BurnSync: Your Personal Fitness Assistant

Hsuan-Ying, Liu (Co-first author)[1] and Chi-En, Dai (Co-first author)[1]

[1]National Taiwan University

Abstract In this work, we propose **BurnSync**, a personal fitness assistant to enhance exercise routines. BurnSync is a smartwatch-like device capable of classifying exercise types, automatically recording sets and repetitions, detecting heart rate, providing fitness recommendations, and estimating calorie consumption. With the integration with our custom website, BurnSync can provide a time-efficient and excellent fitness experience for users. Furthermore, BurnSync is also an open-source project, and we provide a detailed presentation for reference. [1] [2] [3] [4]

Keywords: AIoT, Human-centered computing, Human computer interaction(HCI), Machine Learning, Embedded System, Fitness Tracker

1 Motivation & Introduction

1.1 Motivation

In today's society, health has become a growing concern, and exercise is recognized as a key factor in maintaining physical and mental well-being. While

[1] Main repository: `https://github.com/lsy1205/AIoT_Final_Project.git`

[2] Custom website: `https://github.com/lsy1205/BurnSync_website.git`

[3] Exercise classification model deployed on Hugging Face Space: `https://huggingface.co/spaces/AIOT12345/IMU_CLASSIFY`

[4] Our presentation: `https://docs.google.com/presentation/d/1sYJubLL72miAi9WDsaDhsZYGOqvKxebexM1-4gYhk2g/edit?usp=sharing`

according to the analysis of World Health Organization (WHO), up to 31% of adults and 80% of adolescents fail to meet the recommended levels of physical activity.[5] Modern people have busy schedules and an abundance of entertainment activities take up all of the time, which leads to little time for exercise. The lack of time, money, and motivation also keep people away from doing exercise. To address these challenges, we propose **BurnSync**, a smart device which is affordable, highly accessible, and seamlessly connected.

1.2 Background Introduction

Fitness Tracking

Fitness tracking has always been a popular topic. Advancements in machine learning have significantly expanded the potential applications of fitness tracking, enhancing accuracy and introducing novel functionalities such as exercise analysis and feedback[5].
Fitness tracking can be done with a variety of means, but the most common ways are by camera[5] and by inertial measurement unit (IMU)[7]. The drawbacks of cameras are form factor and privacy issues. In this work, we choose to use IMU. Although video-based methods may provide richer information, IMUs are compact and suitable for wearable devices, allowing portability and ease of use. IMU can also cause less privacy issues. We will discuss more about IMU later.

Inertial Measurement Unit

Inertial measurement units (IMUs) are invented for over 10 years. There are several advantages of IMUs, including it is low-cost, small, and robust. IMUs can measure a body's acceleration, angular rate, and sometimes the orientation. They can be used in a variety of domains, like in vehicles, smartphones, robotics, and of course wearable devices like our work. With the development of machine learning, IMU data can also be used for many tasks like gait analysis, gesture recognition and activity recognition. There are many previous works discussing the applications of IMUs. [9]

Arduino Nano 33 BLE Sense Rev 2

Arduino is an open-source electronic prototyping platform. The most often seen is Arduino Uno board. In comparison with Arduino Uno, Arduino Nano is a compact board which is suitable for wearable devices. Arduino Nano 33

[5] WHO analysis: `https://www.who.int/news-room/fact-sheets/detail/physical-activity`

BLE Sense Rev2[6] also integrate several sensors including IMU, microphone, temperature and humidity sensor, and so on. In this work, we only use the IMU sensor, and thus there is a potential to integrate more sensors for other functions, which we will leave it as future work.

2 System Overview

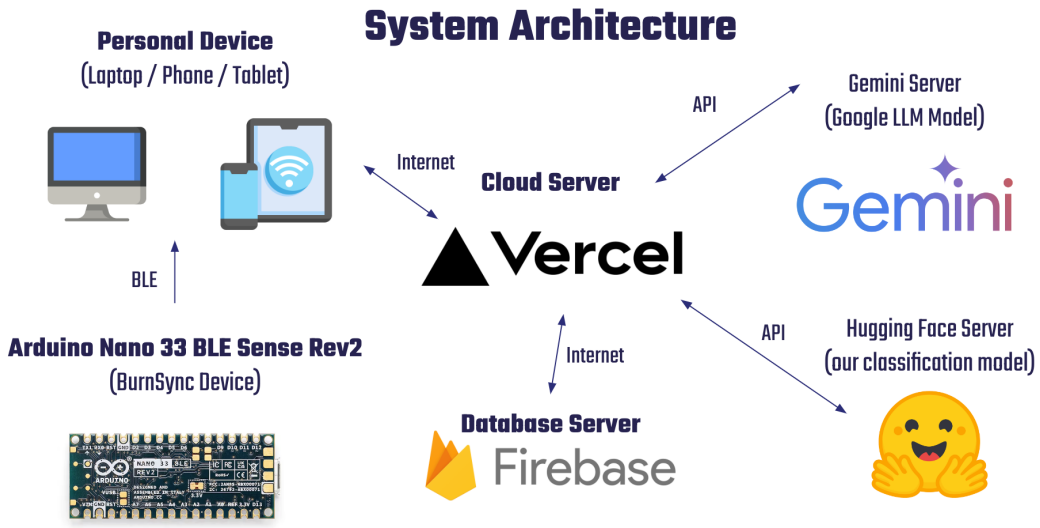


Figure 1: The high level system architecture of BurnSync

2.1 Architecture Overview

To have an overview of BurnSync, please refer to **Figure 1**. BurnSync is a device based on Arduino Nano 33 BLE Sense Rev2. After collecting the data from sensors, Arduino Nano will interact with personal devices through Bluetooth Low Energy (BLE). Since we use a custom website to connect with the internet, personal devices need only support BLE and have internet connectivity, making the system compatible with laptops, smartphones, and tablets. Then, the personal device will connect to our website through the internet. Our website is hosted on **Vercel**, which is a platform for web application. The website will store user data in **Firebase**. We also connect our website with **Gemini** through api. Gemini is a large language model (LLM) developed by Google. Last but

[6] To know more about Arduino Nano, visit the official website: `https://docs.arduino.cc/hardware/nano-33-ble-sense-rev2/`

not least, we deploy our exercise classification model on **Hugging Face Space**, so our website can interact with the model through api.

