Comparative Analysis of Speech Foundational Models

Panchal Nayak[1] , Shripad Ghone[2] , Anshuman Panda[3] ,
Aditya Raj[4] , Aditya Katoch[5] , Jayaram Reddy A[6]

School of Computer Science Engineering and Information Systems , Vellore Institute Of Technology , Vellore , 632014 , Tamilnadu , India

*Corresponding author(s). E-mail(s): ajayaramreddy@vit.ac.in ;

Contributing authors: panchal.nayak2021@vitstudent.ac.in ;

ghoneshripad.dipak2021@vitstudent.ac.in ;

anshuman.panda2021@vitstdent.ac.in ;

adityaraj.katoch2021@vitstudent.ac.in ;

aditya.raj2021c@vitstudent.ac.in ;

***Abstract*—Emotion detection in crowd scenarios is a challenging yet crucial task with applications in various fields such as marketing, entertainment, and security. We describe a comprehensive comparative analysis of different machine learning models focusing on their accuracy and F1-score metrics for crowd emotion detection.The evaluated models include wavLM, wave2vec, Hubert, X vectors (both with and without MFCC features), KNN, Random Forest, XGBClassifier, Neural Network, and ResNet. A dataset of crowd audio recordings annotated with emotions is used to train and test each model.The primary objective of this study is to identify the most effective model for crowd emotion detection tasks. To achieve this, we meticulously analyze the performance of each model across various metrics, paying particular attention to accuracy, which are commonly used to evaluate classification tasks. Our experimental results reveal compelling insights into the effectiveness of different machine learning models for crowd emotion detection. Notably, the X-vector with MFCC demonstrates superior performance compared to other models across all evaluated metrics. Research and practical applications of these findings have important implications for emotion recognition. By highlighting the effectiveness of the X-Vector, this study provides valuable guidance for researchers and practitioners seeking to develop more accurate and reliable crowd emotion detection systems. Moreover, our study sheds light on feature engineering and model selection as important factors in improving emotion recognition models, which may lead to future innovations.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Emotion detection in crowd scenarios represents a burgeoning field with profound implications for crowd behavior management, safety, and responsive robotics. There has been extensive research focused on individual emotion detection, but a dedicated investigation of crowd emotions presents unique challenges and opportunities. Traditionally, emotion detection methodologies have centered on singleperson analysis, leveraging various modalities such as facial expressions, voice cues, and body language. However, the transition to crowd settings introduces complexities such as the fusion of multimodal features and the temporal constraints inherent in dynamic group interactions. As a result of the sheer volume of data and the need for real-time processing, existing approaches fail to detect emotions in crowd contexts, even when applied to individuals. This paper addresses these challenges by presenting a comparative analysis of machine learning models tailored

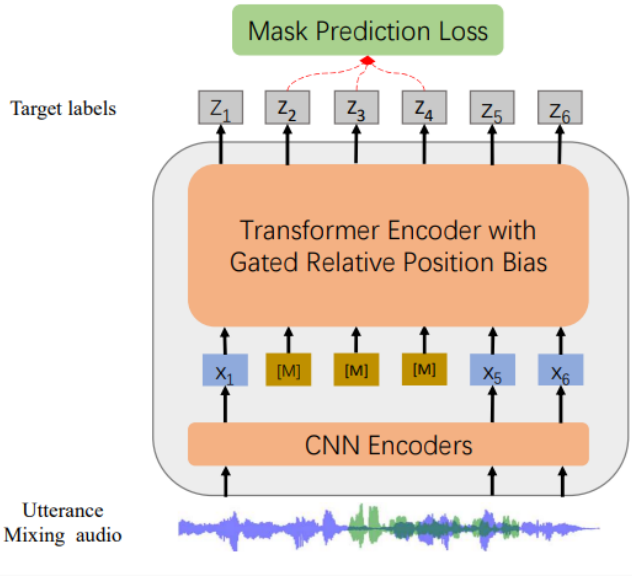
Identify applicable funding agency here. If none, delete this.

for crowd emotion detection. Through meticulous evaluation of accuracy, we assess the performance of prominent models such as WavLM, wave2vec, Hubert, X-vectors (with and without MFCC features)), KNN, Random Forest, XGBClassifier, Neural Network, and ResNet. Analyzing these models, we provide insight into their ability to capture crowd emotions at their

nuanced best. In subsequent sections, we detail the methodology employed for model training and evaluation, followed by a comprehensive discussion of our findings. By shedding light on the strengths and limitations of different machine learning approaches, we aim to inform future research endeavors and contribute to the development of more robust crowd emotion detection systems.

