Data Collection (Engine Team)

Task: In the environment, there are several other factors that affects the environments such as weather, vehicles, tools. etc. Project Echo needs more data on those factors, followed by some models to train those data.

Deliverables: A document and Code Implementation of newly found data and a most suitable model in training them.

The importance of this task for Project Echo: The echo simulator is a system created by Project Echo that replicates the Otway's National Forest environment that monitors wildlife and also the interactions with the environment. This task is important for Project Echo by enhancing the systems ability to be able to differentiate between different environmental sounds. By being able to classify these sounds accurately, it can help make the system more comprehensive and reliable simulation for the Otway's National Forest ecosystem and it can also help researchers analyzing important data.

Purpose: A systematic and organized directory structure is created for storing raw audio files, spectrograms, and categorized datasets.

Project Aim: Our aim for this project is to develop a audio classification system using Convolutional Neural Networks (CNNs). The project involves collecting and preprocessing audio data through techniques like bandpass filtering, normalization, silence removal, and conversion into spectrograms saved as RGB images. To enhance model performance, the dataset is augmented with methods such as Gaussian noise addition, time stretching, pitch shifting, and time shifting. A CNN model is designed with convolutional layers, batch normalization, and pooling to extract features effectively. The system is trained using the augmented spectrogram data and evaluated on unseen audio samples to classify them accurately, providing a scalable solution for environmental sound analysis.

Directory Structure Setup

- Automatically generates subdirectories such as:
    - Raw audio categories (Animal Sounds, Environment).
    - Processed spectrogram categories (train, validation, Mixed Sounds).

    - There includes separate subdirectories for training and validation for both animal and environmental sounds.
- **Functions:**
    - **setup_directories(base_dir):** Sets up all required directories.
    - **create_directories(base_dir, subdirs):** Creates subdirectories if they don't already exist.

Audio Preprocessing

Purpose: Enhance audio data quality and diversity for effective analysis and classification. Ensures the audio is clean, normalized, and relevant for training.

- Bandpass Filtering: frequencies between 500 Hz and 8000 Hz are only retained, Important for distinguishing animal and environmental sounds.
    - **Example:** apply_bandpass(y, sr, low=500, high=8000)
- Normalization: Audio signals are scaled between -1 and 1, ensuring uniform amplitude and preventing distortions during training.
    - **Example:** normalize(y)
- Silence Removal: silent segments are detected and removed using amplitude thresholds (top_db=20), and this ensures classifier focuses only on meaningful parts of the audio data.
    - **Example:** remove_silence(y, sr)

Spectrogram Generation

Purpose: audio signals are converted into visual representations (Mel spectrograms) suitable for processing by convolutional neural networks.

- Mel spectrograms are generated with a fixed size (128x128) and this is important for compatibility with the input layer of the CNN.
- The power spectrograms are converted to the decibel scale (logarithmic) and this is important for dynamic range visualization.
- Spectrograms are saved RGB images.
- **Example:**
    - **saved_spectrogram(y, sr, output_path, size=(128, 128)):** Converts audio to spectrogram and saves as an image.
    - **test_spectrograms(input_folder, output_folder):** Generates spectrograms from test audio files.

Audio Augmentation

Purpose: The dataset diversity is expanded and ensures robustness through introducing realistic variations to the audio data.

- Uses the audiomentations library to apply transformations:
    - AddGaussianNoise: Simulates real-world background noise.
    - TimeStretch: the tempo is altered while preserving the pitch.
    - PitchShift: the pitch is changed without affecting the tempo and this mimics the variability in sound sources.
    - Shift: Time-shifts the signal, introducing temporal variations.
- Each audio file is augmented multiple times (e.g., 5 versions), and this helps expand the dataset size to include more variations and results in improving the model's generalization.
- **Example:**
    - **processing_audio(input_dir, output_dir, augmentations):** Applies augmentations and generates spectrograms.

Mixed Sound Generation

Purpose: Prepare the model to classify audio in noisy, real-world conditions by combining different sound types.

- The animal sounds and environmental sounds are combined to create mixed audio clips.
- The mixed signals are normalized to prevent amplitude imbalances.
- The mixed audio is converted into spectrograms for further processing.
- **Example:**
    - **generate_mixed_sounds(animal_dir, env_dir, output_dir):** Creates mixed audio samples and saves spectrograms.
    - **mixed_spectrograms(mixed_dir, target_dirs):** Distributes mixed spectrograms into specific directories.

Dataset Splitting

Purpose: The data is split into training and validation subsets to be able to evaluate the model's generalization ability.

- Divides automatically the augmented data into training and validation sets using the standard 80%-20% split.
- **Example:**

- **split_data(input_dir, output_dirs, test_size=0.2):** Splits spectrogram data into training and validation directories.

