Mitigating Null-Class Dominance in Multiclass Inertial-Based Activity Recognition

HASCA-WEAR Challenge - A Technical Report of Team HARMA

Ricarda H. Link
University of Mannheim
Mannheim, Germany
ricarda.link@uni-mannheim.de

Heiner Stuckenschmidt
University of Mannheim
Mannheim, Germany
heiner@informatik.uni-mannheim.de

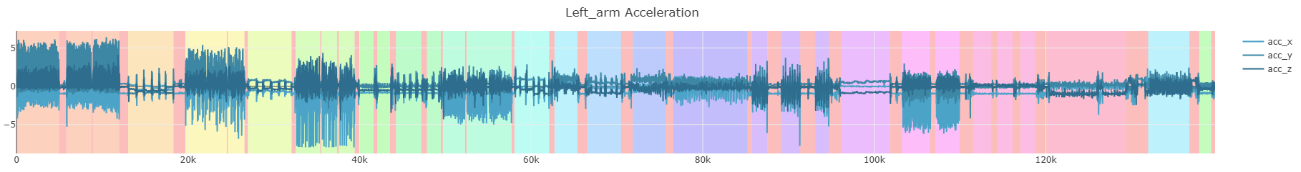


Figure 1: Visualization of WEAR accelerometer data at sensor location: left arm of an exemplary subject

Abstract

This technical report presents the methodology and results of Team HARMA's participation in the HASCA-WEAR Challenge. The challenge uses the WEAR dataset, which involves classifying 18 sports activities using inertial-based data. Additionally, the dataset includes a prevalent "null" class, resulting in 19 classification targets overall. The dominance of the null class introduces class imbalance, a key challenge in activity prediction. In this report, we focus on mitigating difficulties posed by this imbalance. To this end, we explore preprocessing strategies, including adjusted undersampling, which we adopt for the final challenge submission, as well as postprocessing techniques such as optimized thresholding based on prediction confidence. This report outlines our methodology, design choices, and evaluation results. Our approach uses an ensemble of gradient boosting methods applied to statistical and frequency-domain features extracted from raw sensor data.

CCS Concepts

• Human-centered computing → Ubiquitous and mobile computing; • Computing methodologies → Classification and regression trees.

Keywords

Machine Learning; Human Activity Recognition; Wearables; Sensor location; WEAR Dataset

ACM Reference Format:
Ricarda H. Link and Heiner Stuckenschmidt. 2025. Mitigating Null-Class Dominance in Multiclass Inertial-Based Activity Recognition: HASCA-WEAR Challenge - A Technical Report of Team HARMA. In *Companion of the 2025 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp Companion '25), October 12–16, 2025, Espoo, Finland.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3714394.3756192

1 Introduction

Human Activity Recognition (HAR) based on inertial-based data is a practical research topic with applications in areas like sports monitoring, healthcare, and wearable devices, [4]. The HASCA-WEAR Challenge provides a structured way to test different HAR approaches on a real-world dataset, the WEAR dataset by Bock et al., [2].

This section presents the dataset and classification task, highlighting specific characteristics that make the task more challenging. Section 2.2 details feature engineering, Section 3 introduces the predictive models, and Section 4 explores strategies for handling the null class. Section 5 presents results, with the key finding being that balancing misclassifications between null and non-null classes significantly improves performance. Section 6 summarizes the findings.

1.1 Description of the Task

The WEAR dataset [2] comprises recordings of 18 different outdoor sports activities, including various forms of jogging and stretching exercises. The data was collected from 22 participants across 11 different locations, capturing realistic variations in performance and environmental conditions. The dataset contains two types of data: visual recordings from a first-person GoPro camera and inertial-based measurements from accelerometers. However, the HASCA-WEAR Challenge restricts participants to using only the inertial-based data for activity recognition. Inertial-based data is collected from four sensor placements: the left and right arms, and the left and right legs. Each sensor captures acceleration along the three orthogonal axes (x, y, z), measured in units of gravitational acceleration ($1g = 9.81\,\text{m/s}^2$). The sampling rate is 50 Hz, corresponding to 50 data points per second for each axis. The training data consists of sequences where multiple activities are performed consecutively, with intervals between activities where none of the 18 predefined activities is performed, labeled as the null class.

1.2 Specific Characteristics of the Challenge Data

The challenge hosts provide a training and a challenge dataset which differ significantly from each other. The key differences between the two are outlined below.