II. Methodology

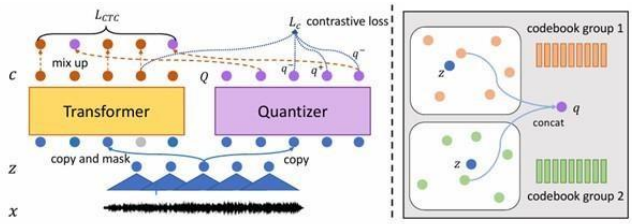
A. WavLM



The WavLM (Waveform Language Model) architecture integrates a convolutional feature encoder and a Transformer encoder. The feature encoder comprises seven blocks of temporal convolution, followed by layer normalization and a GELU activation layer. Each block has 512 channels with specific strides and kernel widths, resulting in representations of about 25ms of audio sampled every 20ms. The convolutional output serves as input to the Transformer encoder, which includes a

convolution-based relative position embedding layer with specified parameters. Additionally, gated relative position bias is employed to adaptively adjust the relative position bias based on the content of the speech, enhancing the model's ability to capture contextual information. The model leverages large-scale unsupervised data from diverse domains, including datasets like GigaSpeech and VoxPopuli, to enhance its robustness. To stabilize training, a simple trick is applied to mitigate overflow issues in large models trained with mixed precision, ensuring stability during training.

B. UniSpeech



The UniSpeech model architecture is designed to tackle the challenge of learning robust representations for automatic speech recognition (ASR) tasks, particularly in low-resource settings. It comprises three main components: a convolutional feature encoder (X_Z), a Transformer context encoder (Z_C), and a vector quantizer (Z_Q). The convolutional feature encoder processes raw audio data through seven blocks of temporal convolution, followed by normalization and activation layers. This results in representations that capture essential features of the audio, with each block producing outputs representing about 25ms of audio sampled every 20ms.

Following the feature encoding stage, the Transformer context encoder refines the extracted features to output context representations. This component plays a crucial role in capturing the contextual information necessary for accurate speech recognition. Additionally, the model includes a vector quantizer, which discretizes the continuous representations into speech units. This step enhances efficiency and allows for a more compact representation of the audio data.

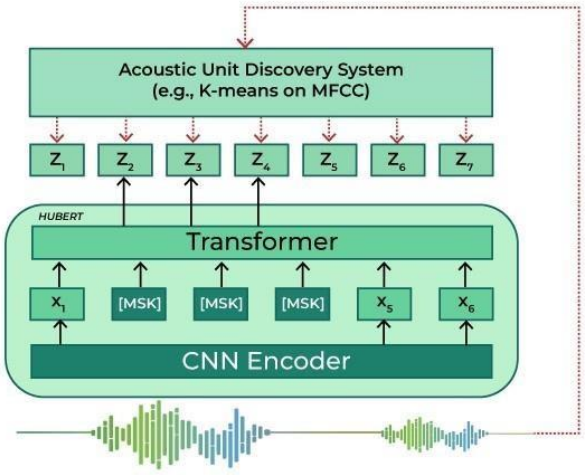
UniSpeech employs a multitask learning approach with unified representation to enhance the learning process. During pre-training, the model is trained on both high-resource labeled datasets (L) and low-resource datasets (M). It leverages three main objectives: Phonetic CTC loss, Contrastive loss, and Contrastive loss on dataset M. These objectives aim to align contextual representations with phonetic units, close the gap between contextual and discrete features, and adapt the model to the target language or domain.

To further enhance the learning process, UniSpeech introduces a novel approach during pre-training. It replaces continuous representations with their quantized versions in the CTC loss calculation. By forcing representations from both supervised and unsupervised learning to project into the same space, the model improves its overall performance and generalization capabilities. After pre-training, the model is finetuned on dataset N using the CTC loss, where the pretrained CTC layer is replaced with a new layer representing the target vocabulary. This fine-tuning process helps tailor the model to the specific characteristics of the target dataset, further improving its performance in low-resource settings.