2.2 Function Overview

BurnSync has 5 main functions, exercise classification, sets/repetitions recording, calories estimation, fitness recommendation, and heart rate detection. We will go through them one by one.

Exercise Classification

BurnSync can classify four types of exercises: push-ups, sit-ups, squats, and dumbbell exercises. We use the 6 axes of IMU data to classify which kind of exercise user is performing.

Sets/Repetitions Recording

BurnSync is capable of recording the number of sets and repetitions of the 4 types of exercise, which is convenient for users and can hence improve the fitness experience of users.

Calories Estimation

On our custom website, we perform calories estimation according to the number of repetitions. The equation is as follow.

$$\text{Calories Burned} = \text{Exercise Coefficient} \times \frac{\text{Weight(kg)}}{70} \times \text{Age Coefficient} \times \text{Gender Coefficient} \times \text{Repetitions}$$

The exercise coefficient is push-up:0.5, sit-up: 0.3, squat: 0.32, dumbbell: 0.4.
The age coefficient equals to $1 - (\text{age} - 25) \times 0.005$. The smallest value is 0.75.
The gender coefficient is 1 for male and 0.9 for female.
Please note that this equation only provide a really rough estimation.

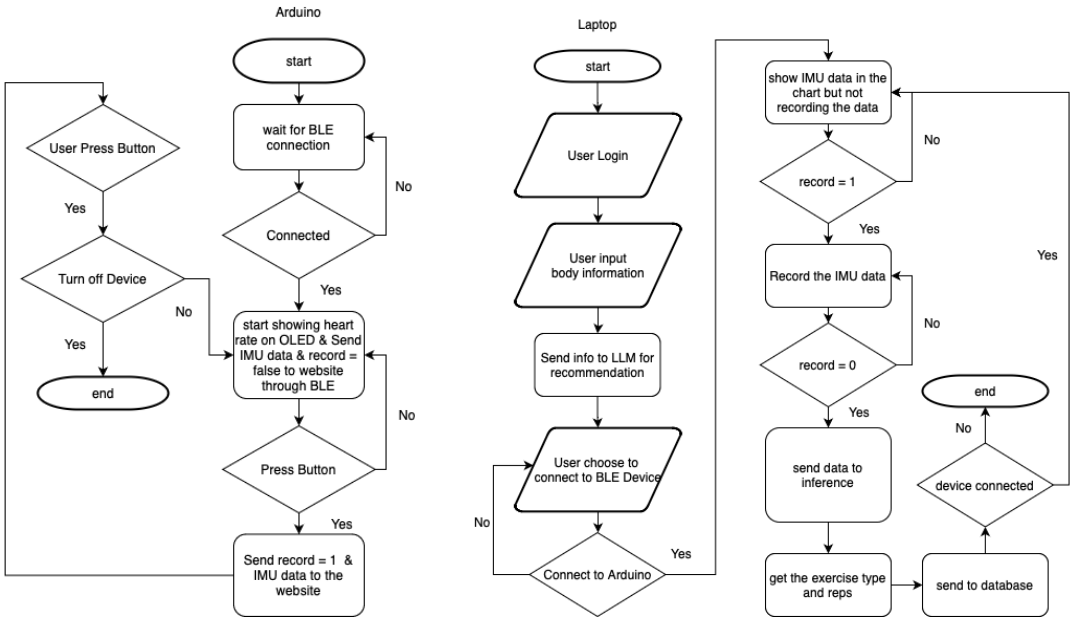
Fitness Recommendation

We integrate LLM for fitness recommendation, so our website will recommend the number of sets and repetitions for users according to their personal information and the target calories they want to burn.

Heart Rate Detection

Our device is able to detect the average beats per minute (BPM) of the user and display on the OLED module, which allows users to monitor their body status while doing exercise.

2.3 User Flow Overview



(a) The flow chart of BurnSync device

(b) The flow chart of personal device

Figure 2: The flow chart of our devices

Let's start with the flow chart on the left hand side, which is the Arduino part. First our device will wait for BLE connection. If connected, it will start to show the heart rate and send real time IMU data to the website. If user presses the button, it will send the variable **record** as 1. After the user presses the button again, the variable **record** will be toggled back to 0.

For the flow chart on the right hand side, which is the laptop (personal device) side, first, user will login and input his/her body information. Then the user can choose to ask LLM for personal recommendation. Next, the user can connect to the Arduino, that is to say, the BurnSync device. The website will show the real time IMU data in the chart. If the variable **record** equals to 1, the website will record the IMU data. If the variable **record** toggles back to 0, the falling edge will induce the website to stop recording IMU data and send the recorded data to inference. After inference, we will get the type of exercise and the number of repetitions. The data will then be sent to the database for record.

