the latent representations and train the network to identify the correct latent speech units among a set of distractors. This enables the model to learn robust and context-aware representations of speech without explicit supervision.

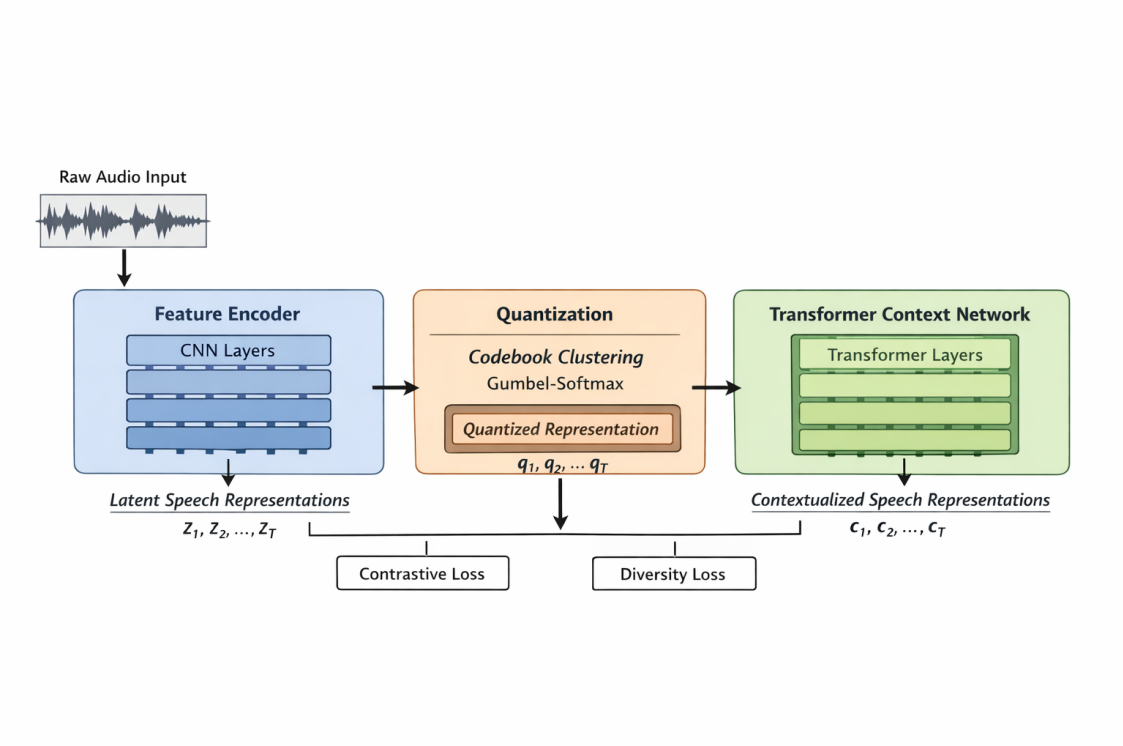


Figure 2.11: Architecture of Wav2Vec 2.0 model.

1. **Feature encoder:** the feature encoder is responsible for extracting low-level representations from the raw audio signal. It consists of a multi-layer convolutional neural network (CNN) that processes the waveform input and produces a sequence of latent speech representations. Each convolutional block includes temporal convolution, layer normalization, and a GELU activation function. The encoder maps the input audio signal X into a sequence of latent vectors $Z = \{z_t\}_{t=1}^{T}$, where T represents the number of time steps. These latent representations serve as the input to the subsequent context network.
2. **Context network:** the context network is based on the Transformer architecture and is designed to capture long-range dependencies in the speech signal. It takes the latent representations generated by the feature encoder as input and produces contextualized representations $C = \{c_t\}_{c=1}^{T}$. Unlike traditional approaches that rely on fixed positional embeddings, Wav2Vec 2.0 uses convolutional layers to model relative positional information. This allows the model to better capture temporal relationships in speech signals. The use of self-attention enables the Transformer to integrate information from the entire sequence, resulting in more expressive and context-aware representations.
3. **Quantization module:** the quantization module converts continuous latent representations into discrete speech units. This is achieved using product quantization, where multiple codebooks are used to represent the latent vectors. For each time step, a discrete representation is selected from each codebook and concatenated to form the final quantized vector. To enable end-to-end training, the model employs the Gumbel-Softmax technique, which allows differentiable sampling of discrete codebook entries. The resulting quantized representations serve as targets in the self-supervised learning objective, enabling the model to learn meaningful speech units.

- **Training strategy:**

1. **Masking:** during pre-training, a portion of the latent representations is randomly masked before being passed to the Transformer. Specifically, contiguous spans of time steps are replaced with a learned mask embedding. This strategy is inspired by masked language modeling and forces the model to infer missing information from context.
2. **Contrastive learning:** the core of the Wav2Vec 2.0 training process is a contrastive learning objective. For each masked time step, the model is required to identify the correct quantized latent representation from a set of candidate representations that includes both the true target and several distractors. The objective function maximizes the similarity between the contextual representation and the correct quantized target while minimizing its similarity with negative samples. As illustrated in Figure 2.12, the model learns to distinguish the true quantized representation from a set of negative samples through this contrastive learning mechanism.

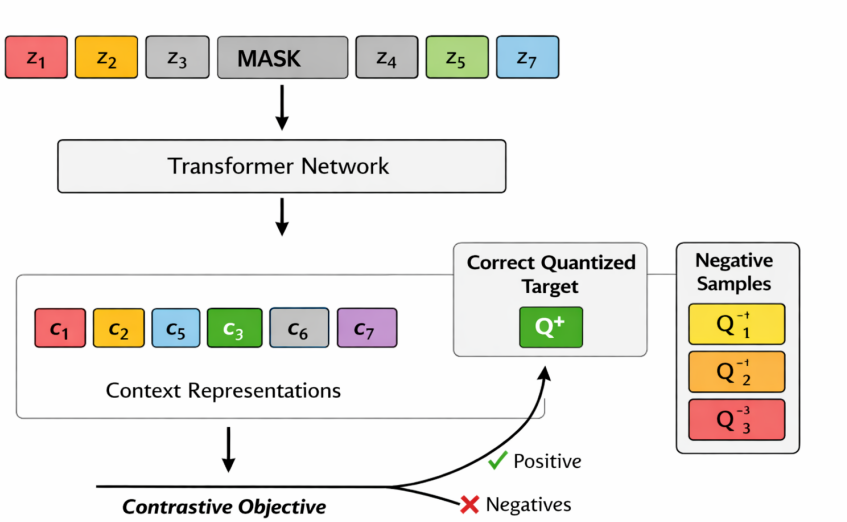


Figure 2.12: Self-supervised training process in Wav2Vec 2.0.

This encourages the model to learn discriminative and robust representations of speech.

3. **Diversity loss:** in addition to the contrastive loss, a diversity loss is introduced to ensure efficient utilization of the quantization codebooks. This loss encourages the model to use all available codebook entries uniformly, preventing the collapse of representations to a limited subset of discrete units.

- **Fine-tuning:** after pre-training, the model is fine-tuned on labeled data for specific downstream tasks. A linear classification layer is added on top of the context network, and the entire model is optimized using the Connectionist Temporal Classification (CTC) loss.

  Fine-tuning allows the model to adapt the learned representations to the target task while leveraging the knowledge acquired during pre-training. Notably, the feature encoder is often kept frozen during this stage to preserve the learned representations .

  During pre-training and fine-tuning, Wav2Vec 2.0 is evaluated on several widely used speech datasets. For unlabeled data, the model is trained on the LibriSpeech corpus (960 hours of audio) as well as the larger LibriVox dataset, which contains approximately 53,000 hours of speech audio. For fine-tuning, different labeled subsets of LibriSpeech are used, ranging from extremely low-resource settings (10 minutes, 1 hour, and 10 hours of labeled data) to larger configurations such as the 100-hour subset and the full 960-hour dataset. Additionally, the model is evaluated on the TIMIT dataset for phoneme recognition tasks. Experimental results demonstrate that Wav2Vec 2.0 achieves strong performance across all settings, particularly in low-resource scenarios. For instance, when trained with only 10 minutes of labeled data, the model achieves a word error rate (WER) of approximately 4.8% on clean test data and 8.2% on noisy data. With the full 960-hour labeled dataset, performance improves significantly, reaching around 1.8% and 3.3% WER on clean and noisy test sets, respectively. These results highlight the effectiveness of self-supervised pre-training in reducing the dependence on large amounts of labeled data while maintaining state-of-the-art performance

- **Advantages:**
  - **Data efficiency:** the model achieves strong performance even with extremely limited labeled data (e.g., 10 minutes of annotated speech).
  - **End-to-end learning:** unlike previous approaches, it jointly learns feature representations and discrete speech units.
  - **Scalability:** performance improves significantly with more unlabeled data.
  - **State-of-the-art results:** it outperforms prior methods on benchmarks such as LibriSpeech in both low-resource and high-resource settings .
- **Limitations:**
  - **High computational cost:** pre-training requires substantial computational resources and large-scale datasets.
  - **Task specificity:** the model is primarily designed for speech recognition and may require adaptation for other tasks such as audio deepfake detection.
  - **Complex training procedure:** the combination of masking, quantization, and contrastive learning increases implementation complexity.

2.6.2.2 HuBERT

This section is adapted from this paper [45], titled "HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units", with a focus on the aspects most relevant to this work.

HuBERT (Hidden-Unit BERT) is a self-supervised framework designed to learn high-quality speech representations directly from raw audio signals. The model is pre-trained on large amounts of unlabeled speech and then fine-tuned using a small amount of labeled data for downstream tasks such as Automatic Speech Recognition (ASR). HuBERT introduces a masked prediction objective based on hidden cluster units, allowing the model to learn robust acoustic and contextual speech representations without manual annotations.

- **Architecture:** HuBERT follows a multi-stage architecture, as shown in Figure 2.13, that composed of four main components: a waveform encoder, a Transformer encoder, an offline clustering module, and a prediction head. The overall pipeline starts with raw waveform input, which is transformed into latent features, contextualized through a Transformer network, and trained to predict masked hidden units generated by clustering. The key idea behind HuBERT is to mask portions of the speech sequence and train the model to recover the cluster labels of the masked frames using surrounding context.

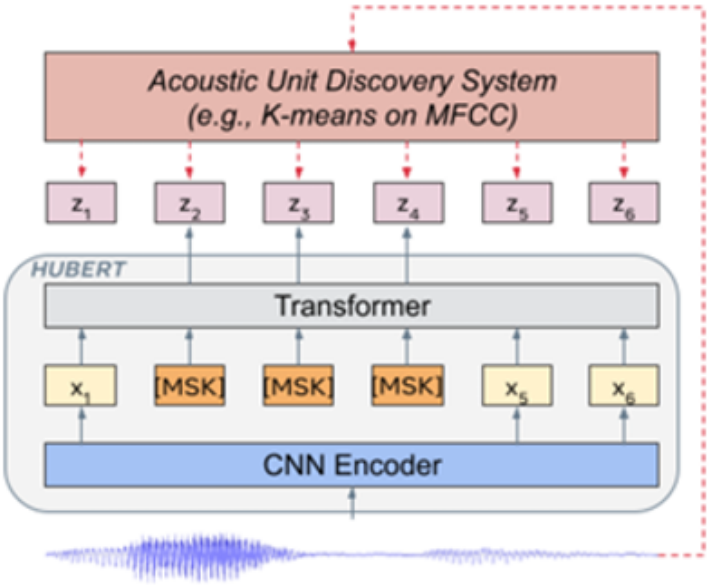


Figure 2.13: Architecture of HuBERT model[45].

a) Waveform encoder: the waveform encoder is responsible for extracting low-level acoustic representations from raw speech. It consists of a multi-layer convolutional neural network (CNN) with seven temporal convolution layers. These layers progressively downsample the audio signal and convert it into latent feature vectors at a 20 ms frame rate. These latent features are then passed to the Transformer encoder.

b) Transformer encoder: the Transformer encoder captures long-range temporal dependencies and contextual information in speech. It receives the latent representations produced by the waveform encoder and processes them using multiple self-attention layers. HuBERT provides three model sizes: **BASE:** 12 Transformer layers **LARGE:** 24 Transformer layers **X-LARGE:** 48 Transformer layers The Transformer outputs contextualized speech representations that are used for masked prediction.

c) Offline clustering module: instead of using manually labeled targets, HuBERT generates pseudo-labels using unsupervised clustering methods such as K-means. In the first iteration, K-means is applied on MFCC features. In later iterations, K-means is applied on HuBERT learned representations. These cluster assignments act as target labels during self-supervised training.

d) Prediction head: a projection layer followed by a softmax classifier predicts the correct cluster label for each masked frame. The model compares the predicted distribution with the pseudo-label generated by clustering.

- **Training strategy:**

a) Masking: during pre-training, a percentage of time steps are randomly selected as starting positions, and spans of consecutive frames are masked. The masked frames are replaced with a learned mask embedding before being passed to the Transformer. This forces the model to infer missing content from surrounding context.

b) Masked prediction objective: unlike wav2vec 2.0, HuBERT predicts only the cluster labels of masked frames. The model minimizes cross-entropy loss between predicted labels and clustering targets. This strategy encourages the model to learn: acoustic representations of visible speech segments long-range temporal dependencies phonetic and linguistic structure of speech sequences As illustrated in Figure 2.14, The model HuBERT masks parts of the input representations and leverages a Transformer to use surrounding context for predicting the correct cluster labels, while being trained to minimize the Cross-Entropy Loss between predicted and target outputs.

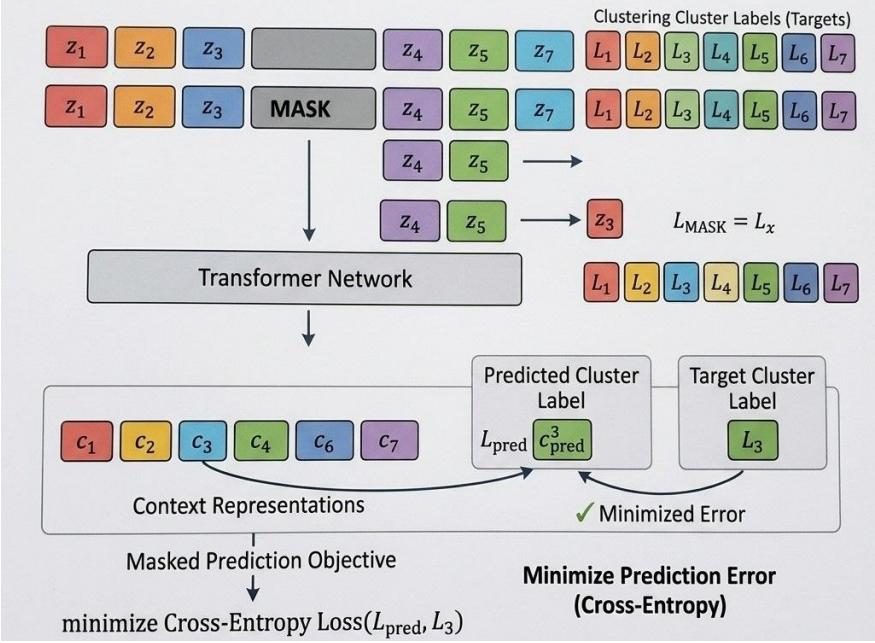


Figure 2.14: Self-Supervised Training Process in HuBERT

This encourages the model to learn discriminative and robust representations of speech.

c) Iterative refinement: HuBERT improves target quality through multiple training iterations:

1. Train first model using K-means labels from MFCC features
2. Extract better latent features using the trained model
3. Re-cluster these new features
4. Retrain HuBERT with improved labels This iterative refinement significantly improves representation quality and recognition performance.

d) Pretraining data: HuBERT was trained using:

LibriSpeech: 960 hours of speech

Libri-Light: 60,000 hours of speech

- **Fine-tuning:** After self-supervised pre-training, HuBERT is fine-tuned for supervised speech recognition tasks. A randomly initialized softmax classification layer is added on

top of the Transformer encoder, and the model is optimized using Connectionist Temporal Classification (CTC) loss. During fine-tuning, the convolutional waveform encoder is usually kept frozen.

Fine-tuning data splits: HuBERT was evaluated under different labeled data settings:10 minutes, 1 hour, 10 hours, 100 hours, and Full 960 hours

Performance: Experimental results show that HuBERT matches or outperforms wav2vec 2.0 in many low-resource and high-resource settings. The X-LARGE model achieved significant Word Error Rate (WER) reductions on difficult test subsets.

- **Advantages:**
  - Strong performance: HuBERT achieves state-of-the-art ASR results, especially when labeled data is limited.
  - Simpler objective than wav2vec 2.0: It avoids contrastive learning, negative sampling, and Gumbel-softmax scheduling.
  - Iterative improvement: cluster labels become better over time, leading to stronger learned representations.
  - Scalability: HuBERT successfully scales to very large models up to 1 billion parameters.
  - General-purpose representations: the learned features can be transferred to multiple speech tasks beyond ASR.
- **Limitations:**
  - External clustering dependency: HuBERT requires a separate K-means clustering step before training.
  - Multi-stage pipeline: training is not fully end-to-end because clustering and representation learning are separate processes.
  - High computational cost: large-scale pre-training on 60k hours of speech and billion-parameter models requires significant resources.
  - Quality depends on pseudo-labels: poor initial cluster assignments may reduce final performance.

2.6.2.3 WavLM

This section is adapted from the original WavLM framework proposed by Chen et al. [46], with a focus on the aspects relevant to this work. WavLM is a self-supervised speech representation model designed to learn universal and robust representations from large-scale unlabeled audio data. Unlike earlier self-supervised models that primarily focused on automatic speech recognition (ASR), WavLM aims to support a broader range of speech processing tasks, including speaker verification, speech separation, and emotion recognition. By integrating both speech content and speaker-related information during pre-training, WavLM provides a unified framework for full-stack speech processing. This makes it particularly suitable for complex real-world scenarios involving noise, overlapping speech, and multiple speakers.

- **Architecture:** The architecture of WavLM follows a structure similar to earlier self-supervised models, consisting of a feature encoder and a Transformer-based context network. The feature encoder processes raw audio signals and extracts latent speech representations, which are then fed into the Transformer to generate contextualized representations. However, WavLM introduces key enhancements to improve modeling capacity, including modifications to positional encoding through a gated relative position bias mechanism. These improvements allow the model to better capture temporal dependencies and adapt dynamically to different speech patterns.

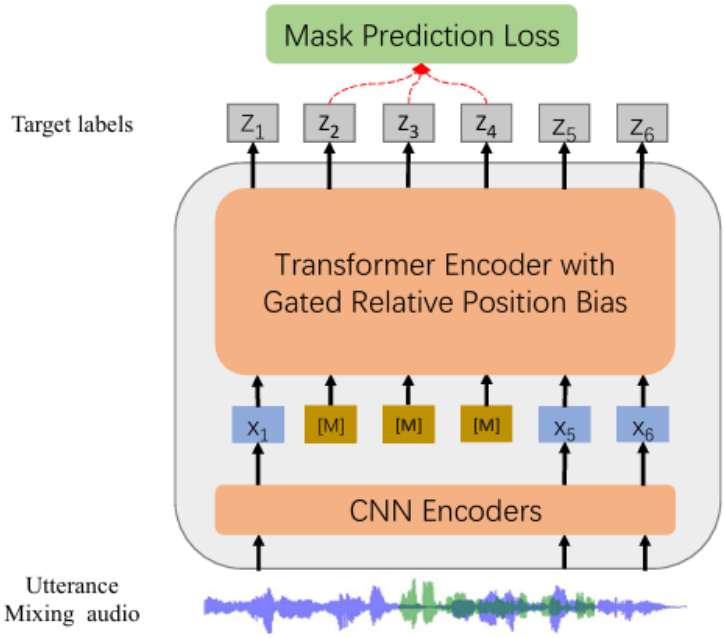


Figure 2.15: WavLM model architecture.[45]

a) Feature encoder: The feature encoder in WavLM is implemented as a multi-layer convolutional neural network that converts raw waveform inputs into latent feature representations. This encoder typically consists of several temporal convolution layers, which progressively reduce the temporal resolution while increasing the representational capacity. The resulting latent vectors provide a compact and informative representation of the speech signal, which serves as input to the Transformer-based context network.

b) Transformer context network: The context network in WavLM is based on the Transformer architecture and is responsible for capturing long-range dependencies within the speech signal. It processes the latent representations generated by the feature encoder and produces contextualized embeddings that incorporate both local and global information. A key innovation in WavLM is the use of gated relative position bias, which enhances the model's ability to encode positional information in a flexible and adaptive manner. This mechanism improves the modeling of temporal relationships and contributes to better performance across diverse speech tasks.

c) **Masked speech denoising and prediction:** A central contribution of WavLM lies in its training objective, which combines masked speech prediction with denoising capabilities. During pre-training, portions of the input representations are masked, and the model is required to predict the corresponding targets. Unlike previous approaches, WavLM extends this idea by introducing noisy and overlapped speech as input, thereby encouraging the model to learn robust representations under challenging conditions. In addition to predicting masked units, the model is trained to recover clean speech representations from corrupted inputs. This includes scenarios where multiple speakers overlap or where background noise is present. By learning to separate and reconstruct meaningful speech components, WavLM captures both linguistic content and speaker-specific characteristics, making it more versatile than earlier models.

- **Training strategy:** The training strategy of WavLM involves the simulation of realistic acoustic conditions through the addition of noise and overlapping speech. During pre-training, clean speech signals are artificially mixed with noise or with other speech signals to create challenging input scenarios. These mixtures are generated with varying overlap ratios and energy levels, ensuring diversity in the training data. The model is then trained using a masked prediction objective, where it predicts discrete targets corresponding to the original clean speech. This approach enables WavLM to learn invariant and robust representations that generalize well across different environments. The combination of masking and denoising objectives plays a crucial role in enhancing the model's performance in real-world applications.

- **Pre-training data:** WavLM is trained on a large and diverse collection of unlabeled speech data totaling approximately 94,000 hours. This dataset includes multiple sources such as Libri-Light, GigaSpeech, and VoxPopuli, which together provide a wide range of speakers, recording conditions, and linguistic variations. The scale and diversity of the training data contribute significantly to the model's ability to generalize across tasks and domains, making it one of the most robust self-supervised speech models to date.

- **Fine-tuning and downstream tasks:** After pre-training, WavLM can be fine-tuned for a variety of downstream tasks using labeled data. The model has been extensively evaluated on the SUPERB benchmark, which includes a wide range of speech processing tasks such as automatic speech recognition, speaker verification, emotion recognition, and speech separation. Experimental results demonstrate that WavLM achieves state-of-the-art performance across many of these tasks, highlighting its effectiveness as a general-purpose speech representation model.

- **Advantages:** WavLM offers several notable advantages over previous self-supervised models:

  – General-purpose representation learning: The model is designed to support a wide range of speech processing tasks, including speech recognition, speaker verification, and speech separation.

  – Robustness to noise and overlap: By incorporating noisy and overlapped speech during training, WavLM learns representations that remain effective in real-world acoustic

conditions.

- Speaker-aware representations: The model captures both linguistic content and speaker-related information, improving performance in tasks involving speaker identity and interaction.
- Scalability: Training on large-scale and diverse datasets allows the model to generalize well across different domains and conditions.
- State-of-the-art performance: WavLM achieves superior results on benchmarks such as SUPERB, demonstrating its effectiveness across multiple speech tasks.

- **Limitations:** Despite its strengths, WavLM has several limitations:
  - High computational cost: Pre-training requires significant computational resources and large-scale datasets.
  - Training complexity: The combination of masking, denoising, and data augmentation strategies increases the complexity of the training process.
  - Data dependency: The model's performance heavily relies on the availability of large and diverse unlabeled datasets.
  - Task adaptation requirements: Although versatile, the model may still require careful fine-tuning and optimization for specific applications such as deepfake detection.

2.6.3 SSL-based deepfake detection frameworks

Self-supervised learning (SSL) models typically follow the following pipeline:

a) SSL feature extraction: In the first stage, as shown in Figure 2.16, a pre-trained self-supervised model (e.g., WavLM, XLS-R, or HuBERT) acts as a front-end encoder on the raw signal (audio, or audio-visual frames) [47][48]. The model outputs multiple hidden-state layers, often at frame-level resolution; analyses show that lower layers tend to capture the most discriminative artifacts for deepfake detection [49].

