MLPC Classification Experiments Report

Team BED

Ferit Cakmak **Ahmet Arıkan** **Kerem Gülpınar** **Mahmut Baha Cogal**

Contributions

Ferit Cakmak led Classification Experiments, performed statistical analyses, and implemented core code components. Kerem Gülpınar supported the coding process through testing and code reviews. Mahmut Baha Cogal revised the task, wrote the report, and managed the LaTeX document. Ahmet Arıkan prepared the presentation and contributed to report development.

1 Labeling Function

The dataset provides a multilabel structure for each audio file in the shape $(58, 220, 1)$, representing 58 classes across 220 temporal frames (each frame represents 120ms time period). Hence the individual audio length differs, so the temporal frame length differs too. To evaluate label quality, we selected three of the most diverse and three of the least diverse files based on the number of active label classes per file. These covered a mix of simple (e.g., `cough`) and complex (e.g., `speech`, `bell`, `airplane`) acoustic scenes.

1.1 (a) Class Correspondence and Label Consistency

We compared the free-text annotations with the label matrices. Although the temporal frames from the label matrix and annotations do not align perfectly, they are semantically compatible. For instance, segments labeled as "cough" in the text often show clear activation in the corresponding label channel. Audible inspection also confirmed that many of the labeled events are clearly present in the indicated regions. This provides confidence that label functions applied over the annotation set do broadly reflect the intended classes.

We observed that label matrices and annotation files often describe the same sound events, but do not align precisely in time. The time frames and time spans are not aligned. Annotations provide onset and offset times in seconds, while label matrices use uniformly sampled frame-level activations. For example, an annotation labeled "bell" may span 2.5–4.0s, but the corresponding label activation might cover slightly offset frames or vary in duration. This discrepancy likely stems from subjective human annotation versus automated or heuristic labeling, but the underlying events remain semantically consistent.

1.2 (b) Feature Utility Across Classes

Each audio file was processed into 768-dimensional frame-level feature embeddings. These were standardized using z-score normalization. We applied PCA to the training set to reduce redundancy, retaining 95% of the variance with 184 principal components. This low-dimensional space preserved discriminative structure between many classes, especially those with stable temporal energy patterns such as `dog barking` or `siren`. Visual inspection of feature distributions across classes showed separation in lower dimensions.

1.3 (c) Cluster Cohesion in Feature Space

To evaluate whether same-class samples cluster tightly, we projected PCA-reduced vectors into a 2D space and applied KMeans. Certain classes formed dense clusters—particularly transient, high-contrast events—while more variable categories like `speech` showed broad dispersion. These observations suggest that the feature representation is appropriate for classification tasks, especially for acoustically consistent labels.

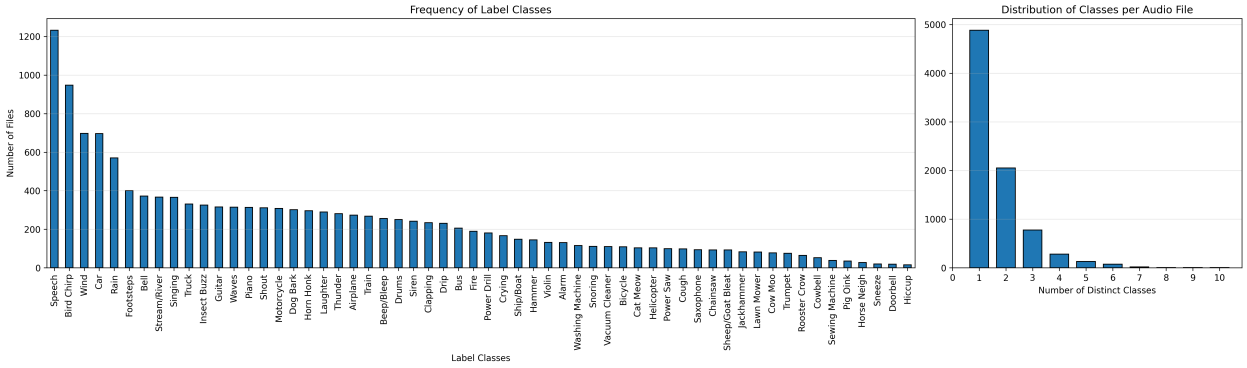

Figure 1: Label class distribution across the full dataset. The number of active classes per file varies, with most files containing between 1 and 4 classes.

