engineering proceedings

MDPI

Proceeding Paper

Embedded AI for Tennis Performance Analysis Using Racket-Mounted Inertial Sensors †

Timothy Malche

Department of Computer Applications, Manipal University Jaipur, Jaipur 303007, India; timothy.malche@jaipur.manipal.edu

† Presented at the 13th International Electronic Conference on Sensors and Applications, 18–20 November 2026; Available online: https://sciforum.net/event/ECSA-13.

Abstract

Detailed performance feedback in tennis mostly uses camera-based analysis tools and laboratory motion capture systems, which are expensive and require dedicated infrastructure. Wearable motion sensors provide a simple and practical solution. When combined with machine learning models running directly on a microcontroller, known as TinyML, they can classify stroke types locally without needing an internet or cloud connection. This paper proposes a racket-mounted embedded sensor system that classifies three tennis stroke types entirely on device. A six-axis IMU combining a three-axis accelerometer and a three-axis gyroscope was mounted on a racket handle. Data was obtained from the public TennisEye dataset, recorded at 100 Hz from seven players across three stroke classes including forehand, backhand, and serve. The dataset contains 1980 strokes, with each stroke represented by a one-second recording. A 1000 ms window was applied. A spectral block produced 144 features, which were provided to a neural network with hidden layers of 32, 16, and 8 units. The neural network was trained, quantized to 8-bit integers, and deployed on an Arduino Nicla Sense ME. Results were relayed via Bluetooth Low Energy to a Raspberry Pi gateway and sent to a cloud server using MQTT. The model achieved 90.4% classification accuracy based on a held-out test set of 396 strokes, with a macro F1 score of 0.891 across all three stroke classes. Quantization reduced accuracy by 1.3 percentage points compared with the float32 version of the model. The system used 9 ms per inference, 4.4 kB of RAM, and 19.5 kB of flash on a Cortex-M4F running at 80 MHz. Volley strokes were excluded because the dataset contained only 33 examples. A racket-mounted IMU running on-device TinyML inference delivers accurate stroke classification without cloud infrastructure. The platform demonstrates that tennis stroke recognition can be performed directly on a low-cost embedded device. Future directions include recording data on the target board, adding an idle class for continuous operation, and testing a convolutional model. Beyond classification, the same on-device signals were used to derive stroke-level performance indicators. Peak angular velocity correlated with the optically measured outgoing ball speed at rho = 0.75 for serves, 0.65 for forehands, and 0.54 for backhands and revealed both between-player differences in swing intensity and within-session trends consistent with fatigue and warm-up.

Keywords: TinyML; inertial measurement unit; tennis; stroke classification; embedded machine learning; edge computing; sports analytics; quantization

Check for updates

Academic Editor: Francisco Falcone

Published: 12 November 2026

Copyright: © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Useful feedback can help players improve their skills, but many existing sports-analysis tools are not practical for everyday use. Professional camera-based tracking systems provide detailed information about each shot, but they are expensive and require a fixed installation. Laboratory motion-capture systems offer precise measurements but are difficult to use on a tennis court. Consequently, many recreational players lack a simple way to record and review their training sessions.

Inertial sensors provide a more practical alternative. A six-axis inertial measurement unit (IMU) is inexpensive, energy efficient, and small enough to attach to a racket or wrist without interfering with play. Previous studies have shown that tennis strokes can be classified using inertial signals alone. However, systems that transmit raw sensor data to a phone or cloud service still depend on connectivity, consume additional power, and raise concerns about data storages.

TinyML enables machine-learning models to run directly on microcontroller-based devices. This allows stroke classification to occur on the racket, even without network access, while only session summaries need to be stored or transmitted. Recent studies have demonstrated tennis stroke classification on Arduino-class hardware, and open-source projects have provided complete racket-mounted TinyML implementations. However, reproducibility remains a challenge because many reported results use private datasets or provide insufficient implementation details, making it difficult to compare systems or reproduce their performance.

