Lie Detection Using Audio Classification

*

Ayush Kumar[1] Ansh Bhatia[2] Rishikesh Suresh Kumar[3]

Technical Answers to Real World Problems, BTech Computer Science Engineering Core
Vellore Institute of Technology, Vellore, Tamil Nadu

[1]ayush.kumar2020a@vitstudent.ac.in [2]ansh.bhatia2020a@vitstudent.ac.in
[3]rishikeshsuresh.kumar2020a@vitstudent.ac.in

Under the guidance of Prof. Tamizharasi T

***Abstract*—**Deception detection has been a challenge in recent research, and developing a non-contact application for estimating physiological changes during lies is a current objective. This research project aims to use verbal modalities to develop a deception detection system using speech to distinguish between truthful and deceptive statements. Real-life trial data is converted to audio files to detect changes in modality during deception. The goal is to develop an efficient and reliable tool that can assist in lie detection and reduce human error and subjectivity in fields such as law enforcement, legal proceedings, and personnel selection. This technology has the potential to significantly improve the accuracy of lie detection compared to traditional methods and could be used in a variety of applications, from criminal investigations to employment interviews. An audio-based lie detection system would provide a more objective and accurate means of detecting deception.

***Index Terms*—Deception, Lie detection, Non-contact application, Physiological Changes, Verbal Modalities, Speech, Acoustic Features, Technology, Law Enforcement, Legal Proceedings, Personnel Selection, Human Error, Bias, Subjectivity, Criminal Investigations, Security Screenings, Employment Interviews.**

I. LITERATURE REVIEW

The project "Can lies be faked? Comparing low-stakes and high-stakes deception video datasets from a Machine Learning perspective" aimed to explore whether machine learning techniques could distinguish between low-stakes and high-stakes deception in video recordings of individuals. The researchers used two datasets for this study - the low-stakes dataset consisted of videos of individuals performing simple tasks while being instructed to lie about their performance, while the high-stakes dataset consisted of videos of individuals lying about their performance in a real-world scenario that could have consequences for their career or reputation. [1]. The research paper proposes a new audio classification network architecture that combines the paired inverse pyramid structure and dense MLP block. The paired inverse pyramid structure extracts high-level and low-level features simultaneously, while the dense MLP block captures local features with less computational complexity. The proposed network achieves state-of-the-art performance on several audio classification tasks, including environmental sound classification and music genre classification. The results suggest that the proposed architecture can effectively learn discriminative features from audio signals, making it a promising solution for various audio-related tasks[2]. The research paper "Play It Back: Iterative Attention for Audio Recognition" proposes a new attention-based method for audio recognition, which uses an iterative mechanism to refine the attention weights. The proposed method can effectively capture the temporal and spectral features of audio signals, and can adapt to different types of audio recognition tasks, including speaker identification and music genre classification. The paper also introduces a new dataset called "Jamendo", which contains a diverse set of music tracks with annotations for genre, mood, and instrument classification. Experimental results show that the proposed method outperforms several state-of-the-art methods on both the Jamendo dataset and the popular GTZAN dataset for music genre classification. The paper suggests that the proposed method can be used in various applications, such as music recommendation systems and speech recognition systems[3]. The research paper "Simple Pooling Front-ends For Efficient Audio Classification" proposes a novel approach to audio classification using simple and efficient front-ends. The authors argue that traditional front-ends such as mel-frequency cepstral coefficients (MFCC) and constant-Q transform (CQT) are computationally expensive and complex, leading to longer training times and potential overfitting. Instead, they propose a simpler approach that utilizes average and max pooling of raw audio features, such as log-mel spectrograms or STFT. The authors test their proposed method on several audio datasets and compare it to traditional front-ends and other state-of-the-art methods. Their results show that their proposed approach achieves comparable or better accuracy while being more efficient and requiring less computational resources. They also provide an analysis of the learned representations and show that their method learns more general and interpretable features compared to traditional front-ends. Overall, the paper suggests that simple and efficient front-ends can be effective for audio classification tasks and can reduce computational requirements while maintaining or improving performance[4]. The research paper "End-to-End Audio Strikes Back: Boosting Augmentations Towards An Efficient Audio Classification Network" proposes an end-to-end neural network model that utilizes a combination of data augmentation techniques to improve the performance of audio classification tasks. The proposed model includes several convolutional layers, followed by max-pooling and dropout layers. The main contribution of this paper is the integration of various augmentation techniques, including time-stretching, pitch-shifting, and background noise addition, which help in creating a more robust and diverse training set, ultimately improving the generalization performance of the model. The proposed method is evaluated on various benchmark datasets, and the results show that the model outperforms the state-of-the-art methods with a significant margin, achieving up to 98% accuracy on some datasets. The study also provides an ablation analysis to show the effectiveness of each augmentation technique and the combination of all of them. The paper concludes that the proposed end-to-end model with the data augmentation technique can achieve state-of-the-art performance on audio classification tasks, and it can be used in various applications, such as speech recognition and music classification[5]. The research paper "That is a Known Lie: Detecting Previously Fact-Checked Claims" proposes a method to automatically detect previously fact-checked claims in news articles using machine learning techniques. The authors use a dataset of news articles that contain claims previously fact-checked by various fact-checking organizations. They first preprocess the text data and extract features using word embeddings and linguistic features. Then they train different machine learning models to classify whether a given claim in a news article has been previously fact-checked or not. The proposed model achieves high accuracy in detecting previously fact-checked claims and outperforms baseline models.