3 Hardware

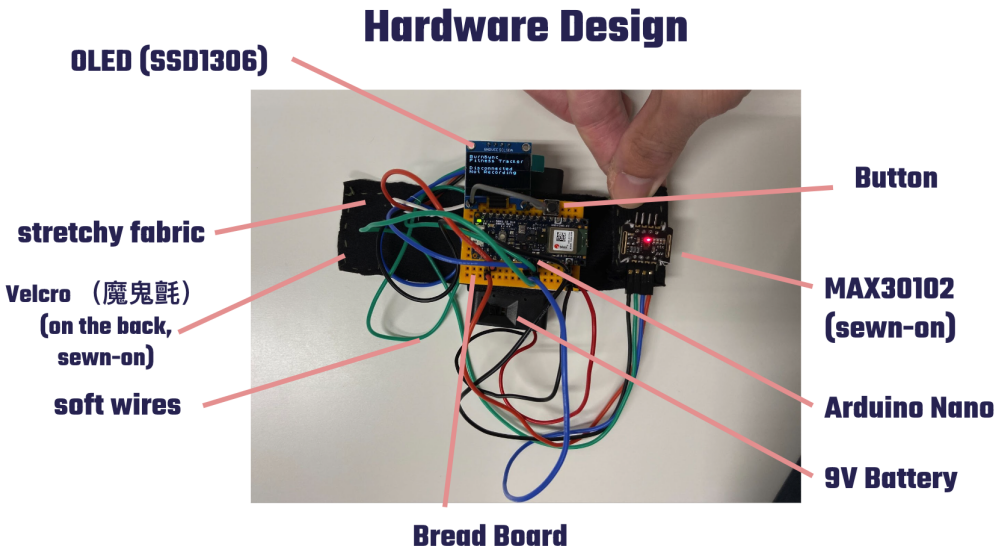


Figure 3: The hardware design of BurnSync

3.1 Hardware Design

Our device, **BurnSync**, is an Arduino Nano based device. In order to do fitness tracking, we want our device to fix on users' wrist. However, different users have different size of wrists. Another issue is that while doing exercise, users typically sweat and it is not easy to fix the position of the device. To deal with these problems, we come out with a special hardware design as **Figure 3**. First, we use a 9V battery as power supply according to the specification of Arduino Nano. To fix on users' wrist, we use velcro, strechy fabric and soft wires to facilitate users' activity. Above the strechy fabric is the bread board which is used to fix Arduino Nano, OLED, and the button. MAX30102 module is sewn on one end of the fabric so it will be close to users' base of the wrist. By such design, BurnSync is able to fix on most of users' wrist. While during exercise, our device is possible to slide slightly on users' wrist. Fortunately, due to the characteristic of IMU and the robustness of our classification model, this sliding will not affect our classification accuracy.

3.2 Modules

In our device, we use 3 main modules as in **Figure 4b**. As our block diagram **Figure 4a**, we use I^2C protocol to communicate with 2 modules and use BLE

to communicate with laptop or other personal devices. We will introduce the 3 modules below.

(a) The block diagram of BurnSync

(b) The modules used in the block diagram

Figure 4: The hardware architecture of BurnSync

IMUs

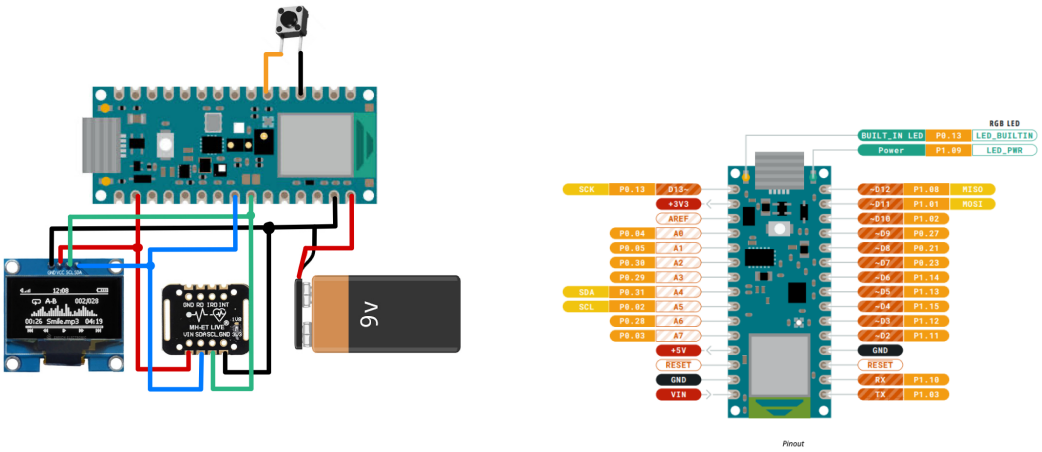
The IMUs we use are **BMI270 and BMM150**, which are the inbuilt modules of Arduino Nano. The IMUs includes an accelerometer,a gyroscope, and a magnetometer. Each measurement device offers 3 axes, which in total provide 9 axes. However, we only take 6 axes as our data since the magnetometer measures the Earth's magnetic field, offering heading information relative to magnetic north, which we consider not necessary for fitness tracking.

Max30102

Max30102 module measures user's heart rate with a technology called **photoplethysmography (PPG)**[3]. It detects blood volume changes in the microvascular bed of tissue. The operating principle of Max30102 is that it emits light into the skin and then measure the amount of light absorbed or reflected by the blood vessels. With this module, we can detect user's heart rate in a non-invasive approach.

OLED

As an end product, our device should be user-friendly, so we use **SSD1306** OLED module to show status and measurement for users. On the OLED, it displays the status of whether BLE is connected and whether the IMU data is being recorded or not. The heart rate measured by Max30102 module will also be shown on the OLED.



(a) The circuit diagram of BurnSync

(b) The connector pinout of Arduino Nano

Figure 5: The circuit diagram and pinout of Arduino Nano

3.3 Circuits

The circuit for BurnSync is quite simple, one can refer to our circuit diagram in **Figure 5a**. BurnSync only use a digital pin, the SCL, SDA pin, the GND pin and the V_{in} pin of Arduino, since the IMU and BLE is already inbuilt in the Arduino Nano. **Table 1** shows how to connect the pins of the modules. If one wants to know more about the pinout of Arduino Nano, one can checkout the datasheet of Arduino Nano 33 BLE Sense Rev2.[7]

Table 1: The pins of each module, the pins in the grid refer to the pins on Arduino Nano, X means not available

Module	+(Vcc)	-(GND)	SCL	SDA
Battery	V_{in}	GND	X	X
Button	D3	GND	X	X
Max30102	3.3V	GND	A5	A4
SSD1306	3.3V	GND	A5	A4

4 Software

4.1 Overview

In this section, we will discuss the software implementation of BurnSync, including our Arduino code implementation, classification implementation,

[7] Datasheet: `https://docs.arduino.cc/resources/datasheets/ABX00069-datasheet.pdf`

website hosting, and LLM integration.

4.2 Arduino Nano 33 BLE Sense Rev2

Our device is based on **Arduino Nano 33 BLE Sense Rev2** . We develop our device with **Arduino IDE**, which can be downloaded from the official website of Arduino. [8]

We use several libraries as listed below to configure our modules. With these libraries, we are able to configure the modules of our device, and we are also able to configure the pins of I^2C protocol and BLE communication. For more implementation details, please refer to our github repository.

- ArduinoBLE.h
- ACROBOTIC_SSD1306.h
- Arduino_BMI270_BMM150.h
- Wire.h
- MAX30105.h
- heartRate.h

4.3 Machine Learning

Architecture

Although our system doesn't has hard real-time constraint, we want the model to predict fast while maintain the accuracy. Due to the above mentioned, we choose a light weight model architecture called **AttnConv**, which consists of two convolution layers and an attention module. The 6-channel IMU signals pass through two 1D convolution layers(with multi-channels) and attention module for feature extraction, then they pass through the fully connected layer for classification. The structure of our model is shown as **Figure 6**.