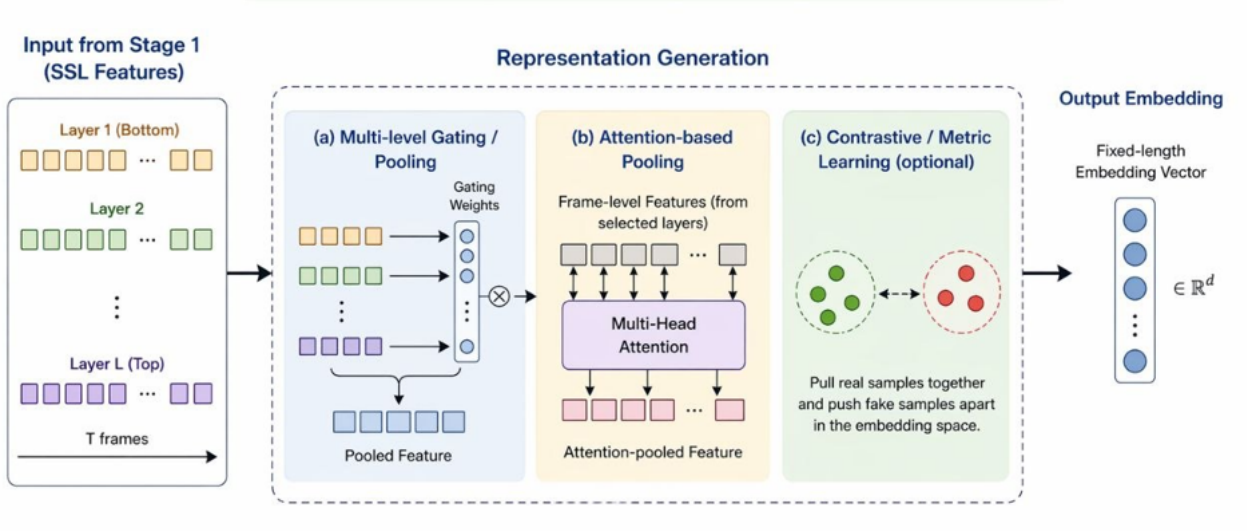


Figure 2.16: SSL feature extraction.

b) embedding generation: In the second stage, SSL features are transformed into a compact, task-oriented embedding, as illustrated in Figure 2.17. Common strategies include:

Gating / multi-level pooling: (e.g., Multi-level SSL Feature Gating) that selectively aggregates useful layers or time frame [47].

Attention-based pooling :(e.g., Multi-Head Attention Pooling, graph-aware attention) to compress frame-wise features into a fixed-length vector [47][48].

Contrastive / metric-learning–inspired embeddings: (in some multimodal or image-based works) that push real and fake instances apart in the embedding space [50][51].

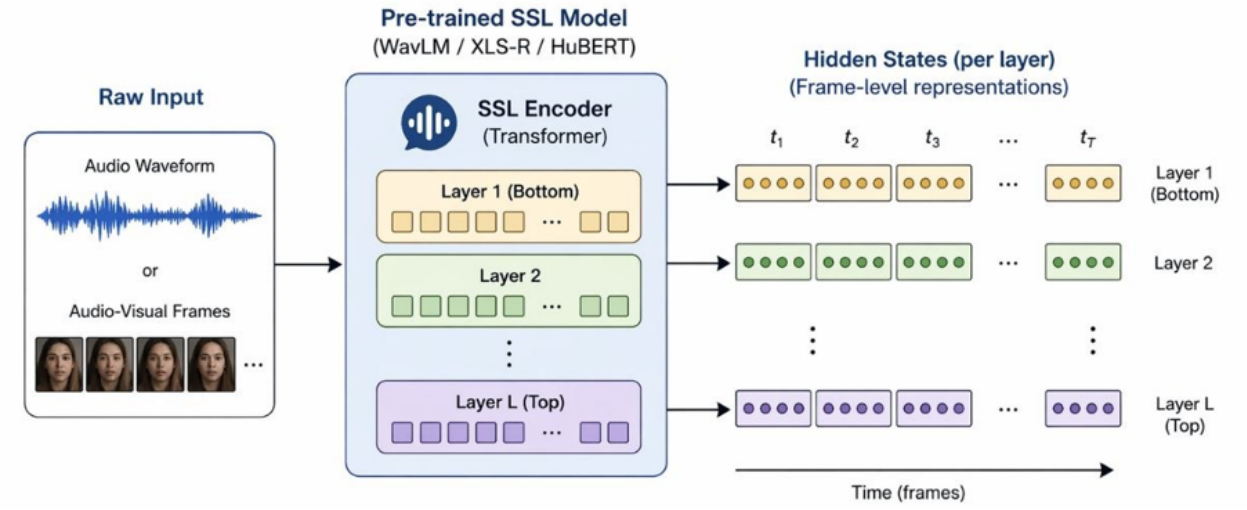


Figure 2.17: SSL embedding generation.

c) classification: Figure 2.18 illustrates how the final stage consumes the SSL-derived embedding and applies a lightweight classifier such as:

- A simple MLP or linear layer for binary deepfake classification [47].
- A multi-modal classifier combining SSL audio embeddings with handcrafted visual features when working on audio-visual deepfakes.[48]

Training often uses a combined loss (e.g., cross-entropy plus a regularization term like CKA or boundary-aware losses) to improve generalization across unseen attacks and languages. [47][48]

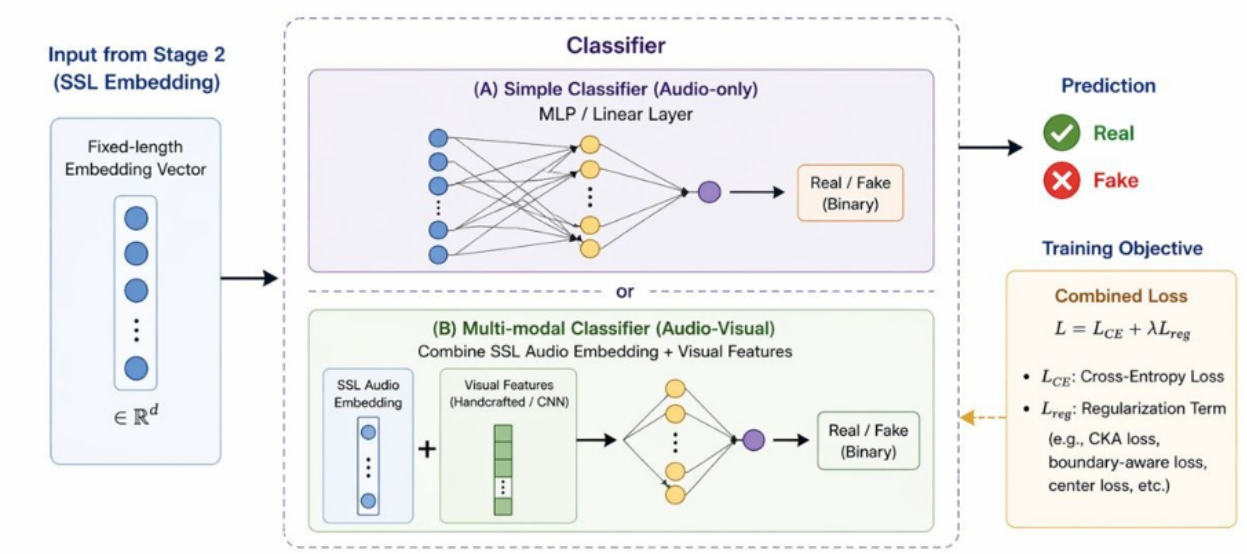


Figure 2.18: SSL classification.

2.7 Transfer learning with SSL models

Transfer learning with self-supervised learning (SSL) models uses a model pre-trained on large unlabeled data via SSL to initialize downstream tasks, then fine-tunes it on a smaller labeled dataset. This reduces annotation needs while improving generalization, especially in low-data regimes like medical imaging or audio processing [52][53].

Role of SSL in transfer learning:SSL trains models on pretext tasks (e.g., predicting missing image patches or audio segments) so they learn rich, domain-relevant representations without labels. These pre-trained weights are then loaded as a backbone and adapted (via fine-tuning or linear probing) to a target task, leveraging the SSL-learned features [52][54].

SSL-based transfer learning performance: Empirical studies show that SSL-pretrained models often transfer better than supervised pre-trained models because the SSL objective avoids overfitting to narrow label-dependent features. This leads to more robust features on downstream tasks, particularly when the target dataset is small or differs from the source domain [53][55].

2.8 Evaluation metrics

The following evaluation metrics are adopted from [56]: confusion matrix, accuracy, precision, recall, and F1-score.

2.8.1 Confusion matrix

The confusion matrix is a cross table that records the number of occurrences between two raters, the true/actual classification and the predicted classification, as shown in Figure 2.19, the columns stand for model prediction whereas the rows display the true classification.The classes are listed in the same order in the rows as in the columns, therefore the correctly classified elements are located on the main diagonal from top left to bottom right and they correspond to the number of times the two raters agree.

		PREDICTED		
	Classes	Positive (1)	Negative (0)	Total
ACTUAL	Positive (1)	TP = 20	FN = 5	25
	Negative (0)	FP = 10	TN = 15	25
	Total	30	20	**50**

Figure 2.19: Example of Confusion Matrix [56].

2.8.2 Accuracy

Accuracy is one of the most popular metrics in classification and it is directly computed from the confusion matrix. Referring to Figure 2.19:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.1}$$

The formula of the Accuracy considers the sum of True Positive and True Negative elements at the numerator and the sum of all the entries of the confusion matrix at the denominator. True Positives and True Negatives are the elements correctly classified by the model and they are on the main diagonal of the confusion matrix, while the denominator also considers all the elements out of the main diagonal that have been incorrectly classified by the model.In simple words, consider to choose a random unit and predict its class, Accuracy is the probability that the model prediction is correct.

2.8.3 Precision

The Precision is the fraction of True Positive elements divided by the total number of positively predicted units (column sum of the predicted positives). In particular, True Positive are the elements that have been labelled as positive by the model and they are actually positive, while False

Positive are the elements that have been labelled as positive by the model, but they are actually negative.

$$\text{Precision} = \frac{TP}{TP+FP} \tag{2.2}$$

Precision expresses the proportion of units our model says are Positive and they actually Positive. In other words, Precision tells us how much we can trust the model when it predicts an individual as Positive.

2.8.4 Recall

Precision expresses the proportion of units our model says are Positive and they actually Positive. In other words, Precision tells us how much we can trust the model when it predicts an individual as Positive.The Recall is the fraction of True Positive elements divided by the total number of positively classified units (row sum of the actual positives). In particular False Negative are the elements that have been labelled as negative by the model,but they are actually positive.

$$\text{Recall} = \frac{TP}{TP+FN} \tag{2.3}$$

The Recall measures the model's predictive accuracy for the positive class: intuitively, it measures the ability of the model to find all the Positive units in the dataset.

2.8.5 F1-score

F1-Score assesses classification model's performance starting from the confusion matrix, it aggregates Precision and Recall measures under the concept of harmonic mean.

$$\text{F1-Score} = \left(\frac{2}{\text{precision}^{-1}+\text{recall}^{-1}}\right) = 2\left(\frac{\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}\right) \tag{2.4}$$

The formula of F1-score can be interpreted as a weighted average between Precision and Recall, where F1-score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall are equal onto the F1-score and the harmonic mean is useful to find the best trade-off between the two quantities.

2.8.6 Equal Error Rate (EER)

EER is a widely used metric for evaluating ASV and other biometric systems in binary classification tasks. It represents the error rate at the threshold where the False Acceptance Rate (FAR) equals the False Rejection Rate (FRR). In the context of detecting audio deepfakes, FAR is the probability of misclassifying an actual fake audio sample as genuine, while FRR is the probability of misclassifying an actual genuine audio sample as fake.[12] Given all the detection scores of a particular system, the FAR and FRR at a specific threshold θ can be defined as [12]:

$$FAR(\theta) = \frac{\#\{\text{fake trials with score} > \theta\}}{\#\{\text{total fake trials}\}}, \tag{2.5}$$

$$FRR(\theta) = \frac{\#\{\text{genuine trials with score} \leq \theta\}}{\#\{\text{total genuine trials}\}}. \tag{2.6}$$

where FAR(θ) and FRR(θ) are monotonically decreasing and increasing functions of θ, respectively, and the EER is where the two curves intersect. One may also see the EER as the crossover error rate and the FAR and FRR as the false alarm rate and miss rate in other studies [12].

In achieving a low EER, the goal is to minimise both the FAR and FRR simultaneously,indicating that the system can maintain a balance between sensitivity and specificity while achieving a low overall error rate. The EER is typically expressed as a percentage,with lower values indicating better system performance [12].

2.9 Comparative analysis and limitations of existing approaches

The table below represents a Comparison between audio deepfake detection approaches:

Table 2.2: Comparison of audio deepfake detection approaches.

Attribute	**Traditional ML**	**Supervised DL**	**Self-Supervised Learning (SSL)**
Main Techniques	GMM, SVM, HMM with MFCC, LFCC, spectral descriptors	CNN, RNN, LSTM, Transformer-based models	Wav2Vec 2.0, HuBERT, WavLM
Feature Extraction	Handcrafted features	Automatically learned from spectrograms or raw audio	Learned from unlabeled speech representations
Data Requirement	Small to medium labeled datasets	Large labeled datasets	Large unlabeled datasets with limited labeled data for fine-tuning
Advantages	Computationally simple, fast training, interpretable models	High detection accuracy and ability to capture complex spectral and temporal dependencies	Strong generalization, robustness to unseen attacks, and reduced need for annotations
Limitations	Strong dependence on handcrafted features and poor real world generalization	Risk of overfitting and high dependency on labeled data	Requires large scale pretraining and significant computational resources
Robustness	Low	Medium to High	Very High
Computational Cost	Low	High	Very High during pretraining
Example of Previous Work	[58]	[59]	[60]

Traditional machine learning approaches remain computationally efficient but heavily depend on handcrafted features and often struggle against unseen spoofing attacks. Deep learning methods improve detection accuracy through automatic feature learning but typically require large labeled datasets. SSL-based approaches overcome many of these limitations by leveraging large-scale unlabeled speech corpora, producing more robust representations and better generalization. However, SSL models still require significant computational resources and may exhibit complementary strengths, motivating the use of ensemble learning strategies.

Despite the remarkable progress achieved by deep learning and self-supervised learning approaches, several challenges remain unresolved. Existing systems often suffer from limited cross-dataset generalization, high computational requirements, and a lack of interpretability. Furthermore, most studies rely on a single SSL model, which may not fully exploit complementary speech representations. These limitations motivate the development of an explainable ensemble framework that combines multiple SSL models to improve robustness, generalization capability, and transparency.

2.10 Explainable artificial intelligence for audio deepfake detection

2.10.1 Introduction to explainable AI

The rapid advancement of deep learning models has led to remarkable performance in speech-related tasks. However, these models are often considered black boxes due to their lack of interpretability. This limitation is particularly critical in audio deepfake detection, where understanding the reasoning behind model predictions is essential for trust, security, and forensic analysis [61].

Explainable Artificial Intelligence (XAI) aims to address this issue by providing methods that make machine learning models more transparent and interpretable. Interpretability refers to the ability to explain or present a model's behavior in understandable terms to humans. XAI methods can be categorized into model-specific and model-agnostic approaches, as well as local and global explanation techniques [61].

In audio deepfake detection, XAI plays a crucial role by enabling the identification of discriminative acoustic features, revealing artifacts present in synthetic speech, and enhancing the overall transparency and reliability of the detection models [61].

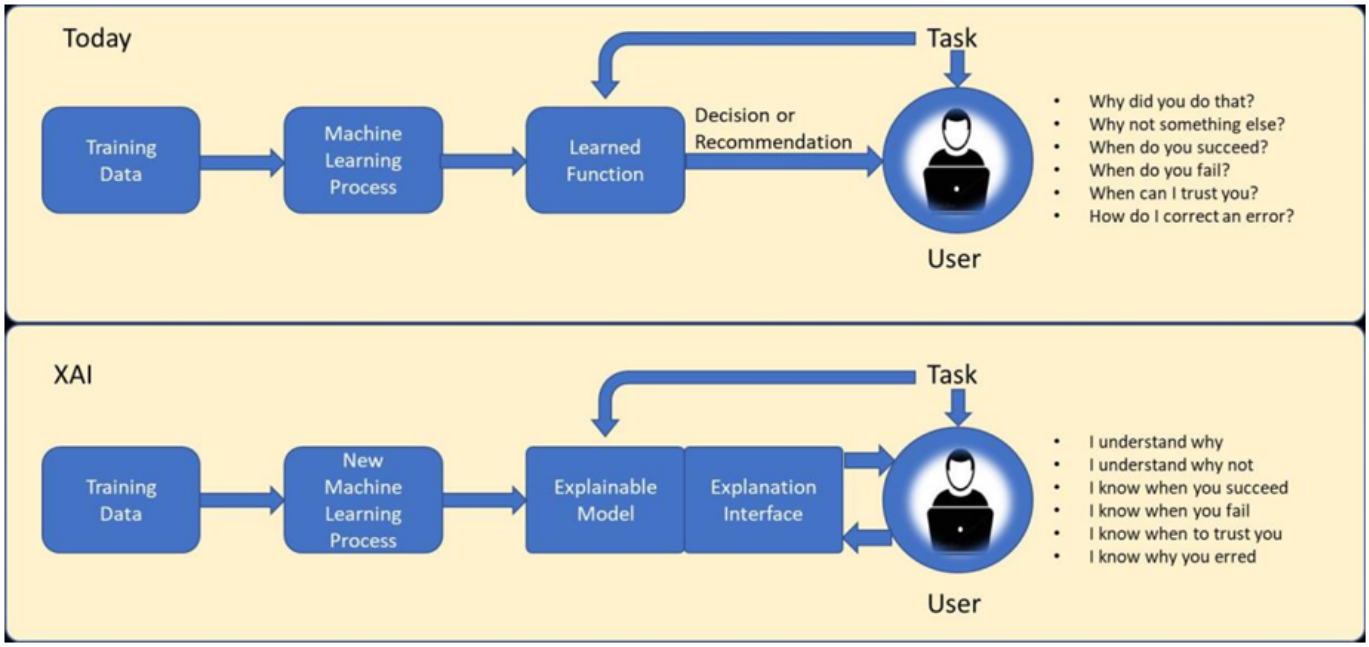


Figure 2.20: Explainable Artificial intelligence (XAI) Approach [62].

2.10.2 Local interpretable model-agnostic explanations (LIME)

Local Interpretable Model-agnostic Explanations (LIME), introduced by Ribeiro et al in their 2016 paper "Why Should I Trust You? Explaining the Predictions of Any Classifier" [63], is a widely used technique for explaining individual predictions of any machine learning model. LIME approximates a complex model locally with an interpretable model by generating perturbed samples around a given instance and observing the corresponding predictions. It then fits a simple model to approximate the behavior of the original model in the neighborhood of the instance.

In audio deepfake detection, LIME can be applied to spectrogram representations or acoustic features such as MFCCs, where it highlights the time–frequency regions that contribute most to classification decisions. This approach offers several advantages, including its model-agnostic nature, ease of interpretation, and effectiveness in providing local explanations of individual predictions. However, it also presents notable limitations, such as instability caused by random perturbations, sensitivity to feature correlations, and its restriction to local interpretability without offering a global understanding of the model behavior.

2.10.3 Shapley additive explanations (SHAP)

SHAP (Shapley Additive Explanations) is a unified framework for interpreting machine learning model outputs by assigning importance values, known as Shapley values, to each feature such as pixels, frames, or phonemes. It is based on cooperative game theory, where the contribution of each feature is computed by considering different combinations of features and measuring their impact on the model's prediction. This approach provides both local interpretability, by explaining individual predictions, and global interpretability, by analyzing the overall behavior of the model [64].

In the context of audio analysis and audio deepfake detection, SHAP assigns importance scores to spectral components, enabling the identification of anomalies such as robotic tones or unnatural frequency patterns commonly found in synthetic or manipulated audio[64].

2.10.4 Gradient-Based Saliency Methods

A saliency map is a visual representation that high-lights key input areas that influence a model's output. It is used in various fields, including image classification, medical imaging, attention mechanisms, etc. Saliency maps en-hance interpretability across several modalities in DD. For images and videos, it emphasizes altered facial areas, revealing artifacts caused by generative networks [64].

In audio, they can be utilized on spectrograms to uncover significant frequency distortions, highlighting artificial speech patterns. In text-based DD, saliency maps can assist transformer-based models in highlighting essential words or phrases that influence classification [64].

2.10.5 Comparative Analysis

To better understand the characteristics of the most widely used Explainable Artificial Intelligence (XAI) techniques, the table below presents a comparative overview based on their methodology, scope, model accessibility, strengths, and common application domains.

Table 2.3: Comparison of XAI Methods.

Method	Type	Scope	Access	Strength	Used in
LIME	Perturbation	Local	Model-agnostic	Simplicity	Tabular data, text classification, image classification
SHAP	Game-theoretic	Local + Global	Both	Theoretical guarantees	Tabular data, NLP, computer vision
Saliency	Gradient	Local	Model-specific	Efficiency	CNN-based models, image analysis, audio/speech models

Each XAI method offers distinct advantages depending on the application scenario. LIME provides simple local explanations for a wide range of models, SHAP offers both local and global interpretability with strong theoretical foundations, while Saliency methods are particularly effective for deep learning models due to their ability to directly highlight the input regions that contribute most to a prediction. Consequently, the choice of an XAI technique should be guided by the model architecture, interpretability requirements, and computational constraints.

In this work, Gradient Saliency was selected because the proposed audio deepfake detection framework is based on deep neural networks operating on spectrogram representations of speech signals. Gradient-based attribution methods are well suited for such architectures, as they reveal the time-frequency components that most influence the model's decision. Furthermore, recent studies

have demonstrated the effectiveness of gradient-based explanations for analyzing deepfake speech detectors. For instance, Staněk et al. [75] employed Integrated Gradients to localize the acoustic evidence used by deepfake detection models, showing that gradient-based attribution techniques can provide meaningful insights into the features driving classification decisions. Therefore, Gradient Saliency was adopted in this study to enhance the interpretability of the proposed model while maintaining compatibility with deep learning-based audio analysis.

2.11 Conclusion

In this chapter, we presented a comprehensive review of the state of the art in audio deepfake detection. We began by introducing the concept of deepfakes, with a particular focus on audio, and highlighted their associated risks in security, politics, and digital communication. We then explored the main audio deepfake generation techniques, including Text-to-Speech (TTS) and Voice Conversion (VC), which enable the creation of highly realistic synthetic speech and constitute the primary sources of spoofed audio. Subsequently, we reviewed commonly used benchmark datasets, such as ASVspoof, and discussed their limitations, particularly in terms of data diversity and generalization to real-world scenarios.

A central part of this chapter focused on detection approaches. We first examined traditional machine learning methods based on handcrafted feature extraction, and spectrogram-based features, combined with preprocessing, feature selection, and classical classifiers. While effective, these approaches remain limited in capturing complex patterns. We then emphasized deep learning approaches, which leverage input representations including Convolutional Neural Networks (CNNs), temporal modeling techniques, as well as attention mechanisms and Transformer-based models for improved performance.

Furthermore, we highlighted self-supervised learning models, such as Wav2Vec 2.0, HuBERT, and WavLM, which learn rich speech representations from large-scale unlabeled data and significantly enhance generalization capabilities. We also reviewed Explainable Artificial Intelligence (XAI) approaches, including LIME, SHAP, and Gradient-Based Saliency methods, which provide valuable insights into model decision-making processes and contribute to improving the transparency, interpretability, and trustworthiness of audio deepfake detection systems.

Generally, despite the significant progress achieved through machine learning and deep learning techniques, audio deepfake detection remains a challenging and evolving field. Key issues, such as dataset limitations, robustness, and generalization to unseen attacks, persist and motivate the development of more advanced approaches. Based on the limitations identified in existing approaches, particularly regarding generalization and interpretability, the next chapter introduces an explainable ensemble framework that combines multiple self-supervised learning models. The proposed approach aims to exploit complementary speech representations while providing greater transparency through explainable artificial intelligence techniques.

Chapter 3

Design of an Explainable Self-Supervised Ensemble Learning Framework for Audio Deepfake Detection

3.1 Introduction

This research aims to develop an ensemble learning approach that combines multiple pre-trained transformer models to improve the accuracy of audio deepfake detection tasks. By using the ASVspoof dataset, our approach is evaluated on realistic spoofed and genuine speech samples to ensure its effectiveness in detecting manipulated audio signals. Ensemble learning techniques are employed to combine the outputs of several models, potentially enhancing overall performance and improving the robustness and generalization capability of the proposed system.

This chapter presents the methodology adopted in this work. It describes the three pre-trained transformer based models used in the ensemble, namely Wav2Vec 2.0, HuBERT, and WavLM, detailing their architectures and main components. The selection of these models is justified by their strong performance in speech representation learning and their ability to provide complementary feature extraction capabilities for detecting deepfake audio.

In addition, this chapter explores different ensemble learning methods, including hard voting, soft voting, weighted soft voting, stacking, bagging. The implementation details of these techniques are presented along with the motivations behind their use in leveraging the strengths of individual models while reducing their limitations, with the aim of building a more efficient and reliable audio deepfake detection system.

