Emotion Recognition from Audio Using CNN-Based Spectrogram Analysis

Team: Hongyao Shao (U27125599), Penny Lin (U45036174)

Elevator Pitch: Train a convolutional neural network (CNN) to recognize emotions from audio clips using spectrograms.

Context:

- Spectrograms: a visual representation of the spectrum of frequencies of a signal as it varies with time, commonly used in audio analysis tasks: speech and emotion recognition
  Link: https://en.wikipedia.org/wiki/Spectrogram

Methods:

- Data Processing: convert audio clips into spectrograms
- Convolutional Neural network: use CNN to process spectrogram images and classify them into different emotion categories
- Improving generalization: use methods like data augmentation, dropout layers, and transfer learning to improve generalization if the training data is limited or low-variance.

Data Source: We plan to use publicly available datasets of audio clips such as:

- Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)
  - https://zenodo.org/record/1188976
- Toronto Emotional Speech Set (TESS)
  - https://tspace.library.utoronto.ca/handle/1807/24487
- Kaggle Speech Emotion Recognition

- https://www.kaggle.com/datasets/dmitrybabko/speech-emotion-recognition-en

Code Resources:

- TensorFlow CNN tutorials
- PyTorch CNN tutorials
- ChatGPT or Github Copilot for coding assistance as needed

What's New:

- We will create our pipeline to convert raw audio files into spectrogram images
    - Experimenting with different types of spectrograms to determine which types work the best for emotion recognition
    - Using libraries to write custom code to handle audio loading, preprocessing, and spectrogram generation
    - Split the dataset
- Developing or adapting CNN architecture suited for emotion recognition from spectrograms
    - Decide the number of layers, types of layers, and activation functions
    - Experiment with different architectures to find the best-performing model
- Experiment with different data augmentation techniques to improve model performance
    - Adding noise to audio clips to simulate real-world
    - Modify tuning parameters, such as batch size and number of epochs
    - Exploring different optimizers and how they affect training stability
- Transfer learning (optional)
    - If we have time, we will explore transfer learning by adapting a pre-trained model.
        - Modify the pre-trained model that is suited to spectrogram inputs
- Evaluate the different impacts on model performance
    - Compare the difference between different spectrogram types

    - Evaluating the performance of different data augmentation methods
    - Evaluating how model accuracy and generalization are affected by hyperparameter selections

Plan:

- Milestone 1 (4/1): Complete convert audio clips to spectrograms, and split the data into training, validation, and test sets.
    - Split the data into training (70%), validation (10%), and testing (20%).
- Milestone 2 (4/8): Build and train the CNN model. Experiment with different hyperparameters.
- Milestone 3 (4/15): Evaluate the model, and experiment with data augmentation.
- Milestone 4 (4/22): Complete final model training and experiments and test with real life audio.
- Final Deliverable (4/30): Summary paper and lightning talk.

Proposed demonstration or evaluation:

- We will demonstrate the effectiveness of our model by evaluating its accuracy on the test set of audio clips. We will use metrics like precision, recall, and F1-score to assess the quality of emotion classification.
- We will demonstrate the generalizability of the model and avoid overfitting by monitoring performance of accuracy and loss during training.
- We will record real-life audio clips with clearly identified emotions, test them using our model, and verify if the predicted emotions match the actual intended emotions.

Experiments:

- Data augmentation: We will vary data augmentation methods to observe their effects on accuracy and loss, and evaluate how each improves model generalization, such as
    - Noise injection: Adding varied types and levels of background noise.

    - Pitch shifting: Modifying pitch at different intervals.
    - Speed modifying: Slowing down or speeding up the audio clip.
- Regularization methods: We will experiment with regularization techniques including dropout, early stopping, and transfer learning and monitor changes in training and validation accuracy and loss to determine what is the best way to reduce overfitting.
- Learning rate and optimization techniques: We will experiment with different learning rates and optimizers to find optimal training strategies.
- Audio clip length: We will evaluate how the length of audio clips affects model performance. We might classify the audio clips by different length and test if shorter or longer clips have higher accuracy.