MLPC 2025 Task 3: Frame-Level Classification Experiments

May 16, 2025

1 Data Split Strategy and Rationale

1.1 Data Partitioning for Model Development and Evaluation

The dataset, comprising 8230 audio files with associated pre-extracted frame-level features, was divided at the **audio file level**. A fixed random seed (`RANDOM_STATE = 42`) was employed for all splitting operations to guarantee reproducibility, and the dataset was allocated as follows:

- **Training Set:** 5761 files (70% of total), used for model parameter learning.
- **Validation Set:** 823 files (10% of total), used for hyperparameter tuning and optimal epoch selection during final model retraining.
- **Test Set:** 1646 files (20% of total), reserved for a single, final unbiased performance evaluation of the selected models.

1.2 Potential Information Leakage Factors and Mitigation

Information leakage across data splits can inflate performance metrics. Potential risks with a file-level split include:

- **Source Recording Overlap:** Multiple feature files originating from segments of the same longer audio recording could be distributed across different splits.
- **Consistent Environmental/Speaker Characteristics:** Similar leakage can occur if groups of files share unique recording environments or distinct speaker traits.

Addressing these risks: A random file-level split was implemented. While this carries a potential for leakage, the consistent use of `RANDOM_STATE = 42` ensures fair comparison across models.

1.3 Methodology for Unbiased Final Performance Estimates

Unbiased final performance estimates were ensured by:

1. Sequestering the test set from all training and tuning processes.
2. Using the validation set exclusively for hyperparameter selection and best epoch identification during model development and retraining.
3. Evaluating the final selected model (best architecture, hyperparameters, and epoch from retraining) only once on the test set.

2 Audio Features

2.1 Feature Selection and Rationale

All experiments utilized the **complete set of 13 pre-extracted audio feature types**. These include: `embeddings`, `melspectrogram`, `mfcc`, `mfcc_delta`, `mfcc_delta2`, `flatness`, `centroid`, `flux`, `energy`, `power`, `bandwidth`, `contrast`, and `zerocrossingrate`. These were concatenated frame-wise, forming a 942-dimensional input vector per 120ms frame (log: `INPUT_DIM: 942 from 13 features`). This inclusive approach was chosen to establish comprehensive baselines by providing models with maximal available information. Detailed individual feature utility analysis is addressed in Section 1 (Labeling Function).

2.2 Feature Preprocessing Applied

The provided features were used with minimal direct preprocessing:

- **No Global Scaling/Normalization:** No dataset-wide global feature scaling was applied post-loading. Model components like Batch Normalization (in CNN1D) offer some robustness to feature scale variations.
- **Sequence Alignment and Padding:**
    - (i) *Intra-file Alignment:* Within each audio file, feature streams were aligned to a common frame count (minimum length among them), primarily by truncation.
    - (ii) *Batch Padding:* Variable-length sequences were zero-padded within each mini-batch by the `collate_fn` to match the longest sequence in that batch.

3 Evaluation Framework

3.1 Choice and Justification of Evaluation Criterion

The primary metric for model selection and comparison was the **mean per-class Balanced Accuracy (BACC)**. This was chosen due to:

- The multi-label nature of the task (58 classes per frame).
- Significant class imbalance common in sound event datasets; BACC mitigates misleadingly high standard accuracy by averaging sensitivity and specificity for each class ($BACC_c = 0.5 \times (TPR_c + TNR_c)$), then averaging these per-class BACC scores.

The training loss function was Binary Cross-Entropy with Logits (`nn.BCEWithLogitsLoss`).

3.2 Baseline and Optimal Performance Benchmarks

- **Baseline Performance:** A theoretical random guessing baseline yields 0.5 mean per-class BACC. Our empirical baseline, the best MLP model, achieved a Test BACC of **0.8008**.
- **Optimal Performance:** The theoretical maximum BACC is 1.0. Our experiments achieved a current best Test BACC of **0.8201** with the CNN1D model.

4 Experiments and Model Performance

4.1 Experimental Setup Overview

- **Dataset & Features:** 5761 train / 823 val / 1646 test files; Input: 942 features/frame, 58 classes.
- **Training Protocol:** Adam optimizer, 'ReduceLROnPlateau' scheduler. Hyperparameter search: 15 epochs (scheduler patience 3). Final retraining: 30 epochs (scheduler patience 6). Batch size 32.
- **Evaluation Metric:** Mean per-class BACC (validation set for selection, test set for final score).

4.2 Multi-Layer Perceptron (MLP) Results

MLPs with one hidden ReLU layer were evaluated.