Finally, the chapter presents a practical single-audio inference pipeline that enables the proposed framework to classify previously unseen speech recordings. In addition, a strategy for handling long audio recordings is introduced through segment-based analysis and majority voting, improving the robustness and applicability of the system in real-world scenarios involving variable-length audio samples.

3.2 General presentation of the proposed framework

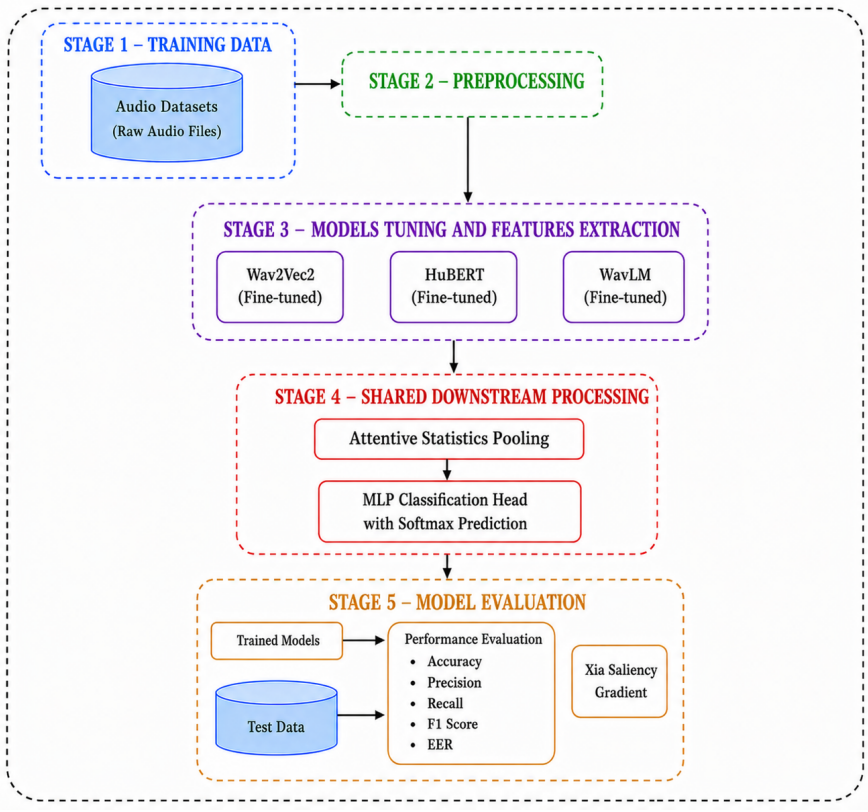
The proposed model architecture for audio deepfake detection is built on a multi-branch self-supervised learning framework that processes raw audio from the ASVspoof 2019 dataset through a unified pipeline.

Each input waveform is preprocessed by converting it to mono, resampling it to 16 kHz, truncating it to a fixed duration, and applying amplitude normalization. The signals are then padded into batches with corresponding attention masks, and augmented with background noise using random signal-to-noise ratio (SNR) levels to improve robustness.

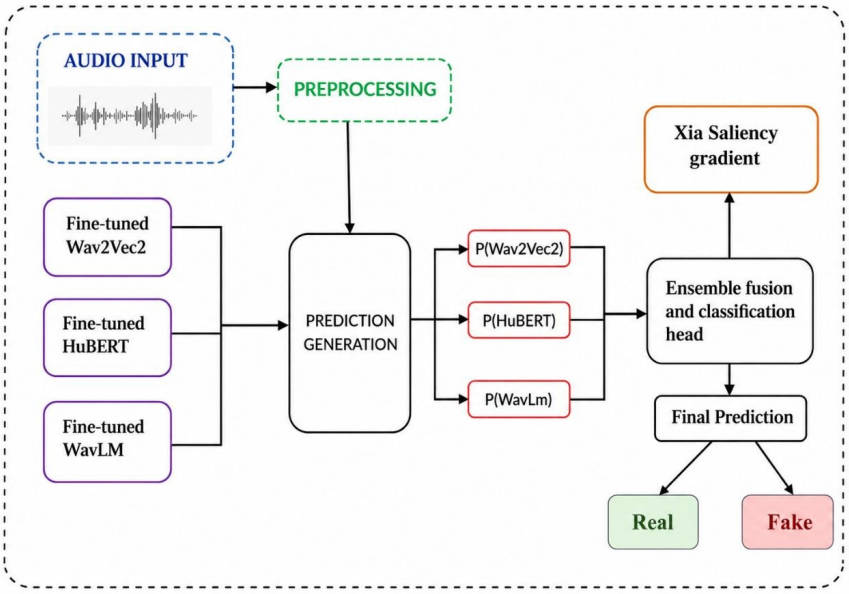
The processed audio is simultaneously fed into three pretrained speech models Wav2Vec2.0, HuBERT, and WavLM which are individually fine-tuned on the target task prior to integration, enhancing their discriminative capability for spoof detection. Each branch produces contextualized representations that are aggregated using attentive statistics pooling (capturing weighted mean and standard deviation), followed by a shared SLS classifier head that outputs bonafide/spoof logits and probabilities.

The predictions from the three models are then combined using multiple ensemble strategies (e.g., soft voting, logit averaging, and majority voting), with the optimal method selected based on development set performance and subsequently applied to the evaluation set for final reporting using both classification metrics, including Accuracy, Precision, Recall, and F1-score, and the security-oriented metric Equal Error Rate (EER), which is widely used in deepfake detection tasks to measure the trade-off between false acceptance and false rejection errors

The selected ensemble pathway is also reused for explainability through gradient-based saliency maps, enabling the identification of the most informative temporal regions and providing insight into the model's decision-making process, as well as for consistent single-audio inference.



(a) Training process



(b) Classification process

Figure 3.1: General schema of the proposed audio deepfake detection framework

3.3 Base models

We propose a deepfake audio detection system that operates on raw speech signals processed through a unified and model-agnostic pipeline. First, each audio signal is loaded and converted to mono when necessary. It is then resampled to a fixed sampling rate of 16 kHz to ensure consistency across all inputs. Waveform normalization is applied to stabilize amplitude variations, and extremely short audio samples are discarded to avoid unreliable predictions. For long recordings, two strategies are used: during training, audio inputs are limited to a fixed maximum duration, while during inference, long audio files are divided into overlapping segments to preserve temporal structure while handling variable-length inputs.

To improve generalization, the training pipeline optionally applies data augmentation by adding random white Gaussian noise to the input audio. A feature extraction module converts raw waveforms into model-ready representations, while padding is applied to standardize batch lengths. An attention mask is also generated to distinguish valid audio frames from padding, ensuring that the model does not learn from artificial padded regions.

The processed input is then passed to a pretrained self-supervised speech representation backbone (e.g., Wav2Vec 2.0, HuBERT, or WavLM). These models typically consist of a convolutional feature encoder that extracts low-level acoustic representations from the waveform, followed by a Transformer-based context network that captures long-range temporal dependencies across the speech sequence. These backbones are pretrained using self-supervised learning objectives that learn robust speech representations without requiring large labeled datasets.

On top of the pretrained encoder, a task-specific classification head is added. It begins with an Attentive Statistics Pooling (ASP) layer, which learns attention weights over temporal frames. These weights are used to compute a weighted mean and weighted standard deviation of frame-level embeddings, which are then concatenated into a fixed-length utterance-level representation. Dropout regularization is applied to reduce overfitting.

The resulting embedding is passed through a Multi-Layer Perceptron (MLP) classifier composed of linear layers, non-linear activations, and normalization components. The final layer outputs logits for two classes: bonafide (genuine speech) and spoof (manipulated speech). These logits are converted into probabilities using the softmax function, and final predictions are obtained using argmax. During training, the model is optimized using the AdamW optimizer with discriminative learning rates applied to different parts of the network (backbone and classification head). A cosine annealing learning rate scheduler is used to gradually reduce the learning rate over epochs. Gradient clipping is applied to stabilize training and prevent exploding gradients. The model is trained using cross-entropy loss. The evaluation process includes multiple performance metrics such as accuracy, F1-score, ROC-AUC, Precision-Recall AUC, and Equal Error Rate (EER), which is a key metric for spoof detection tasks. A checkpointing mechanism is implemented to save both the latest and best-performing models based on validation EER. This allows full training resumption, including model weights, optimizer state, scheduler state, and training history.

During evaluation, the system generates a confusion matrix and multiple diagnostic plots, including ROC curves, Precision-Recall curves, Detection Error Tradeoff (DET) curves, and training history visualizations. These artifacts are automatically saved for analysis and reporting purposes. The system also includes an Explainable AI (XAI) module based on gradient-based saliency methods and Integrated Gradients (when available). This module identifies the most influential temporal regions in the waveform that contribute to the model's decision. Saliency maps are visualized

alongside the raw waveform, and the most important regions are extracted as interpretable time segments.

For inference, the system supports both single-file and long-form audio classification. Long audio recordings are split into overlapping segments, each of which is independently processed by the model. The final decision is obtained by aggregating segment-level predictions using either mean probability fusion or majority voting, along with confidence estimation for each class. Overall, the system integrates self-supervised speech representation learning, attentive pooling mechanisms, supervised classification, advanced optimization strategies, training stabilization techniques (such as freezing and scheduling), explainability tools, and segment-level inference aggregation. This makes it a robust and practical framework for real-world deepfake audio detection under diverse and variable conditions.

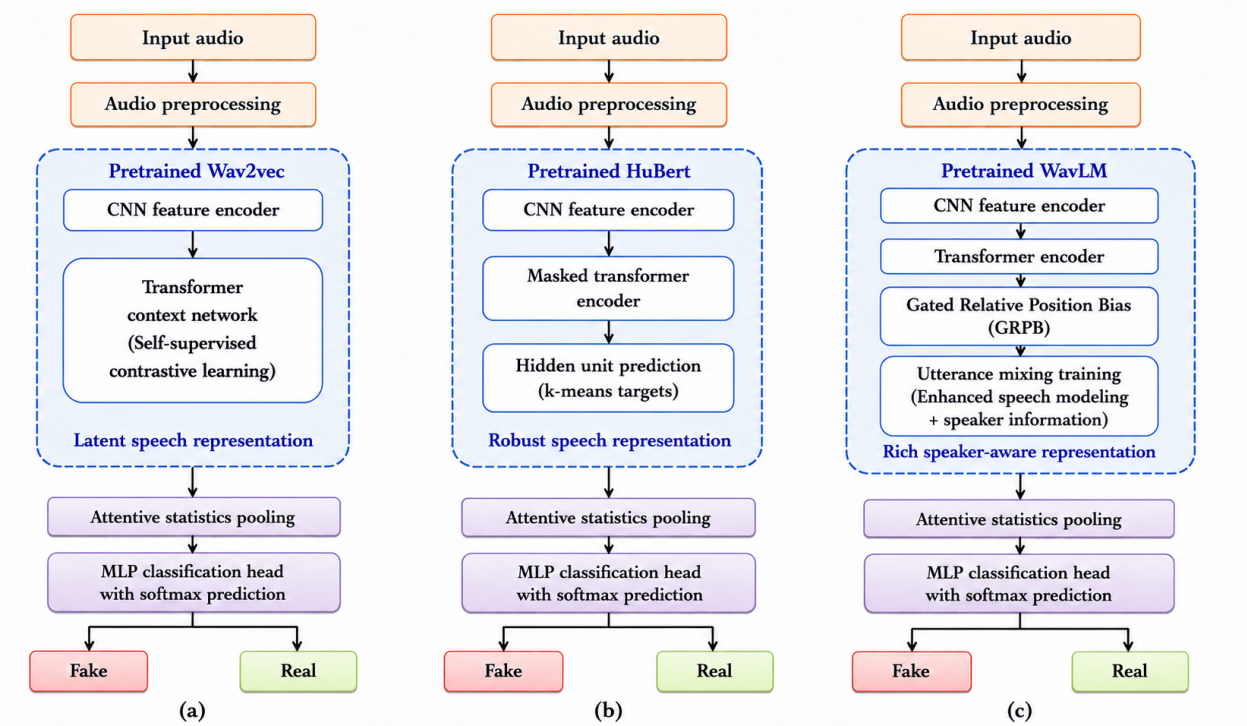


Figure 3.2: Schemas of base models used for the ensemble learning: (a) Wav2Vec 2.0 model, (b) HuBERT model, and (c) WavLM model.

The Wav2Vec 2.0 model: Wav2Vec 2.0 is a self-supervised speech representation learning model that learns powerful acoustic features directly from raw audio waveforms. It consists of a convolutional neural network (CNN) feature encoder that extracts low-level representations, followed by a Transformer network that models long-range temporal dependencies across the speech sequence. The model is pretrained using a contrastive learning objective, where masked latent speech representations are predicted by distinguishing the correct quantized targets from a set of negative samples. This allows the model to learn robust and informative speech embeddings without requiring large amounts of labeled data. A key strength of Wav2Vec 2.0 lies in its ability to learn high-quality general-purpose speech features that transfer effectively to downstream tasks such as deepfake detection.

The HuBERT model: HuBERT (Hidden-Unit BERT) is a self-supervised speech representation model that learns by predicting discrete hidden units obtained through clustering-based pseudo-labeling. Unlike contrastive methods, HuBERT first applies a clustering algorithm (e.g., k-means) to acoustic features in order to generate pseudo-labels, which serve as targets during training. The model is trained in a masked prediction framework, where it learns to predict these hidden units for masked regions of the input sequence. Its architecture includes a convolutional feature encoder followed by a Transformer encoder that captures contextual information over time. A key advantage of HuBERT is its iterative refinement process, where improved representations lead to better pseudo-labels, resulting in increasingly stable and semantically meaningful speech embeddings.

The WavLM model: WavLM is an advanced self-supervised speech representation model designed to improve robustness in real-world and noisy audio environments. It shares a similar architecture with Wav2Vec 2.0 and HuBERT, consisting of a CNN feature encoder followed by a Transformer-based context network. However, WavLM introduces significant improvements in its pretraining strategy, including speech denoising and multi-task learning objectives that enhance its ability to handle overlapping speech and background noise. A key architectural innovation in WavLM is the use of Gated Relative Position Bias, which improves the modeling of temporal relationships by dynamically controlling how positional information influences attention weights. This mechanism allows the model to better capture long-range dependencies while remaining robust to variations in speech structure. As a result, WavLM produces more stable and context-aware representations, making it particularly effective for challenging speech-related tasks such as deepfake audio detection in real-world conditions.

3.4 Ensemble methods

After fine-tuning the base models on ASVspoof 2019 dataset, we explored various ensemble learning techniques to combine their predictions and leverage their collective strengths. The ensemble approaches investigated in this study are as follows:

3.4.1 Mean Soft Voting

Mean Soft Voting is an ensemble learning technique used to combine the outputs of multiple classification models. It averages the predicted probability scores of each class from all models and then chooses the class with the highest average probability. This approach generally provides more reliable and robust predictions because it takes into account the confidence level of each model rather than only their final decisions. As shown in Figure 3.3, an audio input is provided to three independently fine-tuned self-supervised speech models: Wav2Vec 2.0, HuBERT, and WavLM. Each model extracts different acoustic and contextual representations from the same audio signal and generates its own prediction probabilities through a prediction generation stage. These outputs are represented as Prediction 1, Prediction 2, and Prediction 3. Rather than relying on a single model, the system applies a soft voting mechanism that averages the class probabilities produced by the three models. By integrating the strengths of Wav2Vec 2.0, HuBERT, and WavLM, the ensemble reduces individual model biases and errors, leading to a more accurate and stable final decision. The class with the highest averaged probability is selected as the final predicted class.

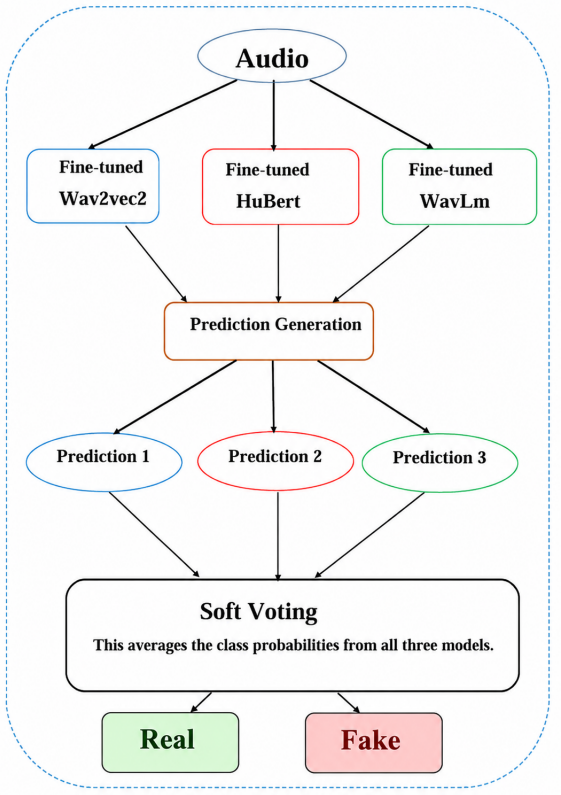


Figure 3.3: Proposed soft voting ensemble method

3.4.2 Weighted Soft Voting

Weighted Soft Voting is an ensemble learning technique that combines the probability outputs of multiple classifiers while assigning different importance weights to each model. Unlike standard soft voting, where all models contribute equally, weighted soft voting multiplies each model's predicted class probabilities by a predefined weight that reflects its performance, reliability, or confidence. The weighted probabilities are then averaged, and the class with the highest final score is selected as the prediction. This approach allows stronger models to have a greater influence on the final decision, often leading to improved classification performance. Figure 3.4 presents a weighted soft voting ensemble framework for audio classification. Initially, an audio input is processed by three independently fine-tuned self-supervised learning models: Wav2Vec 2.0, HuBERT, and WavLM. Each model extracts distinctive speech representations and generates a set of class probability predictions through the prediction generation stage. These outputs are represented as Prediction 1, Prediction 2, and Prediction 3. Before combining the predictions, a specific weight (W1, W2, and W3) is assigned to each model according to its classification performance or importance. The weighted soft voting module then computes a weighted combination of the predicted probabilities, giving greater influence to models with higher weights. By leveraging the complementary strengths of the three fine-tuned models while emphasizing the most reliable ones, the ensemble produces a more accurate and robust final classification. The class associated with the highest weighted probability score is ultimately selected as the system's output.

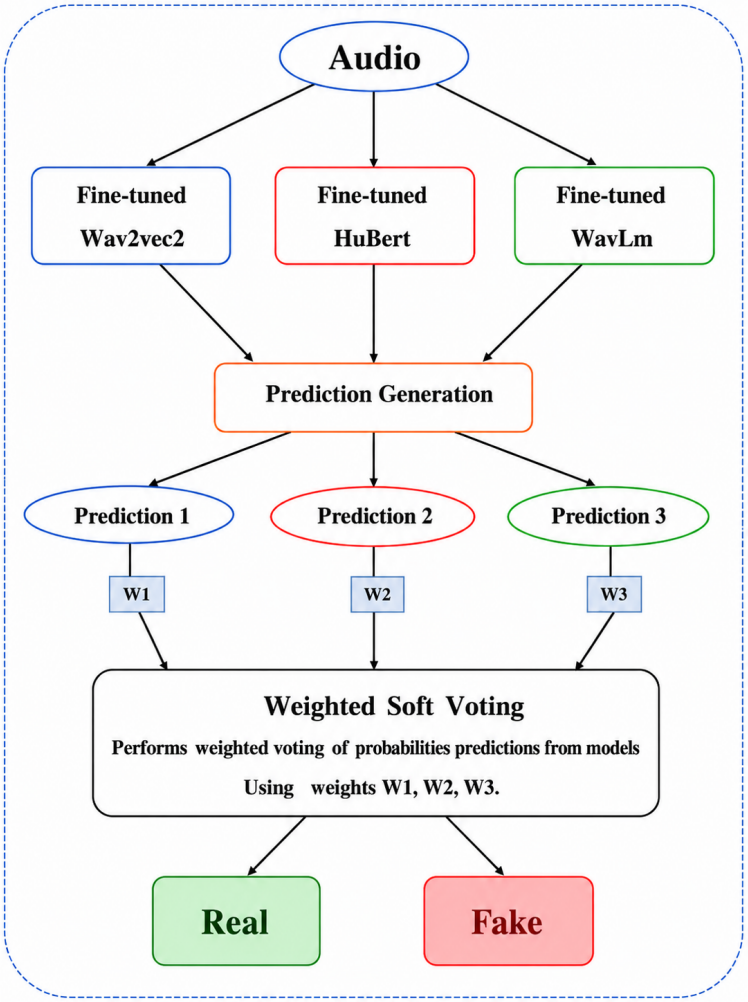


Figure 3.4: Proposed Weighted Soft Voting ensemble method

3.4.3 Logit Averaging

Logit Averaging is an ensemble learning technique that combines the raw output scores (logits) of multiple classifiers before converting them into probabilities. A logit is the unnormalized output produced by a model prior to the application of the softmax function. In logit averaging, the logits generated by different models are averaged element-wise to form a single ensemble logit vector. The softmax function is then applied to the averaged logits to obtain the final class probabilities, and the class with the highest probability is selected as the prediction. This method often provides better calibration and preserves more information than averaging probabilities directly, as the combination occurs before the non-linear softmax transformation.

Figure 3.5 illustrates a logit averaging ensemble framework for audio classification. First, the input audio signal is processed independently by three fine-tuned self-supervised speech models: Wav2Vec 2.0, HuBERT, and WavLM. Each model generates a vector of logits, denoted as Z_1, Z_2, and Z_3, representing the confidence scores for each target class before probability normalization.

These logits are then combined through an ensemble layer that computes their element-wise average, producing a single averaged logit vector:

$$Z_{\text{avg}} = \frac{Z_1 + Z_2 + Z_3}{3}. \tag{3.1}$$

The resulting ensemble logits are subsequently passed through a softmax function to convert them into class probability distributions:

$$P(y = i) = \frac{\exp\left(Z_{\text{avg},i}\right)}{\sum_{j=1}^{C} \exp\left(Z_{\text{avg},j}\right)} \tag{3.2}$$

where C denotes the total number of classes and $P(y = i)$ represents the probability assigned to class i.

By aggregating the raw confidence scores of the three models prior to probability estimation, the ensemble effectively captures complementary information from each model and reduces individual prediction variance. Finally, the class associated with the highest probability is selected as the output of the audio classification system:

$$\hat{y} = \arg\max_{i} P(y = i). \tag{3.3}$$

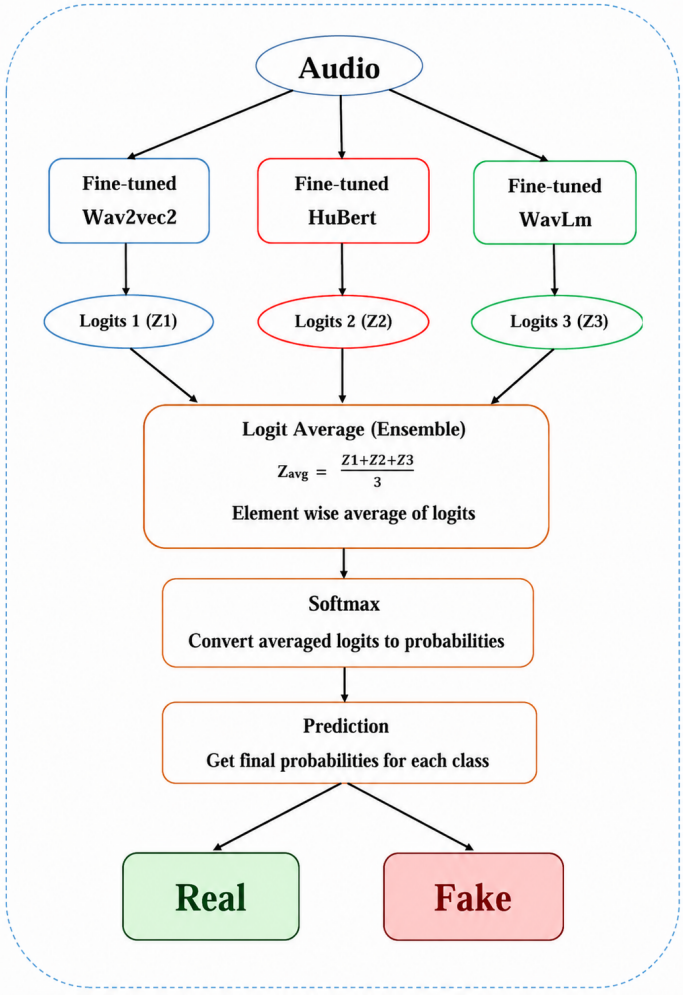


Figure 3.5: Proposed Logit average ensemble method

3.4.4 Hard Majority Voting

Hard Voting (Majority Voting) is an ensemble learning technique that combines the final class predictions of multiple classifiers by selecting the class that receives the highest number of votes. Unlike soft voting, which relies on predicted probabilities, hard voting considers only the class labels predicted by each model. The class predicted by the majority of the models is chosen as the final output. This approach is simple, computationally efficient, and effective when the individual models exhibit complementary strengths and make independent prediction errors.