2. Related Work

2.1. Inertial Sensing for Tennis Stroke Analysis

Inertial tennis-analysis systems mainly use wrist- or racket-mounted sensors. Wrist-based studies have classified forehand, backhand, serve, and other stroke types using accelerometer and gyroscope features [1,2]. Whiteside et al. [3] achieved 93.2% accuracy across nine stroke types, while Wu et al. combined wrist sensing with an IoT system to assess player skill.

Racket-mounted sensors capture impact vibrations more directly. Pei et al. reported 98% shot detection and 96% stroke recognition, while Zhao et al. [4] classified five stroke types and estimated ball speed using the dataset adopted in this study. Direct comparisons found similar performance for wrist and racket placement [5], and Park et al. [6] achieved 88.2% accuracy using the non-dominant wrist, only 1.9 percentage points below the dominant wrist. Liu et al. integrated the sensor into a vibration dampener to estimate stroke type, ball speed, and impact location [7–10].

2.2. TinyML on Microcontroller-Class Hardware

In tennis, Dabbous et al. deployed a three-class convolutional model on an Arduino Nano 33 IoT with a 65 ms inference time, while open-source projects have provided complete racket-mounted TinyML pipelines. Microcontroller machine-learning tools now support efficient on-device inference. TensorFlow Lite Micro provides a lightweight runtime, CMSIS-NN offers optimized Cortex-M kernels, and Edge Impulse, used in this study, combines data processing, model training, and deployment. However, quantization loss and limited labeled data remain common challenges [11–19].

The present study contributes a reproducible evaluation using a public dataset together with classification and deployment-resource results.

https://doi.org/10.3390/engproc2026163108

3. Materials and Methods

3.1. Dataset

The study used the TennisEye dataset [20], containing 2013 strokes from seven players. The dataset originally contained five stroke types, but the two volley classes had only 33 examples and were excluded. The final dataset contained 1980 samples from three classes, forehand groundstroke, backhand groundstroke, and serve, as summarized in Table 1.

Table 1. Class distribution after exclusion of the volley categories. Each window is 1.000 s of six-axis inertial data sampled at 100 Hz.

Class	Training Windows	Test Windows	Total	Share (%)
Forehand	814	203	1017	51.4
Serve	455	114	569	28.7
Backhand	315	79	394	19.9
Total	1584	396	1980	100.0

The source dataset reports one participant characteristic only. Players were placed into three self-reported categories by playing frequency: coach, playing several times per week; regular player, playing once per week; and casual player, playing zero to two times per month. No sex, age, anthropometric, or physical-condition variable was released with the data, and the license requires users not to attempt to reverse the anonymization of the subjects. The absence of demographic covariates is therefore an inherent limitation of using a secondary dataset and prevents any analysis of performance by sex or physical condition. The seven records contain 49, 289, 186, 302, 628, 381, and 145 strokes respectively after volley removal, and their stroke mixes differ considerably: two players contributed serves only or predominantly, while two others contributed no serves at all.

3.2. Signal Characteristics and Preprocessing

The MATLAB records were converted into individual CSV files containing timestamps and six sensor channels. Each stroke was processed using a 1000 ms window and stride. Sensor values reaching the accelerometer and gyroscope limits were retained. Spectral and statistical features were then extracted and supplied to the classifier. The complete preprocessing and model settings are provided in Table 2. It should be noted that the accelerometer was configured to ±16 g and reaches that limit in 79.8% of strokes, so its peak value is censored rather than measured. The gyroscope reaches its ±2000°/s limit in only 10.3% of strokes. Peak angular velocity is therefore the quantity used for the intensity analysis reported in Section 4.3.

Table 2. Complete impulse configuration. All values are as entered in Edge Impulse Studio and are sufficient to reconstruct the pipeline from the public dataset.