1.2.1 Lack of Temporal Context. Instead of full sequences of activities, as in the training data, the challenge set consists of randomly sampled, 1-second sliding windows, which contain 50 accelerometer measurements per axis. This limits the ability to compute features that rely on longer time spans or movement patterns over several seconds. Additionally, no time-based postprocessing, such as smoothing or majority voting across adjacent time windows, which could help stabilize predictions, is possible.

1.2.2 Availability of Only One Sensor Location at a Time. Each test sample contains data from only one sensor location out of the four available locations which were simultaneously provided in the training data. The sensor location is provided and does not have to be estimated. This constraint makes it harder to incorporate full-body motion information, but reflects real-world use cases where only a limited number of sensors (e.g., on a smartwatch) are available.

1.2.3 Unseen Test Subjects. The challenge set includes data from four participants who do not previously appear in the WEAR dataset. This introduces additional variability due to personal differences in movement patterns and sensor placement, and increases the difficulty of generalizing the model to unseen individuals.

1.3 Null Class Dominance

The null class (label 0), this is the class that includes all activity segments that do not correspond to one of the 18 defined workout classes (labels 1-18), presents a significant challenge. As shown in Figure 2, the null class appears notably more often than any individual activity class. The decoded activities are the following: null: 0, variations of jogging: 1-5, variations of stretching: 6-10, variations of push-ups: 11, 12, variations of sit-ups: 13, 14, burpees: 15, variations of lunges: 16, 17, and bench-dips: 18.

2 Data Processing and Feature Extraction

2.1 Preprocessing of Training Data

2.1.1 Generation of One-Second Samples. As previously described, the nature of the training data differs significantly from that of the test data. To make the two datasets compatible, we preprocess the training data by segmenting it into one-second intervals. These intervals are extracted from the continuous activity recordings of each subject. Each second contains measurements of 3D accelerometer data (a^x, a^y, a^z).

Importantly, each sensor location (e.g., left arm, right leg) is treated independently to meet the requirements of the challenge data format. This means that from a single one-second time segment across all four sensor positions, we obtain four separate training samples, one for each location.

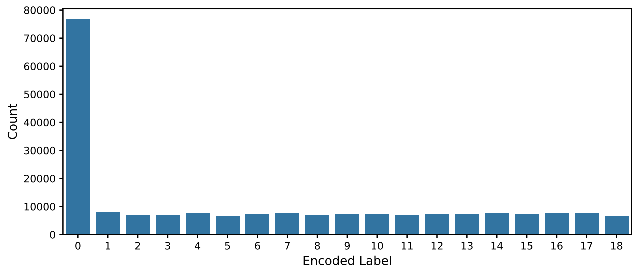


Figure 2: Distribution of class labels in the processed training set. Label 0 corresponds to the null class, while labels 1 to 18 represent the workout activities. The null class is assigned to data segments not belonging to any of the predefined activities.

2.1.2 Subject-Based Train-Test Split. To evaluate model generalization, we perform a subject-based train-validation-test split. Given that the training set includes subject IDs, we allocate approximately 70% of the 22 individuals to the training set, and the remaining 30% are evenly split between validation and test sets. Since each subject contributes a similar amount of data, this results in a roughly 70/15/15 split of the overall one-second samples. This setup mimics the challenge condition where the model must generalize to unseen participants, which is crucial, as the execution of activities often varies with individual body characteristics, habits, and preferences.

2.2 Feature Engineering

Our approach relies on engineered features derived from the raw accelerometer data. Each sample consists of 50 time steps across the three axes, resulting in $3 \cdot 50$ measurements per sample. From this data, we extract features that summarize key statistical and frequency-related characteristics of the signal. We calculate features as suggested by last year's challenge winners, Van Der Donckt et al. [7] and extend them with other common statistical features. We first examine single-axis features, followed by multi-axis features.

2.2.1 Single-Axis Features. We compute features separately for each of the three axes (x, y, and z). These features are categorized into two groups. The first are time domain features which describe the statistical properties of the signal over time. The second are frequency domain features which capture the periodic characteristics of the signal by applying a Fourier transform. This is particularly useful for activity recognition, as many physical activities exhibit repetitive patterns.