Figure 3.6 illustrates a hard voting ensemble framework for audio classification. First, the input audio signal is processed independently by three fine-tuned self-supervised speech models: Wav2Vec 2.0, HuBERT, and WavLM. Each model extracts relevant acoustic and contextual features and generates its own class prediction through the prediction generation stage. These outputs are represented as Prediction 1, Prediction 2, and Prediction 3. The individual predictions are then forwarded to the hard voting module, where each model contributes one vote to its predicted class. The ensemble selects the class label that receives the majority of votes among the three models. By combining the decisions of multiple fine-tuned models, the hard voting strategy reduces the impact of incorrect individual predictions and improves the overall robustness of the classification system. The class with the highest number of votes is ultimately selected as the final output.

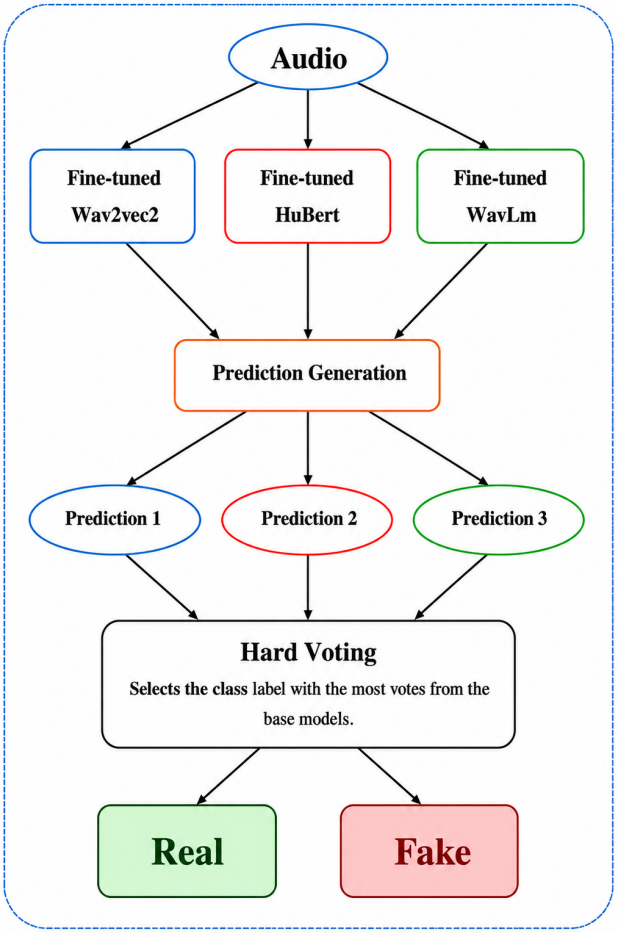


Figure 3.6: Proposed Hard Voting Majority ensemble method

3.4.5 Max Confidence

Max Confidence Ensemble is an ensemble learning technique that selects the prediction produced by the model with the highest confidence score. In this approach, each model independently generates a probability distribution over the target classes using the softmax function. The confidence score is defined as the maximum probability assigned to the predicted class. Rather than combining probabilities or votes, the ensemble compares the confidence scores of all models and chooses the prediction from the model with the highest confidence. This method assumes that the most confident model is more likely to provide the correct classification and can be particularly effective when confidence estimates are well calibrated.

The Figure 3.7 illustrates a max confidence ensemble framework for audio classification. First, the input audio signal is processed independently by three fine-tuned self-supervised speech models: Wav2Vec 2.0, HuBERT, and WavLM. Each model generates a vector of logits, denoted as Z_1, Z_2, and Z_3, which are subsequently transformed into probability distributions P_1, P_2, and P_3 through the softmax function. For each model, the highest probability value is extracted as the confidence score, and the corresponding class label is identified as the predicted class. The ensemble module then compares the confidence scores produced by the three models and selects the prediction associated with the maximum confidence value. By choosing the prediction from the model that expresses the greatest certainty, the max confidence strategy aims to leverage the strongest decision among the ensemble members while maintaining a simple and computationally efficient decision process. The selected class is ultimately returned as the final output of the audio classification system.

$$\hat{y} = \arg \max_{m \in \{1,2,3\}} \left(\max_{c} P_m(c) \right), \tag{3.4}$$

where $P_m(c)$ denotes the probability assigned to class c by model m, and the final prediction is taken from the model that produces the highest confidence score.

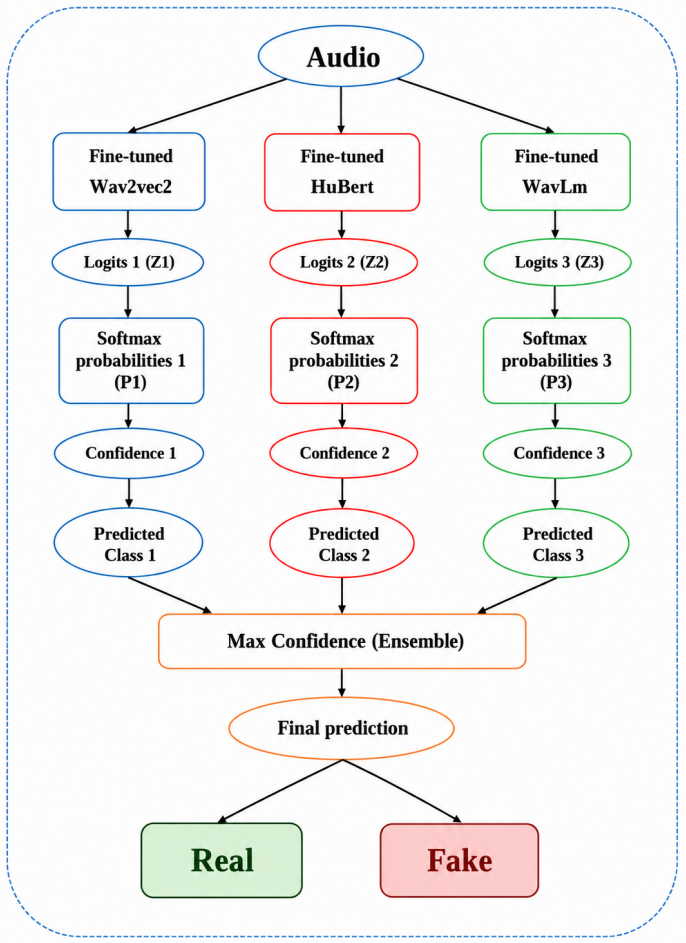


Figure 3.7: Proposed max confidence ensemble method

3.4.6 Bagging Soft Voting

Bagging Soft Voting is an ensemble learning technique that combines bootstrap aggregating (bagging) with weighted soft voting to improve prediction stability and robustness. In this approach, multiple ensemble instances are generated by randomly sampling base models with replacement from a pool of classifiers. As a result, the same model may appear multiple times within a sampled ensemble, while others may be omitted. For each sampled group, a weighted soft voting strategy is applied, where the predicted class probabilities are combined according to predefined model weights. The outputs of all bagged ensembles are then averaged to produce the final prediction. By aggregating the results of numerous randomly generated ensembles, bagging soft voting reduces the dependence on any single model combination and improves the overall reliability of the classification system.

The Figure 3.8 illustrates a bagging soft voting framework for audio classification using three fine-tuned self-supervised speech models: Wav2Vec 2.0, HuBERT, and WavLM. Given an input audio signal, multiple bagging iterations are performed by randomly selecting models with replacement from the available model pool. For example, one iteration may sample {Wav2Vec 2.0, Wav2Vec2.0, HuBERT}, while another may sample {HuBERT, WavLM, WavLM}. During each

iteration, the selected models generate class probability predictions, which are combined using weighted soft voting. This process is repeated for a predefined number of bagging rounds, denoted by the parameter `bagging_estimators`, where the number of bagging estimators is set to 101 in this work., meaning that 101 bagged ensembles are generated. The probability outputs from all iterations are subsequently averaged to obtain the final ensemble prediction. This two-stage aggregation process exploits both the diversity introduced by bootstrap sampling and the confidence information provided by weighted soft voting, resulting in a more stable and robust classification decision.

$$P^{(b)}(c) = \frac{\sum_{m=1}^{M_b} w_m P_m(c)}{\sum_{m=1}^{M_b} w_m}, \tag{3.5}$$

where $P^{(b)}(c)$ is the weighted soft-voting probability of class c obtained during bagging iteration b, $P_m(c)$ is the probability predicted by model m, and w_m is the corresponding model weight.

The final bagging soft-voting probability is computed as

$$P_{\text{final}}(c) = \frac{1}{B} \sum_{b=1}^{B} P^{(b)}(c), \tag{3.6}$$

where B denotes the number of bagging iterations ($B = 101$ in this work). The final predicted class is then given by

$$\hat{y} = \arg\max_{c} P_{\text{final}}(c). \tag{3.7}$$

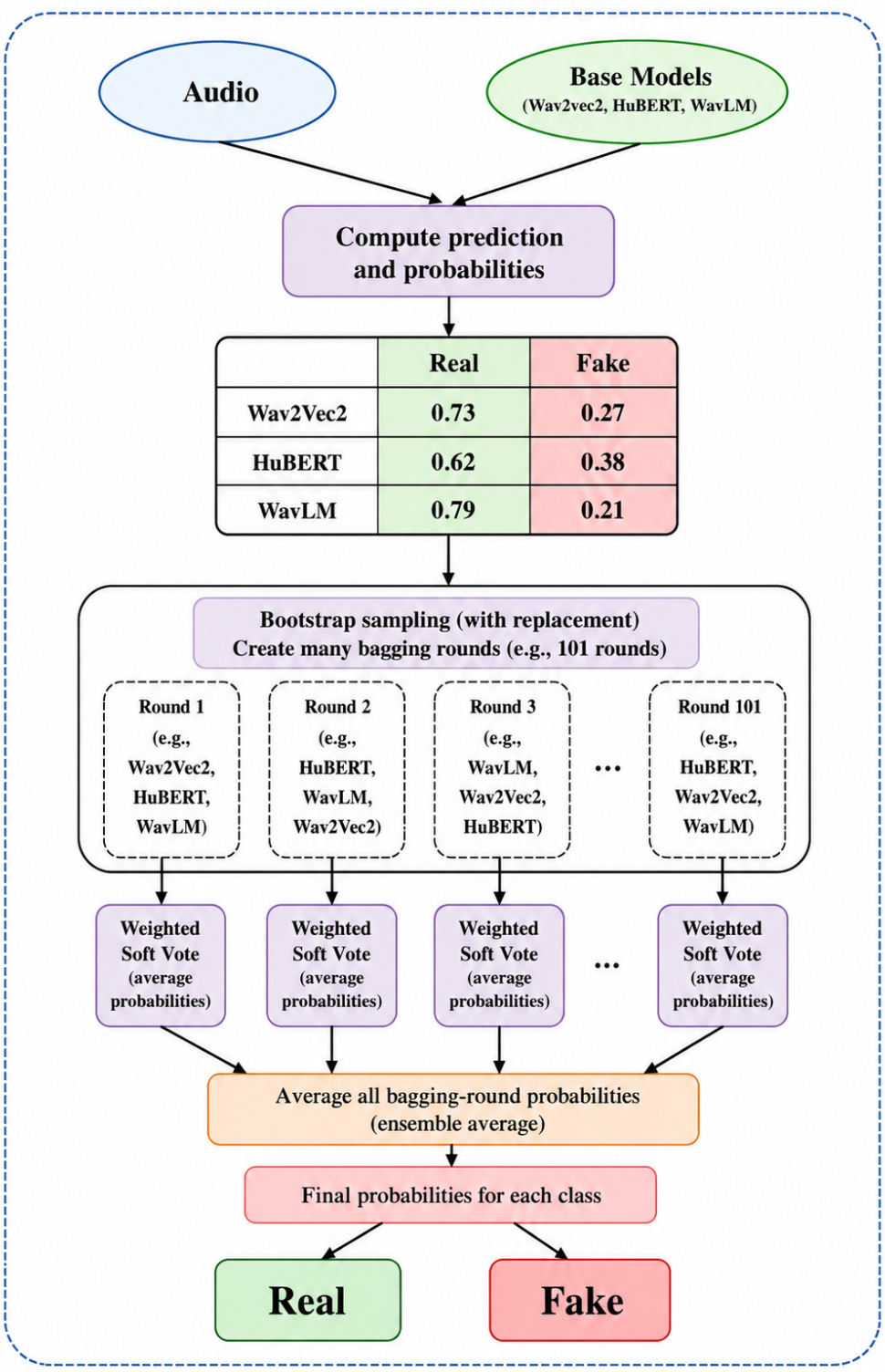


Figure 3.8: Proposed Bagging Soft Voting ensemble method

3.4.7 Stacking with Logistic Regression

Stacking with Logistic Regression is an ensemble learning technique that combines the predictions of multiple base models through a trainable meta-model. Unlike voting-based methods, which rely on predefined aggregation rules such as averaging probabilities or selecting the majority class, stacking learns how to optimally combine the outputs of the base classifiers from data. In this approach, the predictions generated by the base models are used as input features for a second-level classifier, known as the meta-model. The meta-model learns the relationships between the outputs of the base models and the true class labels, allowing it to identify which model should be trusted under different circumstances.

In this work, Logistic Regression is employed as the meta-model due to its simplicity, interpretability, and effectiveness in combining heterogeneous prediction sources. Since the stacking features consist of low-dimensional prediction scores, probabilities, and logits generated by the base models, a linear classifier is sufficient to capture their relationships while reducing the risk of overfitting. Furthermore, Logistic Regression requires fewer hyperparameters and lower computational cost than more complex alternatives such as Support Vector Machines (SVMs). These advantages have made Logistic Regression one of the most commonly adopted meta-learners in stacking ensembles. According to Seng et al [71]. Logistic Regression is the most frequently used meta-learner in the stacking literature for classification problems. Therefore, it represents a robust and well-established choice for learning the optimal combination of the base model predictions.

Figure 3.9 illustrates a stacking ensemble framework for audio classification using three fine-tuned self-supervised speech models: Wav2Vec 2.0, HuBERT, and WavLM. First, each model processes the input audio signal and generates prediction-related outputs, including class probabilities, logits, and spoof scores. During the training phase, these outputs are collected from the development set and concatenated to form a feature vector for each audio sample. The resulting feature vectors serve as inputs to a Logistic Regression meta-model, which learns how to combine the strengths of the three base models. Rather than applying a fixed averaging strategy, the meta-model automatically discovers patterns in the prediction behavior of each classifier. For example, it may learn to assign greater importance to HuBERT and WavLM when their predictions strongly agree, or to rely on HuBERT when Wav2Vec 2.0 exhibits low confidence. Once trained, the Logistic Regression model is applied to the outputs generated by the base models on the evaluation set to produce the final classification decision. By learning data-driven combination rules, stacking can exploit complementary information among the base models and often achieve higher predictive performance than traditional ensemble methods.

$$\mathbf{x} = [P_{\text{W2V}}, P_{\text{HuBERT}}, P_{\text{WavLM}}, Z_{\text{W2V}}, Z_{\text{HuBERT}}, Z_{\text{WavLM}}, S_{\text{W2V}}, S_{\text{HuBERT}}, S_{\text{WavLM}}], \tag{3.8}$$

where P, Z, and S denote the probabilities, logits, and spoof scores produced by the base models, respectively.

The Logistic Regression meta-model estimates the probability of class membership as

$$P(y = 1 \mid \mathbf{x}) = \frac{1}{1 + \exp\left(-(\mathbf{w}^T \mathbf{x} + b)\right)}, \tag{3.9}$$

where $\mathbf{w}$ represents the learned weight vector and b is the bias term.

The final prediction is obtained as

$$\hat{y} = \arg\max_{c} P(y = c \mid \mathbf{x}). \tag{3.10}$$

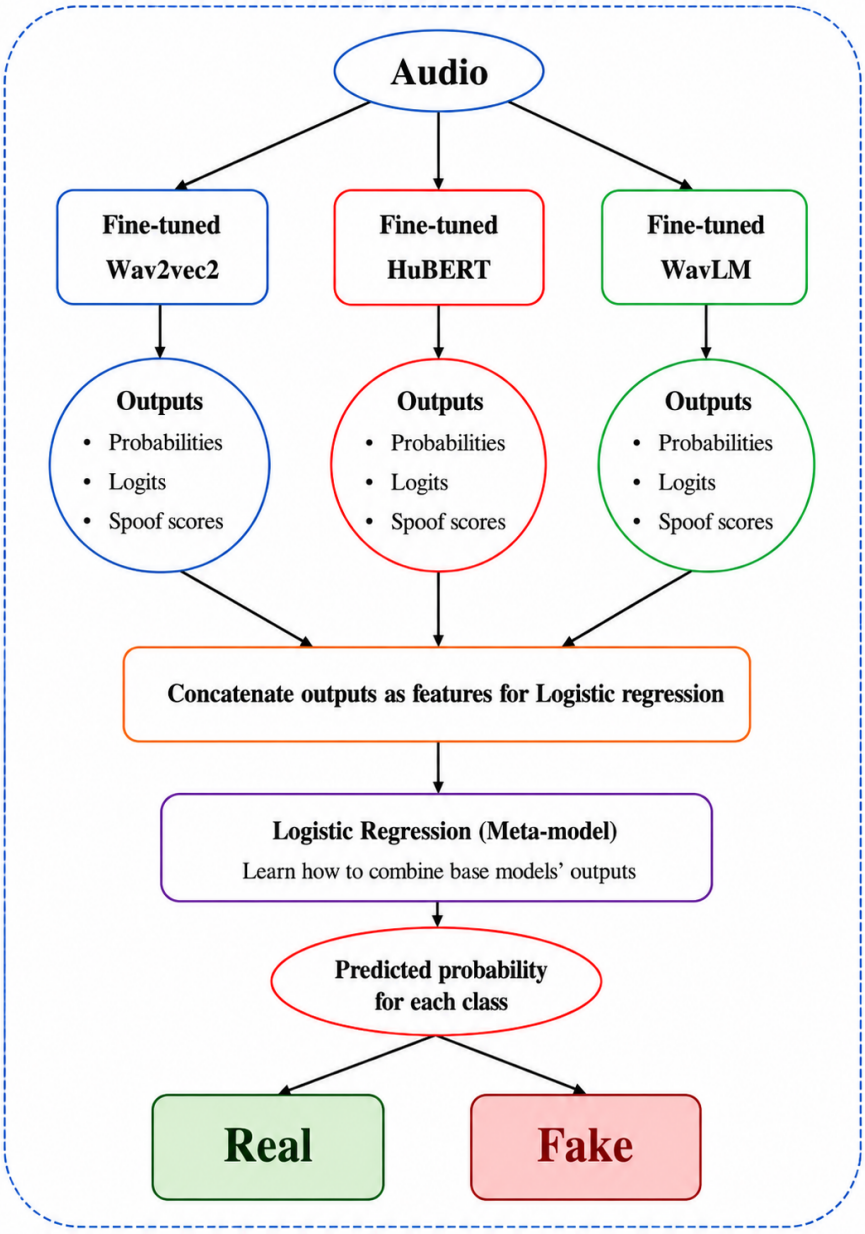


Figure 3.9: Proposed Stacking Logistic Regression ensemble method

3.5 Single-Audio Inference Pipeline

To demonstrate the practical applicability of the proposed anti-spoofing system, a single-audio inference pipeline was developed. The pipeline enables the classification of an unseen speech recording as either bona fide or spoof by applying the trained ensemble model. First, an input

audio file is loaded and validated. The system accepts several common audio formats, including FLAC, WAV, MP3, OGG, and M4A. The recording is converted to mono channel, resampled to a sampling rate of 16 kHz, normalized, and truncated to a maximum duration of 6 seconds. Very short recordings are rejected to ensure sufficient speech information for reliable classification. The preprocessed waveform is then independently processed by the three pretrained self-supervised speech encoders: Wav2Vec 2.0, HuBERT, and WavLM. Each encoder is coupled with the same classification architecture, composed of an Attentive Statistics Pooling layer followed by a fully connected classifier head. Each branch produces a pair of output logits and the corresponding posterior probabilities for the two classes. After obtaining the predictions from the individual models, the outputs are combined using the ensemble strategy selected during the evaluation phase. The inference system supports several ensemble techniques, including soft voting, weighted soft voting, logit averaging, hard majority voting, maximum-confidence selection, bagging, and stacking with logistic regression. By default, the best-performing ensemble method identified during the validation is automatically used for inference. Finally, the ensemble generates the predicted class label together with its confidence score. The system outputs a human-readable decision, indicating whether the input recording is classified as REAL (bonafide) or FAKE (spoof), along with the associated probability estimate.

Handling Long Audio Recordings

The proposed inference pipeline is designed for audio segments of up to 6 seconds. For recordings that exceed this duration, a practical consists of dividing the signal into a sequence of shorter overlapping or non-overlapping segments (e.g., 3-second windows). Each segment is classified independently by the ensemble model, and the final decision is obtained through majority voting across all segment-level predictions. An experimental evaluation of this approach showed that it can improve the robustness of the system. In some cases, a long recording was incorrectly classified when processed as a single input, whereas the segmentation-based approach correctly identified most of the individual speech segments, leading to the correct final decision after majority voting. This behavior suggests that spoofing artifacts may be localized in time and that segment-level analysis can reduce the impact of irrelevant or less informative portions of the recording. Therefore, segment-based inference combined with majority voting represents a promising extension for improving the reliability of the proposed anti-spoofing framework when dealing with long-duration audio samples.

3.6 Conclusion

In this chapter, we presented the design of the proposed SSL-based ensemble framework for audio deepfake detection. We first introduced the overall architecture of the system, including the preprocessing pipeline, data preparation strategy, and the integration of multiple self-supervised learning models. The proposed framework relies on three powerful transformer-based models, namely Wav2Vec 2.0, HuBERT, and WavLM, which were selected for their ability to learn rich and complementary speech representations from raw audio signals.

Furthermore, we described the different ensemble learning techniques investigated in this work, including Soft Voting, Weighted Soft Voting, Logit Averaging, Hard Voting, Max Confidence, Bagging Soft Voting, and Stacking with Logistic Regression. These methods were designed to combine the strengths of the individual models while reducing their weaknesses, with the objective

of improving robustness, generalization capability, and detection performance.

In addition, we presented a practical single-audio inference pipeline that allows the trained ensemble to classify unseen speech recordings and provide confidence scores for its predictions. We also discussed a segment-based strategy for handling long audio recordings, where recordings exceeding the predefined duration are divided into shorter segments and classified independently before applying majority voting. This approach improves robustness and extends the applicability of the proposed framework to real-world audio samples of varying lengths.

In general, the proposed framework provides a comprehensive approach for audio deepfake detection by leveraging both the representation power of self-supervised transformer models and the advantages of ensemble learning.

The next chapter will present the experimental setup and evaluation results, where the performance of the proposed models and ensemble strategies will be analyzed and compared using different classification metrics, including Accuracy, Precision, Recall, and F1-score, and the security-oriented metric Equal Error Rate (EER).

Chapter 4

Results and Discussion

4.1 Introduction

This chapter presents the experimental evaluation and analysis of the proposed audio deepfake detection framework. The objective of these experiments is to assess the effectiveness of the individual self-supervised learning models, namely Wav2Vec 2.0, HuBERT, and WavLM, as well as the different ensemble learning strategies proposed in this work. The evaluation is conducted using the ASVspoof 2019 Logical Access (LA) dataset and is based on both classification metrics, including Accuracy, Precision, Recall, and F1-score, and the security-oriented metric Equal Error Rate (EER), which is widely used in spoofing detection tasks to measure the trade-off between false acceptance and false rejection errors.

This chapter first describes the hardware and software environment used for training and evaluation, together with the experimental settings and implementation details. It then presents the performance of the individual models, followed by a comparative analysis of the ensemble methods, including Soft Voting, Weighted Soft Voting, Logit Averaging, Hard Voting, Max Confidence, Bagging Soft Voting, and Stacking with Logistic Regression. Furthermore, ablation studies and cross-dataset evaluations are conducted to investigate the contribution of each model and the generalization capability of the proposed framework. Finally, explainability results based on Gradient Saliency are presented to provide insights into the decision-making process of the ensemble model and to identify the speech regions that contribute most significantly to the classification outcomes.