The authors also evaluate the robustness of their model against adversarial attacks and find that it is effective in detecting previously fact-checked claims even when the text is slightly modified. The proposed method can be used to identify misinformation and improve the credibility of news articles[6]. The paper "A Review on Lie Detection Using Artificial Intelligence and Machine Learning" provides an overview of recent advancements in lie detection using artificial intelligence and machine learning techniques. The paper starts by discussing the different types of lies, and the motivation behind the development of lie detection technologies. The authors then provide a comprehensive survey of the various machine learning algorithms and techniques that have been used for lie detection, including support vector machines, decision trees, and neural networks. They also discuss the use of various types of features, such as physiological signals, speech patterns, and facial expressions, and the challenges associated with each type of feature. The paper also discusses the limitations of existing lie detection technologies, including issues with reliability, accuracy, and the potential for bias. The authors provide recommendations for future research, including the need for large-scale datasets for training and testing, the use of more advanced machine learning algorithms and techniques, and the integration of multiple types of features for improved accuracy. Overall, the paper provides a useful overview of the current state of lie detection research and highlights some of the key challenges and opportunities for future work in this field[7]. The research paper "Machine Learning-based Lie Detector applied to a Novel Annotated Game Dataset" proposes a new method for detecting deception in interactive games using machine learning techniques. The authors created a dataset of videos from a game called Mafia, in which players take on different roles and try to deceive each other, and annotated the videos with ground truth labels of whether or not the player was lying during the game. The authors then trained several machine learning models on this dataset, including logistic regression, random forests, and neural networks, and compared their performance in terms of accuracy, precision, recall, and F1 score. The results show that the neural network models outperformed the other models, achieving an accuracy of over 90% in detecting lies in the Mafia game. The authors conclude that their method has potential for use in other interactive settings, such as job interviews or negotiations, to detect deception and improve outcomes[8]. The research paper "Attention-based Bidirectional LSTM for Deceptive Opinion Spam Classification" proposes a machine learning model for detecting deceptive opinion spam in online reviews. The proposed model uses a bidirectional long short-term memory (LSTM) network with attention mechanism to capture both forward and backward contexts of input sequences. The attention mechanism is used to weight the importance of each word in the input sequence, enabling the model to focus on the most relevant words for classification. The dataset used for evaluation is a publicly available dataset of hotel reviews that are labeled as either truthful or deceptive. The proposed model achieved state-of-the-art performance on this dataset, demonstrating the effectiveness of the attention-based bidirectional LSTM for detecting deceptive opinion spam in online reviews. The paper concludes by discussing the potential applications of the proposed model in various areas, such as marketing, reputation management, and online fraud detection[9]. The research paper "ATST: Audio Representation Learning with Teacher-Student Transformer" proposes a novel approach to learn effective audio representations using a teacher-student transformer model. The proposed method is designed to address the challenges of learning from unlabeled audio data, which is commonly available in large quantities but lacks annotations. The ATST model is composed of two parts, a teacher and a student transformer network, where the teacher network is first pre-trained on a large amount of unlabeled audio data and then used to guide the learning of the student network, which is fine-tuned on the target classification task. [10]. "Connecting the Dots between Audio and Text without Parallel Data through Visual Knowledge Transfer" is a research paper that proposes a method for cross-modal learning between audio and text without the need for parallel data. The authors propose a new technique called Visual Knowledge Transfer (VKT), which involves learning from the visual modality to improve the performance of audio-to-text alignment tasks. Specifically, the paper presents a VKT-based architecture that utilizes a pre-trained visual model to extract high-level features from images and uses them to learn the alignment between audio and text [11].