Input Settings	Spectral Analysis Settings	Classifier Settings
Axes: 6 (ax, ay, az, ωx, ωy, ωz) **Sampling frequency:** 100 Hz **Window size:** 1000 ms **Window stride:** 1000 ms	**Zero padding:** Enabled **Scaling factor:** 0.0005 **Filter:** 6th-order low-pass **Cut-off frequency:** 15.234 Hz **Analysis:** FFT, length 128 **Logarithmic spectrum:** Enabled **Output features:** 144 (24 per axis)	**Topology:** 144-32-16-8-3, fully connected **Training cycles:** 200 **Learning rate:** 0.001 **Batch size:** 32 **Validation split:** 20% **Activation:** ReLU (hidden), softmax (output) **Regularization:** L1 activity, $\lambda = 1 \times 10^{-5}$ **Loss:** Categorical cross-entropy **Optimizer:** Adam ($\beta_1 = 0.9$, $\beta_2 = 0.999$) **Quantization:** int8, per-channel

3.3. Training Procedure

The compact classifier shown in Figure 1 uses 144 input features and three ReLU hidden layers with 32, 16, and 8 units, followed by a three-class softmax output. The network contains 5331 trainable parameters.

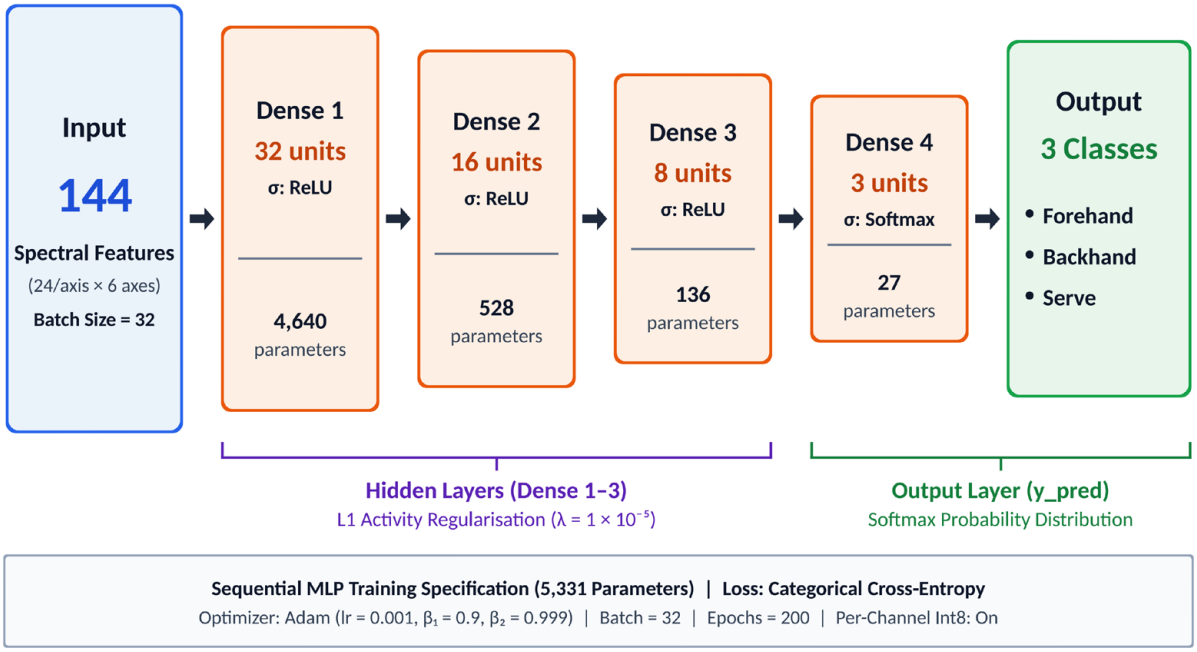


Figure 1. Classifier architecture.

Training used Adam with categorical cross-entropy loss for 200 cycles, a learning rate of 0.001, and a batch size of 32. Twenty percent of the training data was reserved for validation. Post-training INT8 quantization was applied, and both float32 and INT8 results are reported.

3.4. Deployment Target

The deployment target is an Arduino Nicla Sense ME [21], and selected for its inertial front end, is a Bosch BHI260AP smart sensor hub because its accelerometer and gyroscope match the measurement ranges used to record the TennisEye dataset. The target can therefore be configured with the same nominal measurement limits as the training data, reducing differences in signal clipping that could occur with hardware using a wider measurement range, as discussed in Section 5. The module is also light, and it is a self-contained system.