4.2 Software and tools

The main software platforms and tools used throughout this study are presented as follows:

4.2.1 Google Colab

Colaboratory, or 'Colab' for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary Python code through the browser, and is especially well suited to machine learning, data analysis and education [65]. We utilized Google Colab as the main development and experimentation environment for implementing, training, and evaluating the proposed audio deepfake detection models. Its cloud-based Jupyter notebook interface provided access

to GPU resources, enabling efficient fine-tuning of transformer-based speech models, including Wav2Vec2, HuBERT, and WavLM. Furthermore, the platform supported the evaluation of various ensemble learning approaches and facilitated the analysis of model performance on the audio datasets.

4.2.2 Kaggle

It is an online community of data scientists and machine learning engineers. Kaggle allows users to find datasets they want to use in building AI models, publish datasets, work with other data scientists and machine learning engineers, and enter competitions to solve data science challenges. It also provides cloud-based notebooks similar to Google Colab, allowing users to write, run, and share Python code directly through a web browser without requiring local installation [66]. We used Kaggle to obtain the DeepVoice dataset, which was used to evaluate the performance of our models.

4.2.3 Hugging Face Transformers

The Hugging Face Transformers library is an open-source framework that provides a unified interface for accessing, training, fine-tuning, and deploying state-of-the-art pretrained models across multiple modalities, including natural language processing, computer vision, audio, video, and multimodal tasks. The library centralizes model definitions and offers compatibility with major deep learning frameworks, enabling researchers and developers to easily use pretrained models[67]. In this work, all the pretrained self-supervised speech representation models Wav2Vec 2.0, HuBERT, and WavLM were accessed through the Hugging Face Transformers library, which provides a unified framework for utilizing state-of-the-art pretrained speech models.

4.2.4 Python

Python is a high-level, interpreted, and general-purpose programming language designed to emphasize code readability, simplicity, and developer productivity. It supports multiple programming paradigms, including object-oriented, procedural, and functional programming, and provides an extensive ecosystem of libraries for scientific computing, data analysis, machine learning, and deep learning applications [68].

In this work, Python was used as the primary programming language for the development and evaluation of the proposed audio deepfake detection system. Its rich machine learning ecosystem enabled efficient implementation of data preprocessing, feature extraction, model training, and performance evaluation. In particular, the PyTorch library was employed for deep learning model development and optimization, NumPy for numerical computations, Matplotlib for data visualization, and Scikit-learn for performance evaluation metrics.

4.2.5 Google drive

Google Drive is a cloud-based file storage and synchronization service developed by Google that enables users to store, organize, access, and share files across multiple devices through an internet connection [69].

In our work, Google Drive was used as the central storage platform for managing all artifacts generated during the training and evaluation of the audio deepfake detection models. For each self-supervised learning model (Wav2Vec 2.0, HuBERT, and WavLM), the platform stored the latest training checkpoints, including the current epoch number, evaluation metrics, optimizer state, and early stopping counter information. This allowed interrupted training sessions on Google Colab to be resumed from the last saved state without losing progress.

Additionally, the best-performing checkpoint of each model was preserved separately for final evaluation and ensemble learning. Google Drive also stored the experiment configuration files, training history, prediction outputs, performance reports, and visualization results, including loss curves, confusion matrices, DET, ROC, and Precision–Recall curves, as well as Explainable Artificial Intelligence (XAI) visualizations and summaries.

For the final ensemble model, the best saved models of Wav2Vec 2.0, HuBERT, and WavLM were loaded from Google Drive and used to generate the ensemble predictions for audio deepfake detection.

4.3 Dataset used

Due to computational resource limitations, the experiments were not conducted on the complete ASVspoof2019 LA dataset (the data is presented in Section 2.4.1.1). The models were trained and evaluated using the free version of Google Colab, which provides limited GPU memory and execution time, making it impractical to process the full dataset within the available computational budget. To overcome this constraint, a stratified (balanced) sampling strategy was adopted to reduce the dataset size while simultaneously addressing the class imbalance problem.

The selection process was based on the official ASVspoof2019 protocol files, which provide the filename, the class label (bonafide or spoof), and, for spoofed samples, the corresponding attack type. All bonafide (real) samples were retained, while the number of spoof (fake) samples was reduced to match the number of bonafide samples. Instead of selecting spoof samples completely at random, the sampling process ensured that samples from every attack type were included in the final dataset. Approximately the same number of samples was selected from each attack category whenever possible. If an attack category contained fewer samples than required, all of its samples were retained, and the remaining positions were filled by randomly selecting additional spoof samples from the other attack categories. This approach reduced the dataset size while preserving the diversity of spoofing attacks.

Similarly, a balanced evaluation subset containing 2000 bonafide and 2000 spoof samples was created using the same procedure. Finally, a fixed random seed (of 42) was used throughout the sampling process to ensure the reproducibility of the experiments.

Table 4.1: Number of samples after the balanced sampling strategy.

Dataset Partition	Bonafide	Spoof	Total
Training	2580	2580	5160
Development	2548	2548	5096
Evaluation	2000	2000	4000

The ASVspoof 2019 Logical Access (LA) evaluation set includes a diverse collection of ad-

vanced Text-to-Speech (TTS) and Voice Conversion (VC) attacks [70]. Since the attack-wise analysis in this work is conducted on the evaluation set, Table 4.2 summarizes the characteristics of each spoofing attack to facilitate the interpretation of the experimental results.

Table 4.2: ASVspoof 2019 Logical Access (LA) Evaluation Set Attack Types [70].

Attack	Category	Description
A07	TTS	Neural TTS with WORLD vocoder and GAN-based post-filter.
A08	TTS	Neural TTS using a neural source-filter waveform model.
A09	TTS	LSTM-based statistical parametric TTS with Vocaine vocoder.
A10	TTS	End-to-end neural TTS (Tacotron 2) with WaveRNN vocoder.
A11	TTS	End-to-end neural TTS (Tacotron 2) with Griffin-Lim waveform generation.
A12	TTS	Autoregressive WaveNet-based neural TTS.
A13	VC+TTS	Hybrid voice conversion and TTS with moment-matching conversion.
A14	VC+TTS	Hybrid voice conversion and TTS using ASR bottleneck features and STRAIGHT vocoder.
A15	VC+TTS	Hybrid voice conversion and TTS using ASR bottleneck features and WaveNet vocoder.
A16	TTS	Reference attack using the same waveform concatenation algorithm as A04.
A17	VC	VAE-based voice conversion with waveform filtering.
A18	VC	Non-parallel i-vector/PLDA-based voice conversion.
A19	VC	Reference attack using the same transfer-function voice conversion algorithm as A06.

4.4 Hyperparameters tuning

Table below summarizes the main hyperparameters used in the proposed model, along with their respective roles during tuning and explainability analysis:

Table 4.3: Description of the hyperparameters used in the proposed models.

Hyperparameter	**Definition**
batch_size	Represents the number of training samples processed before updating the model parameters.
num_train_epochs	Specifies the total number of complete passes through the training dataset during training.
encoder_learning_rate	Defines the learning rate used to update the encoder parameters during optimization.
head_learning_rate	Defines the learning rate used to update the task-specific prediction head during training.
weight_decay	Represents the regularization coefficient applied to model weights to reduce overfitting.
dropout	Specifies the probability of randomly deactivating neurons during training to improve model generalization.
patience	Defines the number of consecutive epochs without performance improvement before early stopping is triggered.
seed	Represents the value used to initialize random number generators, ensuring reproducible experimental results.
optimizer	Specifies the optimization algorithm used to update model parameters based on computed gradients.
num_workers	Defines the number of parallel worker processes used for data loading and preprocessing.
max_seconds	Specifies the maximum allowed training time, expressed in seconds.
freeze_feature_encoder	Defines the number of initial epochs during which the feature encoder remains frozen and its parameters are not updated.
train_noise_std	Represents the standard deviation of the Gaussian noise added to training data for data augmentation and robustness enhancement.
xai_samples_per_class	Specifies the number of samples selected from each class for Explainable Artificial Intelligence (XAI) analysis.
xai_num_samples	Defines the total number of samples used to generate explanations in XAI experiments.
xai_steps	Represents the number of interpolation steps used by attribution methods, such as Integrated Gradients, to compute feature importance scores.
xai_internal_batch_size	Specifies the batch size used internally during XAI computations to improve memory efficiency and computational performance.

The hyperparameter values reported in this study were selected through extensive experimental evaluation on the development set. Multiple configurations of learning rates, batch sizes, dropout rates, weight decay values, and training strategies were investigated. The final configuration was chosen as it provided the best trade-off between convergence stability, generalization performance, and computational efficiency.

Table 4.4: Hyperparameters of Wav2Vec2, HuBERT, and WavLM.

Hyperparameter	**Wav2Vec2**	**HuBERT**	**WavLM**
Batch Size	4	4	4
Epochs	15	10	15
Encoder Learning Rate	1e-05	2e-05	8e-06
Head Learning Rate	5e-05	1e-04	4e-05
Weight Decay	0.01	0.01	0.02
Dropout	0.4	0.4	0.3
Patience	3	3	3
Seed	42	42	42
Optimizer	AdamW	AdamW	AdamW
Number of Workers	2	2	2
Max Seconds	6.0	6.0	6.0
Freeze Feature Encoder Epochs	1	1	1
Train Noise Std	0.002	0.004	0.004
XAI Samples Per Class	8	8	8
XAI Num Samples	16	16	16
XAI Steps	16	16	16
XAI Internal Batch Size	1	1	1

Table 4.4 summarizes the final hyperparameter configurations adopted for the three transformer-based models. Although the models share several settings, such as batch size, patience, optimizer, and maximum audio duration, some parameters were adjusted individually to accommodate the characteristics of each architecture. In particular, different learning rates, dropout values, weight decay coefficients, and training epochs were selected to achieve optimal performance and stable convergence. These configurations were determined experimentally and contributed to improving the models' generalization capability on unseen audio samples.

4.5 Experimental results and discussion

This section presents and discusses the experimental results obtained from the proposed approach.

4.5.1 Wav2Vec2

The following results evaluate the performance of the Wav2Vec2 model on the selected dataset.

4.5.1.1 Overall Evaluation Performance

Table 4.5 summarizes the performance of the proposed Wav2Vec2-based anti-spoofing system on the ASVspoof 2019 LA dataset.

Table 4.5: Performance of the fine-tuned Wav2Vec2 model.

Metric	Train	Dev	Eval
Accuracy (%)	93.90	98.55	96.10
Precision (%)	100.00	99.80	99.89
Recall (%)	87.79	97.29	92.30
F1-score (%)	93.50	98.53	95.95
EER (%)	5.31	0.67	1.68

The model achieved an evaluation accuracy of 96.1%, demonstrating strong spoofing detection capability. The high precision of 99.89% indicates that nearly all utterances classified as spoofed were indeed attacks. Furthermore, the recall value of 92.3% confirms excellent separability between bonafide and spoofed speech. The low Equal Error Rate (EER) of 1.68% further highlights the effectiveness of the proposed system in balancing false acceptance and false rejection errors.

4.5.1.2 Confusion Matrix Analysis

Figure 4.1 presents the confusion matrix obtained on the evaluation set.

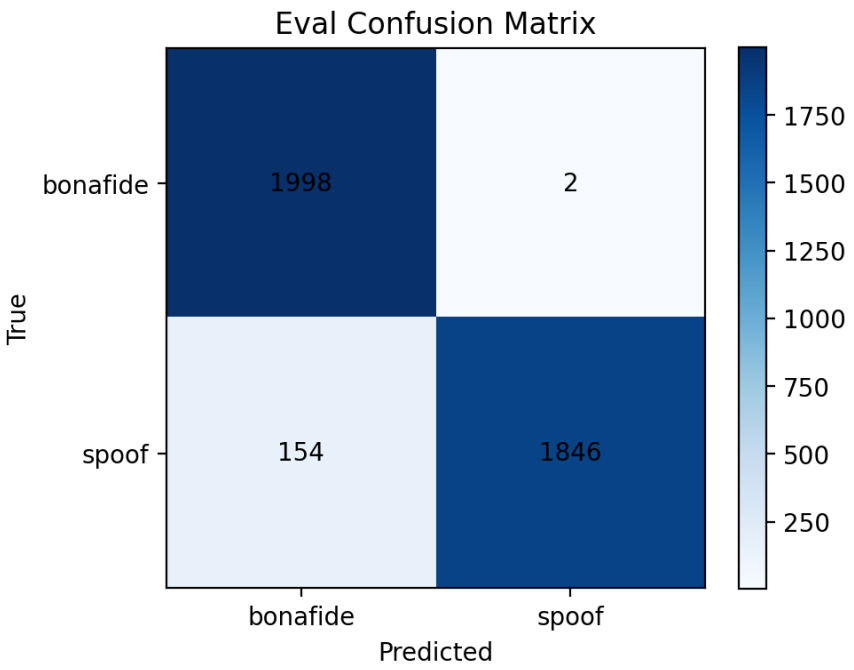


Figure 4.1: Confusion matrix of Wav2Vec 2.0 model

.

The model correctly classified 1998 out of 2000 bonafide utterances and 1846 out of 2000 spoofed utterances. Only two genuine utterances were incorrectly classified as spoofed, while 154 spoofed utterances were misclassified as bonafide.

This behavior is reflected by the high specificity of 99.9% and the very low false acceptance rate (FAR) of 0.1%.

These results indicate that the model is highly conservative when accepting speech as genuine. Such behavior is desirable in security-sensitive speaker verification systems, where accepting a spoofed utterance can have more severe consequences than rejecting a legitimate one. However, the larger number of false negatives suggests that some spoofing attacks remain challenging to detect.

4.5.1.3 Generalization Analysis

To evaluate the generalization capability of the proposed Wav2Vec 2.0-based anti-spoofing system, performance on the development and evaluation sets was compared, as summarized in Table 4.6. Furthermore, the learning curves presented in Figure 4.2 provide insight into the training dynamics and convergence behavior of the model. Together, these results enable an assessment of both the model's learning process and its ability to generalize to previously unseen data.

Table 4.6: Generalization performance from the development set to the evaluation set of the Wav2Vec 2.0 model.

Metric	Dev	Eval	Diff.
Accuracy (%)	98.55	96.10	-2.45
F1-score (%)	98.53	95.95	-2.58
EER (%)	0.67	1.68	+1.01

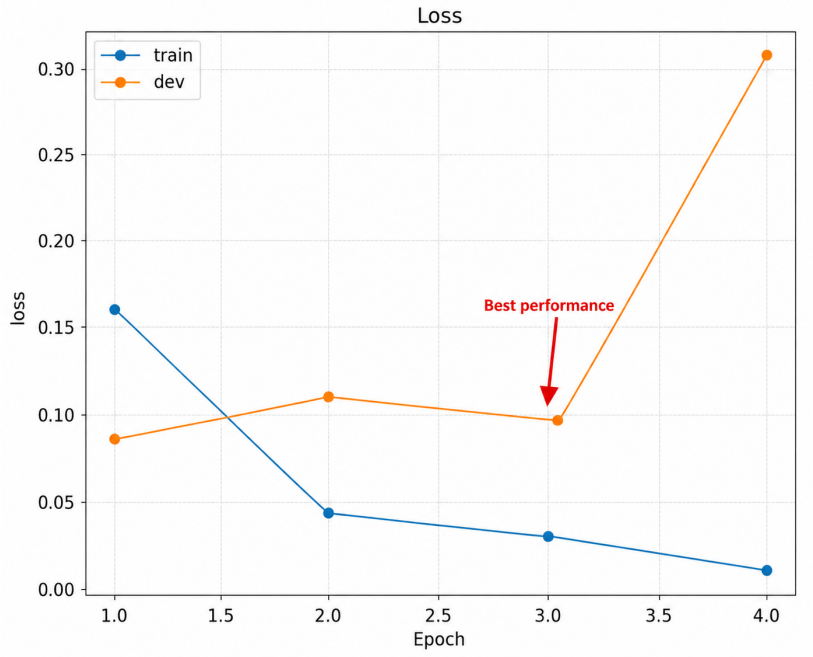


Figure 4.2: Training and development loss curves of the Wav2Vec 2.0 model.

A moderate decrease in performance is observed when moving from the development set to the unseen evaluation set. The accuracy decreases from 98.55% to 96.10%, while the F1-score decreases by 2.58 percentage points. Similarly, the EER increases from 0.67% to 1.68%, reflecting the greater difficulty of the evaluation data, which contains previously unseen utterances and attack conditions.

The training and development loss curves show stable convergence during the initial epochs. The training loss decreases consistently, while the development loss reaches its minimum around Epoch 1 and remains relatively stable before increasing noticeably at Epoch 4. This rise in development loss indicates the onset of overfitting; however, the early stopping mechanism with a patience of three epochs prevented unnecessary additional training and preserved the model corresponding to the best development performance. Furthermore, the relatively small gap between development and evaluation results suggests that the model maintains good generalization capability despite the observed overfitting trend in the later stage of training.

Generally, the model achieves an accuracy above 96%, maintains a ROC-AUC greater than 0.99, and keeps the EER below 2% on the evaluation set, indicating that it can effectively discriminate between bona fide and spoofed speech while preserving good robustness to unseen conditions.

4.5.1.4 Attack-wise Analysis

The attack-wise analysis provides insight into which spoofing techniques are easier or more difficult to detect.

a) Best Detected Attacks

Table 4.7: Best detected spoofing attacks.

Attack	Accuracy (%)
A08	100.00
A09	100.00
A14	100.00
A11	98.05
A07	98.04

The model successfully detected all samples belonging to attacks A08, A09, and A14. These attack types likely contain artifacts that are sufficiently distinct from natural speech, making them easier to identify.

b) Most Difficult Attacks

Table 4.8: Most difficult spoofing attacks.

Attack	Accuracy (%)
A18	70.32
A19	78.71
A16	88.24
A10	88.89
A17	89.54

Attack A18 proved to be the most challenging spoofing condition, achieving a detection accuracy of only 70.32%, which means that nearly 30% of its samples were incorrectly classified as bonafide. Similarly, attacks A19, A16, A10, and A17 exhibited comparatively lower detection rates than the remaining attack types. These results suggest that such spoofing techniques generate acoustic characteristics that closely resemble those of genuine speech, thereby reducing the discriminative cues available to the model and increasing the likelihood of misclassification. In particular, the difficulty associated with A18 may indicate that this attack produces highly realistic synthetic speech patterns that are harder to distinguish from authentic utterances.

The average spoof detection accuracy across all attack types is approximately 92.3%, which closely matches the overall recall obtained on spoofed utterances. The strong agreement between attack-wise performance and overall recall confirms the reliability and consistency of the obtained results. Despite the performance variations observed across attack categories, the model maintained a high detection rate for most spoofing conditions, demonstrating its robustness and ability to generalize effectively across a diverse range of synthetic and manipulated speech attacks.

4.5.1.5 Discussion

The fine-tuned Wav2Vec 2.0 model achieved strong anti-spoofing performance on the ASVspoof 2019 LA evaluation set, obtaining an accuracy of 96.1%, an F1-score of 95.95%, and an Equal Error Rate (EER) of 1.68%. The confusion matrix shows that 1998 of the 2000 bonafide utterances were correctly classified, resulting in a specificity of 99.9% and a False Acceptance Rate (FAR) of only 0.1%. Among the 2000 spoofed utterances, 1846 were correctly detected, corresponding to a recall of 92.3%. These results indicate that the model is highly effective at preserving genuine speech while maintaining strong spoof detection capability. Attack-wise analysis revealed substantial variability across attack types. Attacks A08, A09, and A14 were detected perfectly, whereas A18 and A19 remained challenging, achieving detection accuracies of 70.3% and 78.7%, respectively. The slight decrease in performance from the development set to the evaluation set suggests reasonable generalization to unseen data. Overall, the obtained ROC-AUC of 0.993 and PR-AUC of 0.995 demonstrate excellent class separability and confirm the suitability of Wav2Vec 2.0 as a backbone for automatic speaker verification anti-spoofing systems.

4.5.2 Hubert

The following results evaluate the performance of the Hubert model on the selected dataset.

4.5.2.1 Overall Evaluation Performance

Table 4.9 summarizes the performance of the proposed Hubert-based anti-spoofing system on the ASVspoof 2019 LA dataset.

Table 4.9: Performance of the fine-tuned Hubert model.

Metric	Train	Dev	Eval
Accuracy (%)	99.81	99.84	98.68
Precision (%)	99.84	99.80	99.54
Recall (%)	99.77	99.88	97.80
F1-score (%)	99.81	99.84	98.66
EER (%)	0.14	0.14	1.00

The results obtained by the fine-tuned Hubert model demonstrate its strong capability for audio deepfake detection. On the evaluation set, the model achieved an accuracy of 98.68% , together with a precision of 99.54% and a recall of 97.80% , indicating reliable identification of both spoofed and bonafide utterances. The corresponding F1-score of 98.66% confirms the balanced performance of the classifier across both classes. Furthermore, the EER value of only 1.00% highlights the effectiveness of the model in reducing false acceptance and false rejection rates, making it a highly reliable solution for detecting synthetic and manipulated speech.

4.5.2.2 Confusion Matrix Analysis

Figure 4.3 presents the confusion matrix obtained on the evaluation set.

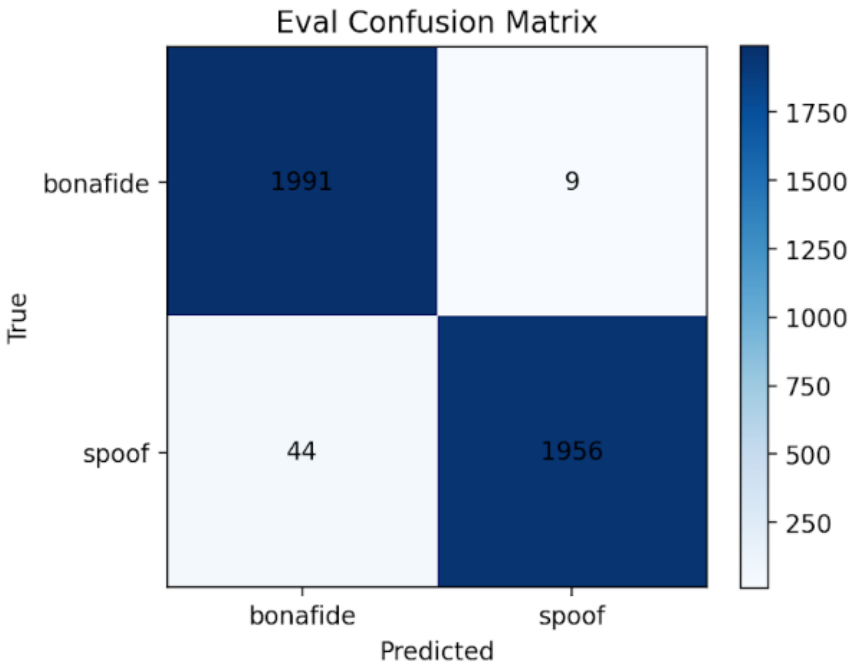


Figure 4.3: Confusion matrix of HuBERT model.

The confusion matrix shows that the fine-tuned Hubert model correctly classified 1991 out of 2000 bonafide utterances and 1956 out of 2000 spoofed utterances. Only 9 genuine utterances were incorrectly classified as spoofed, while 44 spoofed utterances were misclassified as bonafide.

This behavior is reflected by the high specificity of 99.55% and the low false acceptance rate (FAR) of 2.20%.

These results indicate that the model effectively distinguishes between genuine and spoofed speech. The very small number of false rejections demonstrates its ability to correctly recognize authentic utterances, while the limited number of false acceptances highlights its robustness against spoofing attacks. Such behavior is highly desirable in audio deepfake detection and speaker verification systems, where both security and usability are critical. Overall, the confusion matrix confirms the strong generalization capability of the Hubert-based model and is consistent with the low EER value achieved on the evaluation set.

4.5.2.3 Generalization Analysis

To further evaluate the robustness and generalization ability of the proposed Hubert-based anti-spoofing system, its performance on the development and evaluation sets was analyzed, as presented in Table 4.10. Moreover, the learning curves depicted in Figure 4.4 offer a detailed view of the model's optimization process, showing its convergence behavior during fine-tuning. Collectively, these results highlight the effectiveness of the training strategy and confirm the model's capability to generalize well to unseen audio samples while maintaining strong spoofing detection performance.

Table 4.10: Generalization performance from the development set to the evaluation set of the Hubert model.