Data Collection

We collect data by sending IMU data through BLE to computer and save as a csv file. We collect totally 1600 pieces of data by ourselves, every piece of data have 30 data points which is about 3 seconds. We spend about two hours including the resting time each day, and it lasts for five days to collect all of the data.

[8] Arduino IDE download: `https://www.arduino.cc/en/software`

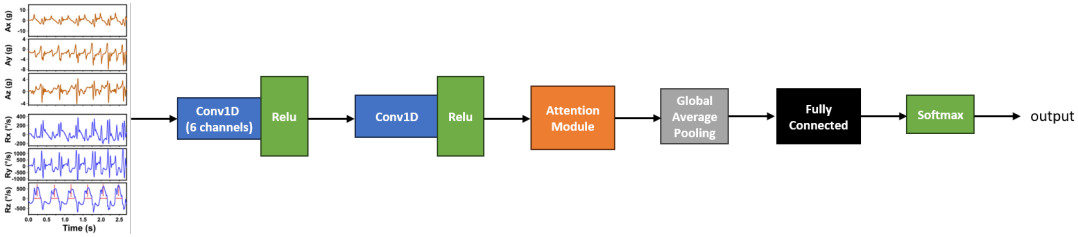


Figure 6: Our model architecture: AttnConv

Dataset

After data collection, we process each piece of data to make it unified with 30 data points. Since there will be some delay during transmission, some pieces of data have data points slightly less than 30 data points or more than 30. We apply zero padding for pieces less than 30, and clipping for pieces more than 30. After data processing, we split the dataset into training set, validation set and testing set, following the proportion of 70%, 15%, and 15% respectively.

Training Process

The configurations of our training process are listed as follow: epoch=100, batch size=8. We train **AttnConv** with cross entropy loss. Optimized by AdamW optimizer with initial learning rate=0.003, and the learning rate are scheduled by cosine annealing method. We validate the model every epoch, and estimate the overall accuracy and validation loss. If the accuracy does not improve for 10 epochs, the training will be early stopped.

Deployment Platform

We deploy the AttnConv model on the cloud, the platform is called Hugging Face Space. The free resource of the cloud platform is 2 CPUs with 16GB RAM. And we called our model to inference from website through Gradio API, We need to upload the inference code, model weights and the model architecture to the platform. After setting up the platform, we only have to feed in the input through API then we can get the prediction results.

4.4 Sliding Window

In the inference phase, we apply a sliding window on the six-channel 1D time sequence data. The six windows will slide concurrently in each channel and then predict the result. In our inference code, we will accumulate the prediction class with 1 in that window. The window size is 30, and the overlap is set to

half of the window size, that is to say, 15. To avoid the misjudgment when we start our exercise or finish our exercise, we set a threshold. If every prediction probability in the model output is less than the threshold, then that window will be classified as "None". Another mechanism is that we calculate the mode for the final statistics. For example: [8,1,2,0] will be transform to [8,0,0,0], due to the normal workout behavior of people, since users typically will not perform two kinds of exercise in a single set.

4.5 Website

We use a custom website as an interface to collect the IMU data from Arduino Nano through BLE and to interact with the internet. We'll introduce the implementation in this section. For more detail information, please refer to our repository.

Frontend

For our frontend, we use **Next.js** and **Tailwind CSS** to create a responsive website which can fit laptop, smartphone and tablet. Next.js is a frontend framework based on React, which is a framework with several powerful features such as server side rendering, file-based routing, and so on. Tailwind CSS is a highly customizable framework supporting responsive website design (RWD).

Backend

When it comes to database, there are several choices such as MongoDB, PostgreSQL, Neo4j, etc. Each DBMS has its own strength and weakness, here we choose **Firebase** since it provide a variety of functions including user authentication and database. We store our data in **Cloud Firestore**, allowing our user to store his/her own personal information and exercise track.

4.6 Large Language Model

We aim to let users get personal recommendation from our project. To achieve this goal, we connect our website with **Gemini** api. The reason we didn't choose openai api is due to the cost. Gemini is a LLM developed by google, providing a sufficient amount of free api usage for development, while openai did not provide any amount for development.

We choose the **Gemini 1.5 Flash** model. We also design a prompt to force Gemini to suggest the repetitions and sets of the 4 exercise types according to users' personal information and target calories to burn. We will discuss the number of tokens and cost in the **Discussion** section. Currently we notice

the delay of Gemini response is quite long. In the future, we do not rule out replacing with other large language models based on performance or response delay.

5 Evaluation

5.1 Overall Testing

In this section, we evaluate our AttnConv model on the testing set which is the 15% split from the original dataset. This testing set is the mixture of user1 and user2, both left hand and right hand, high position and low position.

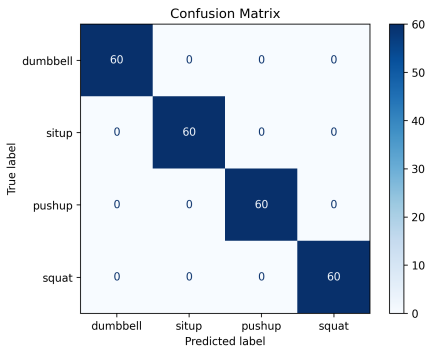


Figure 7: Confusion Matrix for the Test Set

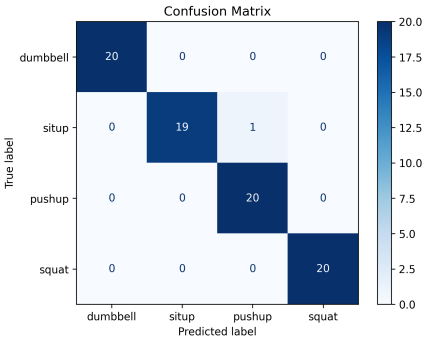
We achieve an overall accuracy of **100%**. From the overall accuracy and the confusion matrix in **Figure 7**, we can conclude that our model is well trained and can have a high performance. However, the issues of wearable devices are always the robustness of cross-users and that of variation in wearing position. To test the robustness of our model, we conduct further evaluations under different settings, as detailed in the following section.

