Shot Classification in Padel Using Classical Machine Learning on Positional Patterns Only

1st Jan Niklas Schulenburg
TH Köln – University of Applied Sciences
Cologne, Germany

2nd Marcel Max Bizon
TH Köln – University of Applied Sciences
Cologne, Germany

***Abstract*—Padel is one of the fastest-growing sports worldwide, yet automated performance analysis remains underexplored due to a lack of annotated datasets. This paper investigates shot classification in padel using only sequential player and ball position data from the PadelTracker100 dataset, relying exclusively on classical machine learning models. Two segmentation strategies are compared: overlapping sliding windows and non-overlapping windows aligned with individual shot events. Sliding windows yield more training instances but lead to severe overfitting and inflated cross-validation estimates that do not transfer to the test set. Non-overlapping windows, despite far fewer instances, generalize considerably more reliably. A CatBoost classifier using this strategy achieves a macro F1-score of 0.615 across five shot categories.**

***Index Terms*—padel, shot classification, classical machine learning, sliding window, sequential segmentation, position data**

I. Introduction

Padel is currently considered a rapidly growing sport worldwide [1]. Given its rapid growth, the demand for media coverage and data-driven match analysis is rising accordingly, yet automated analysis for this sport remained largely unexplored with limited data available [2], [3]. The recently introduced PadelTracker100 [1] dataset represents a significant step forward, providing comprehensive frame-level annotations derived from computer vision pipelines, including ball bounding-boxes, real-world player positions, pose estimations, and shot event labels for professional match footage [1]. While the dataset authors target live match analysis through computer vision and deep learning, this paper takes a different direction, as it makes use of the annotated positional and shot event data only, while applying classical machine learning techniques.

Specifically, the paper investigates whether shot types in padel can be classified from sequential player and ball position patterns alone, deliberately excluding pose estimations. To answer this question, statistical features are extracted from position sequences and are used to train and evaluate classical classifiers on the different shot categories. As the number of annotated shot events in the dataset is relatively small [1], it is additionally examined whether increasing the number of training instances through overlapping sliding window segmentation yields genuine performance gains over fixed, non-overlapping sequential windows, as overlapping windows can be prone to overfitting without actually improving performance, despite increasing the number of training instances [4].

II. Related Work

The automated analysis of racket sports has gained increasing research attention in recent years [5]. Studies on tennis and badminton have explored shot classification through a variety of modalities, including wearable inertial sensors, pose estimation, and ball trajectory data, mainly relying on deep or machine learning architectures [5]–[7]. In padel specifically, research has been significantly hindered by the lack of annotated datasets. Domínguez et al. addressed this gap by creating the first padel shot dataset, collecting IMU sensor data from 12 players performing 13 different shot types using a wristband device, and demonstrated that both classical machine learning and deep learning algorithms can classify padel shots from wearable sensor data [2]. However, no video-based annotated dataset for padel existed until the recent introduction of PadelTracker100, which provides the first large-scale frame-level annotations for ball detections, real-world player positions, pose estimations, and shot events derived from professional match footage [1].

As the problem of classifying shots from sequential position data is closely related to human activity recognition, the methodological approach gets oriented accordingly. Baldominos et al. demonstrate that classical machine learning techniques with hand-crafted features remain competitive with deep learning on structured sensor data for human activity recogntion, and can even outperform deep learning architectures in comparable settings [8]. This is especially true for limited amount of (labeled) data, as deep learning solutions tend to need a larger amount of training data [9]. Given the relatively small number of annotated shot events in the PadelTracker100 dataset [1], classical machine learning is therefore the more suitable choice for this work.

This paper, therefore, applies classical machine learning models with extracted statistical features, following the established activity recognition pipeline consisting of data acquisition, preprocessing, segmentation, feature extraction, and classification [10]. As part of this process, the sequential position data needs to be flattened into feature vectors [8]. The literature shows different approaches for constructing input sequences in human activity recognition, which mainly differ along two dimensions: window length (variable, bound to the duration of each shot, versus fixed) and overlap (non-overlapping versus overlapping) [4]. In this project, variable-

length, non-overlapping windows are used as the baseline segmentation, since each window directly corresponds to one annotated shot. Additionally, fixed-length overlapping sliding windows are evaluated to investigate whether the resulting increase in training instances yields a genuine improvement in classification performance or just results in more overfitting [4].

