Speaker Verification using Machine Learning and Deep Learning Approaches on the VoxCeleb Dataset

Yixiong Chen (ychen646), Zuojun Zhou (zzhou111)
Susan Wang (qwang95), Yuhao Zheng (yzhen130)

December 5, 2024

1 Introduction

Speaker verification is important for secure authentication, voice-controlled devices, and personalized experiences. This project aims to build models that can identify speakers from audio recordings by leveraging both traditional machine learning methods and advanced deep learning techniques.

The VoxCeleb dataset, which is large and noisy, was used to test the models under real-world conditions. Traditional methods like Support Vector Machines (SVMs), Random Forests, XGBoost were compared with fine-tuned deep learning models. Traditional methods are often faster and require fewer computational resources, while deep learning models generally achieve higher accuracy due to their capacity for learning complex, speaker-specific representations from raw audio.

This project compares the strengths and weaknesses of these methods. It highlights the trade-offs between accuracy, computational cost, and robustness. While deep learning usually performs best, traditional methods can still be viable under constrained scenarios. Such insights help guide the choice of approach depending on the application's requirements.

2 Data Description

The VoxCeleb dataset is used as the primary data source for this project. It is a large-scale, real-world audio dataset containing over 1 million utterances from thousands of speakers across diverse demographics and environmental conditions. The dataset is characterized by its scale, diversity, and inclusion of real-world noise and channel variability, making it ideal for evaluating speaker verification systems.

2.1 Key Characteristics

- **Scale and Diversity:** The dataset includes over 1 million audio clips from thousands of different speakers with a wide variety of accents and backgrounds.
- **Real-world Noise and Variability:** The audio recordings often have background sounds, poor quality channels, and are made in different environments, which makes it a challenge to test how strong the models are.
- **Rich Features:** The dataset is not pre-processed, allowing for comprehensive preprocessing, feature extraction, and augmentation techniques to be applied for fair comparisons between machine learning approaches.

2.2 Data Splitting

The dataset is split into training and testing subsets, ensuring that the data from individual speakers does not overlap between the subsets. The audio samples of each speaker are divided at the video level, with 80% used for training and 20% for testing. For all experiments except those involving the abandoned HuBERT model (due to computational constraints), a subset of 200 speakers is consistently used. This ensures that models are evaluated on unseen speakers within a controlled setting, providing a realistic measure of generalization performance.

3 Methodology

This section outlines the steps followed to preprocess the data, extract features, and train machine learning and deep learning models for speaker verification. The methodology ensures consistent preparation and evaluation across all approaches.

3.1 Preprocessing

The raw audio data from the VoxCeleb dataset was standardized to ensure consistency. Audio files were resampled to 16 kHz to maintain a uniform time resolution. Silence and background noise were removed using energy-based thresholding with the `librosa` library, focusing on speech segments. To make the models robust to variations in the data, augmentation techniques such as pitch shifting, time masking, and additive noise were applied.

3.2 Feature Extraction

To represent each audio recording, we extracted a comprehensive set of features using the `librosa` library. The process began with energy-based trimming to remove low-intensity segments that likely contain noise or silence. Specifically, any segment with an energy level below a certain threshold was discarded to reduce computation time and exclude non-informative samples.

After trimming, fundamental frequency (f_0) was estimated using `pyin`, and harmonicity was measured to assess the presence of vocal fold vibrations. If both f_0 mean and standard deviation were zero, and the harmonic-to-noise ratio (HNR) was below a specified threshold (e.g., 0.2), the segment was treated as non-vocal or dominated by noise and thus removed.

For segments considered valid, we extracted a range of traditional audio features:

- **Pitch-Related Features:** Mean and standard deviation of f_0, as well as HNR, to capture vocal quality.
- **MFCCs and Delta-MFCCs:** 13 Mel-Frequency Cepstral Coefficients (MFCCs) and their first-order deltas, averaged across time.
- **Spectral Features:** Spectral centroid, spectral contrast, and chroma features to capture brightness, harmonic distribution, and tonal information of the signal.

- **Zero-Crossing Rate (ZCR):** Reflects the noisiness of the signal.
- **Energy:** Overall energy level of the processed segment.

These features, concatenated into a single vector, served as inputs to our traditional machine learning models. To identify which features contributed most to classification performance, we utilized an XGBoost-based feature selection approach.

Figure 1 illustrates why spectral features are valuable. The spectrogram shows the energy of different frequency bands over time. From this representation, we derive spectral features like centroid and contrast, which help differentiate one speaker's voice from another by capturing distinct frequency characteristics.