5.2 Different Hand

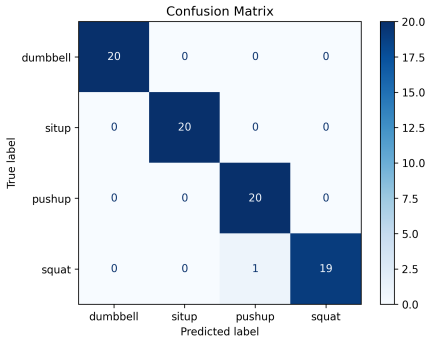
We evaluate our model on both left and right hand of 2 different users to demonstrate that the performance of our system is independent of the user's dominant hand.

Table 2: The classification accuracy of left and right hands

	Left Hand (%)	Right Hand (%)
User 1	98.75	98.75
User 2	100.0	100.0

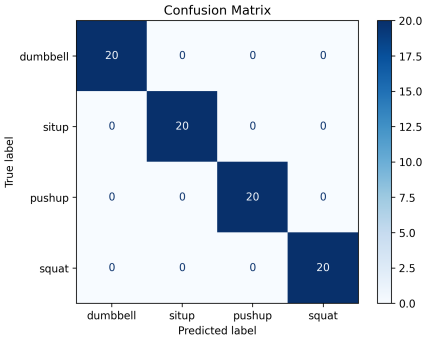


(a) Left Hand

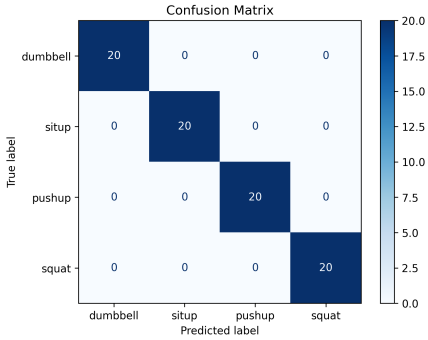


(b) Right Hand

Figure 8: Confusion matrix of User1 on Left and Right Hands



(a) Left Hand



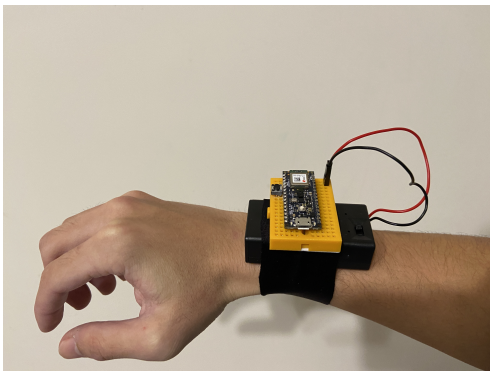
(b) Right Hand

Figure 9: Confusion matrix of User2 on Left and Right Hands

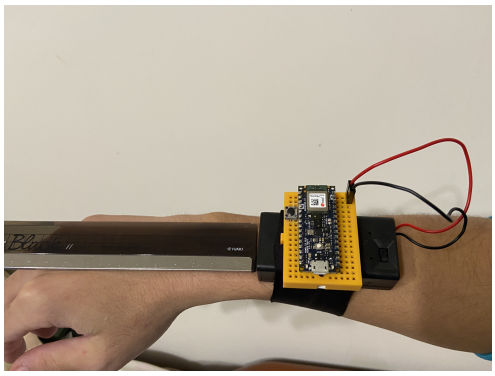
From the comparison of the results between the left and right hands, we can conclude that our system does not restrict the use of either hand. However, there is still an accuracy degradation of approximately 1.25% between user 1 and user 2 as in **Table 2**. This discrepancy is likely due to the training set containing more data from user 2, with the ratio of user 1 to user 2 in the dataset being 1:3. Therefore, optimizing this imbalance will be a key direction for future improvements.

5.3 Different Position

We evaluate our model on both high position and low position of our arms, and we also include different users' result to demonstrate that the performance of our system is independent of the position wearing on our forearm. **Figure 10** shows the device positions on our arms.



(a) Low Position of User1



(b) High Position of User1

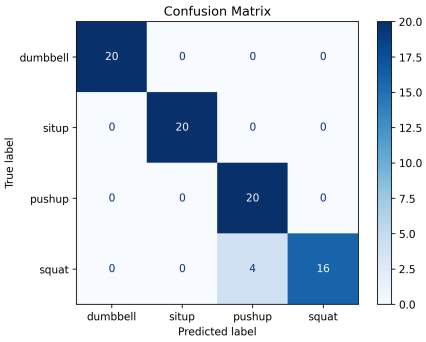
(c) Low Position of User2

(d) High Position of User2

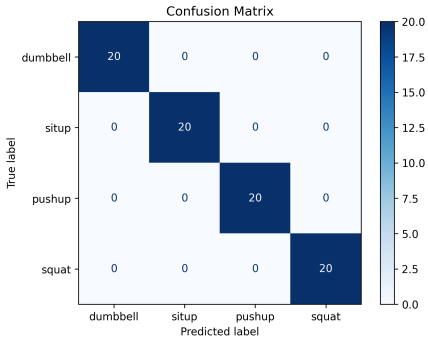
Figure 10: Position of the device on our arms, the low position is close to our wrist, the high position is about 3cm above our wrist. We allow our device to slightly slide on our wrist while performing the exercises.

Table 3: The classification result at high and low positions

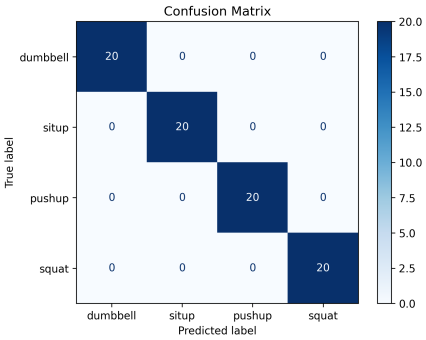
	Low position (%)	High position (%)
User 1	95.0	100.0
User 2	100.0	100.0



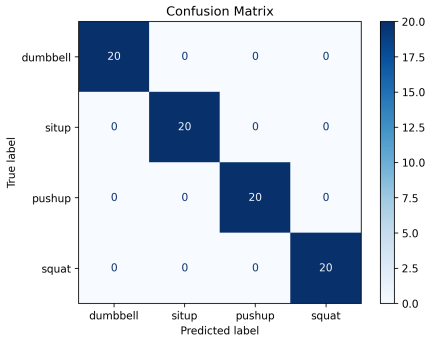
(a) User1 Low Position



(b) User1 High Position



(c) User2 Low Position



(d) User2 High Position

Figure 11: Confusion matrix of 2 users at Low and High positions, we can see that only User1's low position has misclassification.