Let $a_n^j \in \mathbb{R}$ denote the measurement at time step $n \in \{1, \ldots, 50\}$ on axis $j \in \{x, y, z\}$. The full time series for axis j is denoted as $a^j := (a_1^j, \ldots, a_{50}^j)$.

2.2.2 Multi-axis Features. Next, we derive the multi-axis features by combining information across multiple sensor axes to compute a single feature. We define the Signal Magnitude Vector (SMV) as $smv_n := \sqrt{(a_n^x)^2 + (a_n^y)^2 + (a_n^z)^2}$ with $smv = (smv_n)_{n \in \{1,\ldots,N\}}$, which serves as the basis for several features used. In addition, we compute other statistical and orientation-based features such as the tilt angle derived from [1]. We list the computed features in the supplementary material.[1] Based on these features, we aim to predict the correct activity labels using the models introduced in the following section.

3 Feature-based Model

3.1 Classification Model

We evaluated several gradient-boosted decision tree (GBDT) models, which are particularly effective for tabular data due to their ability to handle heterogeneous feature types and complex interactions. The models considered include LightGBM [5], known for its efficiency, GPBoost [6], which extends GBDTs with Gaussian process components to capture structured dependencies. In addition, we included TabM [3], an MLP-based deep learning architecture tailored for tabular data that admits unbalanced data. Among these, LightGBM delivered the most effective and consistent performance under our configurations and tuning efforts, and is therefore used to report the majority of results in the following sections.

Importantly, one single model is trained jointly across all sensor locations, rather than training distinct models for each location.

3.2 Ensemble Model

Building on the strengths of the individual classifiers, we employed an ensemble of M independently trained models to further improve predictive performance and robustness. For each one-second test sample, each model outputs class probabilities $(p_i^m)_{i\in\{0,\dots,18\}}$, where p_i^m denotes the predicted probability that the true label corresponds to class i according to model m. Now for our ensemble method, we choose the final predicted label i^* to be obtained via majority soft voting over the highest label probability

$$i^* := \underset{i\in\{0,\dots,18\}}{\arg\max}\ (p_i^m)_{m\in\{1,\dots,M\}}.$$

We aim to profit from the individual strengths of the models through this ensemble method.

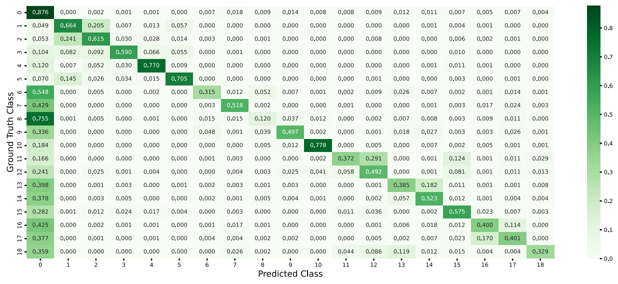
4 Mitigating the Null Class Imbalance

The class imbalance caused by the dominance of the null class affects predictive performance. To illustrate this effect, we present preliminary experimental results, in the form of confusion matrices generated by a LightGBM model, in this section. These initial results reveal that the imbalance causes a high number of misclassifications, where the model incorrectly predicts the null class even though the ground truth corresponds to one of the 18 defined activities (see Figure 3(a)).

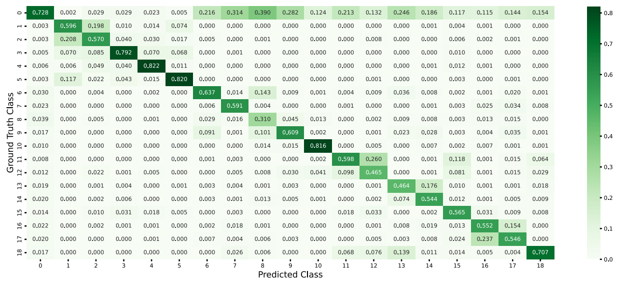
In this section, we explore three strategies to mitigate the impact of the dataset's class imbalance.

The first strategy, presented in Section 4.1, is based on the hypothesis that, unlike activity classes 1 to 18, the null class lacks a consistent behavioral pattern. It likely contains a broad range of motions, which makes it more challenging for the model to learn a clear and reliable representation. Hence, we omit the null class entirely from the training set and then apply a confidence-based projection onto the null class.

[1]Supplementary information about the computed features is available at https://github.com/rilink/HARMA.



