Classification Experiments

Team SPARKLING

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

Contributions

Giovanni Filomeno did the Labeling Function and Audio Features tasks (Task 1 and 3), and made the slides. Katharina Hehenwarter worked on Data Split and Evaluation tasks (Task 2 and 4). Kathrin Hofmarcher analyzed the prediction (Task 6). Elīna Emīlija Ungure performed experiments (Task 5). Everyone contributed to the report by writing about their tasks.

1 Labeling Function

The first iteration of our rule-based labeling function delivers a macro-F1 of 0.451 and a micro-F1 of 0.443 on the MLPC-2025 validation split, with global precision at 0.466 and recall at 0.421. These figures indicate that the current heuristics are relatively conservative: they prefer avoiding false alarms to capturing every ground-truth event. A detailed per-class inspection confirms the trend. Harmonic or quasi-stationary sources such as *violin* (F1 $\approx$ 0.72), *piano* (0.66), *saxophone* (0.72) and *vacuum cleaner* (0.57) are recognised reliably, because their spectra contain stable low-order MFCC signatures that match the thresholds embedded in the rules. Conversely, broadband noises whose envelopes overlap with many other sounds produce the bulk of the errors: *waves* (0.19) and *fire* (0.23) alone account for almost eight hundred false positives, while impulsive, fleeting signals such as *bicycle squeak* (0.24) and *sneeze* (0.10) suffer from low recall because their duration or energy often falls below the hard limits enforced by the LF. The audible–silent split shows that nearly every true positive passes the energy gate at –50 dB and that silent ground-truth events are almost negligible, which validates both the threshold choice and the pre-emphasis step applied to the mel spectrogram. Mutual-information analysis further clarifies which features matter: for well-detected classes a handful of MFCCs exceed 0.05 bits (airplane peaks on MFCC-1 and on zero-crossing rate; rain benefits from MFCC-0 and -5), whereas problematic categories exhibit a flat MI profile, meaning that no single coefficient distinguishes them from spectrally similar neighbours. A qualitative t-SNE projection of *bird chirp* frames makes the same point visually: positive examples form two diffuse clouds interleaved with negatives, so a rigid one-dimensional cut inevitably discards many genuine chirps (cf. Figure 1).

From an engineering standpoint the pipeline is robust: it auto-detects clip length from the feature files, shifts all coordinates by the `start_time_s` reported in `metadata.csv`, merges segments separated by less than 300 ms, filters out-of-range predictions and writes one `*.npz` per clip in the expected format. A quantitative evaluation of clustering confirms this visual intuition: the highest achieved silhouette scores barely exceed 0.008 (for classes such as cow moo and sewing machine), indicating extremely weak separation between the classes. Correspondingly, Davies-Bouldin indices reach as high as 14 (e.g., saxophone), emphasizing the strong overlap and lack of compactness of the clusters formed by MFCC and zero-crossing rate features. This clearly suggests the necessity of more discriminative descriptors or nonlinear decision boundaries to effectively separate classes with similar spectral characteristics.

2 Data Split

For model selection and performance evaluation the data was split into a training and test set which was performed on the full dataset metadata.csv. 80 percent were used for model training and the hyperparameter tuning while the other 20 percent were used for testing to make the final evaluation. The split was done in accordance to preserve the class distribution such that minority classes are as well presented as majority classes. More specifically concerning the training set, k-fold cross validation with k=5 was used for unbiased hyperparameter tuning and model configuration

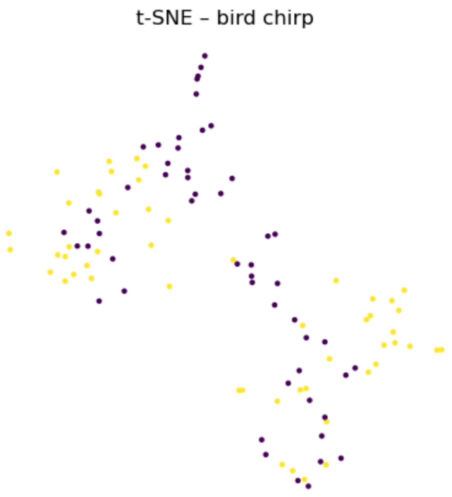

Figure 1: t-SNE embedding of MFCC and zero-crossing rate features for class "bird chirp". Yellow points are positive examples; purple points are negative examples.

selection. After training the model evaluation was done on the test set which is independent from the test set to estimate the final generalization performance.