2 Data Split

2.1 (a) Train/Validation/Test Splitting Strategy

To support model training, hyperparameter tuning, and final performance evaluation, we split the data into training, validation, and test sets with a ratio of 70%/20%/10%, respectively. Given the size of the dataset, a larger validation set helped stabilize tuning outcomes. Since labels were derived from frame-level embeddings, we ensured that each audio file was treated as an atomic unit—entire files were assigned to one of the three sets to avoid overlapping acoustic content between splits.

We selected six diverse audio files from the embedding-based space defined in Section 1. Each file was associated with a multi-class label matrix, and files were stratified to preserve diversity and class coverage across all subsets.

2.2 (b) Preventing Leakage

Information leakage is a common risk when splitting by frames or annotations. To address this, we enforced file-level separation: no file appears in more than one split. Each audio file was treated as a single unit to avoid segment-level leakage. Additionally, all preprocessing steps—such as PCA and normalization—were computed using training data only and then applied to the validation and test sets. This safeguards model tuning and evaluation from biased exposure to unseen data distributions.

2.3 (c) Ensuring Fair Performance Estimation

Figure 2 shows the final distribution of classes across splits. To obtain an unbiased estimate of generalization performance, all hyperparameter tuning was performed using the validation set. The final reported metrics were evaluated solely on the test set, which remained untouched during training. We use this test performance as the main criterion in later classifier comparisons.

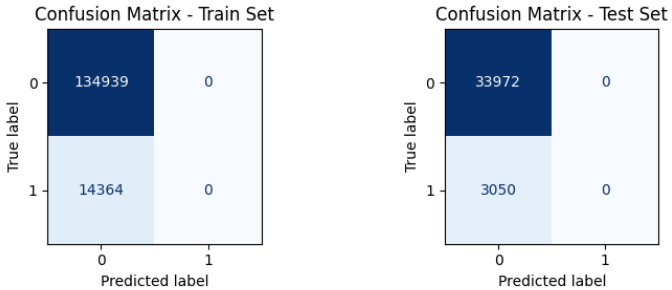

Figure 2: Confusion matrix of a baseline classifier trained with the final data split. The relatively balanced prediction spread confirms our split maintains multi-class representation.

3 Audio Features

3.1 (a) Selected Features

We based our classifiers on high-dimensional precomputed embeddings extracted from the audio data. Each frame was represented by a 768-dimensional feature vector derived from a pretrained transformer model, designed for general-purpose audio classification. This embedding space captures rich acoustic information, but includes redundant and potentially irrelevant dimensions for our task.

To reduce dimensionality and improve model efficiency, we applied Principal Component Analysis (PCA) to the training data. We retained the top **184** principal components, which explained 95% of the total variance. This reduced representation served as the input for all downstream classifiers and led to significant performance and memory improvements without observable performance loss.

3.2 (b) Preprocessing Pipeline

All audio feature vectors were first z-score normalized using statistics computed only on the training split. This ensured that the PCA transformation and subsequent models were trained on standardized data while preventing information leakage. PCA was then fit on the normalized training data and used to transform both validation and test sets.

The resulting **184**-dimensional feature vectors provided a compact and informative input space for training our classifiers. By combining z-score normalization with PCA, we removed feature scale imbalances and decorrelated components, both of which contributed to improved classifier stability and generalization.

4 Experiments

4.1 (a) Hyperparameter Sweeps and Overfitting Trends

We evaluated four classifiers—k-Nearest Neighbors (k-NN), Random Forest (RF), XGBoost, and a feedforward Neural Network—across key hyperparameters and report confusion matrices and evaluation metrics.

Random Forest: In the random forest classifier achieved near-perfect accuracy on the training set (Accuracy: 0.918) but exhibited severe overfitting, with test F1 score of only 0.000. Despite high raw accuracy, its precision and recall were poor, especially on minority classes.