The module is fixed to the shaft as shown in Figure 2. Placement occurs at the handle sensor position used to record the training data, which matters because stroke signatures are placement-dependent.

The proposed system classifies each stroke directly on the racket and stores stroke counts and summary motion data in the on-board memory. After the session, the data can be transferred to a phone using Bluetooth Low Energy. No network connection or second device is required during play. Only summary session data are transmitted, while the raw sensor data remain on the device.

https://doi.org/10.3390/engproc2026163108

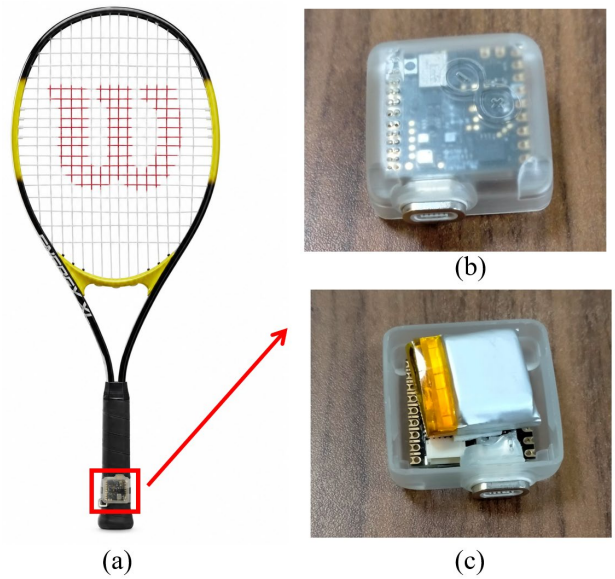


Figure 2. Instrumented racket setup. (**a**) Nicla Sense ME mounted on the racket shaft. (**b**) Original sensor unit. (**c**) Battery connected to board.

4. Results

4.1. Classification Performance

Two data partitions are used in this study. The validation set contains 20% of the training data and is used during training to monitor model performance and support model selection. The test set contains 396 strokes and was kept separate from training. It therefore provides an independent measure of model performance.

The comparison of validation and test results is given in Table 3. The float32 model achieves 90.5% validation accuracy and 90.4% test accuracy. Based on the test set, the macro F1 score is 0.891 and the area under the receiver operating characteristic curve is 0.962. The small difference of 0.1 percentage points between validation and test accuracy shows consistent performance based on both partitions. Therefore, the test accuracy of 90.4% is used as the main result.

Table 3. Classification performance of the deployed model. Quantization to int8 costs 1.3 percentage points of accuracy based on the validation partition, while leaving discrimination, as measured based on the ROC AUC, essentially unchanged.

Partition	Precision	Accuracy (%)	Macro F1	Weighted F1	ROC AUC	Loss
Validation (n = 317)	float32	90.5	0.897	0.905	0.964	0.297
Validation (n = 317)	int8	89.3	0.884	0.892	0.964	0.302
Test (n = 396)	float32	90.4	0.891	0.903	0.962	0.307

Per-class results based on the test partition are given in Table 4. Serves and forehands both achieve F1 scores above 0.92. Backhand is the weakest class with an F1 score of 0.829, driven by a recall of 0.797. A total of 19.0% of backhands are classified as forehands. This asymmetry is consistent with the class imbalance, with forehands outnumbering backhands by 2.6 to 1. Figure 3 presents the corresponding confusion matrices for both partitions, row-normalized so that each row sums to 100%.

The training dynamics are shown in Figure 4. Training accuracy continued to increase, while validation accuracy stabilized after the initial improvement. The subsequent increase in validation loss supports retaining the model weights from the epoch with the lowest validation loss.

Table 4. Per-class metrics on the held-out test partition (float32 model).