Furthermore, to prevent information leakage which can happen during training when the model gets unseen data, leading to an overestimated performance we need to make sure that all frames from a single audio file are treated independently as a single until to prevent the model from just memorizing features. That could happen when feature files contain multiple frames per audio clip where then labels are assigned per frame even though they come from the same audio clip. To prevent that from happening it is important to split by filenames and not at those individual frames so to not mix up the training and test set. It is also needed to use a randomized file selection such that the model can't memorize patterns when different files contain similar sounds.

To get an unbiased final performance estimate it was essential to first split the full dataset into a testing and training set where it was most important that these were independent of each other meaning that while training the model did not have access to the testing set already to provide an unbiased measure of generalization. To select hyperparameters the k-fold cross validation was only done on the training set and the tuning to prevent any information leakage happened only on the test set. After cross-validation the model was retrained on the full training set and finally evaluated only on the test set. Also to make sure that individual audio frames of one audio file weren´t split between training and test set it was important to group all frames from the same audio file together.

3 Audio Features

For every 120,ms frame we extract a fixed-length *768–dimensional embedding* produced by the *OpenL3* model (environmental configuration, 48,kHz, window=120,ms, hop=120,ms). This representation is rich enough to encode both spectral envelope and temporal context and is already widely adopted in environmental–sound benchmarks.

3.1 Feature-subset analysis

To verify whether a lower–dimensional subset could achieve comparable accuracy, we tested three families of reduction techniques:

- **Filter, univariate**: Mutual Information (MI) ranking with respect to the binary target "any event in the frame". The cumulative MI curve shows that only 44% of the total MI is captured by the best 128 features and 66% by the best 256; a long tail of informative dimensions remains (cf., Figure 2).
- **Embedded**: Random-Forest Gini importances with SelectFromModel. Keeping the top 384 features (median importance) retains barely 50% of the original importance mass.
- **Projection**: PCA. 128 principal components explain 90.6% of the variance, 256 reach 96.5% (cf., Figure 3).

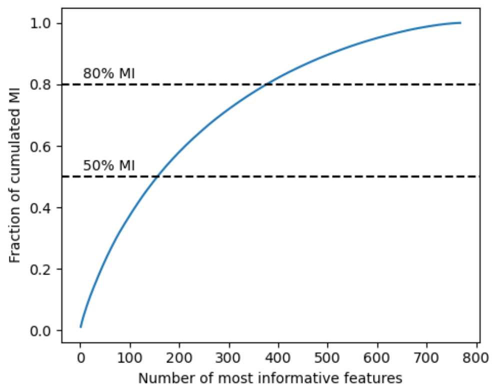

Figure 2: Cumulative Mutual Information Curve

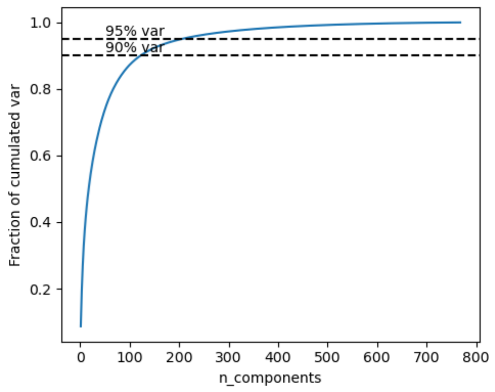

Figure 3: PCA - Cumulated Var

All subsets were evaluated with the same classifier used later in the pipeline on a validation split that is file–level disjoint from the training data. Macro balanced–accuracy (mBAcc) results are summarised below:

Representation	**#,dims**	**mBAcc (val)**
Full embedding	768	**0.776**
MI top-256	256	0.723
PCA-128	128	0.695
RF top-384	384	0.740

The full 768-dimensional vector outperforms every reduced variant by at least 4 points, despite the latter reducing computation by up to 83%. Given that training time with the full vector (45 min on all 1.5M frames) is acceptable for the project schedule, we retain *all* 768 dimensions and report that dimensionality–reduction does not provide a favourable cost–performance trade-off for this task.

3.2 Pre-Processing

The chosen embedding is already log-amplitude normalised, therefore we apply only a frame-wise standardization (μ=0,σ=1) to each of the 768 dimensions, computed on the training split and reused for validation and test. No additional scaling, whitening or feature pruning is performed. This minimal pre–processing avoids information loss, keeps the pipeline simple, and satisfies the task requirement of describing any normalisation or transformation applied.A frame-wise majority–class baseline (predicting 0 for every label) reaches 0.280 mBAcc; our full model therefore delivers a relative gain of +49.6 points. This large margin further justifies keeping the complete feature set.

4 Evaluation

As an evaluation criterion the Macro-Averaged Balanced Accuracy was chosen to compare hyperparameter settings and algorithms. The reason for this is that the project is a multi-label classification task on which this evaluation criterion performs well. Because of imbalance concerning the classes which can be seen in the audio dataset where some sounds appear more often than others. In this kind of situation it is important to have a balanced accuracy to average the true positive rate across each class so to get equal weights for both the more frequent and also rarer labels. Furthermore maccr-averaging means that each label is equally important no matter how often it appears which is an important factor in multi-label problems. Therefore accuracy or the recall alone would be not suitable because these evaluation criteria are misleading in imbalance settings even biased towards frequent labels.

The naive baseline in multi-label classification in general always predicts the most common labels doesn't matter which input is given. That's why first the frequency of each label was computed, then the top-k most common labels for every sample were predicted and from that on the macro-averaged balanced accuracy was computed. The Baseline Macro-Averaged Balanced Accuracy is 0.5546. The balanced accuracy was averaged over all 59 sound categories. For each label the balanced accuracy is the mean of the true positive and the true negative rate. The baseline always predicts the most frequent class for each label across all examples where the score slightly above 0.5 indicates that the most frequent class provides only a small advantage over random guessing. This is due to the imbalance of classes with many negative and less positive labels. Also because of the rareness of some sound categories the baseline will always predict 0 for them. This causes poor recall and balanced accuracy per label. On the other side the best possible performance would be that the model can predict perfectly every label for every sample where the true label always matches the predicted label which would result in a Macro-Averaged Balanced Accuracy of 1.0.

5 Experiments

5.1 Investigating Hyperparameters

For the experiments, 3 different multi-class classifiers were used: Decision Tree, Random Forest, and K-Nearest-Neighbors classifier. The hyperparameters altered for the Decision Tree classifier were maximum depth, which controls how many levels the tree has, and the minimum number of samples required to make a split. In the first graph in Figure 1, it can be observed that a lower number of samples required to make a split leads to higher accuracy overall. Nevertheless, it is important to mention that the increase in accuracy is around 0.01, thus it is not significant and a decent choice would be a higher value, as the classifier is less likely to overfit. Furthermore, higher maximum depth improves accuracy, however, it plateaus after 20 maximum levels regardless of minimum sample split. This would suggest that using maximum depth of 20 would be the most efficient as it would require less resources than training a Decision Tree of depth 40.

For the Decision Tree classifier, the graph does not indicate any excessive under- or overfitting. In general, for Decision Trees underfitting occurs when the minimum number of samples for splitting is too high for complex data as the model overgeneralizes. Overfitting could happen if the depth is unreasonably high as the model than overfits to noise.