III. Dataset

This work uses the PadelTracker100 dataset [1], comprising two professional padel matches from the 2022 World Padel Tour Finals in Barcelona, recorded at 1920×1080 and 30 fps, totaling nearly 100,000 frames (≈55 minutes of footage) [1]. When viewing the dataset's footage, it becomes clear that breaks between points have been cut, meaning that the increasing frame count does not correspond to a linear passing of time.

Ball positions and player poses were annotated semi-automatically using iterative YOLOv8 training and a ViTPose-L model, followed by manual refinement [1]. Player court positions were derived via homography transformation of the detected keypoints into real-world coordinates [1]. Shot events, in contrast, were annotated entirely manually, labelling all frames in a neighborhood around each shot as defined by the racket movement of the performing player, resulting in 40,135 annotated frames across both matches [1]. Therefore, only around 40,000 of the original frames in the dataset are usable for this task.

Each of these frames is labelled with one of six shot categories (backhand, forehand, smash, serve, dropshot, other) or as containing no shot. The class distribution is highly imbalanced, because dropshot is severely underrepresented with only 25 and 66 instances in the women's and men's final [1].

IV. Data Preparation

Following the activity recognition pipeline outlined in Section II [10], this section covers the acquisition and preprocessing steps prior to segmentation and feature extraction.

The ball, homography, and shot annotation JSON files (partly in COCO-Format) were merged into a single DataFrame on a per-frame basis using the frame file name and coressponding IDs in the JSON Files as the join key, resulting in 99,822 frames across both matches. Since only 40,135 frames contain shot annotations and frames without a shot label are not usable for this task, all unlabelled frames were dropped.

The dataset was split into training and test sets using a chronological 80/20 split applied independently to each match, preserving the order of frames. This approach has been selected as it minimizes data leakage between the train and test set.

Plotting the player coordinates as a heatmap and as positions over frames revealed that the player identity assignment in the homography annotations is inconsistent, with player indices switching every few frames, even within the same rally. As a result, the original player columns could not be used to track individual players over time and were discarded. Instead, players were reassigned at each frame based on their court position, grouping them into four quadrants (top-left, top-right, bottom-left, bottom-right) according to their distance from the court's center on both axes. This reassignment is consistent with the structure of padel, where each team occupies one half of the court. If three players are assigned to the same half due to a player crossing over near the net, the player closest to the net within that group is reassigned to the opposing team's side, ensuring that each team consists of exactly two players.

Ball position data is missing for a substantial portion of frames, probably due to occlusion or breaks between points. Across 216 missing ball sequences in the training set, the median gap length was 5 frames, while the mean was considerably higher at 13.42 frames (max: 52), indicating a right-skewed distribution with many short gaps and a smaller number of much longer ones. Missing values were therefore handled differently depending on the length of the gap: gaps of five frames or fewer were linearly interpolated, as they likely correspond to brief occlusions, while longer gaps were dropped, as they are more likely to result from cuts in play rather than missing detections. Notably, missing ball values are more concentrated in frames without a shot (12.7% of frames missing a ball annotation), while all other shot categories are affected only marginally (around 1–2% each), indicating that dropping long gaps has little impact on the classes most relevant to this task.

Additionally pixel-based player coordinates were dropped in favor of the homography-projected real-world coordinates, to better generalizability across different camera setups in future studies. The small number of remaining missing player position values was handled using iterative multivariate imputation.

V. Model Development

As the raw, time-dependent position data is not directly suitable for classical machine learning models, it is segmented into windows, from which statistical features are extracted to obtain a flattened, fixed-length representation [8], [10]. Two segmentation strategies are compared throughout this process, and the resulting feature sets are used to train and evaluate a series of classifiers, leading to the final model.

A. Segmentation

Two segmentation strategies are compared. The first uses fixed-length, overlapping sliding windows of 20 frames with a step size of one frame, where the label of each window corresponds to the shot category of the last frame. The window size of 20 frames was chosen as it is slightly higher than the mean length of a shot event. To avoid windows spanning across discontinuities introduced by previously dropped frames, windows containing non-consecutive frame IDs are discarded. However, since the original match footage itself already contains cuts (e.g., between points), this check cannot guarantee that all windows are free of scene cuts or changes in play. This strategy was applied first, as it does not depend on

the length of individual shot events and yields a substantially larger number of training instances. Applying this segmentation to the training set results in 27,951 windows.

The second strategy uses variable-length, non-overlapping windows, where each window combines one annotated shot event with the immediately preceding sequence of no-shot frames without a change in frame IDs greater than 3. This results in 731 windows in the training set and 173 in the test set, considerably fewer than the sliding window approach. Due to this substantially smaller number of training instances, this strategy is evaluated separately as a comparison after the initial sliding window-based models have been established.