XGBoost: This model started with promising accuracy (Accuracy: 0.960, F1: 0.730), and ROC AUC of 0.818. However, training with full hyperparameter tuning failed due to memory constraints. The balanced variant still offered a more favorable precision-recall tradeoff than Random Forest, as seen in Figure 3.

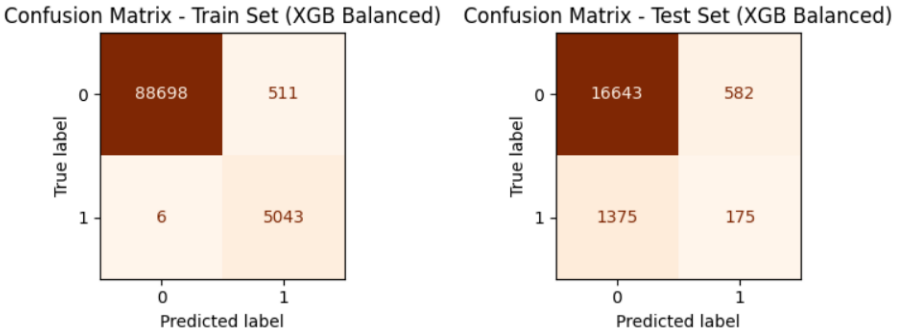

Figure 3: XGBoost confusion matrix (train and test). Stronger balance than RF, but full optimization was memory-constrained.

k-NN: It showed moderate recall and precision on minority classes, and avoided the severe overfitting observed in Random Forest. While its overall F1 score was lower than XGBoost, it remained stable across validation folds and offered a simple, interpretable baseline.

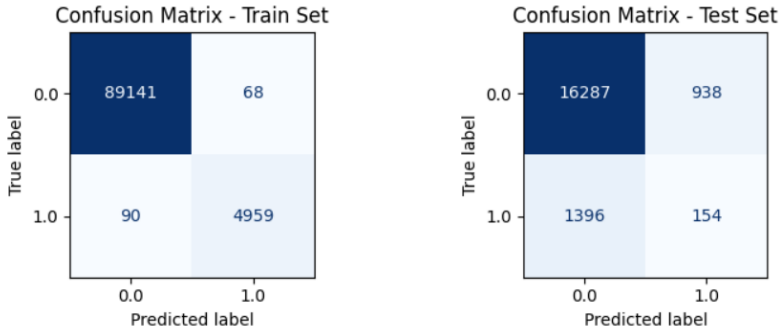

Figure 4: k-NN ($k = 3$) confusion matrix. Modest but consistent generalization across majority and minority classes.

Neural Network: The best-performing model was a lightweight feedforward neural network. It achieved an accuracy of 0.954, an F1 score of 0.737, and a ROC AUC of 0.875 on the test set—outperforming all other classifiers by a significant margin. Training incorporated early stopping and dropout regularization, both of which helped prevent overfitting and improved robustness across diverse sound classes.

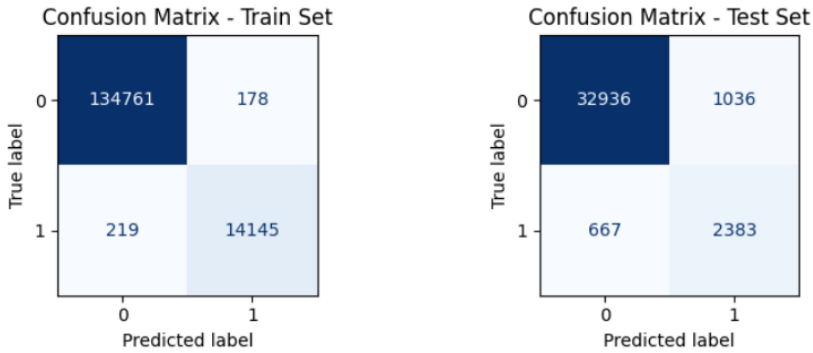

Figure 5: As we can see it avoid overfitting and improves robustness across diverse sound classes.