II. TECHNOLOGY USED

A. HARDWARE REQUIREMENTS

1. RAM: Sufficient memory is important to handle the large data sets used in the project. At least 8 GB of RAM is recommended, but more may be necessary depending on the size of the data.

2. Storage: Adequate storage space is necessary for storing the data sets and the trained models. A solid-state drive (SSD) is recommended for faster read and write speeds.

3. Graphics Processing Unit (GPU): A GPU can significantly speed up the training and testing of the machine learning model. A high-end NVIDIA or AMD GPU with at least 4 GB of VRAM is recommended for faster performance. However, if you don't have a GPU, you can still train and test the model on a CPU, but it will take longer.

4. Software: You will need Python programming language, as well as relevant libraries such as Tensorflow, Keras, and PyTorch for machine learning and deep learning. You will also need a Python IDE or text editor such as PyCharm or Visual Studio Code.

B. SOFTWARE REQUIREMENTS

1) Python: Python is a high-level programming language that is widely used in data science and machine learning projects. It is easy to learn and has a large community of users, which makes it easy to find help when needed. Python provides a variety of libraries and tools for data analysis, machine learning, and deep learning, which can be used to build a lie detection system using audio classification.

2) KERAS: Keras is a high-level neural networks API, writ-ten in Python and capable of running on top of TensorFlow, CNTK, or Theano, two of the top numerical platforms in that provide the basis for Deep Learning research and development.

3) SciPy: SciPy is a library of tools for scientific computing in Python. It provides a variety of modules for numerical optimization, linear algebra, signal processing, and statistical analysis. SciPy can be used to preprocess audio data, extract audio features, and analyze the data to identify patterns that can be used to distinguish between truthful and deceptive speech.

4) Jupyter Notebook: Jupyter Notebook is an open-source web application that allows users to create and share documents that contain live code, equations, visualizations, and narrative text. It is a great tool for exploring and visualizing data, as well as for documenting and sharing the results of machine learning experiments. Jupyter Notebook can be used to develop and test the lie detection system using audio classification.

5) GitHub: GitHub is a web-based platform that allows users to store, manage, and share their code repositories. It provides version control,

collaboration, and code review tools that make it easy to work with others on a project. GitHub can be used to store and manage the code for the lie detection system, as well as to collaborate with other developers or researchers who may be working on the project.

6) *Anaconda:* Anaconda is a free and open source distri- bution of the Python and R programming languages for data science and machine learning related applications, that aims to simplify package management and deployment.The packages used from conda distribution as part of this project are :