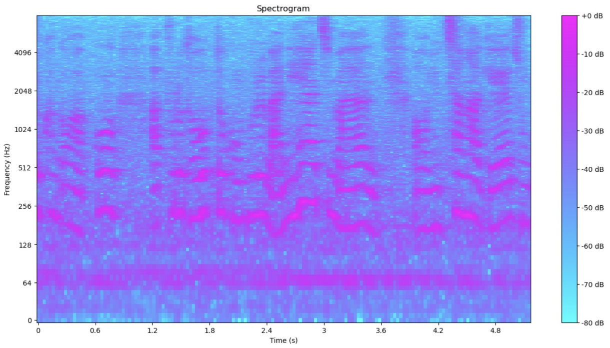


Figure 1: A spectrogram representation of an audio segment

For deep learning models, raw audio data or log-Mel spectrograms were used directly as inputs. These approaches allowed the models to learn features automatically without manual feature engineering.

3.3 Machine Learning Models

Traditional machine learning methods, including Support Vector Machines (SVM), Random Forest, and XGBoost, were trained on the extracted features:

- **SVM:** Tested with linear and RBF kernels. Hyperparameters such as C and γ were optimized using grid search.

- **Random Forest:** Built multiple decision trees with optimized tree depth and the number of estimators to balance performance and complexity.
- **XGBoost:** A more advanced tree-based method, tuned using parameters like learning rate, maximum depth, and subsample ratio.

3.4 Deep Learning Models

Deep learning approaches included both custom-built and pre-trained models:

- **MLP Models:** Three Multilayer Perceptron (MLP) architectures—SmallNet, MediumNet, and LargeNet—were trained from scratch using extracted embeddings.
- **Pre-trained Models:** Wav2Vec and HuBERT, trained on large speech datasets, were fine-tuned on the VoxCeleb dataset (200-speaker subset for Wav2Vec, and an attempt with HuBERT that was later abandoned due to computational constraints). Fine-tuning involved replacing the final layer to classify speakers and training on task-specific data while leveraging pre-trained knowledge.
- **YAMNet:** Pre-trained for sound event detection, YAMNet was fine-tuned on a 200-speaker subset of the dataset using embeddings extracted from audio. The extracted audio features from YAMNet have a dimensionality of 1024.

3.5 Evaluation Metrics

The primary evaluation metric used in this study is **Accuracy**, defined as the percentage of correctly classified samples. Accuracy provides a straightforward and interpretable measure for comparing different models and configurations. Given the large-scale and multi-class nature of the speaker verification task, focusing on accuracy allows for a clear distinction in performance levels between traditional machine learning methods and deep learning models.

In addition to accuracy, we also considered **Training Time** qualitatively, discussing the relative computational overhead associated with training and fine-tuning deep learning models compared to traditional machine learning methods. While not quantified in detail, these discussions help provide insights into the feasibility and resource requirements for deploying each model in real-world scenarios.

By concentrating on accuracy and qualitatively assessing training time, we ensure a clear and consistent basis for comparing models.

4 Experiments

This section provides an overview of the experimental setup and the key observations from initial tests. Detailed model-specific results, including parameter tuning and feature selection, are presented later in the ”Machine Learning Results” section.

4.1 Experiment Setup

All experiments were conducted using the VoxCeleb dataset, which offers a diverse set of speakers and authentic environmental conditions. The dataset was split into 80% training and 20% testing, ensuring no speaker overlap between these subsets. Except for the abandoned HuBERT fine-tuning attempt, a consistent subset of 200 speakers was used across most experiments, enabling a fair comparison of approaches.

For basic machine learning models, hand-crafted features such as Mel-Frequency Cepstral Coefficients (MFCCs) and Zero Crossing Rate (ZCR) were extracted. In contrast, deep learning models utilized raw audio or spectrogram inputs. Traditional models like Support Vector Machines (SVM) and Random Forest were initially trained with default parameters, while pre-trained deep learning models (e.g., Wav2Vec) were later fine-tuned for the speaker verification task.

4.2 Initial Observations

Baseline experiments showed that simple machine learning models provided moderate accuracy with low computational overhead but lacked the complexity needed for high accuracy. Even without extensive task-specific tuning, pre-trained deep learning models generally outperformed these basic baselines, suggesting that learned representations from large datasets offer a strong starting point.