The results here also show that the position on our forearms doesn't affect the performance of our system. Despite the fact that wearing position does not affect too much, the system is optimized for wrist-mounted usage to ensure consistent performance. Please do not wear our device on other positions of our body.

6 Discussion

6.1 Current Problems

6.1.1 Repetition Counting Error

During inference stage, we apply sliding window on the time series data. There are two parameters regarding sliding window. One is **overlap** and the other is **window size**. The window size is fixed when the model is trained, whereas overlap is a flexible parameter we have to tune by ourselves. Overlap will affect the counting significantly, if it is too small then the count will be less than the real repetitions. However, if the overlap is too big then the count will be much more than the real repetitions. We apply overlap=15 so far, while the speed

of motion for every user is not always the same, and thus the overlap should be tuned based on different users to achieve a more accurate repetitions count of each user. In the future we plan to add the function of overlap setting with three options for user: fast(more overlap), normal(overlap=15), slow(less overlap).

6.1.2 Accuracy Discrepancy Among Users

From the evaluation result, we found that user2 have a stable accuracy of 100% while user1 have a few degradation. Apart from the standardization of movements between the two users, we think that the reason that leads to this result is that the original dataset contains more data from user2, which is about 3/4. The model is more adapt to the movement of user2. Nevertheless, the degradation is only around 1.25% to 5%, so it will not seriously impact users' experience if users do the exercise in a normal movement. Due to the time limit and the scope of this project, we did not conduct IRB user study to collect data from more users. To further improve the robustness of our system, we should conduct an user study including participants with different ages, workout experience, gender, and body characteristics. This will help us build a more comprehensive dataset for training, and thus make our model more generalizable and robust.

6.1.3 Diversity in Exercise Classification

Currently, our system supports a limited set of fitness movements, specifically sit-up, push-up, dumbbell, and squat. While this provides a solid foundation, we aim to expand the capability of our device by incorporating more exercises in the future. As long as the IMU data features are sufficiently distinctive, additional movements can be integrated into the system. This will allow our model to recognize a broader range of exercises, enhance its versatility and make it more applicable to diverse fitness scenarios.

6.2 Cost Estimation

Hardware Cost

- **MCU**
  In our project, we currently use Arduino Nano as our MCU, the cost on Amazon is about 44.50 USD, which is approximately 1500 TWD. However, Arduino Nano is definitely a too powerful MCU for our project, since we do not use all the other sensors on the Arduino Nano board.

- **Max30102**
  One can purchase Max30102 at a cost of 3.25 USD, which is approximately 107 TWD.

- **SSD1306**
  One can purchase Max30102 at a cost of 6.99 USD, which is approximately 230 TWD.

To reduce the cost, we can focus on the MCU, since it costs the most in our project. We suggest to use **ESP-32S** as MCU and **MPU6050** as IMU. ESP-32S is a powerful MCU which supports bluetooth and even WiFi. It also supports I^2C, which meets our requirements. ESP-32S costs 9.99 USD on Amazon, which is approximately 330 TWD. With this MCU, we need IMU since it is not inbuilt. MPU6050 is an 6 axes IMU which fits our need. One can buy MPU6050 on Amazon with 4.44 USD, which is approximately 150 TWD.

Software Cost

- **Gemini Api**
  Our prompt integrates users' information, which is about 105 tokens, the reply of Gemini api cost approximately 500 tokens. If there is only one user, the free amount is definitely enough. If we use out the amount, Gemini also provide a "pay-as-you-go" service. The pricing for Gemini 1.5 Flash is input 0.075 / 1 million tokens, output 0.30 / 1 million tokens.

  Let's assume a user workout everyday, each day ask for the suggestion 3 times. The cost of the user per month is $3 \times 30 \times (0.075 \times 105 \div 1,000,000 + 0.30 \times 500 \div 1,000,000) \approx 0.015$ USD, about 0.5 TWD.[9]

- **Website Deploy**
  We use **Vercel** to deploy our website, and the service is definitely enough for at least 100 users. If we would like to upgrade our plan, Vercel provide a **Pro** plan which cost 20 USD/month, that is to say, about 650 TWD.[10]

- **Firebase Firestore**
  Firebase offers a free amount of 1GB storage, read 50,000 times/day, write 20,000 times/day, delete 20,000 times/day, outbound data transfer 10 GB/month, which is definitely enough for at least 100 users. If out of quota, firestore charges by location. In Taiwan, the price is read 0.0345 USD/100,000 documents, write 0.1042 USD/100,000 documents, delete

[9] Gemini api pricing: `https://ai.google.dev/pricing#1_5flash`
[10] Vercel pricing page: `https://vercel.com/pricing`

0.0115 USD/100,000 documents, data storage 0.1725 USD/per GB per month. For more information, please refer to the pricing page.[11]

- **Hugging Face Deploy**
  We use Hugging Face space for our classification deployment, the free plan is also definitely enough for a single user, while if many people want to use the classification model at the same time, we may need GPU during the inference stage. Despite the fact that our model can inference on CPU in a short period of time, when it comes to lots of users, GPU is a necessity. The pro account cost 9 USD/month, about 300 TWD.[12]

Total Cost

To get the sum of our system, there are still some things to take into account. First, considering the 9 Volt battery, the battery case, the stretchy fabric, the button and the soft wire, velcro, the bread board, and the possible encapsulation, our hardware cost should add an extra 250 TWD. Second, although for personal use, the software cost is neglectable, we still add 200 NTD as a buffer. If we use the original Arduino Nano design, our cost could definitely lower than 2500 NTD, and if we take a cost down version, the total cost is as below.
$\text{Total Cost} = 330(\text{ESP-32S}) + 150(\text{MPU6050}) + 107(\text{Max30102}) + 230(\text{SSD1306}) + 250(\text{Others}) + 200(\text{software}) = 1267$ NTD.

This amount is definitely overestimate the cost, and the cost can be even lower when it comes to mass production. We leave this for future when it really comes to production.

6.3 Future Work

LLM Integration

So far, we have primarily utilized LLM as a workout adviser by providing body information and daily calorie goals. However, the current limitation lies in the fact that our system only supports classification of four exercise types and does not fully exploit the vast potential of LLM. Therefore, the future direction is to expand the functionality of LLM integration.