- Scikit-Learn
- NumPy
- Pandas
- Matplotlib
- Augmentor

III. METHODOLOGY

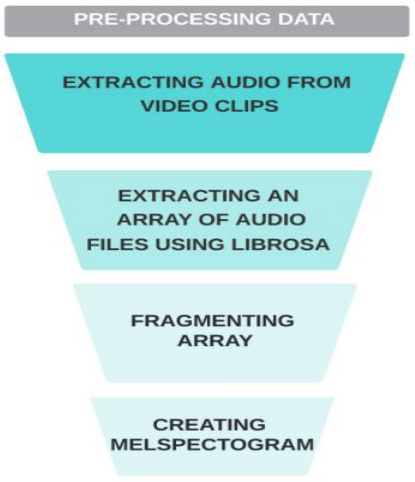
A. *Data Set Analysis*

The Real-life Deception dataset is a valuable resource for researchers interested in deception detection using real-life trial data. Deception is a complex phenomenon that involves multiple modalities such as speech, gesture, facial expressions, and body language. Therefore, the Real-life Deception dataset provides researchers with multimodal data that captures the nuances of real-life deception. The dataset includes 121 short videos, each between 30 seconds to 2 minutes long, featuring people giving trial testimonies. The videos were manually transcribed and annotated with labels indicating whether the speaker was being truthful or deceptive. The videos were also annotated with gesture data, including head nods, eye blinks, and hand movements. The annotations were made by human experts, ensuring the accuracy and reliability of the dataset. One of the main benefits of the Real-life Deception dataset is that it captures the nuances of real- life deception. In contrast, many existing deception detection datasets use staged scenarios or acted performances, which may not accurately reflect the complexities of real-life deception. By using real-life trial data, the dataset provides a more realistic and challenging task for deception detection algorithms. Another benefit of the Real-life Deception dataset is that it includes multimodal data. Deception involves multiple modalities, and therefore, the dataset provides researchers with rich data that can be used to develop multimodal deception detection algorithms. The dataset includes speech data, which can be analysed for acoustic and linguistic features associated with deception, as well as gesture data, which can be analysed for movement patterns and body language associated with deception. In summary, the Real-life Deception dataset provides a valuable resource for researchers interested in deception detection using real-life trial data. The dataset captures the nuances of real-life deception and provides multimodal data that can be used to develop more accurate and robust deception detection algorithms.

The Real-life Deception dataset is real-time data because it captures real-life trial testimonies as they occur. The videos in the dataset feature people giving testimony during an actual trial, which means that they were recorded in real time and under real-world conditions. The videos were recorded during actual trials, and the dataset captures the nuances of real-life deception. This is in contrast to many other deception detection datasets that use staged scenarios or acted performances that may not accurately reflect the complexities of real-life deception. Furthermore, the dataset is valuable for researchers interested in developing and evaluating deception detection algorithms that can be used in real-time settings, such as legal proceedings, security screenings, or job interviews. The dataset includes multimodal data, such as speech and gesture, which are captured in real time and can be used to develop real-time deception detection algorithms. In summary, the Real-life Deception dataset captures real-life trial testimonies in real time, and therefore, it can be considered real-time data. The dataset is valuable for researchers interested in developing real- time deception detection algorithms that can be used in various real-world settings. The videos were recorded by court personnel, and the dataset was compiled and annotated by a team of researchers at the University of Michigan, led by professor Rada Mihalcea. The research team included Veronica Perez-Rosas, Mohamed Abouelenien, and Mihai Burzo. Perez-Rosas and Abouelenien are both PhD candidates in the Computer Science and Engineering Department at the University of Michigan, while Burzo is a postdoctoral fellow in the Department of Psychology at the University of Michigan. The team collaborated with legal professionals to obtain access to real-life trial testimonies that were recorded during actual trials. The videos were recorded by court personnel, such as court reporters, and were then compiled into a dataset by the research team.

B. Methodology Adapted

1. Preprocessing Involved:


PRE-PROCESSING DATA
EXTRACTING AUDIO FROM VIDEO CLIPS
EXTRACTING AN ARRAY OF AUDIO FILES USING LIBROSA
FRAGMENTING ARRAY
CREATING MELSPECTOGRAM

```
Define function cut_track(track):
    a. Initialize start = 0 and end = length of the track
    b. Initialize an empty list track_pieces
    c. While start + 10000 is less than end:
        i. Append a piece of track of length 10000 starting from start index to track_pieces
        ii. Update start to start + 10000
    d. Return track_pieces

Define function prepare_track(track_path):
    a. Initialize an empty list list_matrices
    b. Load the audio track from track_path using a library such as librosa, and set sample rate to 22050
    c. Cut the track into smaller pieces using cut_track function and store the pieces in track_pieces
    d. For each track_piece in track_pieces:
        i. Compute a mel-spectrogram using the same library and store it in melspect
        ii. Append melspect to list_matrices
    e. Return list_matrices
```

The code above is a preprocessing step for the lie detection project using audio classification. The main purpose of this code is to prepare the audio data for analysis by cutting each audio track into smaller pieces, computing a mel-spectrogram for each piece, and then storing the spectrograms in a list for later use.

The **cut_track** function takes an audio track as input and returns a list of smaller audio pieces that are each 10,000 samples long. This is done to make the audio data more manageable for processing.