Subsequent tuning efforts and experiments with pre-trained models revealed that deep learning approaches typically maintained a significant performance advantage over traditional methods.

In general, deep learning models proved more adept at capturing the subtle attributes of speakers' voices, while traditional machine learning methods offered simplicity and faster training times. The following sections detail the fine-grained experiments, including parameter searches, feature selection, and pre-trained model adaptation.

5 Machine Learning Results

This section presents the detailed results of machine learning methods alongside comparisons with deep learning models. The experiments are evaluated primarily by accuracy, providing a clear measure to distinguish among different approaches.

5.1 Baseline Performance

The baseline experiments used default settings for all models and served as a starting point for comparison. Traditional machine learning models, such as SVM and Random Forest, had moderate accuracy levels. Even at baseline, when compared to non-fine-tuned deep models, the deep representations tended to show more promise. Table 1 shows the results of these baseline experiments.

Model	**Validation Accuracy (%)**
Naive Bayes	24.15
Logistic Regression	36.41
Support Vector Machine (SVM)	37.61
Random Forest	33.83

Table 1: Baseline Performance of Models

5.2 XGBoost Model and Parameter Tuning

XGBoost is a tree-based algorithm known for its efficiency and strong performance on structured data. In this project, we used grid search with three-fold cross-validation to find the best hyperparameters for XGBoost on a 200-speaker subset of the VoxCeleb dataset.

5.2.1 Parameter Search

We tested different combinations of key parameters, including:

- **learning_rate**: 0.05
- **max_depth**: {5, 7, 10}
- **n_estimators**: {500, 700, 1000}
- **subsample**: {0.4, 0.6}
- **colsample_bytree**: {0.4, 0.6}
- **gamma**: {0.1, 0.3}
- **reg_alpha**: {0.1, 0.5}
- **reg_lambda**: {2, 5}

The scoring metric was accuracy. By searching over these ranges, we aimed to find a balance that improved accuracy without overfitting.

5.2.2 Best Parameters and Results

The grid search found the best parameters to be:

- **colsample_bytree**: 0.4
- **gamma**: 0.1
- **learning_rate**: 0.05
- **max_depth**: 10
- **n_estimators**: 1000

- **reg_alpha**: 0.1
- **reg_lambda**: 2
- **subsample**: 0.4

With these settings, XGBoost achieved a test accuracy of about 35.35%.

6 Deep Learning Results

6.1 MLP Model Results

Three Multilayer Perceptron (MLP) models, SmallNet, MediumNet, and LargeNet, were evaluated for speaker classification on the same 200-speaker subset. They have 2, 3, 4 fully-connected layers, respectively. We want to see the effect brought by the model capacity given the same features. These models were trained for 30 epochs each, and their performance was assessed based on training loss, validation loss, training accuracy, and validation accuracy. Reflecting the data presented in Table 2, the SmallNet model, with its simple two-layer architecture, achieved a training accuracy of 69.94% and a validation accuracy of 21.99%. The MediumNet model reached a training accuracy of 63.83% and a validation accuracy of 20.48%. The most complex model, LargeNet, achieved a training accuracy of 78.49% and a validation accuracy of 21.17%. While LargeNet attained the highest training accuracy among the three models, its validation accuracy was slightly lower than that of SmallNet, which held the highest validation accuracy at 21.99%. Overall, all three MLP models exhibited relatively modest validation performance, indicating that simple feedforward architectures may not fully leverage the complexity in the speaker verification task.

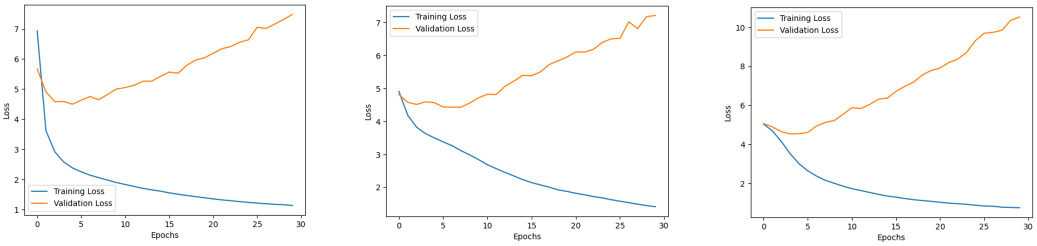
We also found that the training of MLP with deeper layers would become more difficult. When training the three models, the training loss of SmallNet decreases the fastest. The MediumNet struggles at the very begining stage, but still learns successfully at the end. With more parameters and stronger fitting ability, the LargeNet learns more smoothly than MediumNet, but still more slowly than the SmallNet. The learning curves for the three models are shown in Fig. 2.