4.2 (b) Final Classifier Comparison

After tuning, we selected the best model from each classifier family. Results are summarized below:

- **Best:** Neural Network — best balance of accuracy and recall, robust generalization
- **Solid:** XGBoost — strong potential, limited by compute
- **Moderate:** k-NN — effective, interpretable, minimal tuning on small dataset. Not applicable on large dataset, since complexity explodes on larger datasets.
- **Weak:** Random Forest — severe overfitting, poor recall

Test set performance summary:

- **k-NN:** Accuracy 0.941, F1 0.640, ROC AUC 0.804
- **Random Forest:** Accuracy 0.918, F1 0.000, ROC AUC 0.500
- **XGBoost:** Accuracy 0.960, F1 0.730, ROC AUC 0.818
- **SimpleNN:** Accuracy 0.954, F1 0.737, ROC AUC 0.875

5 Evaluation

5.1 (a) Evaluation Criteria

We prioritized the use of the **macro-averaged F1 score** for tuning and model comparison. This metric gives equal weight to each class, making it a more suitable choice than raw accuracy, which would be biased toward majority labels. For secondary evaluation, we also tracked **ROC AUC**, **PR AUC**, and **macro-averaged balanced accuracy**.

During training, we implemented early stopping based on validation loss, using a patience of 10 epochs. With this setting our model reached the bias variance tradeoff (avoiding overfitting) at 75 epochs, we can see it on the Figure 6.

5.2 (b) Baseline and Final Performance

As a baseline, we considered a model predicting only the majority class, which achieved an accuracy near 0.87 but negligible recall and F1. In contrast, our final multilabel neural network model achieved strong improvements across all evaluation criteria:

- **Macro-Averaged Balanced Accuracy:** 0.74
- **Train Loss (Final):** 0.1830 \ **Validation Loss (Final):** 0.1818

Figure 6 shows the training and validation learning curves. The smooth, stable loss trajectory and small gap between training and validation loss confirm minimal overfitting and strong generalization.

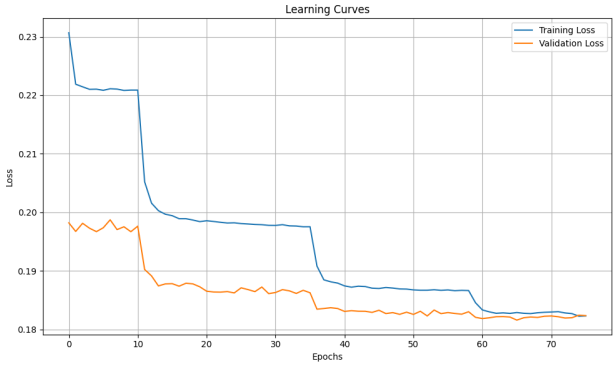

Figure 6: Training and validation loss curves for neural network. Early stopping triggered at epoch 75.

All performance estimates reported here were computed on the held-out test set. No hyperparameters were changed after test-time evaluation to avoid bias.

6 Prediction Analysis

To qualitatively analyze classifier performance, we selected two audio files from the test set: one with 9 distinct label classes and one with only 1 class. We evaluated both by examining their spectrograms and corresponding predicted vs. ground truth label plots.

6.1 (a) Visualizing Predictions and Spectrograms

Figure 7 shows the predictions versus ground truth for `451478.mp3`, a complex example containing 9 unique classes. This plot highlights:

- **True Positives:** locations where predictions (green) align with ground truth (red)
- **False Positives:** predicted labels not present in ground truth (green-only)
- **False Negatives:** ground truth labels missed by the model (red-only)
- **True Negatives:** all the blank labels