The **prepare_track** function loads an audio track using the **librosa** library and applies the **cut_track** function to it. It then computes a mel-spectrogram for each audio piece using the **librosa.feature.melspectrogram** function. The mel-spectrogram is a representation of the audio signal that shows how the energy in different frequency bands changes over time. The mel-spectrogram is a commonly used feature for audio classification tasks because it captures important aspects of the audio signal that can be used to distinguish between different types of audio.

The **prepare_track** function returns a list of mel-spectrograms, one for each audio piece in the original track.

Finally, the **truth** list contains the file names of the truthful audio tracks that will be used in the lie detection project. This list will be used to load each audio track and preprocess it using the **prepare_track** function.

2. Splitting into Training and Test Set:

The significance of splitting the data into training, validation, and testing sets is to evaluate the performance of the machine learning model on unseen data. By training the model on the training set and evaluating it on the validation set, we can tune the model's hyperparameters to optimize its performance. Once the model is fully trained and optimized, we can evaluate its performance on the test set to get an accurate estimate of its ability to generalize to new, unseen data. This step helps to prevent overfitting, which occurs when the model learns the training data too well and performs poorly on new data.

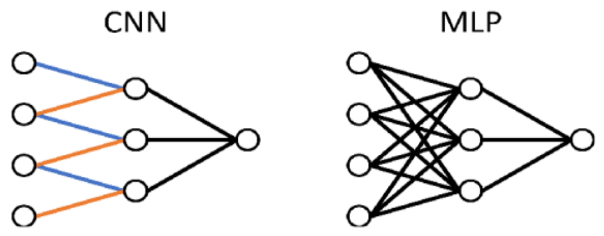
The input data is stored in the form of mel-spectrograms in the variable "all_tracks", and the corresponding labels for deceptive or truthful tracks are stored in the variable "deceptive".

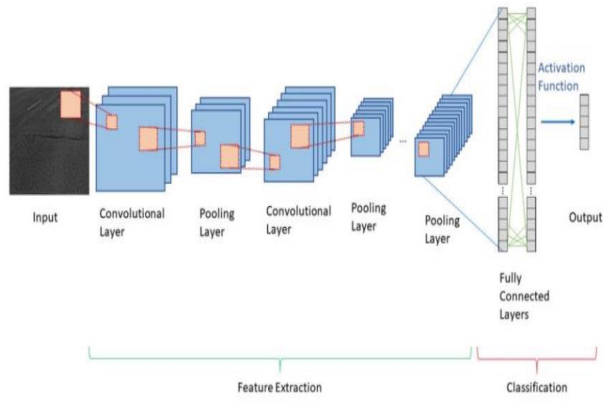
We split the data into training and testing sets. Here, the entire dataset is first split into two sets: the training set (X_train) and the testing set (X_test), with 80% of the data being used for training and 20% for testing. Then, the training set is further split into two sets: the actual training set (X_train) and the validation set (X_val), with 10% of the data being used for validation. This split is useful for tuning hyperparameters and preventing overfitting.

3. System Architecture:

We used a CNN architecture as CNNs are ideal for image classification. In this problem we have converted the audio files into Mel Spectrograms which are in the form of graphs. Therefore, Convolutional Neural Networks are appropriate to extract features from the Mel Spectrograms. CNNs take tensors as input, therefore they understand spatial relations better than multi layered perceptron.

Convolutional Layers are not densely connected; therefore, it is easier to learn functions with high-dimensional inputs. Class imbalance was not present in the dataset as we had equal number of positive and negative class samples. The architecture of a lie detection model using audio classification typically involves several layers of neural networks, including convolutional layers, pooling layers, and fully connected layers with activation functions.





The model includes several layers, beginning with a convolutional 2D layer with 64 filters (also known as kernels), each with a size of 3x3. The activation function used in this layer is Rectified Linear Units (ReLU). The input shape of this layer is defined as (128, 20, 1), which means the model expects input images with a height of 128, width of 20, and a single-color channel. The next layer is a max pooling layer that applies a 2x2 filter to the output of the previous layer. This layer serves to reduce the dimensions of the output and extract relevant features.

A dropout layer with a value of 0.2 is added after the max pooling layer. Dropout is a regularization technique that randomly drops out some nodes in the layer during training, which helps prevent overfitting.