Data Generators

Purpose: Prepare data for the CNN in real-time, reducing memory usage and enabling dynamic augmentations.

- **Training Data Generator:**
  - Real-time augmentations are utilized, for example: rotation, flipping, zoom, and shearing are used to simulate various perspectives of the spectrograms.
  - Rescales pixel values to the [0, 1] range for better gradient propagation.
- **Validation Data Generator:**
  - Only rescales pixel values without applying augmentations to ensure a fair evaluation of model performance.
- **Functions:**
  - **train_generator:** Data generator for training with augmentations.
  - **val_gen:** Data generator for validation.

Convolutional Neural Network (CNN) with Residual Connections

Purpose: A custom CNN architecture is implemented and further optimized for audio spectrogram classification.

- **Features:**
  - Residual Connections: Skip connections prevent gradient vanishing and enhance feature propagation.
  - Batch Normalization: Speeds up training and stabilizes the network by normalizing intermediate layer outputs.
  - Dropout: Helps reduce overfitting by randomly deactivating neurons during training.
  - Global Average Pooling (GAP): Reduces the number of parameters while preserving spatial features.
  - Output Layer: Softmax activation outputs probabilities for two classes (Animal Sounds, Environment).
- **Example:**
  - **cnn_model(input_shape=(128, 128, 3), l2_strength=0.001, dropout_rate=0.4):** Builds and returns the CNN model.

Early Stopping

Purpose: Avoid overfitting by stopping training when validation performance stops improving.

- **Example:**
    - **stop_early:** Implements early stopping callback.

Evaluation Metrics

Purpose: Assess the model's performance and visually display results.

- **Classification Report:** Precision, recall, and F1-score are displayed for both classes.
- **Confusion Matrix:** Correct and incorrect predictions across classes are visualized.
- **Example:**
    - **gen_classification_report(model, data_generator):** Generates the classification report and confusion matrix.

Model Saving and Loading

Purpose: Save and reload the trained model for future use.

- Saves the trained model in HDF5 format (audio_model.h5).
- Implements a function to load the model for evaluating new audio data.
- **Example:**
    - **model.save(model_save_path):** Saves the trained model.
    - **load_trained_model(model_path):** Loads the saved model.

Validating Model

Purpose: Evaluate the model's ability to classify unseen audio data accurately.

- **Spectrogram Generation for Test Data:**
    - The new audio files are converted into spectrograms, ensuring consistency with the training process beforehand.
    - **Example: test_spectrograms(input_folder, output_folder)**
- **Prediction Functionality:**
    - Maps predictions to class labels (Animal Sounds, Environment).
    - Predictions are compared against ground truth labels (from a CSV file).
    - **Example:**
        - **predict(model, spectrogram_folder, ground_truth_csv, class_names)**
    - Provides confidence scores for each prediction.

Results

```
Model loaded successfully.
Processing test audio files: 100%|██████████| 12/12 [00:20<00:00,  1.69s/it]
1/1 [==============================] - 1s 540ms/step
Correct: daintree-bat.png -> Animal Sounds (1.00 confidence)
1/1 [==============================] - 0s 56ms/step
Correct: Desert-wind.png -> Environment Sounds (0.99 confidence)
1/1 [==============================] - 0s 60ms/step
Correct: Giant Banjo Frogs.png -> Animal Sounds (0.99 confidence)
1/1 [==============================] - 0s 58ms/step
Incorrect: Glider_Squirrel_.png -> Predicted: Environment Sounds (0.88 confidence), Actual: Animal Sounds
1/1 [==============================] - 0s 61ms/step
Correct: Golden-Bowerbird-Prionodura-newtoniana.png -> Animal Sounds (0.64 confidence)
1/1 [==============================] - 0s 62ms/step
Correct: Light-rain.png -> Environment Sounds (1.00 confidence)
1/1 [==============================] - 0s 61ms/step
Correct: night-Frog.png -> Animal Sounds (1.00 confidence)
1/1 [==============================] - 0s 49ms/step
Correct: pademelon.png -> Animal Sounds (1.00 confidence)
1/1 [==============================] - 0s 63ms/step
Ground truth not found for shaking-branch-2.png. Skipping.
1/1 [==============================] - 0s 50ms/step
Correct: shaking-branch.png -> Environment Sounds (0.66 confidence)
1/1 [==============================] - 0s 74ms/step
Correct: storm-wind-in-trees.png -> Environment Sounds (0.82 confidence)
1/1 [==============================] - 0s 55ms/step
Correct: water-stream.png -> Environment Sounds (1.00 confidence)

Accuracy: 90.91%
```

As you can see in the code, the model achieved 90% accuracy on the tested datasets. The classification report and confusion matrix shows that the model has some areas to improve. It still struggles to distinguish between some of the environmental sound and the animal sound as you can see from there the model had identified the glider squirrel as environment although it should be animal sounds.