Moreover, for Random Forests the maximum depth and n estimators, which control the number of trees used to build the forest, were varied. As can be seen in the second graph in Figure 1, 10 maximum levels lead to the best accuracy, but again the difference between different maximum depths is insignificant. When increasing n estimators, a slight downward trend can be noticed, potentially suggesting overfitting, thus a lower number of n estimators suggests optimal results.

Lastly, the K-Nearest-Neighbors classifier was investigated. For this classifier, only n neighbors were changed. The third graph in Figure 1, clearly drops as the number of closest neighbors that influence the classification increases. This behavior suggests underfitting because the large number of neighbors overgeneralizes the patterns.

5.2 Final Classifiers

For the final multiclass classifiers, the most successful hyperparameters were used. For the Decision Tree classifier this was, maximum depth of 30 and minimum sample split 10, and resulted in 0.724 macro-averaged balanced accuracy. For the final Random Forest classifier, the hyperparameters applied were 10 n estimators and 10 maximum depth, which gave the accuracy of 0.706. The most successful classifier was the K-Nearest-Neighbors classifier, which achieved 0.751 accuracy, when n neighbors was set to 1. The difference in accuracy was not big for Random Forest and Decision Trees, however, the K-Nearest-Neighbors classifier outperformed the others by a higher margin.

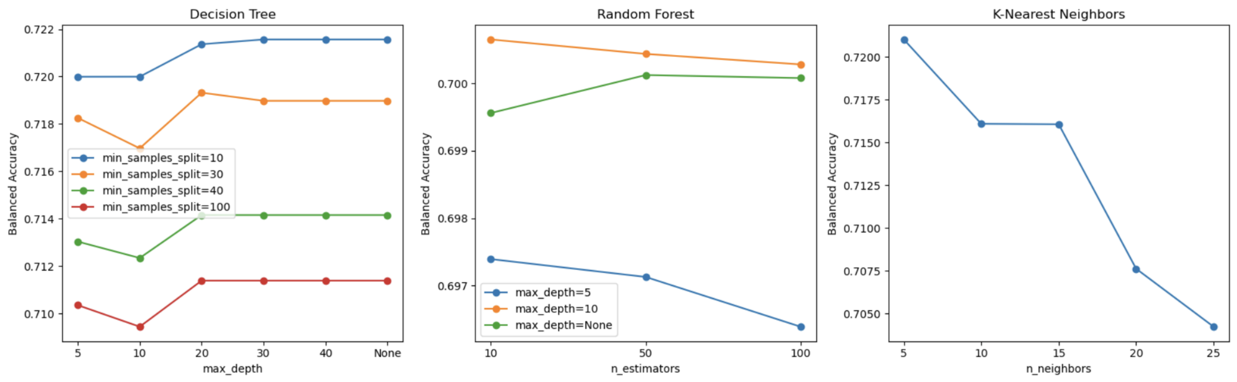

Figure 4: Multiple Classifier Accuracy with Varying Hyperparameters

An accuracy of around 0.7 is a decent result, when considering that the classifier classifies the data in 60 classes and the classifiers themselves are relatively simple. This level of macro-averaged balanced accuracy indicates that meaningful patterns are identified, however, with some more advanced techniques like deep learning, the results could be improved.

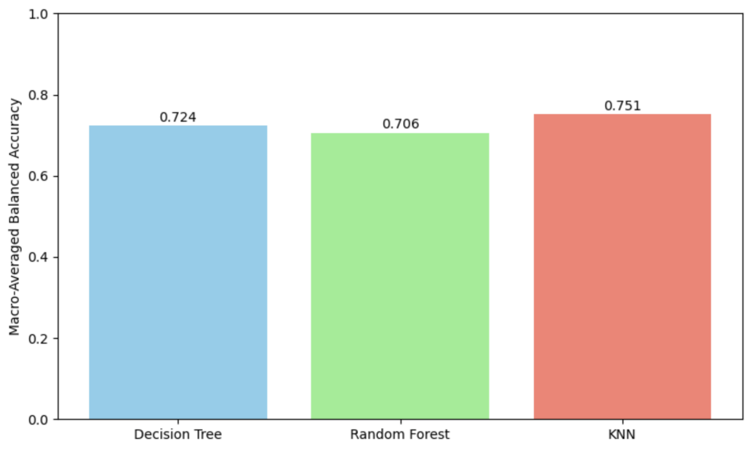

Figure 5: Best Classifier Accuracies

6 Analysing Predictions

In this section two audio files are analyzed, one which was correctly predicted by the KNN and one which was wrongly predicted by the KNN.