Metric	Dev	Eval	Diff.
Accuracy (%)	99.84	98.68	-1.16
F1-score (%)	99.84	98.66	-1.18
EER (%)	0.14	1.00	+0.86

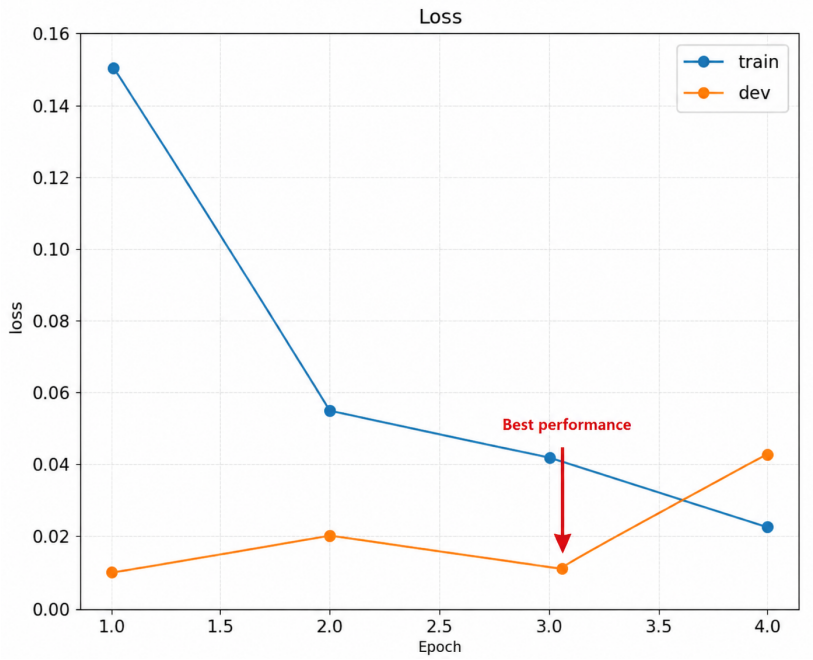


Figure 4.4: Training and development loss curves of the Hubert model.

A slight decrease in performance is observed when moving from the development set to the unseen evaluation set. The accuracy drops from 99.84%to 98.68%, while the F1-score decreases by 1.18%. Similarly, the EER increases from 0.14% to 1.00%, indicating that the evaluation data is more challenging and contains unseen conditions.

The loss curves indicate effective learning, with the training loss steadily decreasing throughout the training process. The development loss reaches its minimum at epoch 3 before increasing slightly at epoch 4, suggesting the onset of overfitting. To mitigate this effect, early stopping with a patience of three epochs was employed, ensuring that the model parameters corresponding to the best development performance were preserved.

Overall, the model maintains strong generalization ability, with high accuracy above 98% and low EER on the evaluation set, showing good robustness to unseen data.

4.5.2.4 Attack-wise Analysis

The attack-wise analysis provides insight into which spoofing techniques are easier or more difficult to detect.

a) Best Detected Attacks

Table 4.11: Best detected spoofing attacks.

Attack	Accuracy (%)
A07	100.00
A08	100.00
A09	100.00
A10	100.00
A11	100.00
A12	100.00
A13	100.00
A14	100.00
A15	100.00
A17	100.00

The Hubert-based model successfully detected all samples belonging to attacks A07, A08, A09, A10, A11, A12, A13, A14, A15, and A17, achieving a perfect detection accuracy of 100% for each attack type. These results indicate that the artifacts introduced by these spoofing techniques are highly distinguishable from genuine speech, enabling the model to identify them without any classification errors.

b) Most Difficult Attacks

Table 4.12: Most difficult spoofing attacks.

Attack	Accuracy (%)
A18	76.77
A19	95.48
A16	99.35

Attack A18 proved to be the most challenging, achieving a detection accuracy of only 76.77%. Approximately 23.23% of A18 samples were incorrectly classified as bonafide. In comparison, attacks A19 and A16 achieved detection accuracies of 95.48% and 99.35%, respectively. Although these values remain high, they suggest that certain spoofing techniques generate speech characteristics that more closely resemble genuine speech, making them more difficult for the model to detect.

The consistent difficulty observed for attack A18 across multiple SSL models suggests that this attack produces acoustic patterns that closely resemble genuine speech, making the discriminative artifacts harder to capture. This observation indicates that highly realistic spoofing attacks remain a significant challenge for current SSL-based anti-spoofing systems. The average spoof detection accuracy across all attack types is approximately 97.80%.

The close agreement between attack-wise accuracy and overall recall confirms the strong generalization capability and robustness of the proposed Hubert-based anti-spoofing system.

4.5.2.5 Discussion

The results obtained from the fine-tuned Hubert model demonstrate its strong effectiveness for audio deepfake detection on the ASVspoof 2019 LA evaluation set. The model achieved an accuracy of 98.68%, an F1-score of 98.66%, and an Equal Error Rate (EER) of only 1.00%, reflecting an excellent balance between detection capability and classification reliability. Analysis of the confusion matrix further highlights this performance, with 1991 out of 2000 bonafide utterances correctly recognized, yielding a specificity of 99.55%, while only 9 genuine utterances were misclassified. In addition, the model successfully detected 1956 out of 2000 spoofed utterances, corresponding to a recall of 97.80%. These findings indicate that the Hubert-based system is capable of accurately preserving genuine speech while effectively identifying manipulated and synthetic audio samples.

A more detailed attack-wise evaluation confirms the robustness of the proposed approach across diverse spoofing conditions. The model achieved perfect detection accuracy (100%) on ten attack types, namely A07, A08, A09, A10, A11, A12, A13, A14, A15, and A17, demonstrating its ability to capture discriminative cues associated with a wide range of spoofing techniques. Nevertheless, performance differences were observed among attack categories. In particular, A18 represented the most challenging attack, reaching an accuracy of 76.77%, whereas A19 achieved 95.48%. This variation suggests that certain spoofing methods generate speech characteristics that more closely resemble genuine utterances, making their detection inherently more difficult.

Regarding generalization, the Hubert model maintained a consistently high level of performance when evaluated on unseen data. Although a slight reduction of 1.16% in accuracy and a moderate increase in EER from 0.14% to 1.00% were observed between the development and evaluation sets, the overall degradation remained limited. This behavior indicates that the model did not excessively overfit the training data and was able to retain its discriminative power under more challenging evaluation conditions. Overall, these results validate the suitability of Hubert as a backbone architecture for automatic speaker verification anti-spoofing systems and highlight its potential for robust real-world detection of synthetic and manipulated speech.

4.5.3 WavLm

The following results evaluate the performance of the WavLm model on the selected dataset.

4.5.3.1 Overall Evaluation Performance

Table 4.13 summarizes the performance of the proposed WavLM-based anti-spoofing system on the ASVspoof 2019 LA dataset.

Table 4.13: Performance of the fine-tuned WavLM model.

Metric	Train	Dev	Eval
Accuracy (%)	99.94	99.84	97.48
Precision (%)	100.00	99.73	99.84
Recall (%)	99.88	99.96	95.10
F1-score (%)	99.94	99.84	97.41
EER (%)	0.00	0.08	0.45

The results obtained by the fine-tuned WavLm model highlight its strong effectiveness in the task of audio deepfake detection. On the evaluation set, the model reached an accuracy of 97.48%, along with a precision of 99.84% and a recall of 95.10%, which reflects a robust ability to correctly identify both bonafide and spoofed speech samples. The resulting F1-score of 97.41% further confirms a well-balanced performance between precision and recall across the two classes. In addition, the low EER value of 0.45% demonstrates a very strong discrimination capability, minimizing both false acceptance and false rejection rates. Overall, these results indicate that the model provides a reliable and effective solution for detecting synthetic and manipulated speech signals.

4.5.3.2 Confusion Matrix Analysis

Figure 4.5 presents the confusion matrix obtained on the evaluation set.

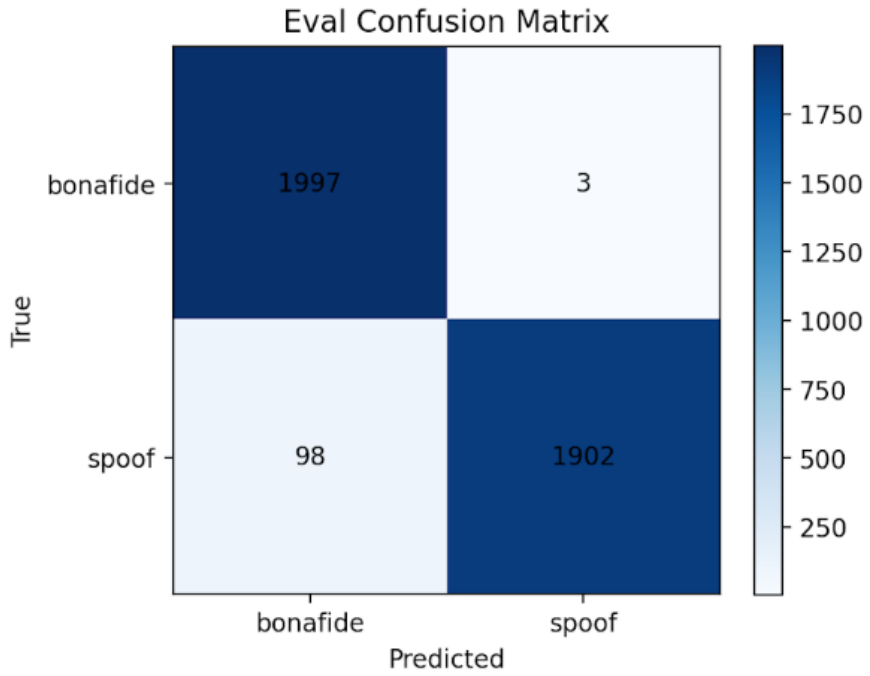


Figure 4.5: Confusion matrix of WavLm model.

The confusion matrix of the fine-tuned WavLM model shows a strong classification performance on the evaluation set. The model correctly identified 1997 out of 2000 bonafide utterances and 1902 out of 2000 spoofed utterances. Only 3 genuine (bonafide) samples were mistakenly classified as spoofed, while 98 spoofed utterances were incorrectly predicted as bonafide.

This behavior is reflected in the very high specificity of the system (99.85%) as well as a low false acceptance rate(4.90%).

These results indicate that the model is highly reliable in recognizing genuine speech, with an extremely small number of false rejections. However, the slightly higher number of spoofed samples classified as bonafide shows that some spoofing attacks are still challenging for the model. Overall, the confusion matrix confirms that the WavLM-based system achieves a strong and balanced performance in distinguishing between bonafide and spoofed speech, making it effective for audio deepfake detection tasks.

4.5.3.3 Generalization Analysis

To further evaluate the robustness and generalization ability of the proposed WavLM-based anti-spoofing system, its performance on the development and evaluation sets was analyzed, as presented in Table 4.14. Moreover, the learning curves depicted in Figure 4.6 offer a detailed view of the model's optimization process, showing its convergence behavior during fine-tuning. Collectively, these results highlight the effectiveness of the training strategy and confirm the model's capability to generalize well to unseen audio samples while maintaining strong spoofing detection performance.

Table 4.14: Generalization performance from the development set to the evaluation set of the WavLM model.

Metric	Dev	Eval	Diff.
Accuracy (%)	99.84	97.48	-2.36
F1-score (%)	99.84	97.41	-2.43
EER (%)	0.08	0.45	+0.37

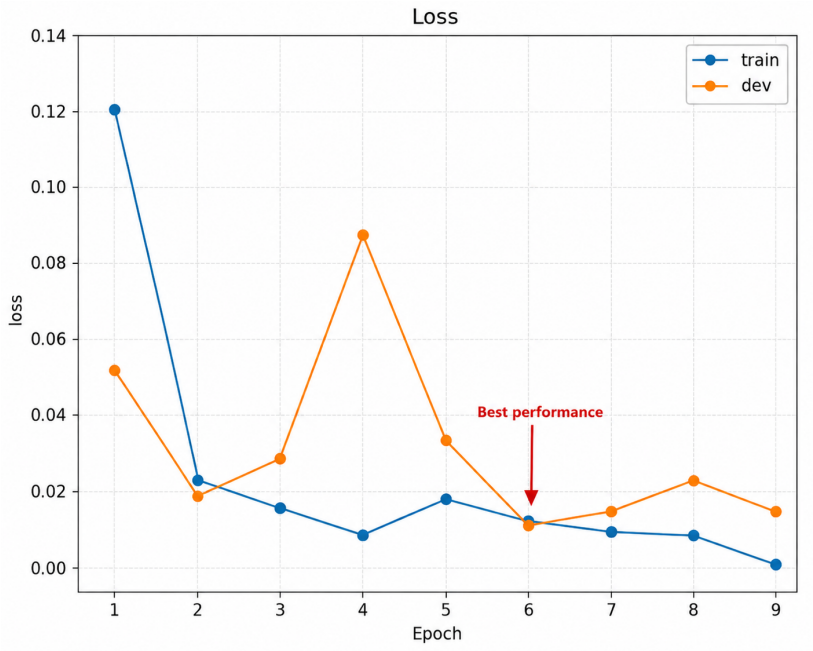


Figure 4.6: Training and development loss curves of the WavLM model.

A slight drop in performance can be noticed when comparing the development and evaluation results of the fine-tuned WavLM model. Specifically, the accuracy decreases from 99.84% on the development set to 97.48% on the evaluation set, while the F1-score shows a reduction of 2.43%. In the same direction, the EER rises from 0.08% to 0.45%, which suggests that the evaluation set introduces more variability and harder unseen conditions.

The training loss shows a consistent downward trend, indicating successful optimization of the model parameters. While the development loss fluctuates during training, it remains relatively

low and follows the overall convergence pattern. The modest separation between the training and development curves suggests limited overfitting and good generalization capability. The application of early stopping (patience = 3) further reduces the risk of overfitting by selecting the model checkpoint with the best development-set performance.

In summary, despite this moderate decline on the evaluation set, the WavLM model still achieves strong results, with accuracy close to 97.5% and a very low EER. This confirms its solid generalization ability and effectiveness in detecting spoofed speech under unseen scenarios.

4.5.3.4 Attack-wise Analysis

The attack-wise evaluation provides a detailed view of the model's robustness against different spoofing algorithms, highlighting clear variations in detection difficulty across attack types.

a) Best Detected Attacks

Table 4.15: Well-detected spoofing attacks.

Attack	Accuracy (%)
A09	100.00
A12	100.00
A13	100.00
A14	100.00
A16	100.00
A17	100.00

Several spoofing attacks such as A09, A12, A13, A14, A16, and A17 are perfectly detected by the model, each reaching an accuracy of 100%. This indicates that the artifacts introduced by these synthesis or conversion methods are highly distinguishable from natural speech, allowing the WavLM-based system to classify them without any errors.

b) Challenging Attacks

Table 4.16: Challenging spoofing attacks.

Attack	Accuracy (%)
A10	57.52
A08	94.23
A18	93.55
A19	99.35
A15	95.42
A07	98.04
A11	98.05

In contrast, attack A10 emerged as the most challenging spoofing condition, achieving a substantially lower detection accuracy of only 57.52%. This result indicates that a considerable

proportion of A10 samples were misclassified as bonafide, suggesting that this attack generates acoustic characteristics that closely resemble genuine speech and therefore reduces the discriminative cues available to the model. Other attacks such as A08 (94.23%) and A18 (93.55%) also show reduced detection performance compared to the remaining attack types, indicating that these spoofing methods produce acoustic patterns closer to real speech. Nevertheless, attacks A19, A07, A11, and A15 still maintain high detection rates above 95%, reflecting generally strong robustness across most conditions.

The average attack-wise accuracy across all spoof categories is approximately:95.36%.

This value confirms that the WavLM-based model maintains strong generalization ability while still showing difficulty with specific spoofing attacks, particularly A10.

4.5.3.5 Discussion

The experimental results demonstrate that the fine-tuned WavLM model provides a highly effective solution for audio deepfake detection on the ASVspoof 2019 LA evaluation set. The model achieved an accuracy of 97.48%, an F1-score of 97.41%, and a remarkably low EER of 0.45%, indicating a strong ability to distinguish between genuine and spoofed speech signals. The confusion matrix analysis further confirms this performance, as the system correctly classified 1997 out of 2000 bonafide utterances and 1902 out of 2000 spoofed utterances. The resulting specificity of 99.85% highlights the model's excellent capability to preserve genuine speech samples, while the FAR of 4.90% shows that only a limited number of spoofed samples were incorrectly accepted as authentic.

The attack-wise evaluation reveals that the model exhibits strong robustness across most spoofing conditions. Perfect detection rates were obtained for six attack types (A09, A12, A13, A14, A16, and A17), demonstrating that the acoustic artifacts generated by these attacks are effectively captured by the learned representations of WavLM. Nevertheless, performance was not uniform across all attack categories. In particular, attack A10 emerged as the most challenging case, achieving an accuracy of only 57.52%, while attacks A08 and A18 also showed comparatively lower detection rates. These findings suggest that certain spoofing techniques generate speech characteristics that are more similar to natural speech, making them harder to distinguish.

From a generalization perspective, the model maintains strong performance when moving from the development set to the evaluation set. Although a moderate decrease of 2.36% in accuracy and 2.43% in F1-score was observed, together with a slight increase in EER from 0.08% to 0.45%, the overall performance remains consistently high. The learning curves indicate stable convergence during fine-tuning, with only a minor gap between training and development losses, suggesting limited overfitting. Overall, these results confirm that WavLM is capable of learning discriminative and transferable speech representations, making it a reliable backbone for anti-spoofing systems operating under unseen and challenging conditions.

4.5.4 Evaluation of ensemble methods

We evaluate the performance of the proposed ensemble learning methods that combine the strengths of three state-of-the-art self-supervised speech models: Wav2Vec 2.0, Hubert, and WavLM. Several ensemble strategies are investigated, including Logit Average, Soft Vote Mean, Soft Vote

Weighted, Hard Voting Majority, Bagging Soft Vote, Max Confidence, and Stacking with Logistic Regression. By leveraging the complementary representations learned by these models, the ensemble approaches aim to improve the robustness and effectiveness of audio deepfake detection. The experimental results on the ASVspoof 2019 dataset provide a comprehensive comparison of the different fusion strategies and highlight their impact on the overall system performance.

Table 4.17: Ensemble learning results on ASVspoof 2019 evaluation set.

Ensemble Methods	Performance Metrics				
	Acc.	**Prec.**	**Rec.**	**F1**	**EER**
Stacking (Logistic Regression)	**99.15%**	**99.95%**	**98.35%**	**99.14%**	0.38%
Logit Average	99.10%	**99.95%**	98.25%	99.09%	**0.30%**
Mean Soft Voting	98.95%	**99.95%**	97.95%	98.94%	0.52%
Weighted Soft Voting	98.95%	**99.95%**	97.95%	98.94%	0.52%
Majority Hard Voting	98.95%	**99.95%**	97.95%	98.94%	0.52%
Bagging Soft Voting	98.95%	**99.95%**	97.95%	98.94%	0.52%
Max Confidence	98.75%	**99.95%**	97.55%	98.73%	0.45%

- **Stacking (Logistic Regression):** achieved the best overall performance, obtaining an accuracy of 99.15% and an F1-score of 99.14%, with a recall of 98.35%. The method integrates base learner outputs through a meta-learning strategy, resulting in strong and consistent performance across evaluation metrics. The EER of 0.38% indicates a low error trade-off.

- **Logit Average:** delivered highly competitive results with an accuracy of 99.10% and an F1-score of 99.09%, while maintaining a precision of 99.95%. It achieved the lowest EER of 0.30%, indicating the most favorable balance between false acceptance and false rejection rates. The recall of 98.25% is slightly lower than that of stacking.

- **Mean Soft Voting:** produced an accuracy of 98.95% and an F1-score of 98.94%. Precision remained at 99.95%, while recall decreased to 97.95%. The EER of 0.52% reflects a higher error rate compared to the best-performing methods.

- **Weighted Soft Voting:** yielded identical results to Mean Soft Voting, with an accuracy of 98.95%, recall of 97.95%, and an F1-score of 98.94%. The weighting strategy did not produce measurable improvement in this setting.

- **Majority Hard Voting:** obtained an accuracy of 98.95%, with an F1-score of 98.94% and a recall of 97.95%. The EER remained at 0.52%, indicating performance similar to other voting-based approaches.

- **Bagging Soft Voting:** achieved an accuracy of 98.95% and a recall of 97.95%, with an EER of 0.52%. The results show no significant improvement over standard voting ensembles.

- **Max Confidence:** recorded the lowest performance, with an accuracy of 98.75% and an F1-score of 98.73%, along with a recall of 97.55%. The EER of 0.45% indicates relatively weaker robustness compared to stacking and logit averaging.

Summary of the Results: The experimental results demonstrate that ensemble learning consistently improves audio deepfake detection performance by leveraging the complementary strengths of Wav2Vec2, HuBERT, and WavLM. Among the evaluated methods, Stacking with Logistic Regression achieved the highest overall classification performance, reaching an accuracy of 99.15%, a precision of 99.95%, a recall of 98.35%, and an F1-score of 99.14%. These results indicate that the meta-learning strategy effectively captures the relationships between the outputs of the base models and combines them in a more adaptive manner than conventional aggregation techniques.

Although Stacking achieved the best overall classification metrics, Logit Average obtained the lowest Equal Error Rate (EER) of 0.30%, compared with 0.38% for Stacking. This suggests that Logit Average provides a slightly better balance between false acceptance and false rejection rates, whereas Stacking offers superior overall classification performance in terms of accuracy, recall, and F1-score.

The voting-based approaches, including Mean Soft Voting, Weighted Soft Voting, Majority Hard Voting, and Bagging Soft Voting, produced nearly identical results. This observation indicates that simple aggregation strategies may not fully exploit the complementary information learned by the individual SSL models. Consequently, their performance remains slightly below that of the more sophisticated fusion methods.

Max Confidence achieved the lowest overall performance among the evaluated ensemble approaches. Since this method relies exclusively on the prediction of the most confident model for each sample, it ignores potentially useful information from the remaining models. As a result, it benefits less from model diversity and exhibits reduced robustness compared with the other ensemble strategies.

In general, the results confirm that learning-based fusion strategies are more effective than static aggregation methods for audio deepfake detection. In particular, Stacking provides the strongest overall classification capability, while Logit Average offers the most favorable error trade-off according to the EER metric.

Confusion Matrix Analysis

Figure 4.7 presents the confusion matrix obtained on the evaluation set.

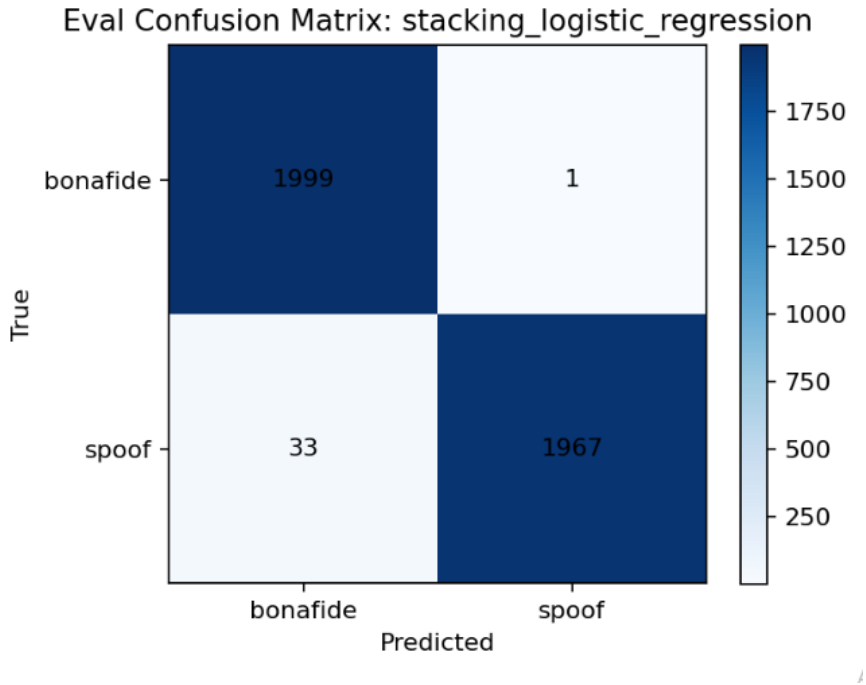


Figure 4.7: Confusion matrix for the stacking ensemble model using Logistic Regression as meta-learner.

The confusion matrix of the stacking ensemble model reveals a highly effective classification performance on the ASVspoof 2019 evaluation set. The model correctly classifies 1999 bonafide utterances and 1967 spoofed utterances, demonstrating a strong discrimination capability between genuine and fake speech samples.

Only a very small number of misclassifications are observed, including 1 bonafide sample incorrectly classified as spoofed speech and 33 spoofed samples incorrectly classified as bonafide. In the context of spoof detection, the latter error type (false acceptance of spoofed speech) is more critical, as it represents security vulnerabilities where fake speech is incorrectly accepted as genuine.

In general, the results correspond to a true negative rate of 1999 samples and a true positive rate of 1967 samples, yielding a total accuracy consistent with the reported value of 99.15%. The very low number of classification errors confirms the robustness of the stacking ensemble approach, and aligns with the low Equal Error Rate (EER) of 0.38% reported in the experimental results.

These findings indicate that the stacking model effectively minimizes both false acceptance and false rejection rates by leveraging complementary information from multiple self-supervised speech representations.

4.5.5 Ablation study

The ablation study was conducted to assess the contribution of each SSL model within the proposed stacking-based ensemble.