C. Wav2Vec

Pre-training in the context of wav2vec involves training a neural network on a task with abundant data and then initializing another network with these learned weights. This approach aims to capture general representations from extensive data, which can then be applied to new tasks with limited data. wav2vec's architecture comprises an encoder network, which maps raw audio to a lower-dimensional representation using a 5-layer CNN with specific kernel sizes and strides, and a context network that contextualizes this representation through a 9-layer CNN. The contrastive loss function is utilized to distinguish true samples from negatives, optimized over multiple time steps, enabling effective training without relying heavily on labeled data. The self-supervised learning aspect of wav2vec involves training the model using inherent labels in the input data rather than external labels. This method allows the model to learn useful representations from the data itself, significantly reducing the reliance on labeled data for training. The output of wav2vec serves as input to acoustic models, enhancing the representation of features compared to other methods. Decoding involves considering various language models, including 4-gram, word-based convolutional, and character-based convolutional models, with word sequences decoded using beam-search algorithms. The results demonstrate the efficacy of pre-training, showcasing substantial reductions in Word Error Rate (WER) even with limited labeled data, with improved performance observed across various tasks and benchmarks.

D. Hubert

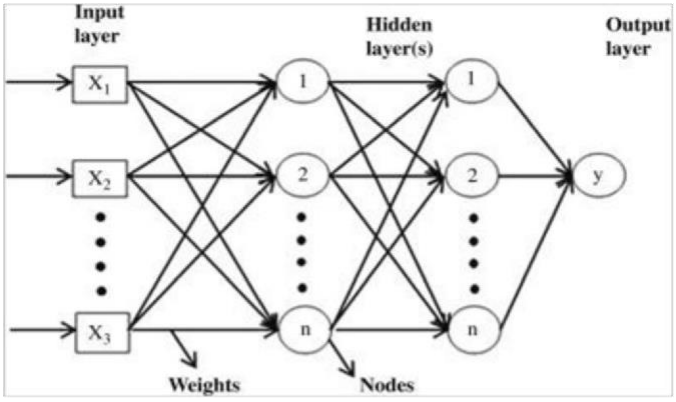


HuBERT, a model for speech recognition, comprises several key components. First, there's a Convolutional Encoder consisting of seven layers with temporal convolutions, reducing input dimensionality and producing latent speech representations. These features are randomly masked for training. Next is the BERT Encoder, the core of the architecture, which transforms the encoded features into hidden unit representations through multi-layer multi-head attention, followed by a feedforward network (FFN) layer. The output is dimensionally expanded based on the model size (Base, Large, X-Large). A Projection Layer adjusts the output dimensions to match the embedding dimension of the clustering step, facilitating cosine similarity calculations for prediction logits.
Then, a Code Embedding Layer converts hidden unit representations to embedding vectors. During ASR fine-tuning, the Projection Layer is replaced with a softmax layer, and a CTC loss function is applied. Finally, a Clustering Layer utilizes kmeans clustering to generate hidden units, initially from MFCC features and later from transformer layer outputs.

E. X-vector

Layer	Layer context	Total context	Input x output
frame1	$[t-2, t+2]$	5	120x512
frame2	$\{t-2, t, t+2\}$	9	1536x512
frame3	$\{t-3, t, t+3\}$	15	1536x512
frame4	$\{t\}$	15	512x512
frame5	$\{t\}$	15	512x1500
stats pooling	$[0, T)$	T	1500Tx3000
segment6	$\{0\}$	T	3000x512
segment7	$\{0\}$	T	512x512
softmax	$\{0\}$	T	512xN

