Individual Final Report

Introduction

The objective of our final project was to develop a neural network model capable of performing multi-label classification of bird species from environmental audio recordings. This challenge was inspired by a Kaggle competition that required identifying bird calls within complex soundscapes where each audio clip could contain calls from multiple bird species. This posed a multi-label classification problem, demanding careful attention to overlapping class signals and imbalanced label distributions.

Our team collaborated extensively across all major phases of the project, including dataset exploration, class distribution analysis, model training and evaluation, result visualization, and final report compilation. We chose to use a **CNN-based architecture, EfficientNet-B0**, trained on mel spectrograms generated from .ogg audio files. Model performance was evaluated using metrics such as **AUC** to ensure robustness across multiple labels.

To improve usability and demonstrate the model's capabilities, we developed an interactive **Streamlit web application**. This app allows users to upload audio files and receive predictions in an accessible, visual format, making our model approachable even for users without technical backgrounds.

We also ensured our codebase followed **good software engineering practices** by organizing the project into well-defined directories for data processing, model training, evaluation, utilities, and application deployment. This modular structure helped streamline collaboration and made the project easy to navigate and maintain.

Additionally, we designed a custom validation framework by splitting the provided training set into training and validation subsets while preserving label distributions. This allowed us to simulate real-world test scenarios and evaluate model performance without relying on Kaggle's hidden test set.

This report outlines my individual contributions, key evaluation results, and the reasoning behind our technical and design decisions throughout the project.

Background

I began by analyzing the unstructured and scattered original code from Kaggle, gaining a solid understanding of the dataset and modeling pipeline, including how audio was converted into mel spectrograms. I then reorganized the project into a cleaner directory structure, separating key components for better readability and collaboration. Additionally, I created the final presentation slides with a focus on visuals for broader audience understanding and wrote the GitHub README to clearly document how to run the project.

Description of Individual Work

My primary responsibility in this project was to streamline the existing modeling pipeline and improve its structure, clarity, and usability. The initial codebase we sourced from Kaggle was highly unstructured, with scripts and functions scattered across files and lacking modularity. I began by carefully reviewing this code to understand the complete workflow, from data loading and preprocessing to model training and evaluation.

A key part of this process involved understanding how raw .ogg audio files were being converted into mel spectrograms an essential step for transforming time-series data into a format suitable for convolutional neural networks. I documented this process internally to ensure the rest of the team could follow and contribute effectively.

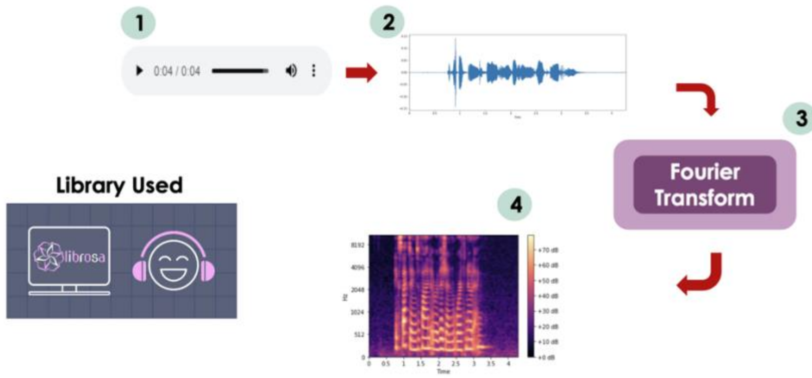
To support collaboration and future maintainability, I reorganized the project into a clean, modular directory structure. I created dedicated folders for data processing, spectrogram generation, model definition and training, utility functions, evaluation scripts, and the Streamlit deployment. This restructuring enabled better separation of concerns and made the codebase more intuitive to navigate.

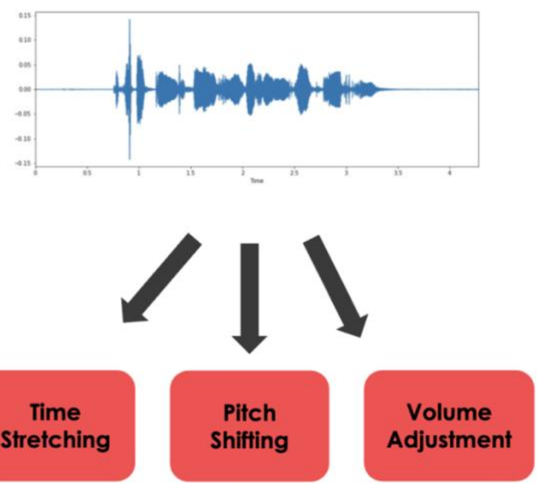
I also led the creation of our final presentation, designing slides that prioritized visuals over text to help a wider audience understand our methodology, findings, and model performance. This involved crafting diagrams and charts that highlighted key concepts without overwhelming viewers with technical jargon.

Finally, I authored the README.md file on our GitHub repository. The README includes a clear overview of the project, step-by-step instructions on setting up the environment, and guidelines on how to run the training pipeline and use the Streamlit app for making predictions. This ensured that external users and evaluators could easily replicate our work without needing to go through the codebase in detail.