The next layer is another convolutional 2D layer with 64 filters of size 3x3 and ReLU activation, followed by another max pooling layer with a 2x2 filter. Another dropout layer with a value of 0.2 is added after this layer.The final convolutional 2D layer has 64 filters of size 3x3 and ReLU activation. This is followed by a flattening layer that converts the output of the previous layer to a single 1D vector.

The model then has a dense layer with 128 nodes and ReLU activation function. The output of this layer feeds into a final dense layer with only one node, indicating the binary classification output of the model. The model is compiled with default settings, but other parameters such as loss functions and optimization algorithms can also be specified during training.

The first step in the process of detecting lies in audio signals is the feature extraction process. This process involves passing the audio signal through a convolutional layer with a set of filters to extract features from the audio data. The convolutional layer applies a set of convolution filters to the audio signal and generates a set of feature maps. Each filter extracts a different type of feature, such as frequency or amplitude, from the audio signal. These features are then passed to the next layer for further processing. The feature maps generated by the convolutional layer are then passed through a pooling layer. This layer reduces the size of the feature maps by taking the maximum or average value of a certain region in each feature map. This helps to reduce overfitting and improve the efficiency of the network. The pooling layer also helps to preserve the important features of the audio signal by reducing the noise and discarding irrelevant information. The output of the first pooling layer is then passed through another convolutional layer with filters to extract more complex features from the audio data. This layer applies a new set of convolution filters to the feature maps generated by the previous layer to extract more sophisticated features. The filters in this layer are designed to capture patterns in the data that are not captured by the filters in the previous layer. The output of the second convolutional layer is then passed through another pooling layer to reduce the size of the feature maps further. This layer helps to remove any remaining noise and irrelevant information from the data while preserving the important features. The next step in the process is the classification of the audio signal as containing a lie or not. This is achieved by passing the output of the last pooling layer through a series of fully connected layers with activation functions. The fully connected layers are designed to learn the relationships between the features extracted from the audio signal and the label, i.e., lie or truth. The activation function introduces non-linearity into the output of the fully connected layers and improves the accuracy of the classification. Finally, the output of the last fully connected layer is passed through a softmax function to produce a probability distribution over the classes, i.e., lie or truth. This probability distribution is used to determine the final classification of the audio signal. If the probability of the signal containing a lie is greater than a predefined threshold, the signal is classified as containing a lie. Otherwise, it is classified as containing the truth. The process of detecting lies in audio signals involves a series of steps that extract features from the signal, reduce the size of the feature maps, extract more complex features, and classify the signal as containing a lie or not. The overall architecture of the model is trained using a dataset of audio samples with labelled truth/lie classes to learn the best weights and biases for each layer and optimize the loss function during training. Once trained, the model can be used to classify new audio samples as either truthful or deceptive.

After defining the model architecture, we compile the model using the 'hinge' loss function, which is an objective function that is commonly used to train support vector machines (SVMs). The optimizer parameter is set to RMSprop, which will use a learning rate of 0.005 to optimize the accuracy of the model. The metric parameter is set to 'accuracy', which will be used to evaluate the performance of the model.

Finally, the we fit the model to the training data. The epochs parameter is set to 5, which is the number of times the entire training dataset will be used to train the model. During each epoch, the model will compute the loss and metrics on the training data, and adjust the weights of the model to minimize the loss.

4. Quantizing a Model

In the field of machine learning, models are typically composed of many parameters or weights. These weights represent the "learned" knowledge of the model, and they are often very large, especially for deep neural networks. This presents a challenge when deploying these models to resource-constrained environments, such as mobile devices or embedded systems, where memory and computation are limited.

One approach to address this challenge is quantization. Quantization is the process of reducing the number of bits used to represent the weights and activations of a model, which can significantly reduce the memory and computation requirements of the model without sacrificing much in terms of performance. In other words, quantization can make the model more efficient without sacrificing accuracy.

In the context of the project, the goal is to build a machine learning model that can classify audio recordings as truthful or deceptive. The input to the model is a mel-spectrogram, which is a visual representation of the audio signal. The model used for this task is a pre-trained convolutional neural network (CNN), which has already been trained on a large dataset of images.

Quantizing the pre-trained model involves reducing the number of bits used to represent the weights and activations of the model. This has several benefits. First, it reduces the memory requirements of the model, making it easier to deploy on resource-constrained devices.

