IBED: Interactive Badminton Entertainment Device

Wenqu Fang*
Yuanxin Pang*
a.fang@kaist.ac.kr
yuanxinpang@kaist.ac.kr
Korea Advanced Institute of Science and Technology (KAIST)
Daejeon, Republic of Korea

CCS CONCEPTS

• **Human-centered computing → Interaction devices**.

KEYWORDS

inertial measurement units, edge computing, embedded machine learning, internet of things

ACM Reference Format:
Wenqu Fang and Yuanxin Pang. 2024. IBED: Interactive Badminton Entertainment Device. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 3 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Figure 1: Lin Dan Playing in 2016 Olympics

Edge computing involves storing and processing data on edge devices, closer to data sources than traditional cloud servers, ensuring lower latencies for applications[4]. Local data processing also reduces the risk of data breaches. A key application is wearable smart sports gear, driven by interest in tracking and analyzing exercise metrics. Current devices like watches and wristbands monitor health data and movements but lack real-time interactive entertainment.

*Both authors contributed equally to this research.

This report proposes a wearable interactive entertainment device for badminton players. The wristband will detect player movements and produce audio responses through a built-in loudspeaker.

The following sections review related literature, detail the product design, outline the development plan, and discuss major challenges.

2 RELATED WORKS

IMU-based motion pattern recognition has been proved possible by lots of works [1] [2]. Specifically, there is a 2021 project that features deploying an ML model on Arduino Nano 33 BLE Sense to evaluate a ping-pong player's form of movement.[3] Our work will be a modification and extension of this project.

3 SYSTEM DESIGN

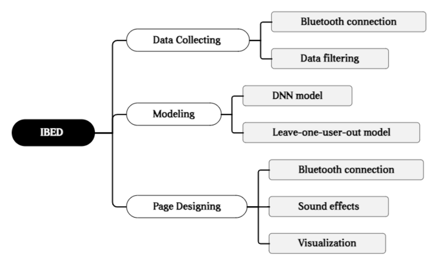

Figure 2: System design

Our product includes three main parts(shown in figure 2). We will introduce each part in detail. First, we will introduce the prototype and technologies we have used in this project.

3.1 System Profile

3.1.1 Prototype. Our product includes an Arduino Nano 33 BLE Sense board on a wrist band with a 200 gram power supply carried in a light-weight running armbag(see Figure3). Nano 33 BLE Sense will recognize the action patterns of badminton players through its IMU sensor and produce fun sound effects through a microphone accordingly.

3.1.2 Technology stacks. **Data collection** involves uploading IMU data via Bluetooth® using Tiny Motion Trainer, with the sampling window and threshold set at the front end. The front end also handles data cleaning to ensure accuracy. In the **IBED system**, inferences run entirely offline on the edge device. The sampling window and threshold are set in the Arduino code, and the probabilities of four patterns are uploaded via Bluetooth® to the frontend.

Figure 3: Prototype

Based on the detected patterns, corresponding sound effects are triggered.

3.2 Data Collection

The data collection was performed in staged settings, targeting four postures, namely full strike, lift, chop, and backhand save, all of which are common badminton techniques. Six sets of data were collected from six different test subjects when a shuttlecock was fed to them and they were instructed to strike it back using a designated stroke technique. Four other sets of data were obtained from four different test subjects when they received the same instructions about stroke techniques as above, but without a shuttlecock. There are in total six test subjects. Experiment settings and test subjects demographics are listed in Table 1.

In total, ten sets of IMU data were collected, all targeting four-stroke patterns, with thirty moves under each pattern. The IMU data sample rate was set as the default 119 Hz. Data was uploaded to Tiny Motion Trainer, a front-end platform via Bluetooth®, where a sampling time window, an inter-sampling interval, and a trigger threshold were set. The time window was set such that each move consisted of 20 pieces of IMU data. The inter-sampling interval was set at 0.2 seconds. The trigger threshold was set as

$$\frac{a_x + a_y + a_z}{4} + \frac{g_x + g_y + g_z}{2000} \geq 0.14 \tag{1}$$

where $a_x, a_y, a_z, g_x, g_y, g_z$ are acceleration and gyroscope readings in x, y, and z axes respectively.