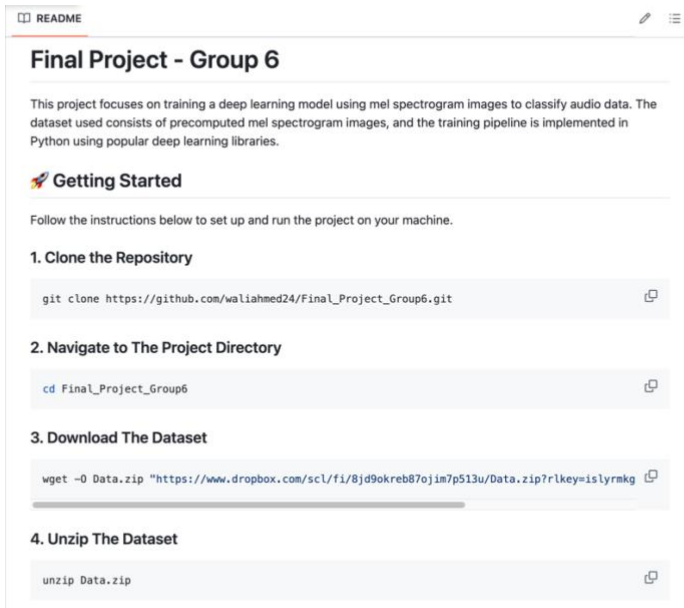
Results

After completing key tasks such as analyzing the unstructured Kaggle codebase, understanding the process of converting audio files into mel spectrograms, restructuring the project into organized directories, and creating supporting materials like the README and presentation the project was successfully executed end-to-end. The model was trained, evaluated, and deployed through a user-friendly Streamlit app. The following screenshots showcase the results and demonstrate the things worked on.






```
│
├── main.py                        # Entry point to run training/inference
├── config/
│   └── class_CFG.py               # Configuration class
│
├── data/
│   └── class_BirdCLEFDatasetFromNPY.py  # Custom dataset class
│
├── models/
│   ├── class_BirdCLEFModel.py     # Model architecture
│   └── utils_model_definition.py  # Extra model components (layers, etc.)
│
├── train/
│   ├── util_run_training.py       # Main training orchestration
│   ├── utils_training_loop.py     # Training and validation loops
│   └── util_collate_fn.py         # Custom collate function for Dataloader
│
├── utils/
│   ├── utils_preprocessing.py     # Audio preprocessing, transformations
│   └── util_set_seed.py           # Seed-setting utility
│
├── test/                          # For future inference scripts / test code
│   └── (optional) inference.py    # Final model evaluation / prediction
│
```



README

Final Project - Group 6

This project focuses on training a deep learning model using mel spectrogram images to classify audio data. The dataset used consists of precomputed mel spectrogram images, and the training pipeline is implemented in Python using popular deep learning libraries.

🚀 Getting Started

Follow the instructions below to set up and run the project on your machine.

1. Clone the Repository

```
git clone https://github.com/waliahmed24/Final_Project_Group6.git
```

2. Navigate to The Project Directory

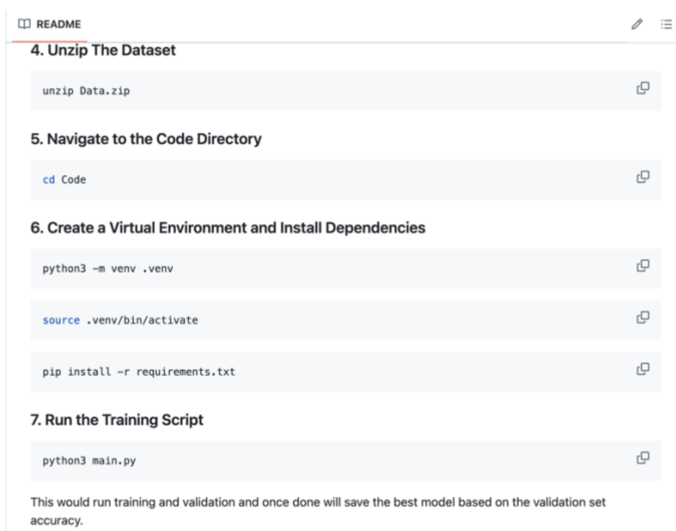
```
cd Final_Project_Group6
```

3. Download The Dataset

```
wget -O Data.zip "https://www.dropbox.com/scl/fi/8jd9okreb87ojim7p513u/Data.zip?rlkey=islyrmkg
```

4. Unzip The Dataset

```
unzip Data.zip
```



README

4. Unzip The Dataset

```
unzip Data.zip
```

5. Navigate to the Code Directory

```
cd Code
```

6. Create a Virtual Environment and Install Dependencies

```
python3 -m venv .venv
```

```
source .venv/bin/activate
```

```
pip install -r requirements.txt
```

7. Run the Training Script

```
python3 main.py
```

This would run training and validation and once done will save the best model based on the validation set accuracy.

Conclusion

This project was a valuable learning experience that strengthened my understanding of real-world machine learning workflows, especially in the context of audio data and multi-label classification. I gained hands-on experience working with spectrogram-based feature extraction, CNN model training, and performance evaluation. Additionally, I learned the importance of clean code organization and effective communication both through writing user-friendly documentation and designing visual presentations. Building and deploying the Streamlit app also gave me practical insight into making machine learning models accessible to non-technical users.

Code Percentage

Attribution % = $\frac{870-300}{870+180} \times 100 \approx 54\%$

References

(Doshi n.d.)

(İdrisoğlu n.d.)

(Velardo n.d.)

Bibliography

Doshi, Ketan. n.d. https://ketanhdoshi.github.io/Audio-Mel/.

—. n.d. https://ketanhdoshi.github.io/Audio-Mel/.

İdrisoğlu, Kadircan. n.d. https://www.kaggle.com/code/kadircandrisolu/efficientnet-b0-pytorch-inference-birdclef-25.

Velardo, Valerio. n.d. https://www.youtube.com/watch?v=4_SH2nfbQZ8&t=2186s.