(a)



(b)

Figure 3: Confusion matrices of the same LightGBM model under two different normalization schemes where the rows represent the ground truth classes and the columns represent the predicted classes. (a) Row-normalized: each row sums to one, showing recall per class. (b) Column-normalized: each column sums to one, illustrating precision per predicted class. The decoded class labels can be revisited in Section 1.3.

In the second strategy, in Section 4.2, we acknowledge that the null class may contain some underlying structure. However, we aim to limit the null class's influence on predictions by introducing an adjusted undersampling technique.

The third strategy, in Section 4.3, again involves a confidence-based thresholding approach, allowing the model to selectively project predictions onto or away from the null class. Now the null class itself was involved in training.

4.1 Leaving out the Null Class from Training and Thresholding

Due to the hypothesized lack of a learnable pattern in the null class, in this strategy we exclude samples labeled as null entirely from the training set. Consequently, we analyze the predicted class probabilities and apply confidence-based thresholding. Specifically, if the maximum predicted class probability falls below a predefined threshold, we project the prediction onto the null class label. This projection is formalized by $\pi_{nn} : [0,1]^{18} \rightarrow \{0,1,\dots,18\}$, where $th_i \in [0,1]$ for $i \in \{0,1,\dots,18\}$ and $th_i = th_j$ for $i,j \in \{1,\dots,18\}$, denoting a confidence threshold. Since the 18 activity classes occur with similar frequencies across the training samples, we choose to assign the same threshold for all classes to reduce computational costs during threshold optimization. For each model $m \in \{1,\dots,M\}$, we define the projection by

$$\pi_{nn} : (p_i^m)_{i\in\{1,\ldots,18\}} \mapsto i^*_{nn} \cdot \mathbf{1}_{[th_{i^*nn},1]}(p^m_{i^*_{nn}})$$
$$\text{with } i^*_{nn} := \underset{i\in\{1,\ldots,18\}}{\arg\max}\ (p_i^m),$$

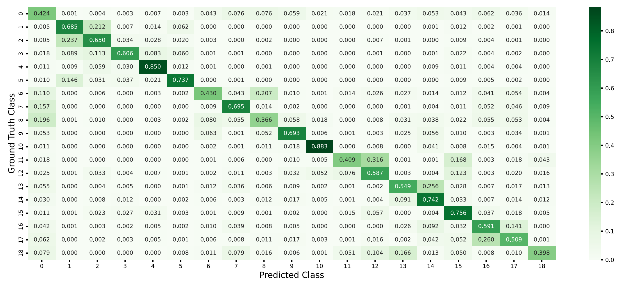
where the subscript "*nn*" indicates the non-null setting, as the null class is absent from both the training data and the predicted outputs. The null class is hence assigned if the above indicator function evaluates to zero, which is the case when no probability $(p_i^m)_{i\in\{1,\ldots,18\}}$ passes the threshold, leading to $\pi_{nn}((p_i^m)_{i\in\{1,\ldots,18\}}) = i^*_{nn} \cdot 0 = 0$.

However, in practice, the null class might indeed contain structured patterns, such as repeating transitions between activities, like getting up or laying down for the next activity, drinking something, or resting phases in general. This observation suggests that excluding the null class entirely discards important information, therefore, the inclusion of the null class in the training set is reconsidered in the next strategy.

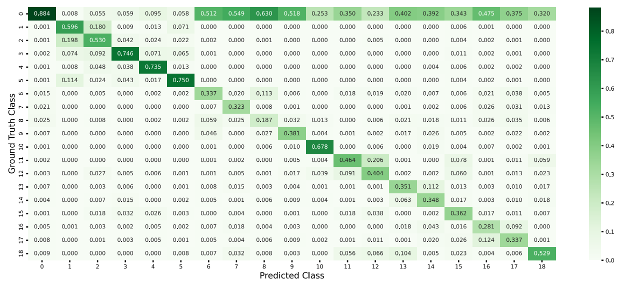
4.2 Undersampled Null Class

Instead of removing the null class, we reintroduce it into the training set. To mitigate the imbalance illustrated in Figure 3(a), we apply undersampling to the majority class, the null class. This adjustment aims to reduce the number of activity samples that are incorrectly classified as null.