The class distributions of both strategies differ structurally. In the sliding window setting, no-shot windows make up the majority of all training windows (over 17,500 of 27,951, roughly 63%), substantially outnumbering even the most frequent shot category (forehand with around 2,500). This imbalance arises because every frame between points produces its own window. In the sequential setting, no-shot frames are never assigned their own class, but are only included as context preceding a shot. The resulting imbalance is therefore limited to the relative frequency of the six shot categories themselves, where dropshot is the most underrepresented class with only 3 training instances, as theire is only a small amount of dropshot-frames in the dataset [1].

B. Feature Extraction

For each window, regardless of segmentation strategy, a fixed-length feature vector is extracted to flatten the sequential position data, in line with the established activity recognition pipeline [10]. For the ball, the start, end, mean, and standard deviation of both x and y coordinates are computed, along with the mean and maximum of their frame-to-frame gradients, the net displacement (delta) between the first and last frame, and the mean and maximum of the ball's bounding box area. For each of the four court-quadrant player positions, the same statistics (mean, standard deviation, start, end, delta, and gradient mean/max) are computed for both coordinate axes. This results in a 72-dimensional feature vector per window. All features are standardized using a StandardScaler fitted on the training set, and shot category labels are encoded numerically.

C. Initial Model Comparison

Three classical classifiers were trained on the sliding window features with no-shot windows and dropshot included: a linear model (SGD), a single decision tree, and an ensemble method (Random Forest). All models were trained using balanced class weights to account for the highly skewed shot distribution and got evaluated using 5-fold cross-validation on the training set. Random Forest achieved the highest mean accuracy (0.668), followed by Decision Tree (0.542) and SGD (0.433). However, comparing training and cross-validation macro F1-scores already revealed substantial overfitting for the tree-based models. Decision Tree and Random Forest both reached a training F1-score of 1.000, while their cross-validation F1-scores dropped to 0.290 and 0.270. SGD showed a similarly large, though smaller, gap (train: 0.346, CV: 0.294). All reported F1-scores are macro-averaged across classes. Balanced weighting during training was used to address the same class imbalance from different angles: balanced class weights prevent the majority class from dominating the training objective, while macro-averaged F1-scores ensure that performance on minority classes is not masked by strong performance on frequent ones during evaluation.

These differences in accuracy and F1 score are most likely rooted in the unbalanced shot distribution. Given the dominant no-shot class, no-shot windows were subsequently removed, restricting the task to the six shot categories. This substantially improved performance for all models (e.g., Random Forest: train F1 1.000, CV F1 0.571), though the gap between training and cross-validation performance remained large. As an additional comparison, a feedforward neural network (MLP) was trained on the same data, showing a similar pattern (train F1: 0.996, CV F1: 0.517), indicating that the overfitting was not specific to tree-based models. All confusion matrizes showed high confusion across all classes.

To address this overfitting, hyperparameter tuning via grid search was performed for both Random Forest and the MLP. For Random Forest, tuning reduced the training F1-score to 0.918 while increasing the cross-validation F1-score to 0.601, indicating a better-generalizing model. For the MLP, despite tuning, the training F1-score remained at 0.992–0.996 with only a modest cross-validation improvement to 0.548–0.560, suggesting that overfitting remained regardless of regularization. Confusion matrices on the validation predictions showed a dominant diagonal for all classes except dropshot, but still a substantial number of misclassifications, particularly between forehand and backhand. Based on these results, Random Forest was selected as the primary model going forward.

D. Refinement

To address the persistent forehand/backhand confusion observed in the confusion matrices, impact features were introduced, capturing the player and ball positions at the moment of impact (the frame where the shot is first registered), along with the distance between the ball and each player at that moment. Repeating the grid search with these additional features improved cross-validation performance (CV F1: 0.618, train F1: 0.998), and feature importance analysis confirmed that several impact-related features ranked among the most informative. However, evaluating this model on the test set revealed a substantial drop in performance (test F1: 0.481), well below the cross-validation estimate, indicating that the cross-validation score itself was overly optimistic due to the large number of highly correlated, overlapping windows distributed across folds. Test-set results were used only to check the cross-validation estimates for overfitting, not as a model selection criterion, and were therefore consulted only late in the pipeline for the respective final models.