Although these validation accuracies remained low, these results provide a baseline for more sophisticated deep learning methods. The similar and

Model	Training Accuracy	Val. Accuracy	Training Speed
SmallNet	69.94%	21.99%	1.08s/epoch
MediumNet	63.83%	20.48%	1.29s/epoch
LargeNet	78.49%	21.17%	1.57s/epoch

Table 2: Performance of MLP Models

relatively low validation accuracies across all three MLP architectures highlight the difficulty of capturing speaker-specific characteristics through basic MLP models. It suggests that deeper architectures, pre-trained models, or more complex network structures might be necessary to significantly improve performance on this dataset.



(a) SmallNet (b) MediumNet (c) LargeNet

Figure 2: Loss curves for SmallNet, MediumNet, and LargeNet.

6.2 Fine-Tuning Pre-Trained Models

Fine-tuning was explored using the HuBERT model, a pre-trained transformer designed for speech tasks, originally intended for a larger set of speakers. Due to very high computational demands and challenges with uninitialized weights, full fine-tuning of HuBERT was abandoned. The time consumption for one iteration is ≈ 17 seconds for a batch of two audio clips, which is equivalent to 203 hours for an epoch. In contrast, focusing on a stable 200-speaker scenario made computational considerations more manageable for other pre-trained models like Wav2Vec.

6.3 Fine-Tuning Wav2Vec Model

Fine-tuning pre-trained models has become an effective strategy for achieving high performance on domain-specific tasks. In this study, the `Wav2Vec` model was fine-tuned for speaker classification on a 200-speaker subset of the VoxCeleb dataset. The model was adapted to classify 200 speaker labels by modifying its output layer.

The fine-tuning process involved training the model for four epochs using an AdamW optimizer with a learning rate of 1×10^{-5}. Training and validation accuracies steadily improved across epochs, reaching a final training accuracy of 84.92% and a validation accuracy of 60.03%. Correspondingly, training and validation losses decreased consistently. These results demonstrate the capability of Wav2Vec models to extract meaningful audio representations that benefit the speaker classification task.

The main challenge encountered during fine-tuning was the computational cost associated with large pre-trained models. Despite these limitations, the model showed strong generalization capabilities, as indicated by the steadily increasing validation accuracy. Figures 3 and 4 illustrate the trends in training and validation loss, as well as accuracy, over epochs. It took us about 7 hours (3.8 iterations per second) to train the model for 10 epochs, and the performance does not plateau at this point.

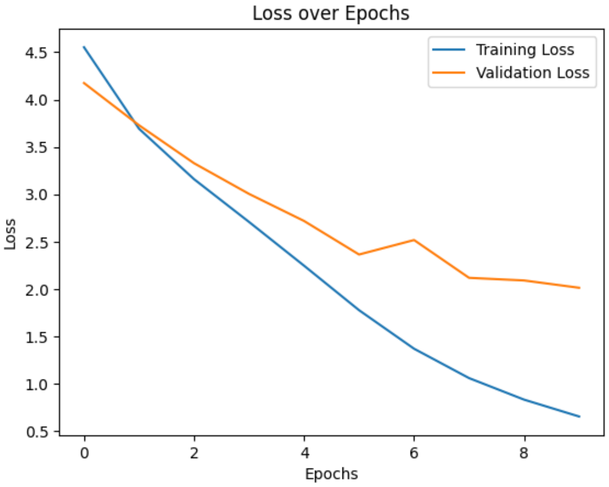


Figure 3: Training and Validation Loss over Epochs for Wav2Vec Model.

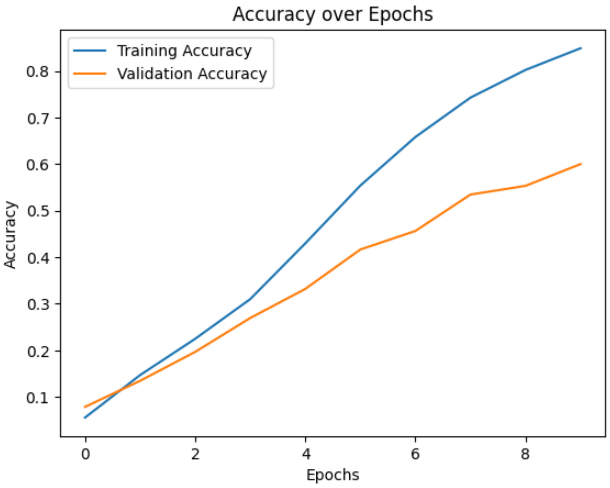


Figure 4: Training and Validation Accuracy over Epochs for Wav2Vec Model.