To determine an appropriate undersampling amount, we compute the average number of samples per class, denoted as $\mu_{\{1,\ldots,18\}}$, over all activity classes $i \in \{1,\ldots,18\}$. We then randomly sample $\mu_{\{1,\ldots,18\}}$ instances from the null class to match this average. This helps balance the training data and is supposed to reduce bias towards the null class.



(a)



(b)

Figure 4: Confusion matrices of the undersampled LightGBM model where the rows represent the ground truth classes and the columns represent the predicted classes. (a) Row-normalized and (b) Column-normalized.

After applying undersampling, the null class no longer dominates the predictions, as seen in the row-normalized confusion matrix in Figure 4(a), effectively reducing the mislabeling of activity samples as null. However, the column-normalized matrix in Figure 4(b) reveals that many actual null class samples are now being confused with activity classes, which can be seen by the more heavily coloured first row, indicating a trade-off introduced by undersampling.

To balance these opposing effects, we introduce an adjustable undersampling method. Instead of undersampling the null class down to the mean class size $\mu_{\{1,\ldots,18\}}$, we define an adjusted undersampling size by adding a tunable hyperparameter adj $\in \mathbb{N}$ to obtain $\mu^{\text{adj}}_{\{1,\ldots,18\}} = \mu_{\{1,\ldots,18\}} + \text{adj}$ which is the adjusted amount of samples we draw from the null class. This provides a way to optimize model performance by finding the ideal trade-off between the two types of misclassification.

4.3 Null Class Thresholding

To address the null class imbalance further, we again introduce a threshold-based method to improve predictive performance. Similar to the approach in Subsection 4.1 we apply confidence-based thresholding. We define the thresholds

$$(th_0, th_1, \ldots, th_{18}) \in [0,1]^{19} \quad \text{with } th_i = th_j \text{ for } i,j \in \{1,\ldots,18\}.$$

Again, since the 18 activity classes occur with similar frequencies across the samples, we choose to assign the same threshold for all classes similar to Section 4.1. The threshold for the null class, th_0, is treated separately and may differ from the shared threshold.

For any model $m \in \{1,\ldots M\}$, we define

$$\pi : (p_i)_{i\in\{0,\ldots,18\}} \mapsto i^* \cdot \mathbf{1}_{[th_{i^*},1]}(p^m_{i^*}) + i^*_{nn} \cdot \mathbf{1}_{[0,th_{i^*})}(p^m_{i^*}) \cdot \mathbf{1}_{[th_{i^*_{nn}},1]}(p^m_{i^*_{nn}})$$

where we again have $i^* = \underset{i\in\{0,\ldots,18\}}{\arg\max}\ (p_i)$ and $i^*_{nn} = \underset{i\in\{1,\ldots,18\}}{\arg\max}\ (p_i)$ for which $i^*_{nn} = i^*$ if $i^* \neq 0$.

Intuitively, if the highest predicted probability p_i corresponds to the null class and exceeds its threshold th_0, the predicted label remains the null class, as expected. However, if the null class has the highest confidence but its value does not exceed th_0, we instead consider the next most confident class i^*_{nn}, excluding the null class, and assign it as the predicted label, provided its confidence exceeds its corresponding threshold $th_{i^*_{nn}}$.

Conversely, if the highest p_i corresponds to any of the activity classes besides the null class, and exceeds its threshold th_i, then i is selected as the predicted label. If the confidence is below the threshold, we do not select the second-best activity class, since it will also be below its threshold. Finally, if no class probability exceeds its respective threshold, the null class is assigned as the predicted label.

This method aims to further reduce the two effects visited in Figures 3(a) and 4(b).

5 Results and Discussion

The classification task on the WEAR dataset is challenged by several factors, with the dominance and ambiguity of the null class being a one of them. To address this, we implemented and evaluated a combination of preprocessing and postprocessing strategies described in the previous section. Next, we report preliminary results on the test set, which are discussed in more detail throughout this section. A simple baseline is established by training a LightGBM model on the full dataset, including all classes. To predict the challenge data, we train the model on the whole training set and derive hyperparameters such as the undersampling adjustment amount using 6-fold cross-validation. This means we trained on all subjects and all one-second sequences available. Our final submission to the Challenge is 'target.csv' where the ensemble method with adjusted undersampling was used.

Table 1: Test set results. Results (0)–(4) were obtained with the best-performing individual model: LightGBM, while (5) represents the ensemble of the models LightGBM, TabM and GPBoost.