To address this, the number of training windows was reduced by increasing the effective step size, retaining only every tenth window. Repeating the grid search on this re-

duced set lowered both the training F1-score (0.861) and the cross-validation F1-score (0.507), bringing them closer together and indicating reduced overfitting, although overall test performance did not substantially improve (test F1: 0.461). Since forehand and backhand remained difficult to distinguish, features representing the ball's position relative to each player at impact were additionally introduced. As the dataset does not indicate which player is left- or right-handed, or which side of the body is used to strike, these features did not improve performance and were ultimately discarded. CatBoost was also evaluated as an alternative ensemble method on the reduced sliding window data, achieving a comparable cross-validation F1-score (0.538) and test F1-score (0.435) to the tuned Random Forest.

As none of these adjustments meaningfully improved generalization on the sliding window approach, the focus shifted to the variable-length sequential windows. Using the standard feature set, Random Forest and CatBoost achieved cross-validation F1-scores of 0.479 and 0.512. Adding the impact features introduced earlier did not meaningfully change cross-validation performance (Random Forest: CV F1 0.489; CatBoost: CV F1 0.494). Since cross-validation on sequential windows shall not be affected by the leakage between overlapping training instances that distorted the sliding window estimates, the test set was consulted at this point as a verification rather than as a primary selection criterion. Test F1-scores closely matched the cross-validation estimates (impact features: 0.500 and 0.507), confirming that the close train-CV gap reflects better generalization. Given the very small number of dropshot instances in the sequential windows (3 in training), this class was subsequently dropped, which further improved performance for both Random Forest (CV F1: 0.592) and CatBoost (CV F1: 0.609).

VI. Final Results

Based on these Findings, CatBoost trained on sequential windows with dropshot removed is selected as the final model, given its consistently strong performance and the more reliable correspondence between cross-validation and test performance observed for this segmentation strategy. Table I summarizes its training, cross-validation, and test performance.

TABLE I
FINAL MODEL PERFORMANCE (CATBOOST, SEQUENTIAL WINDOWS, DROPSHOT REMOVED)

	Train F1	CV F1	Test F1
CatBoost	0.979	0.609	0.615

The close correspondence between the cross-validation and test F1-scores (0.609 vs. 0.615) indicates that the model generalizes reliably to unseen data, in contrast to the sliding window approach discussed earlier, where cross-validation substantially overestimated test performance.

Table II shows the per-class test performance. Smash and serve are classified most reliably (F1: 0.86 each), while backhand and forehand remain the most frequently confused pair (F1: 0.57 and 0.63), consistent with the missing handedness information discussed earlier. The other category, which aggregates shots that do not fit the remaining classes, performs worst (F1: 0.16), likely due to its heterogeneous nature. Testing different confidence thresholds, using the other class as a fallback for low-confidence predictions, did not improve this.

TABLE II
PER-CLASS TEST RESULTS, FINAL MODEL

Class	Precision	Recall	F1
Backhand	0.46	0.75	0.57
Forehand	0.75	0.55	0.63
Other	0.40	0.10	0.16
Serve	0.92	0.80	0.86
Smash	0.80	0.93	0.86

TABLE III
CONFUSION MATRIX, FINAL MODEL (CATBOOST, SEQUENTIAL WINDOWS, DROPSHOT REMOVED), TEST SET

	Predicted				
True	Backhand	Forehand	Other	Serve	Smash
Backhand	30	5	3	0	2
Forehand	23	30	0	0	2
Other	11	4	2	0	3
Serve	0	0	0	12	3
Smash	1	1	0	1	39

The confusion matrix (Table III) reveals that the majority of backhand-forehand confusion is asymmetric: 23 of 55 forehand instances are misclassified as backhand, while only 5 of 40 backhand instances are misclassified as forehand. This asymmetry also explains the near-mirrored precision and recall values for the two classes: backhand shows high recall but low precision (0.75 and 0.46, respectively), as it absorbs many misclassified forehand instances, while forehand shows the inverse pattern (precision: 0.75, recall: 0.55), losing many true instances to backhand while remaining precise on the ones it does predict correctly.

VII. Discussion and Conclusion

The results indicate that shot types in padel can be classified from sequential player and ball position patterns alone with moderate to strong reliability for some classes, but not uniformly across all shot types. Serve and smash, which involve distinctive positonal patterns, are classified reliably (F1: 0.86 each). Smash is typically associated with players positioned closer to the net, while serve follows a distinct positional pattern of no movement of the ball before the impact. Backhand and forehand, by contrast, are both predominantly played from the back of the court and exhibit much more similar positional patterns, which likely contributes to their frequent confusion (F1: 0.57 and 0.63). Without information on player handedness, which side of the body is used to strike, or pose data, positional data alone may not fully classify these two shot types. The other category performs worst (F1: 0.16), likely due

to its heterogeneous composition of shots that do not fit the remaining categories.