---

[11] Firebase pricing page: `https://firebase.google.com/docs/firestore/pricing#network`

[12] Hugging Face pricing page: `https://huggingface.co/pricing?utm_source=chatgpt.com`

1. **Leveraging Pre-trained Foundation Models**
   Utilize pre-trained foundation models that employ contrastive learning between IMU data and textual descriptions. This approach enables open-set exercise recognition, significantly enhancing the diversity of exercise classes that our system can handle.

2. **Transforming IMU Signals into Spectrograms**
   Convert raw IMU signals into spectrograms using techniques like STFT to extract both temporal and frequency domain features. Develop a transformer-based IMU encoder with positional encoding and self-attention mechanisms to process these spectrograms, effectively capturing relationships across time and frequency. Train the encoder with labeled datasets to map spectrograms to specific outputs, such as exercise categories or quality metrics. Validate its performance with metrics like accuracy or RMSE, and integrate the encoder into real-time applications. Additionally, combine the system with LLMs to provide user-friendly feedback and actionable insights.

3. **Fitness Evaluation**
   By taking IMU data as input and creating a custom embedding, it is possible to integrate LLM for fitness evaluation. [4] proves it is feasible to do so. We aim to design a LLM architecture which is able to accept both IMU-derived embedding and textual inputs in the future. By doing so, users can get feedbacks from LLM according to the IMU data while they are performing exercise, which can further help users enhance their fitness performance as a private coach.

Exercise Diversity

Currently, our system supports four basic exercises: sit-ups, push-ups, dumbbell curls, and squats. To further enhance its utility, the system can be expanded to include additional commonly performed fitness exercises. Future works could focus on integrating the following movements.

1. **Lunges**
   Incorporate forward and backward lunges to target the lower body and improve balance.

2. **Plank Variations**
   Add planks and side planks to evaluate core strength and stability during static holds.

3. **Jumping Jacks**
   Include jumping jacks as a simple cardiovascular exercise to broaden functionality.

4. **Step-Ups**
   Integrate step-up exercises to simulate stair-climbing movements, enhancing lower body strength.

5. **Bridge Pose**
   Track bridge exercises to focus on glute activation and lower back strength.

By including these additional exercises, the system can cater to a wider range of fitness routines while maintaining its practicality for home use.

Massive User Study

Currently, our user study involves two participants, testing the device on both hands and at varying positions (upper wrist and lower wrist). While these initial tests have provided valuable insights into the system's robustness, a larger and more diverse user study is necessary to validate the system's generalizability and accuracy. Future works could focus on the following.

1. **Expanding Participant Demographics**
   Include users with different fitness levels, ages, and genders to ensure the system performs consistently across diverse populations.

2. **Testing in Real-World Scenarios**
   Evaluate the system in realistic workout environments, such as gyms, homes, or outdoor settings, to test its reliability under varying conditions.

3. **Longitudinal Studies**
   Conduct long-term studies to observe how the system adapts to user behavior changes over time and how its performance evolves with extended use.

4. **Collecting Larger Datasets**
   Expand the dataset by involving more users and recording various motion patterns to train and refine the classification models, reducing potential biases from limited data.

Accuracy Enhancement

Currently, our system achieves reasonable accuracy by leveraging IMU data from a single device positioned on either hand. However, there is significant po-

tential to further improve classification accuracy and motion capture precision. Future works could focus on the following.

1. **Multi-Device Synchronization**
   Investigate the use of multiple devices placed on different body parts (e.g., both wrists, ankles, or torso) to improve motion capture fidelity. Synchronizing data across these devices could provide a more comprehensive view of body movements, thereby enhancing classification accuracy for complex exercises.

2. **Inference Overlap Calibration**
   Introduce overlap calibration during inference to ensure smoother transitions between overlapping motion segments. By aligning overlapping windows of IMU data, the system can mitigate boundary effects, reduce noise, and improve the consistency of motion detection across frames.

3. **Adaptive Thresholding**
   Implement adaptive confidence thresholds for classification, dynamically adjusting based on motion intensity or signal quality. This could reduce false positives and improve accuracy in scenarios with highly dynamic or subtle movements.

4. **Noise Filtering and Signal Enhancement**
   Apply advanced noise filtering techniques, such as wavelet denoising or low-pass filtering, to preprocess raw IMU signals. Enhanced signal clarity will allow the system to detect subtle movements more reliably.

5. **Model Optimization for User Variability**
   Train models with larger and more diverse datasets to account for inter-user variability, including differences in motion patterns, body size, and exercise habits. This ensures robust performance across a wide range of users.

7 Contribution

7.1 Affordable and Accessible Hardware Development

BurnSync is a low-cost system and can be made below a total expense of **1300 TWD** , making fitness IMU solutions accessible to more users. The integration of essential hardware components like buttons, MAX30102 heart rate modules, SSD1306 OLED displays, and BLE communication provides seamless connectivity and usability.

7.2 Innovative Insights into Fitness IMU Data

We provide our insights about fitness IMU data. we discovered that IMU data is robust to variations, with left or right hand placement not affecting classification results, offering flexibility in user experience. Our work also enabled reliable motion tracking and analysis through IMU data, contributing to deeper understanding and application potential in fitness scenarios.

7.3 AI and LLM Integration with High Development Flexibility

We host our custom machine learning models on platforms like **Hugging Face Space** and integrated **Gradio APIs** for user interaction. We also leverage **Gemini LLM API** to provide intelligent, personalized insights into workout quality and recommendations. Last but not least, we developed a responsive website for a user-friendly interface, with features like authentication, database connections, and mobile-first design, enhancing the system's adaptability and scalability for diverse development needs.

8 Related Work

8.1 Smart Watch

There are many commercial of-the-shelf (COTS) smart watches that are capable of preforming fitness tracking. To name a few, Apple watch, Garmin, Xiaomi are able to do some kind of fitness tracking. However, not all of the smart watches have the function of counting sets and repetitions.

Most of the smart watches have other additional functions, for instance, reading message, alarm, stopwatch function, and thus cost a lot, which is not affordable to many customers. Our device is simple and relative low cost, and there is a really high chance to further reduce the cost of our device. If the customer only want a simple device for fitness tracking, our device will be a better choice than the COTS smart watches.