	Strategy	Macro F1 score
(0)	Baseline	0.573
(1)	No null + Threshold	0.540
(2)	Adj. Undersampling	0.585
(3)	Thresholding	0.584
(4)	Combination of (2) and (3)	0.585
(5)	Ensemble method under (2)	0.594

5.1 Leaving out the Null Class

The strategy suggestion in Section 4.1 did not yield an improvement over retaining the null class during training, as seen in Table 1 (1). Removing the null class labeled samples from training led to a reduced macro F1 score of 48.6%, where only non-null labels were predicted. Incorporating the discussed thresholding and optimizing the threshold th_i for $i \in \{1, \ldots, 18\}$ on a separate validation set with unseen subjects achieved a macro F1 score on the test set of 54%, still falling short of the baseline with the null class included. An example of this optimization process is illustrated in Figure 5. These results challenge the assumption that the null class lacks consistent patterns, suggesting that it may in fact exhibit some regularity.

5.2 Adjusted Undersampling

For the undersampling approach, a range of undersampling levels was tested. As shown in Figure 6, performance, measured by the macro F1 score, peaked at an adjustment of 20,000 samples of the null class, with a cross-validated improvement of 1.2% (Table 1 (2)). This corresponds to an adjusted number of null class samples that is approximately four times greater than the number of samples in the other classes. However, fully undersampling the null class down to the mean size of classes 1 to 18 led to worse performance than the baseline with no undersampling, where the null class remained at its original size (shown in green). Analysis of the confusion matrices showed that while full undersampling reduced false positives for the null class, it also increased false negatives, meaning more actual null class samples were misclassified as activity classes. Given the high

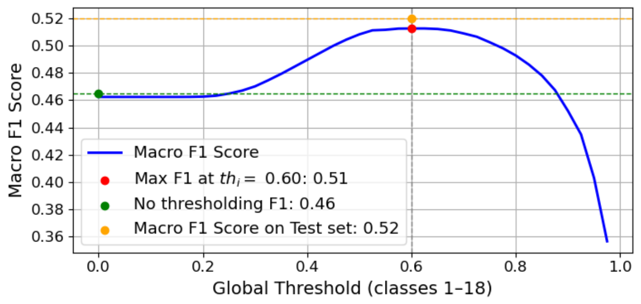


Figure 5: Effect of the optimized global threshold th_i for one exemplary seed. The green marker shows the macro F1 score without the null class or thresholding. The blue line shows validation scores across thresholds, with the red marker indicating the maximum. The orange marker shows the test score using the optimized threshold.

frequency of null class samples, this effect has a greater negative impact in the trade-off discussed in Section 4.2. As a result, the overall macro F1 score with full undersampling is lower than in the case without undersampling, as illustrated in Figure 6.

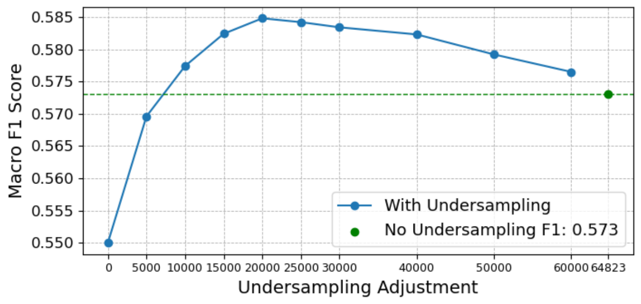


Figure 6: Macro F1 scores for different levels of undersampling applied to the null class, ranging from $\mu_{\{1,\ldots,18\}} + 0$ to $\mu_{\{1,\ldots,18\}} + 60,000$ samples. Best performance is observed at an undersampling level of $\mu_{\{1,\ldots,18\}} + 20,000$ samples. Results are based on cross-validation.

5.3 Confidence-Based Thresholding

As a next strategy, we introduced confidence-based thresholding. The thresholds were optimized on the validation sets with unseen subjects, and the resulting cross-validated performance was evaluated on the test sets. The thresholding approach yields a 1.1% improvement in macro F1 score under cross-validation Table 1 (3).