While overlapping sliding windows yield substantially more training instances, they led to severe overfitting and inflated cross-validation estimates that did not transfer to the test set. Variable-length, non-overlapping sequential windows, despite providing far fewer training instances, generalized considerably more reliably, with cross-validation and test performance closely aligned. This suggests that, at least for this dataset and task, the structural integrity of non-overlapping segmentation outweighs the benefit of a larger but highly correlated training set.

Several limitations should be considered when interpreting these results. The dataset comprises only two professional matches, which limits generalizability to amateur play and to padel more broadly. The severe underrepresentation of dropshot required excluding this class entirely. Furthermore, the lack of handedness information is a structural limitation of the positional data approach used in this work.

This paper presented a approach to shot classification in padel using only positional patterns derived from PadelTracker100, deliberately excluding pose estimation and relying exclusively on classical machine learning models. Future work could incorporate pose estimation or additional wrist-mounted sensor data to better distinguish backhand and forehand shots, extend the analysis to additional matches to improve generalizability, label more different shot types or explore whether the performance gap between segmentation strategies persists as more annotated data becomes available for deep learning approaches.

REFERENCES

[1] R. Bada-Nerin, P. Rodríguez-González, D. Kreibel, V. Teodoro, O. D. Pedrayes, R. Usamentiaga, Y. Fontenla-Seco, P. Ben-Leston, S. Rodriguez-Trillo, M. Jedrzejowski, N. Lozano-García, and F. Mosquera-Bonasorte, "Padeltracker100: A dataset for intelligent player and ball tracking in padel sports," *Data in Brief*, vol. 65, p. 112546, 2026. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352340926000995

[2] G. Domínguez, E. Álvarez, A. Córdoba, and D. Reina, "A comparative study of machine learning and deep learning algorithms for padel tennis shot classification," *Soft Computing*, vol. 27, pp. 12 367–12 385, September 2023.

[3] International Padel Federation (FIP), "Fip world padel report 2025," https://www.padelfip.com/world-padel-report/, 2025, accessed: 2026-06-17.

[4] A. Dehghani, O. Sarbishei, T. Glatard, and E. Shihab, "A quantitative comparison of overlapping and non-overlapping sliding windows for human activity recognition using inertial sensors," *Sensors*, vol. 19, no. 22, p. 5026, 2019.

[5] C. Arthur, R. Chiong, F. U. Din, F. Hajati, S. Chakraborty, D. Paul, M. Welch, and R. G. Crowther, "Making the world's fastest racket sport even better: A systematic review of artificial intelligence-based objective player performance assessment in badminton," Preprint, Research Square, March 2026, version 1.

[6] J. Lin, C.-W. Chang, T.-U. Ik, and Y.-C. Tseng, "Sensor-based badminton stroke classification by machine learning methods," in *2020 International Conference on Pervasive Artificial Intelligence (ICPAI)*, 2020, pp. 94–100.

[7] P. Selvaraju, R. Kumar, and M. Abdullah, "The classification of tennis strokes through machine learning," in *Proceedings of the 7th Asia Pacific Conference on Manufacturing Systems and 6th International Manufacturing Engineering Conference—Volume 2. iMEC-APCOMS 2024*, ser. Lecture Notes in Mechanical Engineering, M. Mohamad Yasin, Z. Ismail, C. Rosyidi, and M. Tokhi, Eds. Singapore: Springer, 2025.

[8] A. Baldominos, A. Cervantes, Y. Saez, and P. Isasi, "A comparison of machine learning and deep learning techniques for activity recognition using mobile devices," *Sensors*, vol. 19, no. 3, p. 521, January 2019.

[9] R. Liu, A. A. Ramli, H. Zhang, E. Henricson, and X. Liu, "An overview of human activity recognition using wearable sensors: Healthcare and artificial intelligence," in *Human Activity and Behavior Analysis*. Springer, 2021.

[10] A. Bulling, U. Blanke, and B. Schiele, "A tutorial on human activity recognition using body-worn inertial sensors," *ACM Computing Surveys*, vol. 46, no. 3, p. 33, 2014.