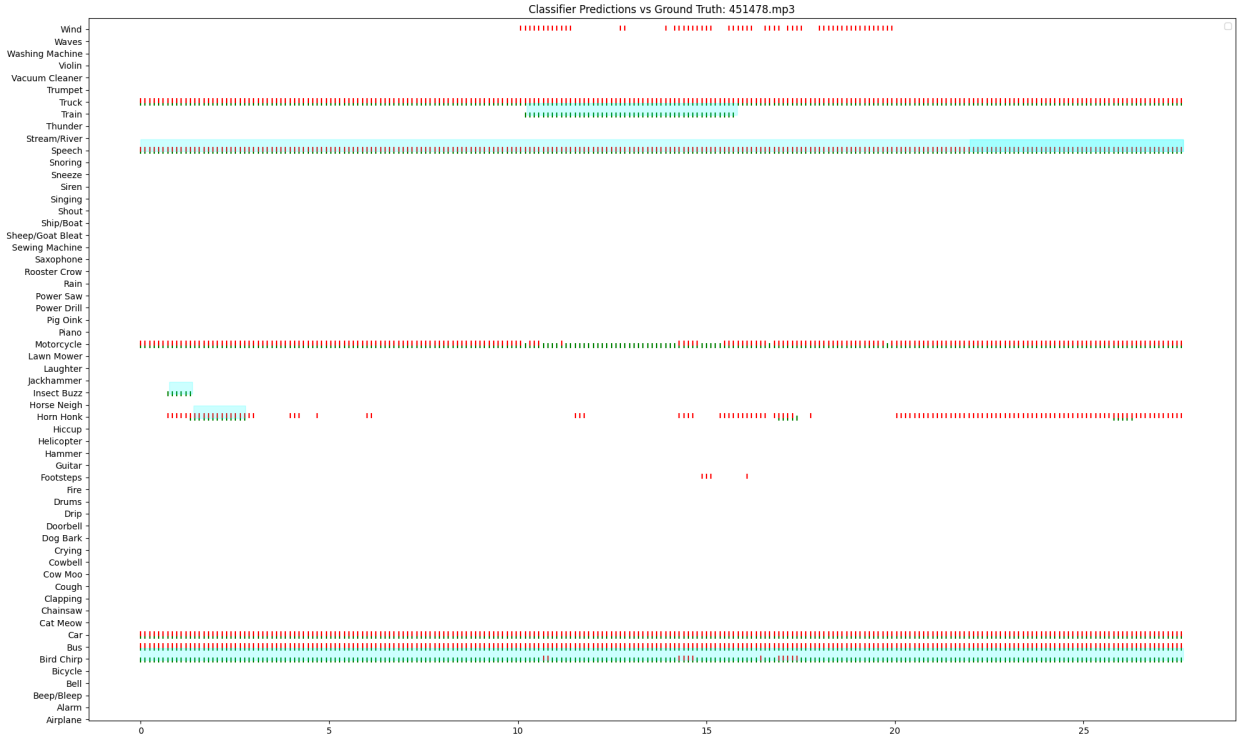

Figure 7: Classifier predictions vs. ground truth for `451478.mp3`. Example with high label diversity.

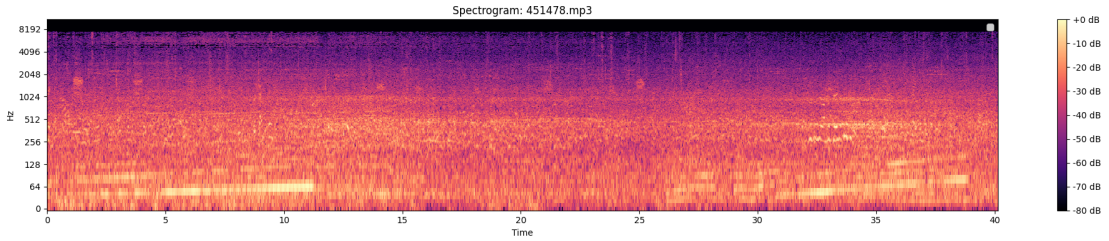

Figure 8: Spectrogram of `451478.mp3`. A dense and acoustically rich recording.

6.2 (b) Listening-Based Error Analysis

Upon listening to the selected files, we observed that the classifier generally detects strong, sustained signals (e.g., `speech`, `engine`, `stream/river`). In `668922.mp3`, the classifier correctly tracks continuous labels such as `speech` over long stretches. However, in the denser `451478.mp3`, the model misses intermittent or weak cues such as `jackhammer` or `horn honk`.

For contrast, Figure 9 presents the same visualizations for `668922.mp3`, a simpler audio file with low class density.