5.4 Combination of Undersampling and Thresholding

The next logical step was to combine adjusted undersampling with confidence-based thresholding. However, this approach resulted in no further improvement Table 1 (4), indicating that both strategies likely address the same underlying challenges in the data. While undersampling slightly outperforms thresholding when used alone,

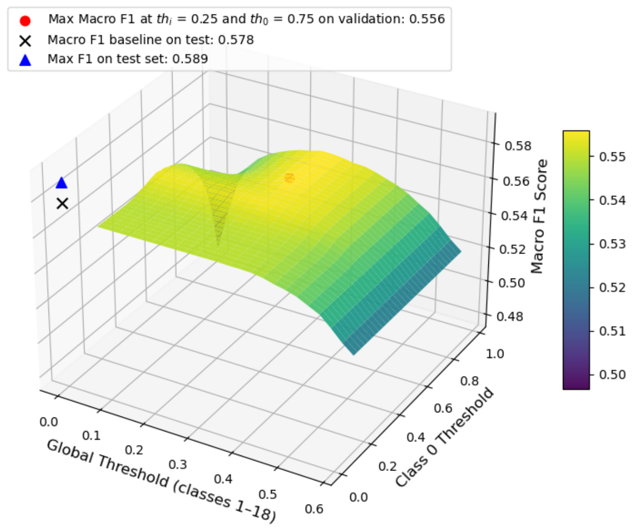


Figure 7: Effect of optimized global threshold th_i for $i \in \{1, \ldots, 18\}$ and null class threshold th_0 for one exemplary seed. The surface shows validation macro F1 scores across thresholds. The red marker indicates the optimal threshold combination, the blue triangle shows the corresponding test set score, the black cross represents performance without thresholding.

the thresholding approach offers a practical advantage, it requires only a single model training since it is a postprocessing measure. In contrast, optimizing undersampling requires retraining the model multiple times since it is a preprocessing measure.

5.5 Ensemble Model Impact

The ensemble approach outperformed individual models (Table 1 (5)). In our ensemble approach, we used adjusted undersampling of three models: LightGBM, TabM and GPBoost and obtained an improvement of 0.9% compared to the LightGBM model with adjusted undersampling alone.

6 Conclusion

The null class in the WEAR dataset introduces a major challenge due to its size and lack of consistency. Basic undersampling alone was not effective, as it shifted misclassifications rather than resolving them. Adjusted undersampling offered a more effective solution by better balancing the class distribution. Confidence-based thresholding approaches also provide a way to improve results when applied independently. However, when adjusted undersampling was already in place, threshold optimization did not yield additional benefits. This suggests that the class imbalance had already been effectively mitigated. Therefore, for our challenge submission, we opted to use only adjusted undersampling. The use of an ensemble model further stabilized predictions. These strategies, to some extent, mitigated the impact of the null class and enhanced overall performance in multi-class activity recognition.

Acknowledgments

The authors acknowledge support by the state of Baden-Württemberg through bwHPC and the German Research Foundation (DFG) through grant INST 35/1597-1 FUGG.

References

[1] Solaiman Ahmed, Tanveer Ahmed Bhuiyan, Taiki Kishi, Manabu Nii, and Syoji Kobashi. 2021. Human activity classification based on angle variance analysis utilizing the Poincare plot. *Applied Sciences* 11, 16 (2021), 7230.

[2] Marius Bock, Hilde Kuehne, Kristof Van Laerhoven, and Michael Moeller. 2024. Wear: An outdoor sports dataset for wearable and egocentric activity recognition. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 8, 4 (2024), 1–21.

[3] Yury Gorishniy, Akim Kotelnikov, and Artem Babenko. 2025. TabM: Advancing Tabular Deep Learning With Parameter-Efficient Ensembling. In *Proceedings of the International Conference on Learning Representations (ICLR).* International Conference on Learning Representations, Singapore EXPO, Singapore, n/a.

[4] Michail Kaseris, Ioannis Kostavelis, and Sotiris Malassiotis. 2024. A comprehensive survey on deep learning methods in human activity recognition. *Machine Learning and Knowledge Extraction* 6, 2 (2024), 842–876.

[5] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems* 30 (2017), n/a.

[6] Fabio Sigrist. 2022. Gaussian process boosting. *Journal of Machine Learning Research* 23, 232 (2022), 1–46.

[7] Jonas Van Der Donckt, Jeroen Van Der Donckt, and Sofie Van Hoecke. 2024. Left-Right Swapping and Upper-Lower Limb Pairing for Robust Multi-Wearable Workout Activity Detection. In *Companion of the 2024 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp/ISWC).* ACM, Melbourne, Australia, 545–550.