Overall, these settings ensured that each move was captured as clearly as possible, without false triggering or overlaps between stroke records.

3.3 Model Design and Experiments

We designed a simple LSTM model(shown in figure 4), training with hyperparameters below:

(1) optimizer=adam
(2) loss=categorical_crossentropy
(3) metrics=[accuracy]
(4) Epoch=50
(5) Batch size = 8

For the experiment, we trained 12 models. For shuttlecock data(6 users), 6 leave-one-user-out models are trained and one general model takes the whole shuttlecock data as input. For no shuttlecock

Layer (type)	Output Shape	Param #
input_6 (InputLayer)	[(None, 1, 120)]	0
lstm_10 (LSTM)	(None, 1, 120)	115680
dropout_10 (Dropout)	(None, 1, 120)	0
lstm_11 (LSTM)	(None, 120)	115680
dropout_11 (Dropout)	(None, 120)	0
dense_10 (Dense)	(None, 120)	14520
dense_11 (Dense)	(None, 4)	484

Figure 4: Model architecture

data(4 users), 4 leave-one-user-out models are trained and one general model takes the entire no shuttlecock data as input. The user profiles are in table 1.

3.3.1 Traning performance. During training, we selected two models for each dataset and found an overfitting problem probably due to the limited number of datasets (shown in figure 5).

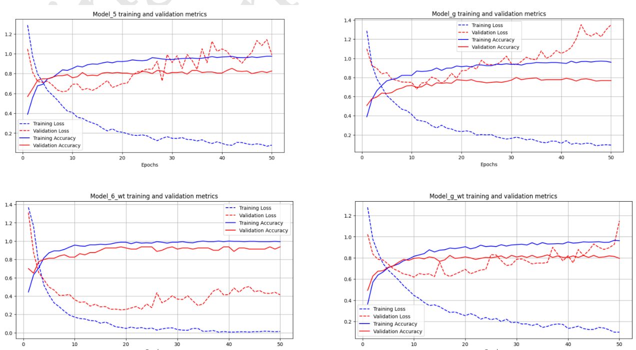

Figure 5: Training performance

3.3.2 Test performance. During the test period, we have two findings from figure 6:

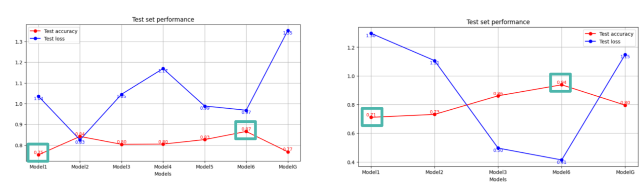

Figure 6: Test performance

(1) **Different** dataset models: shuttlecock models have an average higher performance.
(2) **Same** dataset models : Leave-user-1-out model performs worst while Leave-user-6-out model performs best. It indicates that user 1 may have the most standard gestures which are essential in gesture recognition, while user 6 data may poison the pattern recognition period.

Table 1: Demographics and experiment situations of test subjects

	User1	User2	User3	User4	User5	User6
With shuttlecock	✓	✓	✓	✓	✓	✓
Without shuttlecock	✓	✓	✓			✓
Age	20	23	23	27	23	24
Sex	Male	Male	Female	Male	Male	Male
Proficiency	Intermediate	Intermediate	Intermediate	Professional	Intermediate	Intermediate

3.3.3 Model performance on different gestures. We select both datasets' best, worst, and general models' confusion matrix(shown in figure 7). What we can get from the figure is that shuttlecock models are more advanced in recognizing different patterns than specific special patterns. While the without shuttlecock models are better at specifying special patterns(full_strike and lift), the without shuttlecock models have fewer data which may make user 1 more essential and user 6 more poisonous.