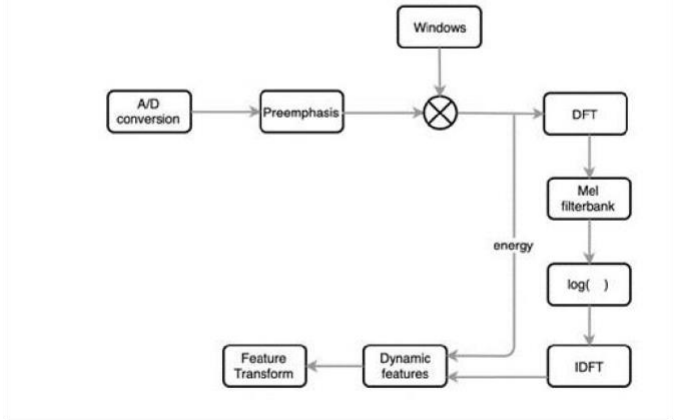
The embedding DNN architecture. x-vectors are extracted at layer segment6, before the nonlinearity. The N in the softmax layer corresponds to the number of training speakers. The features are 24 dimensional filterbanks with a frame-length of 25ms, mean-normalized over a sliding window of up to 3 seconds. The same energy SAD as used in the baseline systems filters out nonspeech frames. The DNN configuration is outlined in Table 1. Suppose an input segment has T frames. The first five layers operate on speech frames, with a small temporal context centered at the current frame t. For example, the input to layer frame3 is the spliced output of frame2, at frames t 3, t and t + 3. This builds on the temporal context of the earlier layers, so that frame3 sees a total context of 15 frames. The statistics pooling layer aggregates all T frame-level outputs from layer frame5 and computes its mean and standard deviation. The statistics are 1500 dimensional vectors, computed once for each input segment. This process aggregates information across the time dimension so that subsequent layers operate on the entire segment. In Table 1, this is denoted by a layer context of 0 and a total context of T. The mean and standard deviation are concatenated together and propagated through segment-level layers and finally the softmax output layer. The nonlinearities are all rectified linear units(ReLUs).The DNNN is trained to classify the N speakers in the training Data.



classification tasks, the class label is determined by majority voting among these neighbors, while for regression tasks, the target value is predicted by averaging the values of the nearest neighbors. Finally, the model's performance is evaluated using appropriate metrics, and hyperparameters such as k and the distance metric may be optimized through techniques like cross-validation to enhance performance.

The XGBoost classifier, short for Extreme Gradient Boost-

F. MFCC

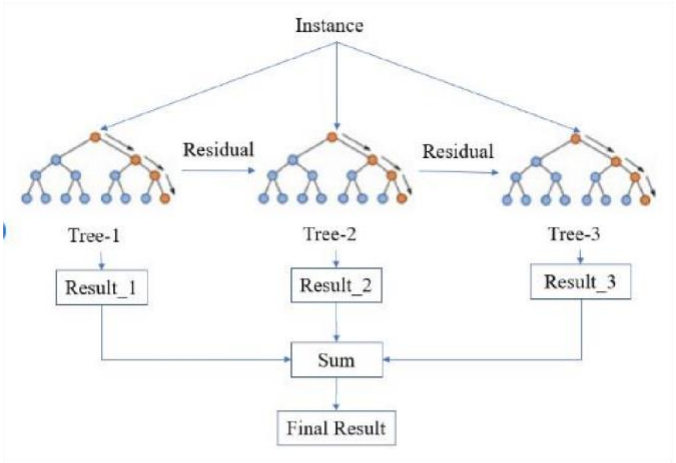


and preemphasis to boost higher frequencies, the process segments audio into frames, applies windowing to reduce spectral leakage, and computes the Discrete Fourier Transform (DFT) for frequency domain analysis. Leveraging the melscale

Mel-frequency cepstral coefficients (MFCC) are a powerful feature extraction technique widely employed in speech and audio processing. Beginning with analog-to-digital conversion

excels in robustness and scalability, making it a popular choice for various machine learning applications.