6.4 Fine-Tuning YAMNet

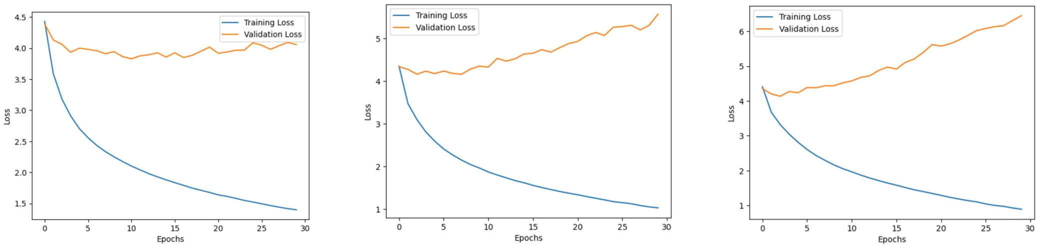
YAMNet, a pre-trained model designed for sound event detection, was fine-tuned for speaker classification tasks using a consistent subset of 200 speakers from the VoxCeleb dataset. The model was loaded from TensorFlow Hub, and its feature extraction capabilities were adapted to classify 200 speaker labels. Input audio files were preprocessed using `librosa` to ensure mono audio and consistent sampling rates, followed by feature extraction through the YAMNet model. The extracted features from YAMNet have a dimensionality of 1024.

After feature extraction for both training and validation datasets, the processed data focused on 200 speakers, resulting in training and validation feature sets of sizes (18,385, 1,024) and (5,424, 1,024), respectively. A classification model (the same as the aforementioned three MLPs) built on top of these embeddings, was trained for 30 epochs with the Adam optimizer. The performance of the three model variants is shown in Table. 3. Based on YAMNet embeddings, we find the three networks performs more stable, where larger models have higher training performance but lower validation performance. We also show their learning curves in Fig. 5. Further optimization and exploration of different network architectures or training strategies could improve its performance for this specific task.

The reason why MLPs on YAMNet embeddings perform better and more

Model	Training Accuracy	Val. Accuracy	Training Speed
SmallNet	64.45%	23.91%	1.21s/epoch
MediumNet	70.88%	21.40%	1.36s/epoch
LargeNet	74.09%	20.08%	1.49s/epoch

Table 3: Performance of MLP Models on YAMNet embeddings.



(a) SmallNet (b) MediumNet (c) LargeNet

Figure 5: Loss curves for SmallNet, MediumNet, and LargeNet with YAMNet embeddings.

predictable than pre-defined features can be inferred as follows: 1) YAMNet embeddings capture hierarchical, high-level features learned from AudioSet, which includes a wide range of acoustic events beyond just speech. These representations encapsulate broader context and nuances of audio signals, which predefined features like MFCCs or ZCR may not capture. 2) Predefined features such as MFCC, ZCR, and RMSE are engineered for specific low-level acoustic properties (e.g., frequency content or signal amplitude). They are not inherently optimized for speaker-related characteristics, while YAMNet embeddings are derived from a model trained to distinguish hundreds of audio events, making them better suited for complex patterns related to speaker characteristics. 3) YAMNet embeddings are higher-dimensional and denser compared to the low-dimensional predefined features. This provides the MLP classifier with a richer feature space to distinguish speakers, enhancing identification performance.

And we hypothesize the reason behind YAMNet that it cannot outperform Wav2Vec model: 1) Domain gap problem. YAMNet is trained on AudioSet, which includes general environmental sounds and audio events. It is not specialized for speech-related tasks or speaker characteristics. But Wav2Vec is specifically pre-trained on speech corpora using self-supervised learning. This focus enables it to better encode features directly relevant to speech and speaker characteristics. 2) Based on MobileNetV1, it is lightweight and designed for efficiency. While Wav2Vec is a much larger model with significantly higher capacity, enabling it to learn and represent more complex patterns in speech and speaker identity. 3) Wav2Vec incorporates sequential context better, thanks to transformer-based architecture or convolutional time modeling. But YAMNet uses a simpler convolutional structure that does not model long-term temporal dependencies as effectively.