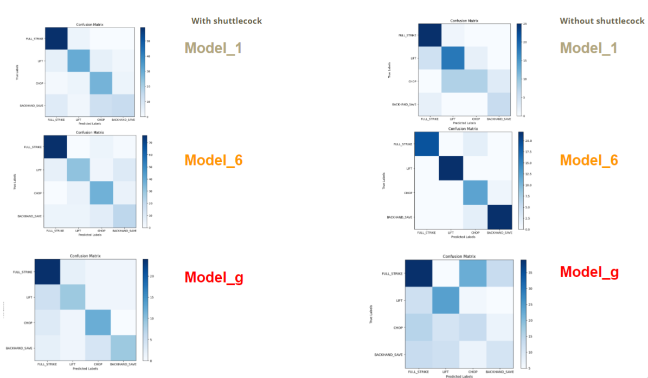

Figure 7: Confusion Matrix

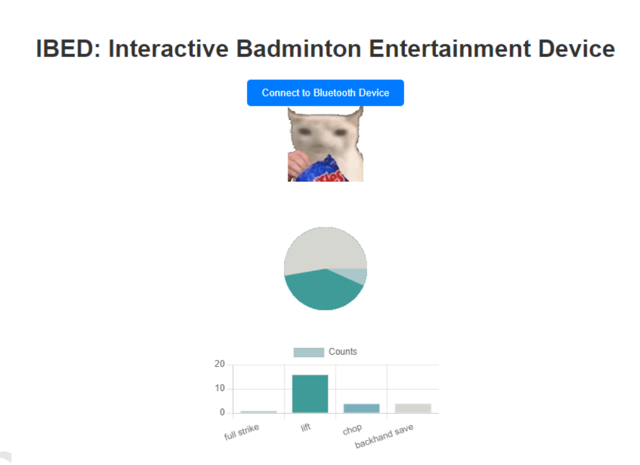

Figure 8: IBED web server demo

3.4 Model Deployment and Web Application

An Arduino program was prepared, quantized, and uploaded to the Arduino Nano 33 BLE Sense board along with the model. The inference is triggered only when the same threshold as was used in the data collection stage is reached. Bluetooth® connect and disconnect handlers were also implemented to reflect its connection status.

A simple web server was implemented (see Figure 8). It sets up a Bluetooth® connection with the Arduino board and receives inference results, i.e. arrays of prediction probabilities of the four patterns to be identified. The one with the highest probability will be reckoned the stroke the player used, and a short sound effect will be played accordingly. At the same time, a pie chart is plot reflecting all probabilities, and a bar chart keeps track of the total number of strokes identified under each pattern.

4 FUTURE WORK

A significant problem encountered in this project is the issue of instability in data transmission during data collection, which poisoned our data at the very beginning, made it hard to perform data cleaning, and therefore undermined our model performance. This is hugely caused by the limited extensibility of the teachable machine platform Tiny Motion Trainer. To resolve this issue, it will be necessary to develop a data collection utility that meets better specifications.

To improve the recognition success rate, it will be also desirable to deploy more traditional signal-processing techniques besides the machine learning approach, such as gravity cancellation and noise filters. Installing additional IMU sensors at different locations on a body can help extract more features from strokes and can potentially increase the model accuracy as well.

ACKNOWLEDGMENTS

To EE488 course staff and Professor Ian Oakley.

REFERENCES

[1] Marcus Georgi, Christoph Amma, and Tanja Schultz. 2015. Recognizing Hand and Finger Gestures with IMU based Motion and EMG based Muscle Activity Sensing. In *Proceedings of the International Conference on Bio-inspired Systems and Signal Processing - Volume 1: BIOSIGNALS, (BIOSTEC 2015)*. INSTICC, SciTePress, 99–108. https://doi.org/10.5220/0005276900990108

[2] Shun Ishii, Kizito Nkurikiyeyezu, Mika Luimula, Anna Yokokubo, and Guillaume Lopez. 2021. *ExerSense: Real-Time Physical Exercise Segmentation, Classification, and Counting Algorithm Using an IMU Sensor.* Springer Singapore, Singapore, 239–255. https://doi.org/10.1007/978-981-15-8944-7_15

[3] Nicholas Patrick, Samuel Alexander. 2021. Table Tennis Bat With Machine Learning AI. https://www.instructables.com/Table-Tennis-Bat-With-Machine-Learning-AI-Arduino-/. License: BSD.

[4] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge Computing: Vision and Challenges. *IEEE Internet of Things Journal* 3, 5 (2016), 637–646. https://doi.org/10.1109/JIOT.2016.2579198