I. XGBClassifier



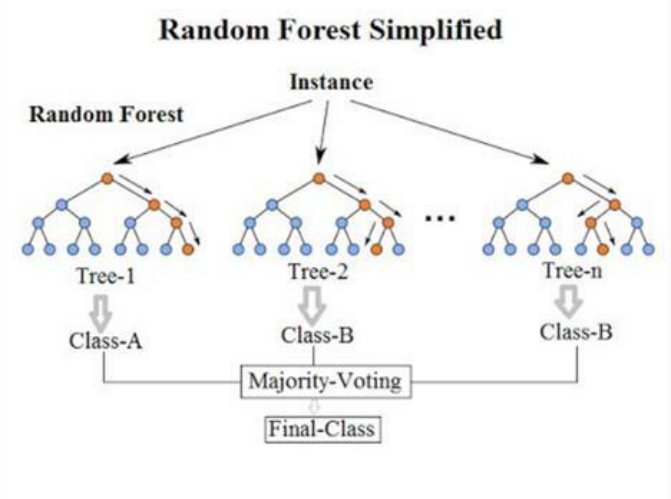
like speech recognition. In essence, MFCC involves steps from initial signal conversion to dynamic feature computation, effectively encapsulating the spectral essence of audio signals. By strategically emphasizing relevant spectral components while discarding noise, MFCC captures salient features crucial for human speech perception. These features, extracted through a series of transformations and enhancements, serve as potent inputs for various machine learning tasks, facilitating tasks such as speaker recognition, emotion detection, and speechto-text conversion with remarkable efficiency and accuracy.

G. KNN

The k-Nearest Neighbors (k-NN) model operates by initializing the value of k, determining the number of nearest neighbors to consider, and optionally selecting a distance metric such as Euclidean distance. There is no explicit training phase; instead, the model memorizes the training data. During prediction, the distance between the test data point and all training data points is calculated using the chosen distance metric. Then, the k-nearest neighbors are identified, and for

to emulate human auditory perception, it constructs a MelFilter Bank to capture relevant frequency information and applies logarithmic compression to mimic human hearing sensitivity. Inverse DFT yields cepstral coefficients, representing spectral characteristics. Finally, dynamic features, including derivatives, enhance discrimination, resulting in 39 distinctive features per audio frame, pivotal for applications

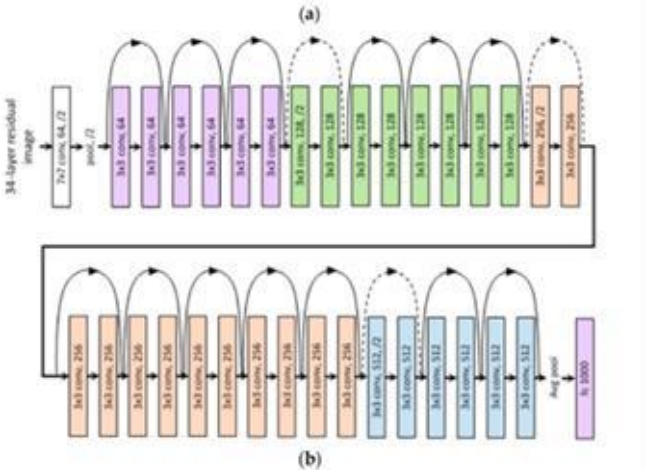
H. Random Forest



The Random Forest algorithm is structured as an ensemble of decision trees, where each tree is constructed using bootstrapped samples from the training data and feature randomization. During training, multiple decision trees are grown deep without pruning to capture complex relationships in the data. At each split in the trees, only a random subset of features is considered, enhancing diversity among the trees and reducing overfitting. For classification tasks, predictions are made by aggregating the votes of all decision trees using majority voting, while for regression tasks, predictions are averaged across all trees. The model's performance is evaluated using appropriate metrics, and hyperparameters are fine-tuned to optimize performance. Overall, Random Forest ing, is structured as a powerful ensemble learning algorithm that leverages the boosting technique to build a collection of weak learners, typically decision trees, in a sequential manner. Each subsequent tree is trained to correct the errors made by the previous ones, with a focus on minimizing a predefined loss function. The algorithm starts with an initial base learner and iteratively adds new trees, optimizing the model's performance by considering the gradient of the loss function. XGBoost incorporates regularization techniques to prevent overfitting, such as shrinkage (or learning rate) and maximum depth constraints for individual trees. Additionally, it utilizes feature importance scores to assess the relevance of input features and can handle missing values internally. Upon completion of training, predictions are made by aggregating the outputs of all trees. The model's hyperparameters, including the number of trees and the learning rate, can be finetuned through techniques like cross-validation to optimize performance for classification tasks. Overall, XGBoost is highly efficient, scalable, and widely used across various domains due to its remarkable predictive capabilities and robustness against overfitting.