8.2 Fitness Tracking & Wearable Devices

Fitness tracking has been a popular theme for several years. The development of machine learning also creates more possibilities for fitness tracking. Some works focus on fitness classification[8][10], while others focus on fitness evaluation[1]. The methods also vary a lot, in addition to IMUs or cameras, some works use bio-impedance [6] to further improve the classification accu-

racy. There is also work using mmWave for exercise classification [10]. We believe there is still lots of potential in this topic, and the improvement of fitness tracking can further improve the life of human being.

8.3 Attention & Convolution Model

In our work, we use IMU data, which is a 1 dimensional data with six channels, and we choose to use a structure combines attention model with convolution model. In [6], they use a similar structure with ours. However, the combination of attention and convolution is not only used in 1 dimensional data, but also in 2 dimensional data[12]. This structure may also be used in other fields.

8.4 LLM

Large language models (LLMs) have demonstrated remarkable capabilities across various applications. Since the success of ChatGPT, people keep trying new applications of LLM. There are several works utilizing LLM for health evaluation [11][2]. We believe LLMs can do much more than what have be done and we leave this for future as previous mentioned.

9 Conclusion

In this work, we propose **BurnSync**, an Arduino Nano based smart device which is able to do exercise classification, sets/repetition counting, calories estimation, fitness recommendation and heart rate detection. We design a custom website and deploy our model on Hugging Face Space. The whole system works as a personal workout assistant which is ubiquitous, highly connected, and low-cost. Our work offers a high flexibility and it is highly potential for future development.

10 Appendix

10.1 Division of Labor

Both

- Project idea brainstorming
- Final report writing

Hsuan-Ying, Liu

- Architecture design
- Hardware design
- Arduino OLED & MAX30102 configuration
- Website design & deployment
- $\frac{1}{4}$ Dataset
- Final presentation ppt & report

Chi-En, Dai

- Arduino IMU & BLE configuration
- $\frac{3}{4}$ Dataset
- Classification model design & training
- Hugging Face model deployment
- Model evaluation
- Proposal presentation

References

[1] Chin-Chih Chang, Chi-Hung Wei, Hao-Wei Wu, and Sean Hsiao. A fitness movement evaluation system using deep learning. In *2023 15th International Conference on Knowledge and Systems Engineering (KSE)*, pages 1–6, 2023. doi: 10.1109/KSE59128.2023.10299493.

[2] Justin Cosentino, Anastasiya Belyaeva, Xin Liu, Nicholas A. Furlotte, Zhun Yang, Chace Lee, Erik Schenck, Yojan Patel, Jian Cui, Logan Douglas Schneider, Robby Bryant, Ryan G. Gomes, Allen Jiang, Roy Lee, Yun Liu, Javier Perez, Jameson K. Rogers, Cathy Speed, Shyam Tailor, Megan Walker, Jeffrey Yu, Tim Althoff, Conor Heneghan, John Hernandez, Mark Malhotra, Leor Stern, Yossi Matias, Greg S. Corrado, Shwetak Patel, Shravya Shetty, Jiening Zhan, Shruthi Prabhakara, Daniel McDuff, and Cory Y. McLean. Towards a Personal Health Large Language Model. *arXiv e-prints*, art. arXiv:2406.06474, June 2024. doi: 10.48550/arXiv.2406.06474.

[3] Alrick B. Hertzman. Photoelectric plethysmography of the nasal septum in man. *Proceedings of the Society for Experimental Biology and Medicine*, 37 (2):290–292, 1937. doi: 10.3181/00379727-37-9543P. URL https://doi.org/10.3181/00379727-37-9543P.

[4] Sheikh Asif Imran, Mohammad Nur Hossain Khan, Subrata Biswas, and Bashima Islam. LLaSA: A Multimodal LLM for Human Activity Analysis Through Wearable and Smartphone Sensors. *arXiv e-prints*, art. arXiv:2406.14498, June 2024. doi: 10.48550/arXiv.2406.14498.

[5] Hitesh Kotte, Florian Daiber, Milos Kravcik, and Nghia Duong-Trung. Fitsight: Tracking and feedback engine for personalized fitness training. In *Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization*, UMAP '24, page 223–231, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400704338. doi: 10.1145/3627043.3659547. URL https://doi.org/10.1145/3627043.3659547.

[6] Mengxi Liu, Vitor Fortes Rey, Yu Zhang, Lala Shakti Swarup Ray, Bo Zhou, and Paul Lukowicz. iMove: Exploring Bio-impedance Sensing for Fitness Activity Recognition. *arXiv e-prints*, art. arXiv:2402.09445, January 2024. doi: 10.48550/arXiv.2402.09445.

[7] Dan Morris, T. Scott Saponas, Andrew Guillory, and Ilya Kelner. Recofit: using a wearable sensor to find, recognize, and count repetitive exercises. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, page 3225–3234, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450324731. doi: 10.1145/2556288.2557116. URL https://doi.org/10.1145/2556288.2557116.

[8] Xiaojie Mu and Cheol-Hong Min. Wearable sensing and physical exercise recognition. In *2022 IEEE World AI IoT Congress (AIIoT)*, pages 413–417, 2022. doi: 10.1109/AIIoT54504.2022.9817342.

[9] Aparajita Saraf, Seungwhan Moon, and Andrea Madotto. A survey of datasets, applications, and models for imu sensor signals. In *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*, pages 1–5, 2023. doi: 10.1109/ICASSPW59220.2023.10193365.

[10] Edward M Sitar and Sanjib Sur. Millifit: Millimeter-wave wireless sensing based at-home exercise classification. In *2022 18th International Conference on Mobility, Sensing and Networking (MSN)*, pages 150–154, 2022. doi: 10.1109/MSN57253.2022.00036.

[11] Jiankai Tang, Kegang Wang, Hongming Hu, Xiyuxing Zhang, Peiyu Wang, Xin Liu, and Yuntao Wang. ALPHA: AnomaLous Physiological Health Assessment Using Large Language Models. *arXiv e-prints*, art. arXiv:2311.12524, November 2023. doi: 10.48550/arXiv.2311.12524.

[12] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. CBAM: Convolutional Block Attention Module. *arXiv e-prints*, art. arXiv:1807.06521, July 2018. doi: 10.48550/arXiv.1807.06521.