- **Hyperparameters Tuned:** Hidden dimension ($h \in \{128, 256, 512\}$), learning rate ($lr \in \{10^{-3}, 5 \cdot 10^{-4}, 2 \cdot 10^{-4}\}$). (9 configurations).
- **Key Findings:** Performance improved with larger h ($h = 512$ optimal). $lr = 0.001$ was generally best. Overfitting (Train BACC > Val BACC) was observed (e.g., Cfg1/9 Ep15: Tr 0.795, Val 0.789) but manageable.
- **Best Search Config:** $h = 512, lr = 0.001$. Val BACC: 0.7934.
- **Final Test BACC: '0.8008'** (Loss: 0.0381). Retrained Val BACC peak: 0.7960. Checkpoint: `MLP_testBACC_0.8008.pth`.

4.3 Long Short-Term Memory (LSTM) Results

LSTMs were tested for temporal modeling.

- **Hyperparameters Tuned:** Hidden dim ($h \in \{128, 256\}$), layers ($l \in \{1, 2\}$), dropout ($d \in \{0.25, 0.4\}$), $lr \in \{10^{-3}, 5 \cdot 10^{-4}, 2 \cdot 10^{-4}\}$). (24 configs).
- **Key Findings:** Many LSTMs (20/24) stagnated (Val BACC ≈ 0.751), especially $l = 2$. Effective learning favored $l = 1, h = 256$. Best search ($h = 256, l = 1, d = 0.25, lr = 0.001$) achieved Val BACC 0.7807. During its final retraining, Tr BACC slightly decreased (0.722 $\rightarrow$ 0.690) while Val BACC increased (0.751 $\rightarrow$ 0.797) and Tr Loss decreased, possibly due to dropout effects.
- **Final Test BACC: '0.7986'**. Loss: 0.0447. Checkpoint: `LSTM_testBACC_0.7986.pth`.

4.4 1D Convolutional Neural Network (CNN1D) Results

Two-block 1D CNNs with BatchNorm and Dropout were evaluated.

- **Hyperparameters Tuned:** Filters $f_1 \in \{64, 128\}$, $f_2 \in \{128, 256\}$; kernel $ks \in \{3, 5\}$; dropout $d \in \{0.25, 0.4\}$; $lr \in \{10^{-3}, 5 \cdot 10^{-4}, 2 \cdot 10^{-4}\}$). (48 configs).
- **Key Findings:** CNN1Ds were top performers. $ks = 5$ outperformed $ks = 3$. Larger filter counts and $d = 0.4$ were generally beneficial. $lr = 0.0005$ was often optimal. Best search ($f_1 = 128, f_2 = 128, ks = 5, d = 0.4, lr = 0.0005$) yielded Val BACC 0.8134. Strong generalization was common (often Val BACC > Tr BACC, e.g., Cfg35/48 Ep14: Tr BACC 0.691, Val BACC 0.813), indicating effective dropout.
- **Final Test BACC: '0.8201'**. Loss: 0.0381. Checkpoint: `CNN1D_testBACC_0.8201.pth`.

4.5 Final Model Comparison and Selection

Table 1: Final Test Performance of Optimized Model Architectures.

Model Family	Best Hyperparameters (Search Phase)	Val BACC (Retrain)	Test Loss	Test BACC
MLP	h=512, $lr = 10^{-3}$	0.7960	0.0381	0.8008
LSTM	h=256, l=1, d=0.25, $lr = 10^{-3}$	0.7970	0.0447	0.7986
CNN1D	**f_1=128, f_2=128, ks=5, d=0.4,** $lr = 5 \cdot 10^{-4}$	**0.8151**	**0.0381**	**0.8201**

Hyperparameter abbreviations: h=hidden_dim, l=num_layers, d=dropout_rate, f_1=filters blk1, f_2=filters blk2, ks=kernel_size, lr=learning_rate.

The systematic experiments identify the **1D Convolutional Neural Network (CNN1D)** as the superior architecture, achieving a Test BACC of **0.8201**. This model's proficiency in capturing local temporal patterns, augmented by effective regularization, proved most advantageous. The MLP established a robust baseline (Test BACC 0.8008). LSTMs (Test BACC 0.7986) were more challenging to optimize effectively for this dataset.

Based on these results, the **CNN1D model with parameters: filters block 1=128, filters block 2=128, kernel size=5, dropout rate=0.4, and learning rate=$5 \cdot 10^{-4}$ is selected as the best-performing model.**