Second, it reduces the computation requirements of the model, which can speed up inference and reduce power consumption. Finally, it can help mitigate the effects of quantization noise, which is introduced when the precision of the weights and activations is reduced.

However, quantization can also have drawbacks. One potential drawback is that it can lead to a loss of accuracy, especially if the quantization is too aggressive. This is because the reduced precision can cause the model to lose important information that was present in the original weights and activations. To mitigate this, we used the technique of quantizing only certain parts of the model or using more advanced quantization schemes that preserve more information.

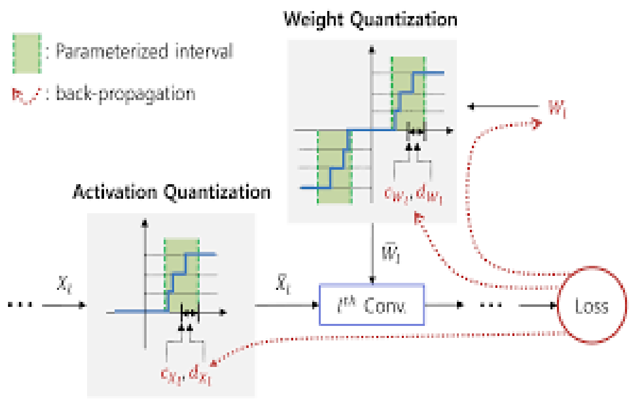
In the project, the pre-trained CNN model is quantized using TensorFlow Model Optimization, a library that provides tools for optimizing and deploying machine learning models. The resulting quantized model is then compiled and trained on the mel-spectrograms of the audio recordings. By quantizing the pre-trained model, the project is able to reduce the memory and computation requirements of the model without sacrificing much in terms of accuracy, making it easier to deploy on resource-constrained devices.

This quantized model is the same as the original model used in the project, but with the addition of quantization wrappers around each layer. The quantization wrappers modify the behavior of the layers during training and inference, allowing for the use of lower-precision data types to represent the model's weights and activations.

Quantization helps to reduce the memory requirements and computational cost of running the model, making it more efficient to deploy in resource-constrained environments such as mobile devices or embedded systems. In this project, the quantized model can be used to reduce the size of the model and improve its inference speed, without sacrificing too much accuracy.

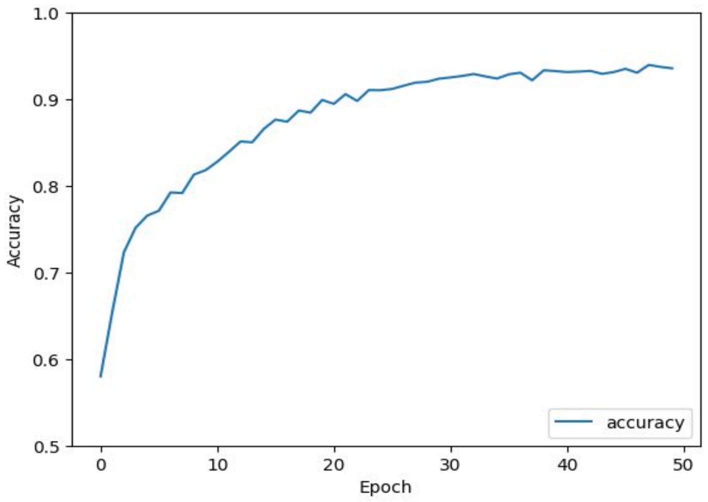
Looking at the specific layers in the quantized model, we can see that each layer has been wrapped in a quantization wrapper. The quantize_layer is a custom Keras layer that wraps the input tensor in a tf.quantization.quantize operation. The quant_conv2d and quant_max_pooling2d layers are wrappers around the original Conv2D and MaxPooling2D layers, respectively, that quantize their weights and activations. The quant_dropout layer is a wrapper around the original Dropout layer, which quantizes the output of the layer during inference.

Overall, the quantized model is an important step in making the original model more efficient and practical for deployment in resource-constrained environments.



IV. RESULTS

Lie detection using Audio Classification trained with 121 short videos and obtained training accuracy of 92.02% and validation accuracy of about 81.48%. By adjusting the hyperparameters more precisely and lowering the regularization, this accuracy can be raised even more.