Table 4.18: Ablation study of the proposed ensemble using stacking logistic regression on the ASVspoof 2019 LA evaluation set.

Configuration	Balanced Accuracy (%)	F1-score (%)	EER (%)
Full Trio (Wav2Vec 2.0+HuBERT+WavLM)	**99.15**	**99.14**	**0.38**
Without Wav2Vec 2.0	99.05	99.04	0.38
Without HuBERT	98.70	98.68	0.33
Without WavLM	98.70	98.68	0.80

Table 4.19: Performance degradation relative to the full ensemble.

Removed Model	Δ BA (%)	Δ F1 (%)	Δ EER (%)
Wav2Vec 2.0	-0.10	-0.10	0.00
HuBERT	-0.45	-0.46	-0.05
WavLM	-0.45	-0.46	+0.42

As shown in Table 4.18, the complete ensemble composed of Wav2Vec 2.0, HuBERT, and WavLM achieved the best overall classification performance, reaching a balanced accuracy of 99.15%, an F1-score of 99.14%, and an EER of 0.38%. These results demonstrate the effectiveness of combining complementary self-supervised speech representations through the stacking framework.

Removing Wav2Vec2 resulted in only a marginal decrease of 0.10 percentage points in both balanced accuracy and F1-score, while the EER remained unchanged. This suggests that although Wav2Vec 2.0 contributes useful complementary information, its impact on the final decision process is relatively limited compared with the other SSL models.

In contrast, removing either HuBERT or WavLM produced a larger degradation of approximately 0.45 percentage points in balanced accuracy and F1-score, demonstrating that both models play a substantial role in the ensemble. Interestingly, removing HuBERT slightly reduced the EER from 0.38% to 0.33%, suggesting that although HuBERT contributes positively to overall classification performance, its influence on the balance between false acceptance and false rejection rates differs from that of the other models. This observation highlights the complex interactions among the base learners within the stacking framework.

The most pronounced effect was observed when WavLM was excluded, causing the EER to increase from 0.38% to 0.80%, more than doubling the error rate. This finding indicates that WavLM provides highly discriminative spoofing-related representations that are particularly beneficial for reducing false acceptance and false rejection errors. The substantial increase in EER confirms its critical role in maintaining the robustness and reliability of the ensemble.

In general, the results confirm that the three SSL models provide complementary information and that combining them through stacking yields better performance than any individual model. Furthermore, the ablation study highlights that WavLM is the most influential contributor to the ensemble, particularly in terms of error-rate reduction, while HuBERT contributes significantly to overall classification performance. Wav2Vec 2.0 provides additional diversity that further strengthens the ensemble and leads to a modest but measurable improvement in performance.

4.5.6 Ensemble Methods versus Base Models

In the following section, a detailed comparison is presented between the results of the individual fine-tuned base models and the best ensemble methods.

Table 4.20: Performance comparison of base models and ensemble methods on the ASVspoof 2019 LA evaluation set.

Base Models/Ensemble Methods	Performance Metrics				
	Acc.	**Prec.**	**Rec.**	**F1**	**EER**
HuBERT	98.68%	99.54%	97.80%	98.66%	1.00%
WavLM	97.48%	99.84%	95.10%	97.41%	0.45%
Wav2Vec 2.0	96.10%	99.89%	92.30%	95.95%	1.68%
Stacking (Logistic Regression)	**99.15%**	**99.95%**	**98.35%**	**99.14%**	0.38%
Logit Averaging	99.10%	99.95%	98.25%	99.09%	**0.30%**
Soft Voting	98.95%	99.95%	97.95%	98.94%	0.52%
Weighted Soft Voting	98.95%	99.95%	97.95%	98.94%	0.52%
Hard Voting	98.95%	99.95%	97.95%	98.94%	0.52%
Bagging Soft Vote	98.95%	99.95%	97.95%	98.94%	0.52%
Max Confidence	98.75%	99.95%	97.55%	98.73%	0.45%

The ensemble methods generally outperform the individual base models on the ASVspoof 2019 LA evaluation set, as shown in Table 4.20. This demonstrates the benefit of combining complementary representations learned by different self-supervised speech models. By aggregating information from multiple models, the ensemble approaches achieve higher robustness and improved classification performance compared with any single base model.

Among all evaluated methods, Stacking (Logistic Regression) achieved the best overall classification performance, obtaining an accuracy of 99.15%, a precision of 99.95%, a recall of 98.35%, and an F1-score of 99.14%. These results surpass those of all individual base models as well as the other ensemble strategies. The superior performance of stacking can be attributed to its meta-learning mechanism, which learns how to optimally combine the predictions of HuBERT, WavLM, and Wav2Vec 2.0 rather than relying on predefined aggregation rules.

Logit Averaging also achieved highly competitive results, with an accuracy of 99.10% and an F1-score of 99.09%. Notably, it obtained the lowest Equal Error Rat (EER) of 0.30%, indicating the most favorable balance between false acceptance and false rejection rates among all evaluated methods. This suggests that simple probability-level fusion can be highly effective when the individual models produce well-calibrated outputs.

Among the individual models, HuBERT achieved the highest overall performance with an accuracy of 98.68% and an F1-score of 98.66%, while WavLM obtained the lowest EER of 0.45%. In contrast, Wav2Vec 2.0 exhibited the weakest standalone performance across most metrics. Nevertheless, its contribution remains valuable within the ensemble framework by providing complementary information that increases model diversity.

In general, the results confirm that ensemble learning significantly enhances audio deepfake detection performance compared with individual SSL models. While Stacking provides the strongest overall classification performance, Logit Averaging achieves the best threshold-based discrimination according to the EER metric, making both approaches particularly effective for the ASVspoof

2019 LA task.

4.5.7 Cross-Dataset Generalization

To assess the generalization capability of the proposed anti-spoofing framework, a cross-dataset evaluation was conducted using the DeepVoice dataset, as discussed in Section 2.4.1.2, this dataset contains synthetic speech samples generated using multiple voice cloning techniques which was not seen during training or validation. This experiment aims to determine whether the model can effectively detect spoofed speech generated under different conditions and synthesis techniques.

Table 4.21: Cross-dataset evaluation setup.

Parameter	Value
Dataset	DeepVoice
Ensemble Method	Stacking Logistic Regression
Audio Aggregation	Mean
Segment Duration	3 s
Segment Overlap	0 s
Audio Files Evaluated	64
Successful Evaluations	64
Segments Evaluated	10,004

Table 4.22: Cross-Dataset Performance Evaluation of the Proposed Stacking Ensemble (Logistic Regression) Across ASVspoof and DeepVoice Datasets.

Metric	**DeepVoice (%)**	**ASVspoof (%)**
Accuracy	81.25	99.15
Precision	92.31	99.95
Recall	85.71	98.35
F1-score	88.89	99.14
EER	37.50	0.38

As shown in Table 4.22, the proposed stacking ensemble achieved an accuracy of 81.25%, a precision of 92.31%, a recall of 85.71%, and an F1-score of 88.89% on the DeepVoice dataset. In comparison, the model obtained 99.15% accuracy, 99.95% precision, 98.35% recall, and 99.14% F1-score on the ASVspoof 2019 LA evaluation set. Although a noticeable decrease in performance is observed when the model is evaluated on DeepVoice, the results indicate that the ensemble retains a substantial ability to distinguish between genuine and spoofed speech in a cross-dataset setting. This degradation is expected due to differences in speech synthesis techniques, recording conditions, and data distributions between the two datasets.

The relatively high precision of 92.31% on DeepVoice suggests that most samples classified as spoofed were indeed spoofed, indicating a low false-positive rate. However, the recall decreased to 85.71%, implying that some spoofed samples were not successfully detected when the model was exposed to previously unseen speech generation methods. Despite this reduction, the F1-score

remained relatively high at 88.89%, demonstrating a balanced trade-off between precision and recall and confirming the model's ability to capture discriminative spoofing-related characteristics that partially generalize beyond the training domain.

Furthermore, the Equal Error Rate (EER) increased substantially from 0.38% on ASVspoof to 37.50% on DeepVoice. This significant increase highlights the challenges introduced by domain mismatch and indicates that the decision boundary learned during training becomes less effective when applied to data originating from a different distribution. Nevertheless, the model maintains acceptable detection performance despite the considerable shift between datasets.

Overall, these findings demonstrate both the strengths and limitations of the proposed ensemble approach. While the model achieves near-perfect performance on the ASVspoof benchmark, its effectiveness decreases when evaluated on an unseen dataset. Nevertheless, the achieved cross-dataset results confirm that the ensemble learns meaningful and transferable spoofing-related features. Further improvements in generalization could be achieved through domain adaptation strategies or by training on more diverse datasets encompassing a broader range of speech synthesis techniques.

To provide a clearer understanding of the prediction mechanism, Figure 4.8 presents a representative example of the complete inference process, from audio segmentation to the final aggregated decision:

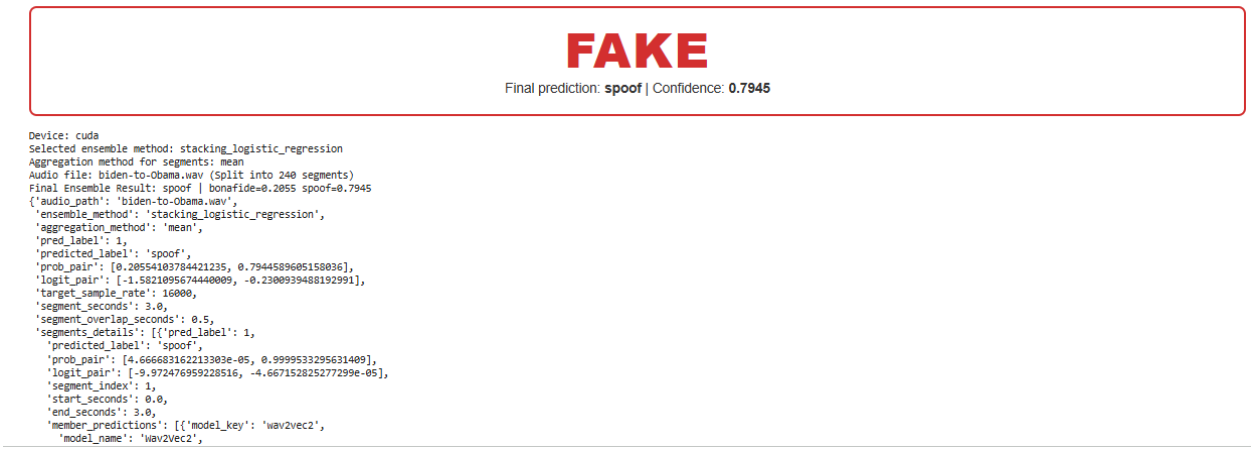


Figure 4.8: Example of the Inference Process and Final Prediction for an Input Audio Sample.

We present the example in Figure 4.8, which illustrates the inference procedure for a single audio recording. Since input recordings may have variable lengths, long audio signals are divided into overlapping 3-second segments with a 0.5-second overlap. Each segment is processed independently by our trained transformer-based anti-spoofing model, producing a prediction along with an associated confidence score. The segment-level outputs are then aggregated to obtain a recording-level decision. This aggregation combines information extracted from all segments, leading to a more reliable and robust final prediction for the entire audio sample. In the illustrated example, the aggregated decision classifies the input recording as spoof with a confidence score of 0.7946.

4.5.8 Explainable AI Analysis

To gain insight into the decision-making process of the proposed ensemble model, Gradient Saliency was applied to a set of correctly classified real utterances from the ASVspoof 2019 LA evaluation set. The generated saliency maps highlight the temporal regions that contributed most strongly to the ensemble predictions. The objective of this analysis is to determine whether the model relies on meaningful speech characteristics rather than irrelevant signal artifacts.

4.5.8.1 Global Analysis of Saliency Maps

A visual inspection of the saliency maps reveals several consistent patterns across the analyzed samples.

First, the highest attribution values are generally concentrated around speech transitions and regions exhibiting rapid temporal variations. The most salient segments consistently occur during intervals characterized by abrupt waveform changes and highly dynamic speech events. This observation suggests that the ensemble model relies primarily on informative acoustic structures rather than stationary portions of the signal.

Second, silent and low-energy regions consistently receive negligible attribution scores. In samples containing extended silence intervals, the saliency values remain close to zero throughout the non-speech segments. This behavior indicates that the classifier does not base its decisions on recording duration or silence-related artifacts, but instead focuses on acoustically relevant speech information.

Another recurrent observation is the presence of salient regions near speech onset and offset boundaries. Several samples exhibit elevated attribution scores at the beginning or end of speech activity. Such regions may contain transient acoustic cues, articulation dynamics, or subtle synthesis artifacts that contribute to the discrimination between bona fide and spoofed speech.

Finally, despite the extremely high confidence of the ensemble predictions, the attribution maps remain highly localized rather than uniformly distributed over the entire waveform. For all analyzed samples, the ensemble assigned a bonafide probability exceeding 99.99%, while relying on a limited number of highly discriminative temporal regions. This behavior suggests that the ensemble has learned meaningful acoustic representations that support its predictions through specific evidence rather than diffuse signal characteristics.

4.5.8.2 Representative Example

Figure 4.9 illustrates the Gradient Saliency explanation obtained for utterance `LA_E_1005795`.

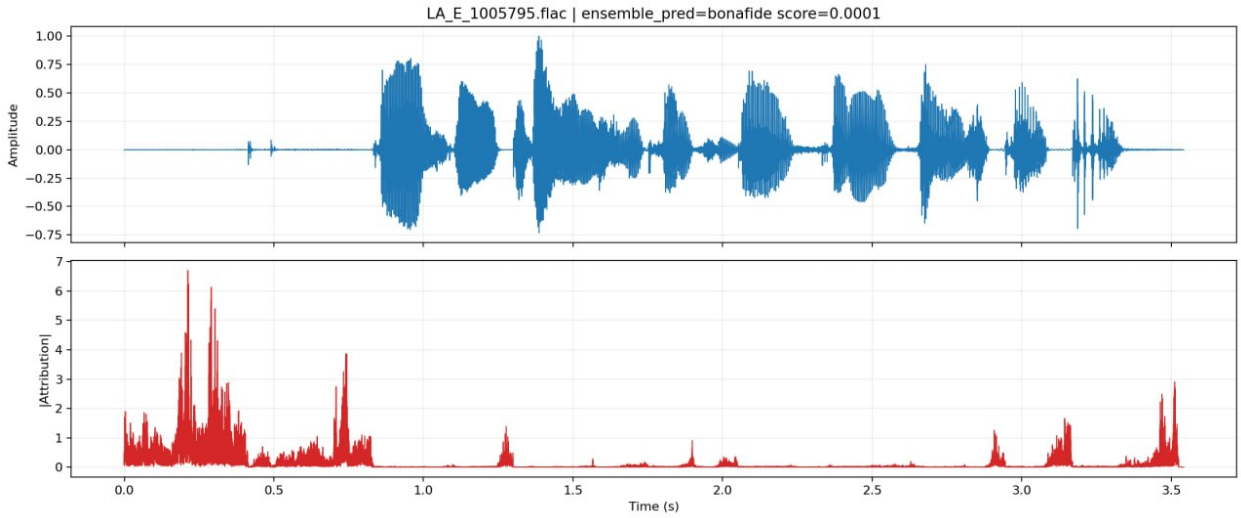


Figure 4.9: Gradient Saliency visualization for the bonafide(real) utterance LA_E_1005795. The red curve represents the attribution score, highlighting the temporal regions that contributed most strongly to the ensemble prediction.

The ensemble classified the sample as bona fide with a confidence of approximately 99.995%. The most influential regions were identified between 0.18 s and 0.31 s, corresponding to prominent speech transitions and rapid amplitude variations. Additional salient regions were observed around 0.74 s. These findings indicate that the ensemble focuses on informative phonetic events and temporal speech dynamics rather than background or silent portions of the recording.

Figure 4.10 illustrates the Gradient Saliency explanation obtained for utterance `LA_E_1000147`.

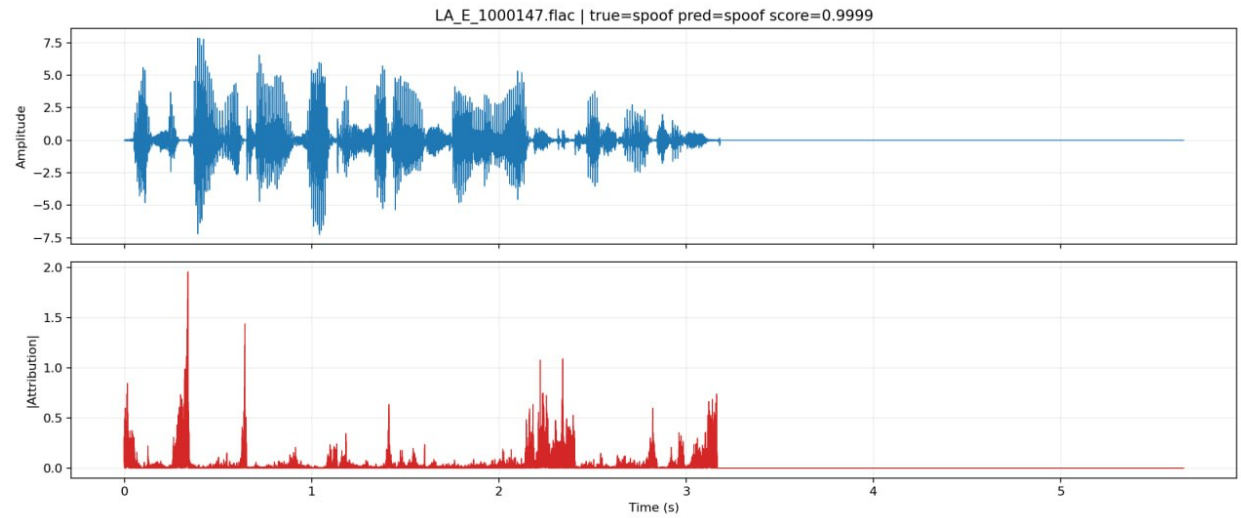


Figure 4.10: Gradient Saliency visualization for the spoof utterance LA_E_1000147. The red curve represents the attribution score, highlighting the temporal regions that contributed most strongly to the ensemble prediction.

The ensemble classified the sample as spoof with a confidence of approximately 99.99%. The

saliency map reveals several highly influential temporal regions distributed throughout the active speech segments, with particularly strong attributions during the early part of the utterance and additional prominent peaks between approximately 2.1 s and 3.2 s. In contrast, the silent trailing portion of the recording receives almost no attribution. These observations suggest that the ensemble primarily relies on discriminative speech characteristics and temporal artifacts associated with spoofed audio rather than on silence or background regions.

Figure 4.11 presents the Gradient Saliency explanation for utterance `LA_E_1026061`, which is a spoofed(fake) sample that was incorrectly classified as bonafide(real) by the proposed stacking ensemble.

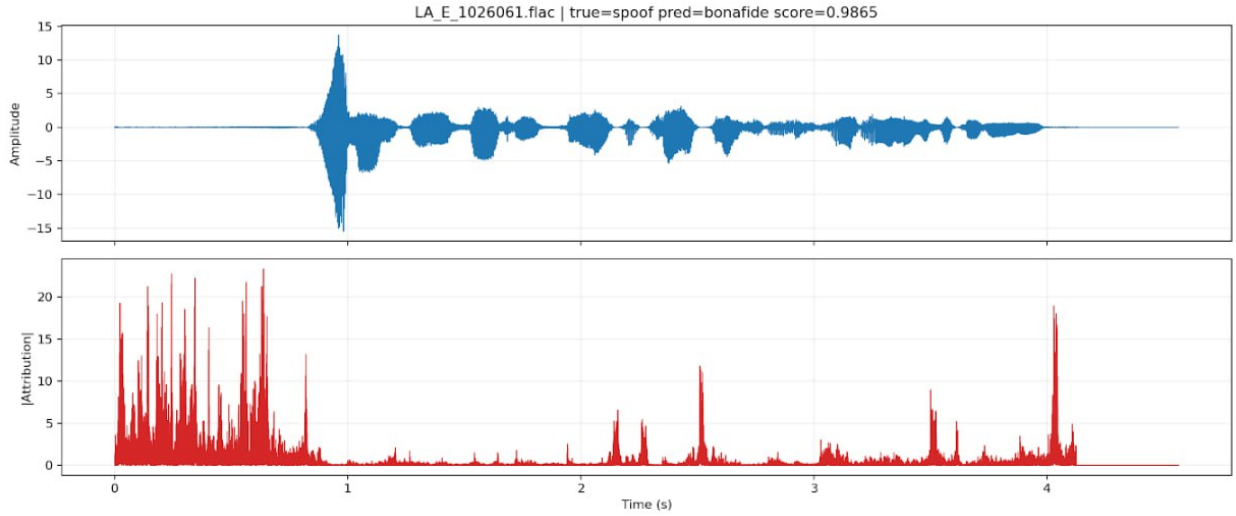


Figure 4.11: Gradient Saliency visualization for the misclassified spoof utterance LA_E_1026061. The true label is spoof(fake), while the ensemble prediction is bonafide(real) with a confidence score of 98.65%. The red curve represents the attribution score highlighting the temporal regions that contributed most strongly to the prediction.

Although the utterance is a spoofed recording, the ensemble assigned a high bonafide confidence of approximately 98.65%, indicating that the spoofing artifacts were not sufficiently captured by the model. The saliency map reveals that the prediction was primarily influenced by several highly attributed regions located in the early part of the signal (approximately 0.0–0.9 s), where strong speech onsets and rapid amplitude variations occur. Additional salient peaks are observed around 2.5 s and 4.0 s.

Interestingly, the model focused on speech segments that appear acoustically natural, while assigning little importance to other regions that may contain subtle spoofing artifacts. This suggests that the synthetic speech characteristics present in this utterance closely resemble genuine speech patterns, making discrimination difficult even for the ensemble model. Such failure cases highlight the challenges of audio deepfake detection and emphasize the importance of improving model robustness and generalization against highly realistic spoofing attacks.

4.5.8.3 Discussion

Generally, the Gradient Saliency analysis demonstrates that the proposed stacking ensemble primarily attends to informative speech regions characterized by strong temporal dynamics, phonetic transitions, and speech-active segments. Silent and low-energy regions consistently receive minimal attribution, suggesting that the model does not rely on trivial dataset-specific cues such as silence duration or recording artifacts.

The concentration of saliency scores around acoustically meaningful events indicates that the ensemble bases its predictions on speech-related characteristics rather than irrelevant portions of

the waveform. Furthermore, the localized nature of the attribution maps, even for highly confident predictions, suggests that the model relies on specific discriminative evidence rather than uniformly distributed signal information.

The analysis of both correctly classified and misclassified samples provides additional insight into the model's behavior. While successful predictions are generally associated with salient speech regions containing informative acoustic cues, misclassified examples reveal situations in which spoofed speech exhibits characteristics that closely resemble natural speech patterns. Such cases highlight the remaining challenges of audio deepfake detection and emphasize the need for further improvements in robustness against highly realistic spoofing attacks.

In general, the saliency analysis increases the transparency and interpretability of the proposed framework by providing evidence that the ensemble focuses on acoustically relevant speech characteristics when making its decisions.

4.6 Comparison with State-of-the-Art Methods

To assess the effectiveness of the proposed ensemble model, its performance was compared with several representative spoofing detection systems evaluated on the ASVspoof 2019 Logical Access (LA) dataset. The comparison is based on the Equal Error Rate (EER), which is one of the most widely adopted evaluation metrics in speaker verification and anti-spoofing research.

EER corresponds to the operating point at which the False Acceptance Rate (FAR) equals the False Rejection Rate (FRR). A lower EER indicates better discrimination between bonafide and spoofed speech samples. Since EER is threshold-independent and commonly reported in the ASVspoof literature, it enables a fair comparison with previously published methods.

It should be noted that, due to computational resource limitations, the experiments were conducted on a subset of the ASVspoof 2019 LA evaluation set rather than the complete evaluation partition. The models were trained and evaluated using the free version of Google Colab, which provides limited GPU resources and execution time. Consequently, processing the entire evaluation set was not feasible within the available computational budget. Although the reported results were obtained on a reduced evaluation subset, the same evaluation protocol and EER metric were employed, allowing a preliminary comparison with existing approaches.

Future work will include evaluating the proposed framework on the complete ASVspoof 2019 LA evaluation set once adequate computational resources become available. Performing experiments on the full dataset would provide a more comprehensive assessment of the model's robustness and improve the fairness of comparisons with previously published studies, most of which report results on the complete benchmark. While the exact performance variation cannot be determined without conducting the experiments, the larger evaluation set would likely yield a more reliable estimate of the true generalization capability of the proposed model by incorporating a broader range of spoofing conditions and speech variability. Therefore, the results reported in this study should be considered indicative rather than definitive with respect to direct state-of-the-art comparisons.

