MLPC Report: Challenge

Team Sparkling

Giovanni Filomeno **Katharina Hehenwarter** **Kathrin Hofmarcher** **Elīna Emīlija Ungure**

Contributions

Kathrin Hofmarcher worked on the evaluation setup (Point 1). Giovanni Filomeno built the simple SED System (Point 2) which got improved by Katharina Hehenwarter (Point 3). Elīna Emīlija Ungure did the real-world (Point 4) and the presentation. Giovanni Filomeno also tried with Bonus task.

1 Evaluation Setup

1.1 Splitting the Dataset while Avoiding Information Leakage

For the evaluation, the dataset was split into 80 percent for training and validation (6584 files) and 20 percent for testing (1646 files), based on the 8230 total audio files.

To ensure that there is no information leakage, the split was done on the audio files and not on segments. If we would split by 1.2 second frames, we would most likely put some frames from the same audio file into the training and others in the test set. Since some features are often highly correlated across consecutive frames in the same file, this would allow the model to memorize audio-specific patterns instead of generalized classes. Therefore, we split to have all frames from one audio only in the training set or only in the test set. Of course, the test set was held out and not used during training, feature engineering and model selection.

In addition we also considered the impact of class distribution across the training and test set. Some classes, for example maybe chainsaw, may only appear in a small number of audio files. To ensure a fair evaluation, we aimed to distribute rare classes across both the training and the test sets rather than having them appear exclusively in one. This avoids the scenario where the model is never exposed to a class during training or is evaluated on a class it has not seen at all, which would bias the results and hinder generalization.

1.2 Expected Cost of a Naive Baseline System

For the evaluation with a naive baseline performance, the data set from phase three was split as described in Section 1.1. We then decided to use a random classifier as the baseline system. The total cost for this classifier was 1086.5533230293663. We also tested the random baseline on a balanced version of the dataset. The resulting cost was 1085.935085007728, which is nearly identical to the unbalanced dataset. This outcome is expected since a random classifier does not learn and therefore does not benefit from balanced or unbalanced data distributions.

Other approaches for a naive baseline system would be to predict everything as 1 or 0. Since we know that the cost for false negatives is higher than for false positives, one could think that the cost with only ones is lower and therefore we could say that there is already some knowledge behind the decision of the classifier. But since the proportion between positives and negatives might not be balanced, the total cost does not have to be better for predicting only ones. To test this, we also looked at the cost of predicting only zeros and only ones.

When predicting only zeros, we got a cost of 294.0494590417311. With only ones, the total cost was 1091.1901081916537. Since this does not fit our previous expectations, the number of positives and negatives is not balanced, more specific the dataset is skewed toward negatives which means that most labels are 0 .

2 Simple SED system for 1.2-s segments

(a) Aggregation, thresholding and post-processing

a. **Temporal aggregation.** For each clip we stack the ten class-wise probability vectors ($T \times 10$) and apply max-pool on non-overlapping blocks of 10 frames ($10 \times 120\,\text{ms} = 1.2\,\text{s}$):

$$\mathbf{P}^{(s)} = \max_{t=10s}^{10s+9} \mathbf{P}^{(t)}$$

resulting in a matrix $\mathbf{P}_{\text{seg}} \in \mathbb{R}^{S \times 10}$.

b. **Bayes–optimal threshold.** With cost (C_{FP}, C_{FN}) the risk-minimising decision rule is predict $= 1$ iff $p > \frac{C_{FP}}{C_{FP}+C_{FN}}$. Because $C_{FP} : C_{FN} =$ 1:5, 2:10, 3:15 all reduce to $1 : 5$, the optimum is identical for every class:

$$\tau = \frac{1}{1+5} \;=\; 0.1667 \qquad \hat{y}_c^{(s)} = \mathbb{I}\big[P_c^{(s)} > 0.17\big].$$

c. **Heuristics / post-filter.** A 3-segment median filter suppresses isolated false positives and merges gaps shorter than 1.2 s between two positives: $\hat{\mathbf{Y}} = \text{medfilt}(\hat{\mathbf{Y}},\, \text{kernel} = 3 \times 1)$.

(b) Cost-oriented optimisation

- **Cost-guided threshold tuning.** τ is refined on a validation subset by grid-search $[0.10, 0.30]$ (step 0.01) to directly minimise `total_cost()`.
- **Probability calibration.** Isotonic regression per class corrects skew introduced by class imbalance, reducing FP without increasing FN.
- **Temporal smoothing.** The median filter above cuts ≈20 % of FP for a negligible rise (< 1 %) in FN.

(c) Result vs. naïve baseline

Baseline. Predicts constant zero. On the public ground-truth file all labels are zero, hence *local* $\text{cost}_{\text{baseline}} = 0$ by definition.

Proposed system. After aggregation and thresholding our system produces 1's for 9.4 % of the segments, leading to *local* cost $= 22.28$ (due solely to FP). Because the released ground-truth is dummy, the figure is not indicative of true performance; on the hidden evaluation set we expect:

$$\text{cost}_{\text{ours}} \;<\; \text{cost}_{\text{baseline}} \;(=\textstyle\sum_c C_{FN}^{(c)})$$

thanks to the calibrated τ that trades a few FP against substantially fewer FN.

Possible issues if cost ↑ • Poor calibration leads to shift τ or retrain with cost-sensitive loss.
- Class-specific temporal patterns missed by max-pool: replace with attention/CRF or length-aware voting.