Class	Precision	Recall	F1
Backhand	0.863	0.797	0.829
Forehand	0.913	0.931	0.922
Serve	0.914	0.930	0.922
Macro average	0.897	0.886	0.891
Weighted average	0.903	0.904	0.903

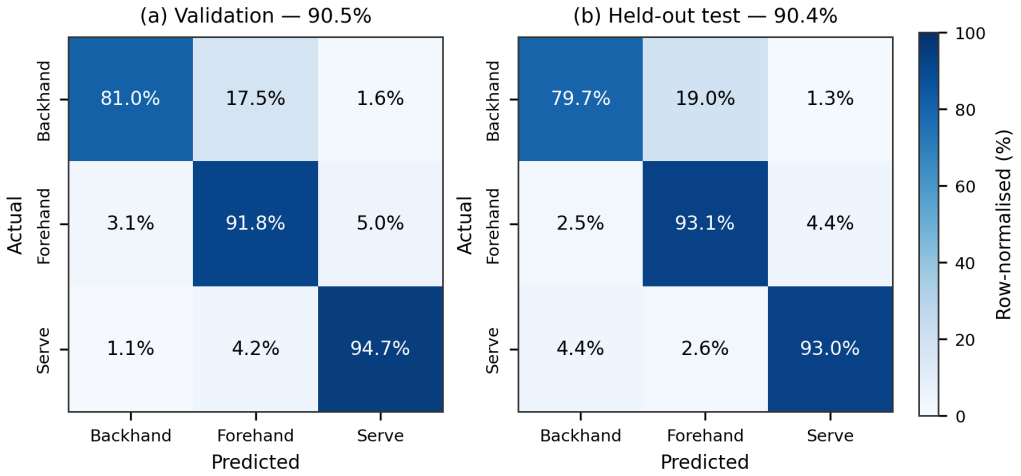


Figure 3. Confusion matrices for (**a**) the internal validation split and (**b**) the held-out test partition.

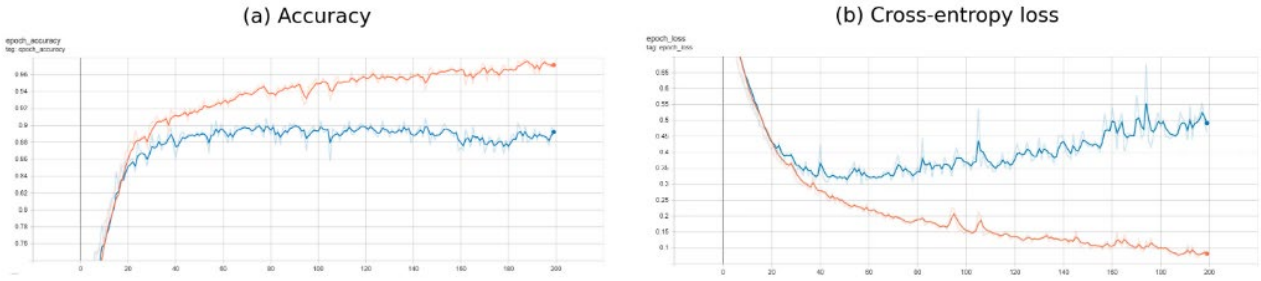


Figure 4. Training and validation dynamics of the classifier over 200 epochs: (**a**) accuracy and (**b**) categorical cross-entropy loss.

4.2. Quantization and On-Device Resource Profile

INT8 quantization reduced validation accuracy slightly from 90.5% to 89.3%, while AUROC remained 0.964. As shown in Table 5, it reduced classifier flash usage from 31.5 kB to 19.5 kB and working memory from 2.1 kB to 1.7 kB, while total latency remained as 9 ms.

Table 5. Estimated on-device resource profile for the int8 and float32 model variants.

Metric	Stage	Quantized (int8)	Unoptimized (float32)
Latency	Spectral features	8 ms	8 ms
Latency	Classifier	1 ms	1 ms
Latency	Total	9 ms	9 ms
Peak RAM	Spectral features	4.4 kB	4.4 kB
Peak RAM	Classifier	1.7 kB	2.1 kB
Peak RAM	Total	4.4 kB	4.4 kB
Flash	Classifier	19.5 kB	31.5 kB
Accuracy	Profiler estimate	88.13%	88.38%

Overall, the proposed pipeline achieved consistent classification performance while meeting the memory and latency requirements for deployment on the Cortex-M4F.