Table 4.23 presents the performance of the proposed ensemble model alongside several representative methods from the literature.

Table 4.23: Comparison of the proposed ensemble model with selected methods on the ASVspoof 2019 LA evaluation set.

Method	EER (%)	Ref.
CQCC-GMM Baseline	9.57	[72]
LFCC-GMM Baseline	8.09	[73]
DKU Fusion System	0.66	[73]
Wav2Vec 2.0 Representation Selective Self-Distillation	0.31	[74]
Deepfake Audio Detection Using Spectrogram-based Features and Ensemble of Deep Learning Models	0.03	[76]
Proposed Ensemble Model	**0.38**	This work

The proposed SSL-based stacking ensemble achieved an EER of 0.38%, outperforming the classical CQCC-GMM and LFCC-GMM baselines as well as the DKU Fusion System. These results demonstrate the strong discriminative capability of the proposed ensemble and highlight the effectiveness of combining complementary self-supervised speech representations extracted from Wav2Vec 2.0, HuBERT, and WavLM.

Compared with recent deep learning approaches, the proposed model achieved competitive performance. Although the method proposed by Pham et al. [76] reported a lower EER of 0.03%, direct comparisons should be interpreted with caution because the reported results were obtained under different experimental settings and evaluation conditions. In addition to its strong detection performance, the proposed framework provides interpretability through Gradient Saliency analysis, offering valuable insight into the decision-making process of the ensemble.

Generally, the obtained results indicate that SSL-based ensemble learning constitutes an effective approach for audio deepfake detection and achieves performance that is competitive with several state-of-the-art systems reported in the literature. However, a complete evaluation on the full ASVspoof 2019 LA benchmark remains an important direction for future work to further validate the generalization and robustness of the proposed framework.

4.7 Conclusion

In this chapter, we presented the experimental evaluation and discussion of the proposed audio deepfake detection framework. We first described the hardware and software environment used for model development, training, and testing, as well as the experimental settings and evaluation metrics adopted throughout the study.

We then analyzed the performance of the three fine-tuned self-supervised learning models, namely Wav2Vec 2.0, HuBERT, and WavLM. For each model, we examined the classification results using accuracy, precision, recall, F1-score, and Equal Error Rate (EER), together with confusion matrix and generalization analyses. The results demonstrated that all three transformer-based models achieved excellent performance in distinguishing bonafide and spoofed speech, while exhibiting complementary strengths that can be exploited through ensemble learning.

Furthermore, we evaluated several ensemble strategies, including Soft Voting, Weighted Soft

Voting, Logit Averaging, Hard Voting, Max Confidence, Bagging Soft Voting, and Stacking with Logistic Regression. The experimental results showed that ensemble learning consistently improved robustness and detection performance by combining the strengths of the individual models. In particular, the Stacking Logistic Regression ensemble achieved the best performance, providing the most effective balance between classification accuracy and security-oriented metric Equal Error Rate (EER).

In addition, ablation studies confirmed the complementary contribution of Wav2Vec 2.0, HuBERT, and WavLM within the proposed framework. Cross-dataset evaluation on the DeepVoice dataset further demonstrated the generalization capability of the ensemble beyond the ASVspoof 2019 LA dataset. Finally, Gradient Saliency analysis provided valuable insights into the model's decision-making process, showing that the ensemble focuses on acoustically meaningful speech regions rather than trivial artifacts or silent segments.

Overall, the experimental findings confirm that combining multiple self-supervised transformer models through ensemble learning constitutes an effective and reliable approach for audio deepfake detection. The proposed framework achieves high accuracy, strong robustness against diverse spoofing attacks, and improved generalization capability, making it a promising solution for real-world audio anti-spoofing applications.

Furthermore, to provide a comprehensive evaluation of the proposed approach, a comparison with several state-of-the-art spoofing detection methods on the ASVspoof 2019 LA dataset was conducted using the Equal Error Rate (EER) metric. The results show that the proposed ensemble model achieves competitive performance compared to recent advanced systems reported in the literature, including both classical baseline approaches and modern self-supervised learning-based methods. Although the evaluation was performed on a subset of the test set due to computational constraints, the proposed framework still demonstrates strong discriminative capability, highlighting its robustness and effectiveness. Overall, this comparison further confirms the relevance and efficiency of ensemble-based self-supervised representations for improving audio deepfake detection performance.

General Conclusion

The rapid advancement of artificial intelligence has significantly improved speech synthesis and voice conversion technologies, enabling the creation of highly realistic audio deepfakes. While these technologies offer numerous beneficial applications, they also pose serious security, privacy, and societal challenges by facilitating voice impersonation, misinformation, and fraud. Consequently, the development of robust and reliable audio deepfake detection systems has become an essential research area.

This thesis addressed the problem of audio deepfake detection through the design and evaluation of an ensemble learning framework based on self-supervised learning (SSL) speech representations. First, the fundamental concepts of speech processing were presented, followed by a comprehensive review of audio deepfake generation techniques, benchmark datasets, detection approaches, and explainable artificial intelligence methods. Particular attention was given to modern SSL models, namely Wav2Vec 2.0, HuBERT, and WavLM, which have demonstrated remarkable capabilities in learning rich speech representations from large-scale unlabeled audio data.

Building upon this theoretical foundation, a complete detection framework was proposed. The framework employed three fine-tuned SSL-based models as base learners and investigated several ensemble learning strategies, including soft voting, weighted soft voting, logit averaging, hard voting, max-confidence fusion, bagging soft voting, and stacking with logistic regression. The objective was to exploit the complementary strengths of the individual models and improve detection robustness and generalization capabilities.

Experimental evaluation conducted on the ASVspoof 2019 Logical Access dataset demonstrated that SSL representations are highly effective for distinguishing bonafide(real) speech from spoofed speech. Among the evaluated approaches, the stacking ensemble using logistic regression achieved the best overall performance, outperforming individual base models and other ensemble methods. The ablation study further confirmed that each SSL model contributed valuable complementary information to the final prediction, validating the effectiveness of the ensemble design. Additionally, cross-dataset experiments on the DeepVoice dataset showed that the proposed framework maintained competitive performance when evaluated on previously unseen data, highlighting its ability to generalize beyond the training distribution.

To enhance transparency and trustworthiness, Explainable Artificial Intelligence (XAI) techniques based on Gradient Saliency were incorporated into the analysis. The generated saliency maps provided insight into the temporal regions that most influenced model decisions, offering a better understanding of the detection process and supporting the reliability of the proposed system.

Despite these promising results, several limitations remain. The experiments were conducted using a limited number of benchmark datasets and focused primarily on logical access attacks. Furthermore, the rapid evolution of speech synthesis technologies continuously introduces new

spoofing techniques that may challenge current detection systems. Therefore, maintaining robust performance against emerging attacks remains an open research problem.

Future work may focus on evaluating the framework on more diverse and recent datasets, investigating domain adaptation techniques to further improve cross-dataset generalization, exploring multimodal deepfake detection approaches, and integrating more advanced explainability methods. Additional research could also examine lightweight architectures suitable for real-time deployment in practical applications such as speaker verification systems, voice assistants, and cybersecurity platforms.

In conclusion, this work demonstrates that combining self-supervised speech representations with ensemble learning constitutes an effective and robust strategy for audio deepfake detection. The proposed framework achieves strong detection performance, improved generalization, and enhanced interpretability, thereby contributing to ongoing efforts to secure voice-based technologies against the growing threat of synthetic speech attacks.

Bibliography

[1] M. A. Redford, The Handbook of Speech Production. Hoboken, NJ, USA: Wiley-Blackwell, 2015.

[2] L. R. Rabiner and B.-H. Juang, Fundamentals of Speech Recognition. Englewood Cliffs, NJ, USA: Prentice Hall, 1993.

[3] S. Hadiby, "Speech recognition: Chapter 1 – Introduction and definitions," M1 Artificial Intelligence course material, Université 20 Août 1955 Skikda, Algeria, 2025.

[4] T. Bäckström, O. Räsänen, A. Zewoudie, P. Pérez Zarazaga, L. Koivusalo, S. Das, E. Gómez Mellado, M. Bouafif Mansali, D. Ramos, S. Kadiri, P. Alku, and M. H. Vali, Introduction to Speech Processing, 2nd ed. Espoo, Finland: Aalto University, 2022. [Online]. Available: https://speechprocessingbook.aalto.fi/

[5] L. R. Rabiner and R. W. Schafer, "Introduction to Digital Speech Processing," Foundations and Trends in Signal Processing, vol. 1, no. 1–2, pp. 1–194, 2007.

[6] RF Wireless World, "Time Domain vs. Frequency Domain: Key Differences Explained." [Online]. Available: https://www.rfwireless-world.com/terminology/time-domain-vs-frequency-domain. [Accessed: 3-Mar-2026].

[7] M. Labied, A. Belangour, M. Banane and A. Erraissi, "An overview of Automatic Speech Recognition Preprocessing Techniques," Proceedings of the 2022 International Conference on Decision Aid Sciences and Applications (DASA), 2022.

[8] M. A. Mazumder and R. A. Salam, "Feature Extraction Technique for Speech Processing: A Review," International Journal of Advanced Trends in Computer Science and Engineering, vol. 8, no. 1.3, pp. 285–292, 2019. doi: 10.30534/ijatcse/2019/5481.32019.

[9] S. Stanković, "Time-Frequency Analysis and Its Application in Digital Watermarking," EURASIP Journal on Advances in Signal Processing, vol. 2010, article 579295, 2010, doi:10.1155/2010/579295

[10] G. Sharma, K. Umapathy, and S. Krishnan, "Trends in audio signal feature extraction methods," Applied Acoustics, vol. 158, p. 107020, 2020, doi: 10.1016/j.apacoust.2019.107020.

[11] K. Jilenkumari Devi and A. Alice Devi, "Automatic Speaker Recognition using MFCC and Artificial Neural Network," 2019. [Online]. Available:https://www.researchgate.

net/figure/MFCC-LPC-and-PLP-feature-extraction-steps_fig1_372444727. [Accessed: 4-Mar-2026].

[12] B. Zhang, H. Cui, V. Nguyen, and M. Whitty, "Audio deepfake detection: What has been achieved and what lies ahead," Sensors, vol. 25, no. 7, p. 1989, 2025, doi: 10.3390/s25071989.

[13] M. Matza, "Fake Biden robocall tells voters to skip New Hampshire primary election," BBC News, 2024. [Online]. Available: https://www.bbc.com/news/world-us-canada-68064247 . [Accessed: 19-Dec-2024].

[14] C. Stupp, "Fraudsters used AI to mimic CEO's voice in unusual cybercrime case," The Wall Street Journal, 2019. [Online]. Available: https://www.wsj.com/articles/fraudsters-use-ai-to-mimic-ceos-voice-in-unusual-cybercrime-case-11567157402 . [Accessed: 19-Dec-2024].

[15] Z. Khanjani, G. Watson, and V. P. Janeja, "Audio deepfakes: A survey," Front. Big Data, vol. 5, Art. no. 1001063, Jan. 2023, doi: 10.3389/fdata.2022.1001063.

[16] Author's own work.

[17] J. J. Bird, "DEEP-VOICE: DeepFake Voice Recognition," Kaggle, 2023. [Online]. Available: https://www.kaggle.com/datasets/birdy654/deep-voice-deepfake-voice-recognition

[18] A. Pascu et al.,"Towards generalisable and calibrated audio deepfake detection with large pretrained models," Proc .Interspeech, 2024, pp. 1234–1238.

[19] Y. Zhang, W. Wang, and P. Zhang, "The effect of silence and dual-band fusion in anti-spoofing system," in Proceedings of the Interspeech 2021, Brno, Czech Republic, Aug. 30–Sep. 3, 2021, pp. 4279–4283.

[20] T. Chen, A. Kumar, P. Nagarsheth, G. Sivaraman, and E. Khoury, "Generalization of audio deepfake detection," in Proceedings of the Speaker and Language Recognition Workshop (Odyssey 2020), Tokyo, Japan, Nov. 1–5, 2020, pp. 132–137.

[21] H. Tak, J. W. Jung, J. Patino, M. Todisco, and N. Evans, "Graph attention networks for anti-spoofing," arXiv preprint arXiv:2104.03654, 2021.

[22] X. Wang and J. Yamagishi, "Investigating self-supervised front ends for speech spoofing countermeasures," arXiv preprint arXiv:2111.07725, 2021.

[23] A. Fathan, J. Alam, and W. H. Kang, "Mel-spectrogram image-based end-to-end audio deepfake detection under channel-mismatched conditions," in Proc. 2022 IEEE Int. Conf. Multimedia and Expo (ICME), Taipei, Taiwan, Jul. 18–22, 2022, pp. 1–6.

[24] T. M. Wani, R. Gulzar, and I. Amerini, "ABC-CapsNet: Attention based cascaded capsule network for audio deepfake detection," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Seattle, WA, USA, Jun. 17–21, 2024, pp. 2464–2472.

[25] X. Li, N. Li, C. Weng, X. Liu, D. Su, D. Yu, and H. Meng, "Replay and synthetic speech detection with Res2Net architecture," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Toronto, ON, Canada, Jun. 6–11, 2021, pp. 6354–6358.

[26] H. Ling, L. Huang, J. Huang, B. Zhang, and P. Li, "Attention-based convolutional neural network for ASV spoofing detection," in Proc. Interspeech 2021, Brno, Czechia, Aug. 30–Sep. 3, 2021, pp. 4289–4293.

[27] R. B. Randall, "A history of cepstrum analysis and its application to mechanical problems," Mech. Syst. Signal Process., vol. 97, pp. 3–19, 2017.

[28] Z. Zhang, X. Yi, and X. Zhao, "Fake speech detection using residual network with transformer encoder," in Proc. ACM Workshop Inf. Hiding Multimedia Secur. (IH&MMSec), Online, Jun. 22–25, 2021, pp. 13–22.

[29] A. Hamza, A. R. R. Javed, F. Iqbal, N. Kryvinska, A. S. Almadhor, Z. Jalil, and R. Borghol, "Deepfake audio detection via MFCC features using machine learning," IEEE Access, vol. 10, pp. 134018–134028, 2022.

[30] A. Firc, K. Malinka, and P. Hanáček, "Deepfake speech detection: A spectrogram analysis," in Proc. 39th ACM/SIGAPP Symp. Appl. Comput. (SAC), Ávila, Spain, Apr. 8–12, 2024, pp. 1312–1320.

[31] Y. Zhang, F. Jiang, and Z. Duan, "One-class learning towards synthetic voice spoofing detection," IEEE Signal Process. Lett., vol. 28, pp. 937–941, 2021.

[32] A. Luo, E. Li, Y. Liu, X. Kang, and Z. J. Wang, "A capsule network based approach for detection of audio spoofing attacks," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Toronto, ON, Canada, Jun. 6–11, 2021, pp. 6359–6363.

[33] C. Wang, J. He, J. Yi, J. Tao, C. Y. Zhang, and X. Zhang, "Multi-scale permutation entropy for audio deepfake detection," in Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP), Seoul, Republic of Korea, Apr. 14–19, 2024, pp. 1406–1410.

[34] D. Kurniawan,"Popular machine learning algorithms (supervised and unsupervised learning)," Medium,2022.Available: https://dedekurniawann.medium.com/popular-machine-learning-algorithms-supervised-and-unsupervised-learning-766120e96d Accessed: Apr. 16, 2026.

[35] S. E. Krüger, M. Schafföner, M. Katz, E. Andelic, and A. Wendemuth, "Mixture of Support Vector Machines for HMM Based Speech Recognition," Otto-von-Guericke University of Magdeburg, IESK, Cognitive Systems, Magdeburg, Germany.

[36] C. Bento, "Hidden Markov Models explained with a real life example and Python code,"Medium(TDS Archive),Nov.5,2023.[Online]. Available: https://medium.com/data-science/hidden-markov-models-explained-with-a-real-life-example-and-python-code-2df2a7956d6 [Accessed: Apr. 16, 2026].

[37] GFG, “What Is the Difference Between Markov Chains and Hidden Markov Models?,” GeeksforGeeks, Jul. 23, 2025. [Online]. Available: https://www.geeksforgeeks.org/what-is-the-difference-between-markov-chains-and-hidden-markov-models/ .[Accessed: Apr. 17, 2026].

[38] F. Mortezapour Shiri, T. Perumal, N. Mustapha, and R. Mohamed, “A Comprehensive Overview and Comparative Analysis on Deep Learning Models,” Journal on Artificial Intelligence, Tech Science Press, Nov. 20, 2024. doi: 10.32604/jai.2024.054314.

[39] GFG, “Introduction to Convolution Neural Network,” GeeksforGeeks, Jul. 11, 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning/introduction-convolution-neural-network/.[Accessed: Apr. 17, 2026].

[40] R. D. Knotts, Natural Language Processing with Transformers and Python: Practical AI Solutions. Packt Publishing, 2023.

[41] A. Vaswani et al., “Attention Is All You Need,” in Proc. 31st Int. Conf. Neural Information Processing Systems (NeurIPS), Long Beach, CA, USA, Dec. 2017, pp. 5998–6008.

[42] T. Bäckström et al., “Introduction to Speech Processing,” 2nd ed., Aalto University, 2022. [Online]. Available: https://speechprocessingbook.aalto.fi/ [Accessed: Apr. 21, 2026].

[43] A. Mohamed et al., “Self-Supervised Speech Representation Learning: A Review,” arXiv preprint arXiv:2205.10643, 2022.

[44] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," arXiv preprint arXiv:2006.11477, 2020.

[45] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed,“HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 29, pp. 3451–3460, 2021.

[46] S. Chen et al., “WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing,” arXiv preprint arXiv:2110.13900, Oct. 2021.

[47] H. M. Tran, D. Lolive, A. Sini, A. Delhay, P.-F. Marteau, and D. Guennec, “Multi-level SSL Feature Gating for Audio Deepfake Detection,” Proc. 3rd ACM Int. Conf. Multimedia in MENA, New York, NY, USA, pp. 19–28, Oct. 27–31, 2025.

[48] I. Kukanov and J. W. Ng, “KLASSify to Verify: Audio-Visual Deepfake Detection Using SSL-based Audio and Handcrafted Visual Features,” arXiv preprint arXiv:2508.07337v1, Aug. 2025.

[49] Y. El Kheir, Y. Samih, S. Maharjan, T. Polzehl, and S. Möller, “Comprehensive Layer-wise Analysis of SSL Models for Audio Deepfake Detection,” Findings of the Association for Computational Linguistics: NAACL 2025, Albuquerque, NM, USA, pp. 4070–4082, Apr.–May 2025, doi: 10.18653/v1/2025.findings-naacl.227.

[50] L. Baraldi, F. Cocchi, M. Cornia, A. Nicolosi, and R. Cucchiara, “Contrasting Deepfakes Diffusion via Contrastive Learning and Global-Local Similarities,” arXiv preprint arXiv:2311.16851, 2023.

[51] Artificial Intelligence Made Simple, “Deepfake Detection: Building the Future,” Artificial Intelligence Made Simple. Available: https://www.artificialintelligencemadesimple.com/p/deepfake-detection-building-the-futureâĄă. Accessed: Apr.24, 2026.

[52] Milvus, “How does SSL relate to transfer learning?,” Milvus AI Reference. [Online]. Available: https://milvus.io/ai-quick-reference/how-does-ssl-relate-to-transfer-learning/. Accessed: Apr. 27, 2026.

[53] M. Marks, M. Knott, N. Kondapaneni, E. Cole, T. Defraeye, F. Perez-Cruz, and P. Perona, “A closer look at benchmarking self-supervised pre-training with image classification,” International Journal of Computer Vision, vol. 133, no. 8, pp. 5013–5025, Apr. 2025. doi: 10.1007/s11263-025-02402-w.

[54] E-SPIN Group, “Self-supervised learning (SSL): Future of scalable, multimodal, and AI,” E-SPIN. [Online]. Available: https://www.e-spincorp.com/self-supervised-learning-ai-future/. Accessed: Apr. 27, 2026.

[55] Anonymous authors, “The Geometry of Self-Supervised Learning Models and Its Impact on Transfer Learning,” Under review as a conference paper at ICLR 2023, 2023.

[56] M. Grandini, E. Bagli, and G. Visani, “Metrics for Multi-Class Classification: An Overview,” arXiv preprint arXiv:2008.05756v1, Aug. 2020.

[57] GeeksforGeeks, “AUC-ROC Curve in Machine Learning,” GeeksforGeeks, Available: https://www.geeksforgeeks.org/machine-learning/auc-roc-curve/âĄă. Accessed: Apr. 27, 2026.

[58] A. Hamza, A. R. Javed, F. Iqbal, N. Kryvinska, A. S. Almahdi, Z. Jalil, and R. Borgohain, “Deepfake Audio Detection via MFCC Features Using Machine Learning,” IEEE Access, vol. 10, pp. 93988–93997, 2022, doi: 10.1109/ACCESS.2022.3231480.

[59] S. A. Khan and D.-T. Dang-Nguyen, “Hybrid Transformer Network for Deepfake Detection,” arXiv preprint, arXiv:2208.05820v1, Aug. 2022.

[60] L. Li, T. Lu, X. Ma, M. Yuan, and D. Wan, “Voice Deepfake Detection Using the Self-Supervised Pre-Training Model HuBERT,” Applied Sciences, vol. 13, no. 13, p. 8488, 2023, doi: 10.3390/app13138488.

[61] F. Doshi-Velez and B. Kim, “Towards A Rigorous Science of Interpretable Machine Learning,” 2017.

[62] N. Mansoor and A. I. Iliev, "Explainable AI for DeepFake Detection," Applied Sciences, vol. 15, no. 725, MDPI, 2025. doi: 10.3390/app15020725.

[63] M. T. Ribeiro, S. Singh, and C. Guestrin, “Why Should I Trust You? Explaining the Predictions of Any Classifier,” Proc. ACM SIGKDD, 2016.

[64] M. Sardar Momin, A. B. Sufian, D. Barman, M. Leo, C. Distante, and N. Damer, "Explainable deepfake detection across different modalities: An overview of methods and challenges," Image and Vision Computing, vol. 163, 105738, 2025. doi: 10.1016/j.imavis.2025.105738.

[65] Google Colab, “Introduction Notebook,” [Online]. Available: https://colab.research.google.com/notebooks/intro.ipynb. [Accessed: Jun. 5, 2026].

[66] Kaggle, “What is Kaggle?,” Kaggle Discussions, 2022. [Online]. Available: https://www.kaggle.com/discussions/general/328265. [Accessed: Jun. 5, 2026].

[67] Hugging Face, “Transformers Documentation,” 2026. [Online]. Available: https://huggingface.co/docs/transformers/main/en/index. [Accessed: Jun. 5, 2026].

[68] Python Software Foundation, “Python 3 Documentation,” [Online]. Available: https://docs.python.org/3/. [Accessed: Jun. 5, 2026].

[69] Google, “Google Drive,” [Online]. Available: https://drive.google.com/. [Accessed: Jun. 5, 2026].

[70] X. Wang, J. Yamagishi, M. Todisco, H. Delgado, N. Evans, T. Kinnunen, K.-A. Lee, V. Vestman, A. Nautsch, M. Sahidullah, and others, “ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech,” *Computer Speech & Language*, vol. 64, p. 101114, 2020.

[71] S. Zian, S. A. Kareem, and K. D. Varathan, "An Empirical Evaluation of Stacked Ensembles With Different Meta-Learners in Imbalanced Classification," *IEEE Access*, vol. 9, pp. 108834–108852, 2021, doi: 10.1109/ACCESS.2021.3088414.

[72] X. Wang, J. Yamagishi, M. Todisco, H. Delgado, A. Nautsch, N. Evans, M. Sahidullah, V. Vestman, T. Kinnunen, K. A. Lee, et al., “ASVspoof 2019: A large-scale public database of synthesized, converted and replayed speech,” Computer Speech & Language, vol. 64, p. 101114, Nov. 2020.

[73] W. Cai, H. Wu, D. Cai, and M. Li,"The DKU Replay Detection System for the ASVspoof 2019 Challenge:On Data Augmentation, Feature Representation, Classification,and Fusion," in Proc. Interspeech 2019,Graz, Austria, 2019, pp. 1023–1027.

[74] J. W. Lee, E. Kim, J. Koo, and K. Lee,"Representation Selective Self-distillation and wav2vec 2.0Feature Exploration for Spoof-aware Speaker Verification,"in Proc. Odyssey Speaker and Language Recognition Workshop,2022.

[75] A. Staněk et al., “What Do Deepfake Speech Detectors Actually Hear?,” arXiv preprint arXiv:2606.10912, 2026.

[76] L. Pham, P. Lam, T. Nguyen, H. Nguyen, and A. Schindler, "Deepfake Audio Detection Using Spectrogram-based Feature and Ensemble of Deep Learning Models," arXiv preprint arXiv:2407.01777, Jul. 2024.