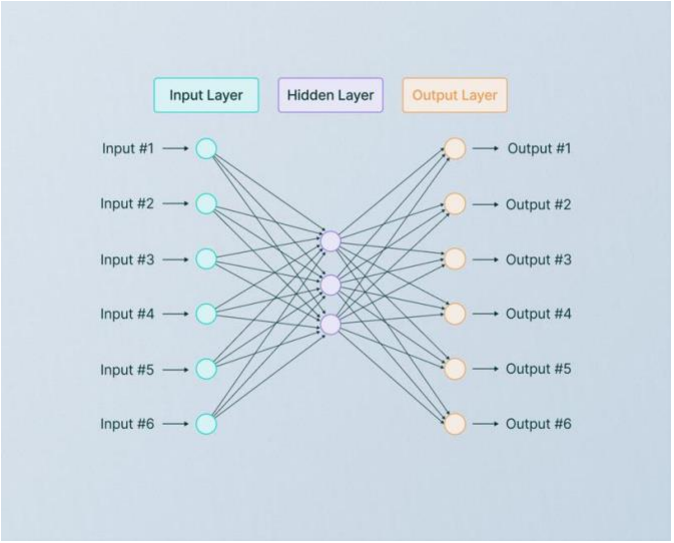
J. ResNet

ResNet (Residual Neural Network) features a deep architecture with skip connections, specifically residual blocks, which facilitate the training of very deep networks. Each residual block contains a shortcut connection that bypasses one or more layers, allowing the network to learn residual functions instead of directly mapping inputs to outputs. This architecture mitigates the vanishing gradient problem and enables the training of deeper networks by preserving useful information and facilitating the flow of gradients during backpropagation. In essence, ResNet's structure promotes easier optimization and convergence in training, leading to improved performance



on various tasks, especially in computer vision applications such as image classification and object detection.

K. Neural Network



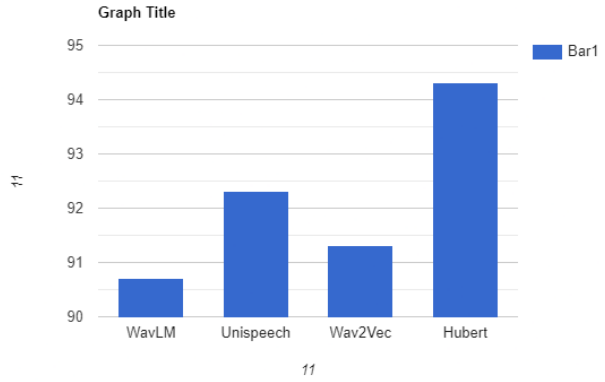
A neural network structure typically consists of interconnected layers of neurons, including an input layer, one

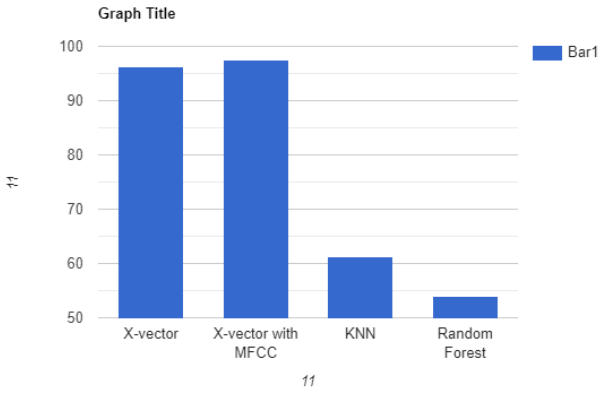
or more hidden layers, and an output layer. The input layer receives raw input data, with each neuron representing a feature. Hidden layers, positioned between the input and output layers, conduct most of the computations, performing weighted sums of inputs followed by activation functions to introduce non-linearity. Multiple hidden layers in deep neural networks enable learning complex representations of input data. Finally, the output layer generates the network's predictions or outputs based on the computations performed in the hidden layers.