4.3. Stroke-Level Performance Indicators

Classification is the indexing step rather than the end product. Once each stroke is labeled on the device, the same one-second window also yields a measure of how hard the stroke was played, and grouping those measures by stroke type, by player, and by position within the session produces performance information without any additional sensing.

Peak linear acceleration cannot serve this purpose here because it is censored, as noted in Section 3.2. Peak angular velocity is used instead, and it is validated against the optical ground truth supplied with the dataset. Peak angular velocity correlates with the outgoing ball speed at rho = 0.750 for serves (n = 569), rho = 0.653 for forehands (n = 1017), and rho = 0.541 for backhands (n = 394), all with $p < 10^{-30}$. A quantity that the microcontroller computes locally therefore tracks a quantity that otherwise requires a six-camera installation.

Table 6 compares the seven players. The median peak angular velocity ranges from 1439 to 2000°/s, and the median outgoing ball speed ranges from 47 to 78 mph. The reporting intensity together with spin separates playing style from playing strength: the player with the lowest swing intensity also records the highest spin, which indicates a topspin baseline game rather than a weaker stroke.

Table 6. Per-player comparison of stroke volume, swing intensity, and optically measured ball outcome. Counts exclude the 33 volleys removed from the study. The dagger marks a median lying on the gyroscope limit, so the true intensity of that player is higher than shown.

Player	Strokes	FH	BH	SV	Median Peak $\lvert\omega\rvert$ (°/s)	Median Ball Speed (mph)	Median Spin (rpm)
P1	49	9	4	36	2000 †	78	0
P2	289	145	69	75	1439	52	728
P3	186	70	66	50	1709	61	651
P4	302	240	62	0	1602	53	814
P5	628	470	158	0	1446	47	972
P6	381	83	35	263	1766	61	0
P7	145	0	0	145	1757	64	344

Within-session tendencies are also visible. Assuming that stroke order reflects acquisition order, one player shows a monotonic decline in both swing intensity and ball speed across the record on both wings, which is the pattern expected from accumulating fatigue, while two other players show increases consistent with warming up. Table 7 lists the trends that reach significance.

Table 7. Within-record trends in swing intensity and ball speed. Only player-and-class combinations reaching $p < 0.05$ are listed. Two players whose strokes were recorded in blocks by stroke type are excluded because the order and stroke type are confounded for them.

Player and Class	n	Rho (Order vs. Peak $\lvert\omega\rvert$)	p	Rho (Order vs. Ball Speed)	p
P4 forehand	240	−0.521	<0.001	−0.291	<0.001
P4 backhand	62	−0.433	<0.001	−0.310	0.014
P6 forehand	83	−0.279	0.011	−0.185	0.093
P5 backhand	158	+0.328	<0.001	+0.194	0.015
P7 serve	145	+0.228	0.006	+0.262	0.001

These results should be read as a demonstration of the feedback that the device can generate rather than as a validated fatigue measure. The dataset publishes no timestamps, so stroke index is only a proxy for elapsed time, and the cohort is too small for the direction of each trend to be attributed to a specific physiological cause. Figure 5 summarizes the three analyses.

