Task 4

Zuriahn Yun

June 5, 2025

1 Task 4:

Predicting the speech sound ("phoneme") from an acoustic feature vector (describing the acoustic content in a small frame of audio)

2 Lead

This task was lead by Zuriahn Yun.

3 Methods

After some initial data exploration, I charted the frequency of each class. From there I applied for feature reductions, I used UMAP, t-SNE and PCA and visualized the results to see if would provide better insight into my dataset. The projections, however, did not yield to any direct results. From there I used baseline models trained on small, stratified subsets of my data. Using the stratify feature in with the train_test_split feature ensures that each subset maintains an equal class distribution, allowing the baselines to learn patterns for every class. This approach also made it easier to tune hyperparameters on a smaller scale before scaling up. Additionally, I applied Z-scored standardization to the features so that the models could form relationships more effectively. Finally, I evaluated performance using accuracy and log loss on the dev set to determine how well each model was performing and whether hyperparameter adjustments were beneficial.

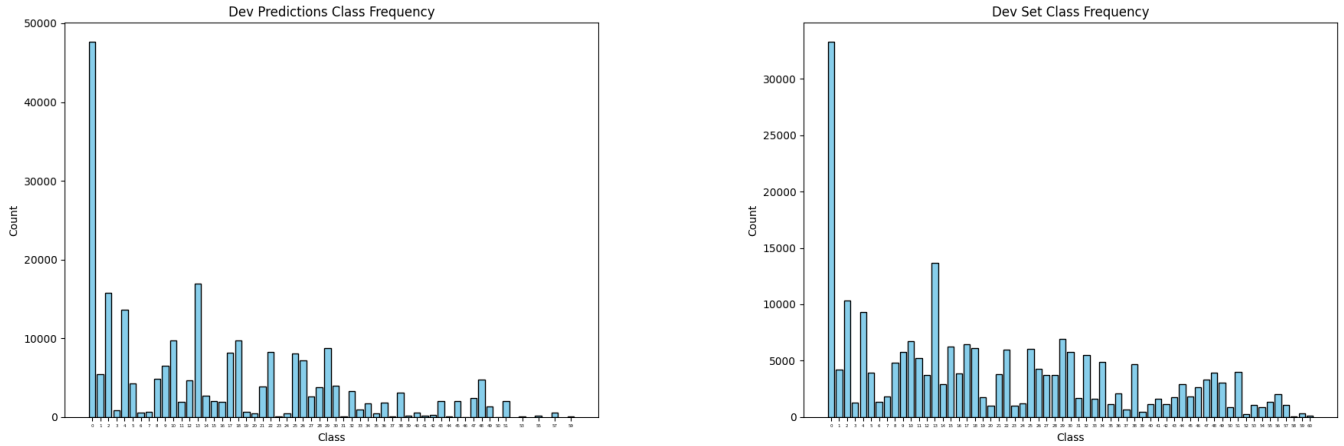
4 Model

The best performing model on the development set was a neural network trained on Z-score standardized data. The architecture consisted of a single hidden layer with 100 neurons. The activation function used was ReLU, with an L2 regularization term of 0.0001. The initial learning rate was set to 0.001, and a constant learning rate schedule was used. Training included early stopping to prevent overfitting. This model was selected through a grid search over multiple hyperparameters, including hidden layer sizes, activation functions, regularization strength and learning rate initialization. The neural network outperformed all previous models, both individually and in comparison, to ensembles. Below is the distribution of class predictions on the dev set for the final model.

5 Results

Figure 1: Development Set Accuracy and Log Loss for Top Performing Models

Model	Dev Accuracy	Dev Log Loss
Neural Network	0.4470	1.9170
NN + XGBoost	0.4402	1.9083
XGBoost	0.4191	2.0761
Logistic Regression	0.4057	2.1242
Random Forest	0.3756	2.3568



(a) Model Predicted Class Distribution

(b) True Class Distribution

Figure 2: Class distribution comparison between model predictions and actual labels.

While the ensemble of the neural network and XGBoost performed the best in terms of log loss the neural network held a lower log loss its accuracy was lower. The standalone neural network achieved the highest accuracy overall and was therefore selected as the final submission model. Throughout my experimentation, I explored multiple models and hyperparameter settings. I spent most of the time tuning neural networks and XGBoost. Several ensemble approaches were attempted but none outperformed the neural network alone or its combination with XGBoost. The class distribution reveals a significant class imbalance, particularly with the model over predicting class 0. This bias likely stems from the natural distribution of the dataset and contributes to the accuracy ceiling.

Models such as SVM were deemed impractical due to excessive training times. For example, even with a stratified subset of 100,000 points from the 2-million-point training set, SVM training took nearly 500 minutes. Due to time constraints and diminishing returns SVMs and other resource heavy models were not prioritized.

Ultimately, I chose the model that consistently achieved higher dev set accuracy and could be trained on the full training set.