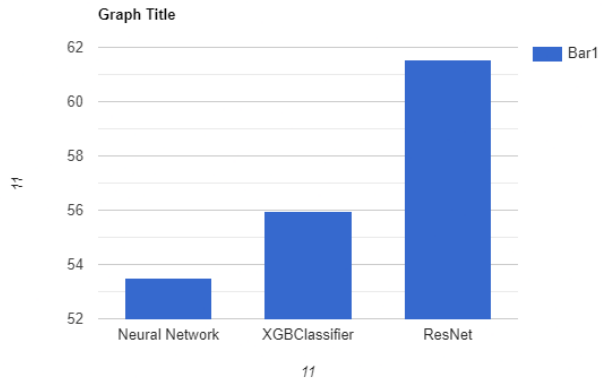
III. Result and Discussion

The training to testing ratio is taken as 80:20 to obtain optimal results. The Dataset includes three folders approval, disapproval and neutral in which the approval folder has 468 audio files, the disapproval files have 240 audio files, and the neutral folder has 3754 audio files. There is total 10 models comparison result based on their Accuracy. Also, the X- vector is model has been used twice, with and without MFCC. The following table represents the Accuracy and F1 score of 6 different models which include WavLM, Unispeech, Wav2Vec, Hubert, X-vector (with and without using MFCC), KNN, Random Forest, XGBClassifier, Neural Network, and
ResNet. From the table's data we found that X-vector with MFCC outperforms other models in terms of accuracy.

S.No.	Model	Accuracy
1.	WavLM	90.7143
2.	Unispeech	92.3214
3.	Wav2Vec	91.3104
4.	Hubert	94.3145
5.	X-vector	96.35
6.	X-vector with MFCC	97.53
7.	KNN	61.30
8.	Random Forest	54.00
9.	Neural Network	53.50
10.	XGBClassifier	55.95
11.	ResNet	61.55







IV. Conclusion

X- Vector model with MFCC outperformed other models among the ten trained and evaluated models. Considering that fact that X-vector model provides almost 99 to 100 percent accuracy, it has proven to be a reliable and effective model.

References

[1] Sanyuan Chen*, Chengyi Wang*, Zhengyang Chen*, Yu Wu*, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong

Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, Furu Wei, ”WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing ” IEEE Journal of Selected Topics in Signal Processing 2022

[2] Kong Aik Lee, Senior Member, IEEE, Qiongqiong Wang, and Takafumi Koshinaka, ”Xi-Vector Embedding for Speaker Recognition” IEEE SPL 2021

[3] Sanyuan Chen, Yu Wu, Chengyi Wang, Zhengyang Chen, Zhuo Chen , Shujie Liu.Jian Wu, Yao Qian, Furu Wei, Jinyu Li, Xiangzhan Yu, ”UNISPEECH-SAT: UNIVERSAL SPEECH REPRESENTATION LEARNING WITH SPEAKER AWARE PRE-TRAINING”, IEEE Access,2021

[4] Steffen Schneider, Alexei Baevski, Ronan Collobert, Michael Auli, ”WAV2VEC: UNSUPERVISED PRE-TRAINING FOR SPEECH RECOGNITION”, INTERPSPEECH 2019

[5] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, Abdelrahman Mohamed, ”HuBERT: SelfSupervised Speech Representation Learning by Masked Prediction of Hidden Units”, IEEE/ACM Transactions on Audio, Speech, and Language Processing,2021

[6] Kaur Gurpreet, Srivastava Mohit and Kumar Amod, "Speaker and Speech Recognition using Deep Neural Network", International Journal of Emerging Research in Management and Technology, vol. 6.118, 2018.

[7] L. Deng and X. Li, "Machine learning paradigms for speech recognition: An overview", IEEE Trans. Audio Speech Lang. Process., vol. 21, no. 5, pp. 1060-1089,May2013.

[8] R. Cowie, E. Douglas-Cowie, N. Tsapatsoulis, G. Votsis, S. Kollias, W. Fellenz, et al., "Emotion recognition in human-computer interaction", IEEE Signal Processing Magazine, vol. 18, no. 1, pp. 3280, 2001.

[9] B. Schuller, B. Vlasenko, F. Eyben, G. Rigoll and A. Wendemuth, "Acoustic emotion recognition: A benchmark comparison of performances", 2009 IEEE Workshop on Automatic Speech Recognition & Understanding, pp. 552-557, 2009.

[10] A. I. Middya, B. Nag and S. Roy, "Deep learning based multimodal emotion recognition using model-level fusion of audio-visual modalities", Knowledge-Based Systems, no. 244, pp. 108580, 2022.