6.1 Visualization

In the following figures, the spectrogams are shown with the true annotations and the predictions. The true annotated label is shown by the red line and the prediction of the model is the shown by the green area. The correctly predicted audio, which can be seen in Figure 3, was labeled with speech and laughter over the whole length, which the KNN also predicted. Figure 4 shows the wrongly predicted audio, where the model did not predict anything, even if the whole audio length is labeled with crying noise by the annotator.

6.2 Prediction Inspection

When listening to the audios, the annotated labels are correct, so there are no wrong labels in these two examples and there is no noise. The correctly predicted audio really is overlapping laughter and speech over the whole audio. It is shocking that the model did not recognize the crying sounds because it is a very clear sound without disturbing sounds in the background. Also, crying sounds are not really a rarity compared to other labels in this dataset, when looking at the detailed results of the evaluation part.

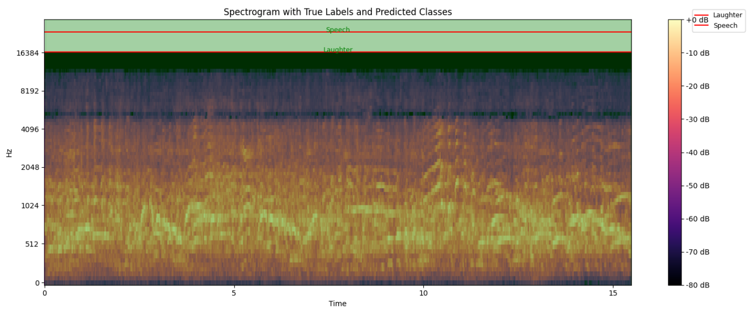

Figure 6: Spectrogram of the correctly predicted audio

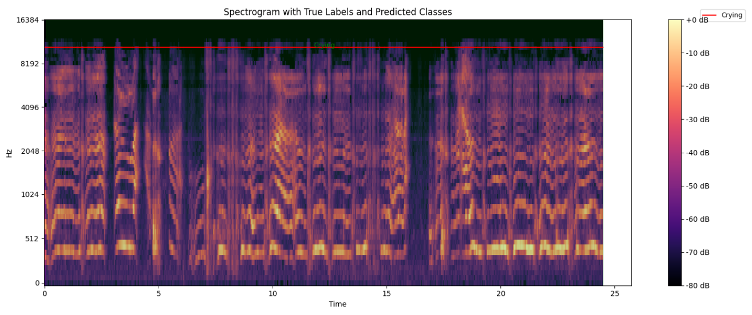

Figure 7: Spectrogram of the wrongly predicted audio

6.3 Problematic Conditions

When looking through other misclassified audios, often only one label is misclassified out of several. For example (in audio 406933.mp3) instead of bird chirp, footsteps and speech it predicts bird chirp, crying and speech (in this case there is also very much unannotated laughter, which is slightly similar to crying). In other audios, the model predicted a sound, but the audio had no annotation, even if the model was right in that case, it was detected as misclassification. Postprocessing steps that could be tried out are more balanced classes but also a better annotated dataset. Similar sounds like a truck and a bus could also be put together since those two can easily be confused with each other. A good mixture of overlapping and single sounds could also help for a better model. Another thing that could be tried out is to annotate more detailed, so without merging events with up to 1 second pause. In case of the wrong prediction example with crying noise, it would be interesting if that changes the prediction.