7 Discussion of Results

This section comprehensively analyzes the experimental findings, comparing traditional machine learning (ML) approaches and deep learning (DL) methods for speaker verification using the VoxCeleb dataset. We delve deeper into the nuances behind their performance differences, the role of feature representations, the impact of computational resources, and the broader implications for real-world deployment.

7.1 Overall Performance Landscape

Across the experiments, deep learning models offered superior performance compared to traditional ML methods. The fine-tuned Wav2Vec model achieved a validation accuracy of about 60.03% on the 200-speaker subset, outperforming baseline ML models such as SVM, Random Forest, Logistic Regression, and Naive Bayes. It also outperformed basic MLP architectures trained solely on hand-crafted features.

In contrast, the best-performing traditional ML approach (SVM) reached about 37.61% accuracy, it falls significantly short of Wav2Vec's performance. This gap underscores the inherent strength of DL models that can learn speaker-specific characteristics directly from raw audio, rather than relying on pre-defined, potentially limiting feature sets.

7.2 Influence of Feature Representation

One key factor differentiating ML and DL models is how features are obtained. Traditional ML approaches rely on hand-crafted features like MFCCs and spectral statistics. Although these features capture basic acoustic patterns, they lack the richness needed to represent the subtle vocal nuances that distinguish one individual's voice from another.

By contrast, DL methods, especially those processing raw audio (e.g., Wav2Vec), learn task-optimized representations end-to-end. This capacity allows DL models to extract intricate patterns directly from the waveform, yielding embeddings that better align with speaker-specific cues. Pre-trained models like Wav2Vec, which leverage large-scale, self-supervised learning on speech data, come equipped with rich acoustic priors that facilitate more effective adaptation to the speaker verification task.

7.3 Hardware and Computational Considerations

The experiments were conducted using different hardware platforms. Traditional ML models were trained on an Apple M2 Pro CPU, benefiting from quick iteration and minimal hardware requirements. In contrast, DL models were trained on an AMD Ryzen 5900X CPU and an NVIDIA RTX 3060 GPU, offering more computational power suitable for training large neural networks and fine-tuning pre-trained models.

8 Conclusion and Future Work

In summary, the comprehensive experiments and extended analyses highlight that while traditional ML methods remain simpler and more resource-friendly, they rarely rival the accuracy of deep learning models in challenging speaker verification scenarios. Deep learning's end-to-end feature learning capability, particularly with models like Wav2Vec that ingest raw audio, proves far more effective in capturing speaker-specific nuances. Although tuning XGBoost or similar ML methods can yield competitiveness under carefully engineered conditions, these instances are the exception rather than the rule.

Looking ahead, the research could focus on making DL methods more resource-efficient, exploring pre-trained embeddings tailored for speaker verification, and refining the training pipelines to reduce computational overhead without sacrificing performance. Integrating ML and DL strategies may also open new avenues, combining the interpretability and speed of ML with the representational power of DL.

References

[1] Chen, T., & Guestrin, C. (2016). *XGBoost: A scalable tree boosting system.* Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. `https://arxiv.org/abs/1603.02754`

[2] Schneider, S., Baevski, A., Collobert, R., & Auli, M. (2019). *wav2vec: Unsupervised pre-training for speech recognition.* In *Advances in Neural Information Processing Systems.* `https://arxiv.org/abs/1904.05862`

[3] Hsu, W.-N., Heigold, G., Bollegala, D., Chen, J., Eisenschlos, J., Raffel, C., & Auli, M. (2021). *HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units. arXiv preprint arXiv:2106.07447.* `https://arxiv.org/abs/2106.07447`

[4] McFee, B., et al. (2015). *librosa: Audio and music signal analysis in python.* Proceedings of the 14th Python in Science Conference. `https://doi.org/10.25080/Majora-7b98e3ed-003`

[5] Plakal, M., Ellis, D., & Google Inc. (2020). *YAMNet: A Pretrained Audio Event Classifier.* `https://github.com/tensorflow/models/tree/master/research/audioset/yamnet`

[6] Praat Developers. *pyin: Pitch tracking.* `https://librosa.org/doc/main/generated/librosa.pyin.html`

[7] Loshchilov, I., & Hutter, F. (2017). *Decoupled Weight Decay Regularization. arXiv preprint arXiv:1711.05101.* `https://arxiv.org/abs/1711.05101`

[8] TensorFlow Hub Team. *TensorFlow Hub.* `https://www.tensorflow.org/hub`