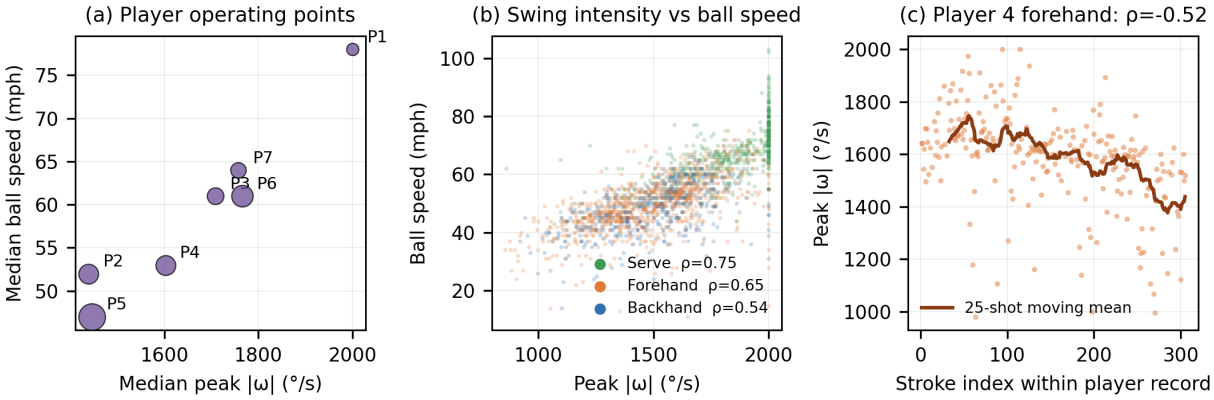


Figure 5. Stroke-level performance indicators derived from the same on-device signals used for classification. (**a**) Median swing intensity and median ball speed for each of the seven players. (**b**) Peak angular velocity against optically measured ball speed by stroke class, with Spearman correlations. (**c**) Within-record trend in swing intensity for the clearest case, with a 25-stroke moving mean.

5. Conclusions

A TinyML pipeline was developed to classify forehand, backhand, and serve strokes from racket-mounted inertial sensor data, achieving 90.4% test accuracy and a macro F1 score of 0.891 on 396 held-out strokes. INT8 quantization reduced validation accuracy by only 1.2 percentage points. The estimated deployment profile includes 4.4 kB of peak RAM, 19.5 kB of classifier flash, and 9 ms of total inference latency on a Cortex-M4F at 80 MHz, indicating that deployment is feasible, although these values require on-device verification. Volleys were excluded because only 33 examples were available. Future work will collect a larger on-court dataset, include idle and volley classes, evaluate raw-input convolutional models, and measure the latency, power consumption, and session duration on the target hardware. The same on-device signals were additionally shown to support stroke-level performance indicators, including a between-player comparison of swing intensity and within-session trends, establishing that the device can report on performance and not only on stroke identity.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The TennisEye dataset analyzed in this study is publicly available from https://github.com/hongyang-zhao/TennisEye (accessed on 12 March 2026) under a non-commercial research license. The SmartTennisRacket dataset referenced in Section 2.2 is available under the MIT license at https://github.com/Fexcatta/SmartTennisRacket (accessed on 12 March 2026).

