1. How to Run the Code
This code implements a multi-layer perceptron (MLP) for frame-level phoneme recognition. Below are the steps to run the model and evaluate its performance.
Steps to Run:
  1. Setup Environment:
    Install the necessary libraries:
    pip install torch numpy sklearn pandas tqdm wandb torchaudio
  2. Login to WandB:
    Login to WandB to track the experiments:
    wandb login --relogin
  3. Train the Model:
    Use train_loader and val_loader to start training:
    python main.py
  4. Check Results:
    After training, the results can be monitored on WandB.

2. Experiment Details
This code is designed to train a multi-layer perceptron (MLP) for frame-level phoneme recognition, taking Mel Frequency Cepstral Coefficients (MFCC) as input and predicting the corresponding phonemes for each frame. Cross-Entropy Loss is used for calculating the loss, and AdamW optimizer is used for updating the model weights. WandB is used to track the experiment progress and performance metrics.

Experiment Process:
Data Preprocessing:

The MFCC features are standardized, and a context window is applied to convert each frame into an input format for the network.
Model Training:

The model is trained for 20 epochs, and the loss and accuracy are monitored during training.
The model checkpoint with the highest validation accuracy is saved during training.
Final Testing:

The trained model is used to predict phonemes on the test set, and the predictions are saved in a CSV file for submission.
3. Model Architecture
The model is a Multi-Layer Perceptron (MLP) that takes the input frames and processes them through the following structure:

Input: (2 * context + 1) * 28 (context size and frame dimension)
Hidden Layers:
Layers with 2048, 1024, 512, 256, and 128 units
Each layer applies Batch Normalization, GELU activation function, and Dropout (0.15)
Output: 42 phoneme classes (based on the PHONEMES list)
4. Hyperparameters Used
The best-performing combination of hyperparameters is as follows:

Epochs: 90
Batch Size: 4096
Learning Rate: 1e-3 (using AdamW optimizer)
Context Size: 25
Scheduler: MultiStepLR (milestones: [60, 80], gamma: 0.1)
Dropout: 0.15
Weight Decay: 0.01

5. Data Loading Method
Dataset Structure:
Train Dataset: Uses MFCC and transcript files from the train-clean-100 directory
Validation Dataset: Uses the dev-clean directory
Test Dataset: Uses the test-clean directory
For each dataset, the MFCC features and corresponding phoneme labels are loaded as Numpy arrays. The input frames are standardized, and padding is applied to create a window of the specified context size.