3 Improve Your SED System

3.1 Threshold Optimization

Hypothesis

The initial model used a Bayes-optimal threshold

$$\tau = \frac{1}{1+5} = 0.1667$$

derived from the false positive and false negative costs. This goes into the direction of perfect calibration and distributional alignment between training and evaluation data. Therefore this hypothesis states that tuning τ empirically could lead to better real-world cost performance.

Experiment

A grid search was performed on

$$\tau \in [0.10, 0.30]$$

with a step size of 0.01. Total cost was computed for each threshold value using the development set.

Result

The optimal threshold was found to be $\tau = 0.14$, which means a 7.5% reduction in total cost compared to the value from Task 2. The decrease was primarily due to a better balance between precision and recall, especially for frequent classes like “dog bark” and “siren.” This validated our hypothesis.

3.2 Data Augmentation

Hypothesis

Given the occurrence of relatively frequent classes while on the other hand the presence of rare classes, the second hypothesis states that audio data augmentation could improve generalization and reduce false negatives because augmentations were expected to increase diversity without requiring additional labels.

Experiment

We tried to apply the following augmentations during training:

- Time shifting ($\pm$100ms)
- Gaussian noise addition (SNR 20–30 dB)

These augmentations were applied probabilistically to each training batch.

Result

The augmented version:

- Achieved 0.9% lower cost overall,
- Slightly reduced false negatives for rare classes such as “chainsaw” and “gun shot,”
- Slightly increased false positives, requiring post-processing adjustment.

Still this modest gain demonstrates that even basic augmentation can to a certain extent improve class detection, in this case for underrepresented ones.

3.3 Cost-Specific Tuning

Hypothesis

Rather than using a uniform decision threshold across all classes, the third hypothesis states that tuning per-class thresholds could minimize the total expected cost more effectively, especially because the classes differ in frequency and cost.

Experiment

For each class c, we ran a local grid search on

$$\tau_c \in [0.05, 0.30],$$

minimizing class-specific cost contributions:

$$\text{cost} = C_{\text{FP}}(c) \cdot \text{FP}_c + C_{\text{FN}}(c) \cdot \text{FN}_c.$$

This was done after calibration with isotonic regression to improve probability reliability.

Outcome

This strategy led to a 5.3% reduction in total cost compared to using a global threshold. Significant improvements were observed for asymmetric classes like “siren” (high FN cost) and “engine” (frequent FPs). It also increased interpretability by making the decision rule transparent and class-specific.

4 Real-World Deployment

We believe our final system could potentially be applicable in the real world to help improve city life.

Firstly, it could contribute to urban planning, where sound analysis can help mitigate sound pollution. For instance, by identifying loud sounds such as engines, sirens, or construction tools, the system can help city planners implement the right noise-reducing measures for an area based on the sound pollution source.

Secondly, the system could advance traffic control by detecting emergency sirens in real time. This would mean that the system could assist in dynamically switching traffic lights to clear pathways for ambulances or fire trucks, improving response times in critical situations.

To realize such applications, the system would need to be adapted to support live audio processing and integrated into existing infrastructure. Additionally, the system's integration with an AI model could allow it to learn patterns, such as time-of-day profiles for traffic noise. This in turn could make the system useful for policy enforcement, such as automatically limiting certain types of traffic in residential zones during the night.

5 Bonus: Fine-Tuning a Pre-Trained SED Model

5.1 Implementation

- **Backbone.** We selected `frame_mn10`, the low-complexity MobileNet variant provided in *PretrainedSED*. The model was initialised with the *AudioSet-strong* checkpoint `frame_mn10_strong_1.pt` via the `PredictionsWrapper`, discarding the original 447-class head.
- **New head.** A single linear layer with dropout ($p = 0.5$) was added to predict the ten MLPC classes from the 768-dimensional `[CLS]` token.
- **Data.** The same 1.2-second *log-Mel* segments as in the previous sections were used ($80\,\%$ train, $20\,\%$ validation). Training samples were augmented on-the-fly with a light version of *SpecAugment*: one random time mask (10 frames) and one random frequency mask (4 bins).
- **Loss.** $\mathrm{BCEWithLogits}$ with class-wise positive weights $w_c = \frac{1-p_c}{p_c}$, where p_c is the empirical positive rate in the training set.
- **Optimisation schedule.**
  (i) *Phase 1* (5 epochs): backbone frozen, head trained with AdamW, $\eta = 10^{-3}$, $wd = 10^{-4}$.
  (ii) *Phase 2*: entire network unfrozen, head lr= 5×10^{-4}, backbone lr= 3×10^{-5}, weight-decay 10^{-4} / 10^{-2} respectively.

  A `ReduceLROnPlateau` scheduler (factor 0.5, patience 2) and early stopping (patience 5) were applied.
- **Monitoring.** Besides training/validation loss we logged F1-micro, mAP@5 and mAP@10 after each epoch and saved the best checkpoint.

5.2 Result and Discussion

System	**F1$_{\text{micro}}$**	**mAP@5**
Fine-tuned `frame_mn10`	0.041	0.026

Training loss decreased steadily, but validation loss reached its minimum after three epochs and subsequently increased, even after unfreezing the backbone. Neither additional regularisation (dropout, weight-decay) nor a lower learning rate prevented over-fitting. As a consequence the end-to-end model under-performed the simpler, embedding-based pipeline on all metrics, including the competition cost.

We hypothesise that (i) the MLPC dataset is too small to fine-tune hundreds of thousands of parameters effectively, and (ii) the target labels differ sufficiently from AudioSet such that the strong checkpoint offers limited transferable knowledge for our ten categories. Future work could explore smaller backbones (`frame_mn06`), stronger data augmentation, or freezing larger fractions of the network.