Acknowledgments: The authors thank the authors of the TennisEye dataset for making their data available for research use.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Kos, M.; Ženko, J.; Vlaj, D.; Kramberger, I. Tennis stroke detection and classification using miniature wearable IMU device. In Proceedings of the 2016 International Conference on Systems, Signals and Image Processing (IWSSIP), Bratislava, Slovakia, 23–25 May 2016; pp. 1–4. [CrossRef]
2. Kos, M.; Kramberger, I. Tennis stroke consistency analysis using miniature wearable IMU. In Proceedings of the 2018 25th International Conference on Systems, Signals and Image Processing (IWSSIP), Maribor, Slovenia, 20–22 June 2018; pp. 1–4. [CrossRef]
3. Whiteside, D.; Cant, O.; Connolly, M.; Reid, M. Monitoring hitting load in tennis using inertial sensors and machine learning. *Int. J. Sports Physiol. Perform.* **2017**, *12*, 1212–1217. [CrossRef]
4. Zhao, H.; Wang, S.; Zhou, G.; Jung, W. TennisEye: Tennis ball speed estimation using a racket-mounted motion sensor. In Proceedings of the 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), Montreal, QC, Canada, 16–18 April 2019; pp. 241–252. [CrossRef]
5. Ebner, C.J.; Findling, R.D. Tennis stroke classification: Comparing wrist and racket as IMU sensor position. In Proceedings of the 17th International Conference on Advances in Mobile Computing and Multimedia (MoMM), Munich, Germany, 2–4 December 2019; pp. 74–83. [CrossRef]
6. Park, J.; Yang, S.; Jo, S. Silent Impact: Tracking tennis shots from the passive arm. In Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (UIST), Pittsburgh, PA, USA, 13–16 October 2024; pp. 1–15. [CrossRef]
7. Ganser, A.; Hollaus, B.; Stabinger, S. Classification of tennis shots with a neural network approach. *Sensors* **2021**, *21*, 5703. [CrossRef]
8. Wu, M.; Fan, M.; Hu, Y.; Wang, R.; Wang, Y.; Li, Y.; Wu, S.; Xia, G. A real-time tennis level evaluation and strokes classification system based on the Internet of Things. *Internet Things* **2022**, *17*, 100494. [CrossRef]
9. Liu, R.; Lu, T.; Yuan, S.; Zhou, H.; Gowda, M. SmartDampener: An open source platform for sport analytics in tennis. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* **2024**, *8*, 118. [CrossRef]
10. Hung, C.-H.; Bai, Y.-W.; Chiang, C.-H.; Wu, H.-Y. An improved design of an automatic detection balance tray with an accelerometer, four servo motors, four force-sensitive resistors and an MCU expansion board. In Proceedings of the 2017 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 8–10 January 2017; pp. 55–58. [CrossRef]
11. Dabbous, A.; Fresta, M.; Bellotti, F.; Berta, R. Arduino Nano-based system for tennis shot classification. In *Proceedings of SIE 2023*; Lecture Notes in Electrical Engineering; Ciofi, C., Limiti, E., Eds.; Springer: Cham, Switzerland, 2024; Volume 1113. [CrossRef]
12. Dabbous, A.; Fresta, M.; Bellotti, F.; Berta, R. Neural architecture for tennis shot classification on embedded system. In *Applications in Electronics Pervading Industry, Environment and Society*; Lecture Notes in Electrical Engineering; Springer: Cham, Switzerland, 2024; Volume 1110. [CrossRef]
13. Warden, P.; Situnayake, D. *TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers*; O'Reilly Media: Sebastopol, CA, USA, 2019.
14. David, R.; Duke, J.; Jain, A.; Janapa Reddi, V.; Jeffries, N.; Li, J.; Kreeger, N.; Nappier, I.; Natraj, M.; Wang, T.; et al. TensorFlow Lite Micro: Embedded machine learning for TinyML systems. *Proc. Mach. Learn. Syst.* **2021**, *3*, 800–811. [CrossRef]
15. Hymel, S.; Banbury, C.; Situnayake, D.; Elium, A.; Ward, C.; Kelcey, M.; Baaijens, M.; Majchrzycki, M.; Plunkett, J.; Tischler, D.; et al. Edge Impulse: An MLOps platform for tiny machine learning. *arXiv* **2022**, arXiv:2212.03332. [CrossRef]
16. Lai, L.; Suda, N.; Chandra, V. CMSIS-NN: Efficient neural network kernels for Arm Cortex-M CPUs. *arXiv* **2018**, arXiv:1801.06601. [CrossRef]
17. Ray, P.P. A review on TinyML: State-of-the-art and prospects. *J. King Saud Univ. Comput. Inf. Sci.* **2022**, *34*, 1595–1623. [CrossRef]
18. Cattaneo, F.; Re, F. SmartTennisRacket: TinyML Model for Detecting Tennis Shots Deployed on an Arduino Nano Board. GitHub Repository. 2023. Available online: https://github.com/Fexcatta/SmartTennisRacket (accessed on 15 August 2026).
19. Aveiro, A. Tinyml-Smart-Tennis-Sensor. GitHub Repository. 2021. Available online: https://github.com/alvarowolfx/tinyml-smart-tennis-sensor (accessed on 15 August 2026).
20. Zhao, H. TennisEye Dataset and Source Code. GitHub Repository. 2019. Available online: https://github.com/hongyang-zhao/TennisEye (accessed on 15 August 2026).
21. Arduino. *Nicla Sense ME Datasheet (ABX00050)*; Arduino S.r.l.: Turin, Italy, 2021. Available online: https://docs.arduino.cc/hardware/nicla-sense-me/ (accessed on 15 August 2026).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.