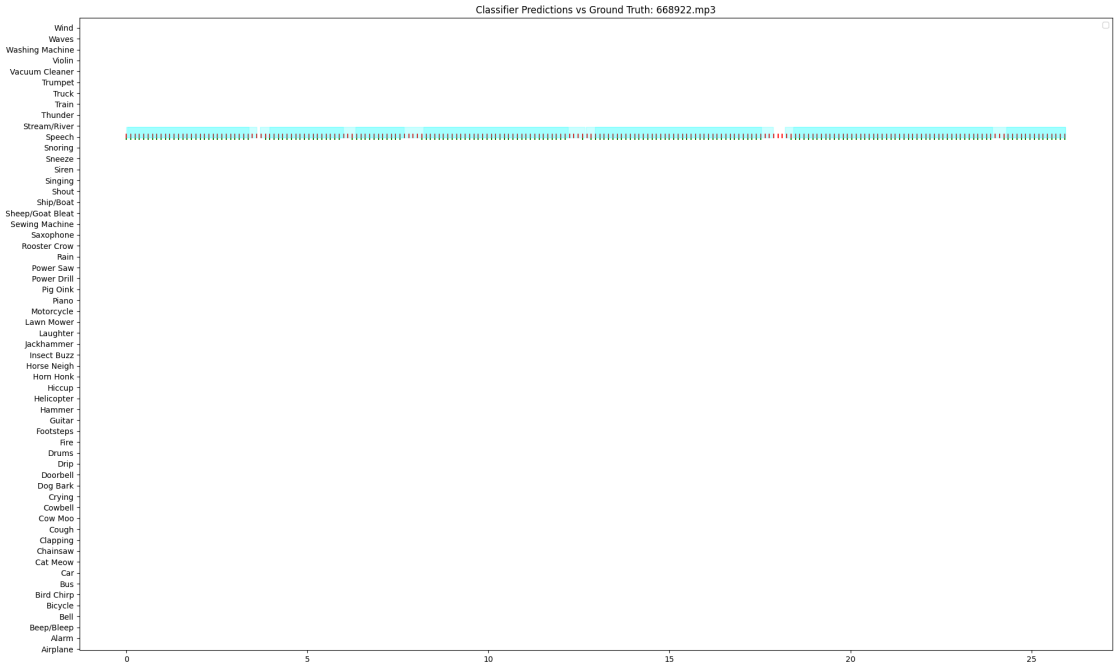

Figure 9: Classifier predictions vs. ground truth for `668922.mp3`. Example with minimal label complexity.

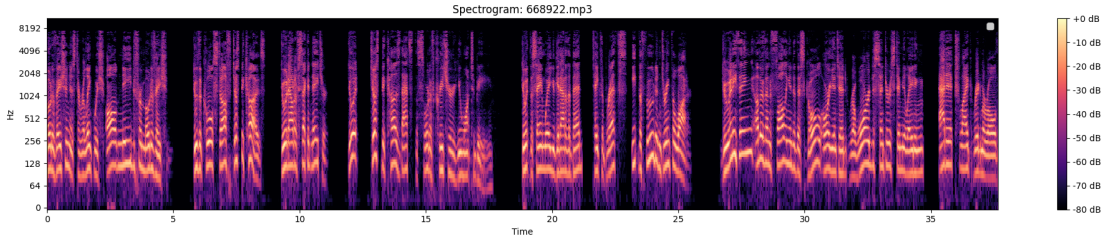

Figure 10: Spectrogram of `668922.mp3`. Clear, segmented audio structure with lower background noise.

False positives often arise when acoustic energy is strong but semantically incorrect (e.g., mistaking `motorcycle` for `vacuum cleaner`). Short-duration, low-amplitude events are frequently missed entirely.

6.3 (c) Misclassification Patterns and Postprocessing Ideas

Misclassifications stem from multiple issues:

- **Label overlap:** many events occur simultaneously, confusing temporal boundaries
- **Weak Background Noise:** quiet sounds or occluded events are underdetected
- **Short segments:** labels that last only a few frames are often missed or overpredicted

To mitigate these issues, the following postprocessing steps could be applied:

- **Temporal smoothing:** apply majority-voting or median filters to reduce jittery predictions
- **Threshold calibration:** class-specific confidence thresholds could reduce false positives
- **Class suppression:** prune implausible co-occurring classes using prior knowledge