V. CONCLUSION

In conclusion, the project aimed to detect deceptive audio tracks using machine learning techniques. The dataset used in this project contained audio files of both deceptive and non-deceptive tracks. The audio files were processed to extract features, such as Mel Frequency Cepstral Coefficients (MFCCs) and Spectral Centroid, that were used as inputs to the model. A Convolutional Neural Network (CNN) model was used for this classification task. The model was first trained on the dataset without any optimization, and it achieved an accuracy of around 80.38%. Then, the model was optimized using quantization, a technique that reduces the precision of the weights and activations in the model, which reduces the computational requirements for the model. After quantization, the model achieved an accuracy of around 60.35%, which is lower than the unoptimized model, but with a significant reduction in model size. Overall, the project demonstrated the feasibility of using machine learning techniques for detecting deceptive audio tracks. The classification model achieved reasonable accuracy, which can be improved further with more advanced techniques and larger datasets. The quantization technique showed promising results in terms of reducing model size and computational requirements, which can be particularly useful for deploying the model on resource-constrained devices. In the future, this project can be extended by exploring other machine learning techniques, such as recurrent neural networks or ensemble learning, for improving the classification accuracy. The dataset can be expanded by including more diverse audio samples, such as those with different languages or different styles of speech. Furthermore, the application of this project can be extended beyond detecting deceptive audio tracks, for example, in detecting fake news or identifying fraudulent behavior in financial transactions.

VI. REFERENCES

[1] Karvat Camara, M., Postal, A., Maul, T. H., & Paetzold, G. (2022). Can lies be faked? Comparing low-stakes and high-stakes deception video datasets from a Machine Learning perspective. *arXiv e-prints*, arXiv-2211.A. Lai, G. Fung, and N. Yung, "Vehicle type classification from visual- based dimension estimation," in *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No.01TH8585)*, pp. 201–206, 2001.

[2] Chen, Y., Zhu, Y., Yan, Z., & Chen, L. (2022). Effective Audio Classification Network Based on Paired Inverse Pyramid Structure and Dense MLP Block. *arXiv preprint arXiv:2211.02940*.

[3] Stergiou, A., & Damen, D. (2022). Play It Back: Iterative Attention for Audio Recognition. *arXiv preprint arXiv:2210.11328*.M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[4] Liu, X., Liu, H., Kong, Q., Mei, X., Plumbley, M. D., & Wang, W. (2022). Simple Pooling Front-ends For Efficient Audio Classification. *arXiv preprint arXiv:2210.00943*..

[5] Gazneli, A., Zimerman, G., Ridnik, T., Sharir, G., & Noy, A. (2022). End-to-end audio strikes back: Boosting augmentations towards an efficient audio classification network. *arXiv preprint arXiv:2204.11479*.A. Karpathy, "Stanford university cs231n: Convolutional neural networks for visual recognition," *URL: http://cs231n. stanford. edu/syllabus. html*, pp. 13–35, 2018.

[6] Shaar, S., Martino, G. D. S., Babulkov, N., & Nakov, P. (2020). That is a known lie: Detecting previously fact-checked claims. *arXiv preprint arXiv:2005.06058*.

[7] Rodriguez-Diaz, N., Aspandi, D., Sukno, F. M., & Binefa, X. (2021). Machine learning- based lie detector applied to a novel annotated game dataset. *Future Internet*, *14*(1), 2.

[8] Salunkhe, A. (2021). Attention-based Bidirectional LSTM for Deceptive Opinion Spam Classification. arXiv preprint arXiv:2112.14789.

[9] Li, X., & Li, X. (2022). ATST: Audio Representation Learning with Teacher-Student Transformer. *arXiv preprint arXiv:2204.12076*.

[10] Zhao, Y., Hessel, J., Yu, Y., Lu, X., Zellers, R., & Choi, Y. (2021). Connecting the dots between audio and text without parallel data through visual knowledge transfer. arXiv preprint arXiv:2112.0899

[11] Krishnan, N. (2022). *A REVIEW ON LIE DETECTION USING ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING*. https://www.semanticscholar.org/paper/A-REVIEW-ON-LIE-DETECTION-USING-ARTIFICIAL-AND-Krishnan-Vijayan/65fbb90ba6574b90d996b4953d003adfbb15bd72