Oulu Zhang

Lab 4 Report - Magic Wand

Part 1: Data Collection

Discussion: Why should you use training data collected by multiple students rather than using your own collected data only? Think about the effectiveness and reliability of your wand.

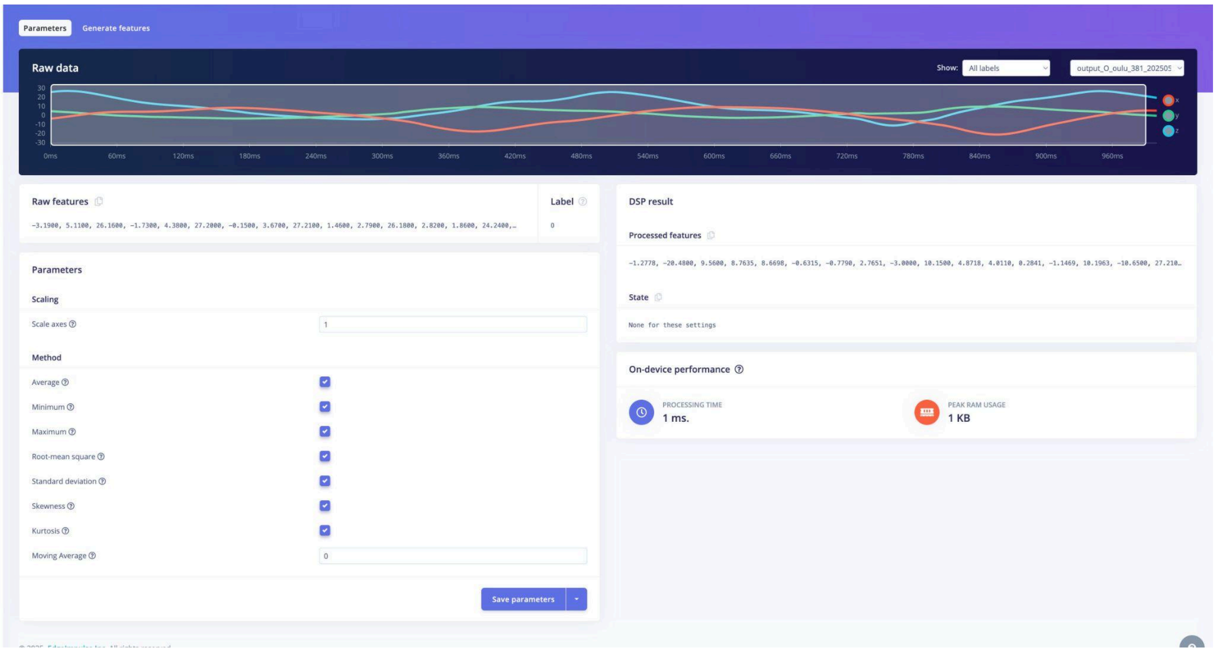
Using training data collected from multiple classmates can significantly improve the accuracy of a gesture recognition model. If the model is trained based only on one user's own data, it is easy to overfit to that user's specific movement habits, such as the size of the hand, the magnitude of the movement, the speed, or subtle differences in sensor placement. In this way, the accuracy of the wand model may drop significantly when it is used against other people, or when the same person uses it in different scenarios.

Part 2: Edge Impulse Model Development

2. Discussion: Discuss the effect of window size.

The window size determines the length of each inference capture. Larger windows capture complete, slow or complex gestures better, but increase the input dimension and model complexity. A smaller window generates more samples and responds faster, but may miss some of the action features, affecting the recognition results. A 1000ms window was chosen for this project to balance recognition accuracy with operational efficiency on the ESP32.

3. Choose your DSP block in the sidebar.
Tune the hyperparameters and visualize the generated features until you are satisfied with the features.

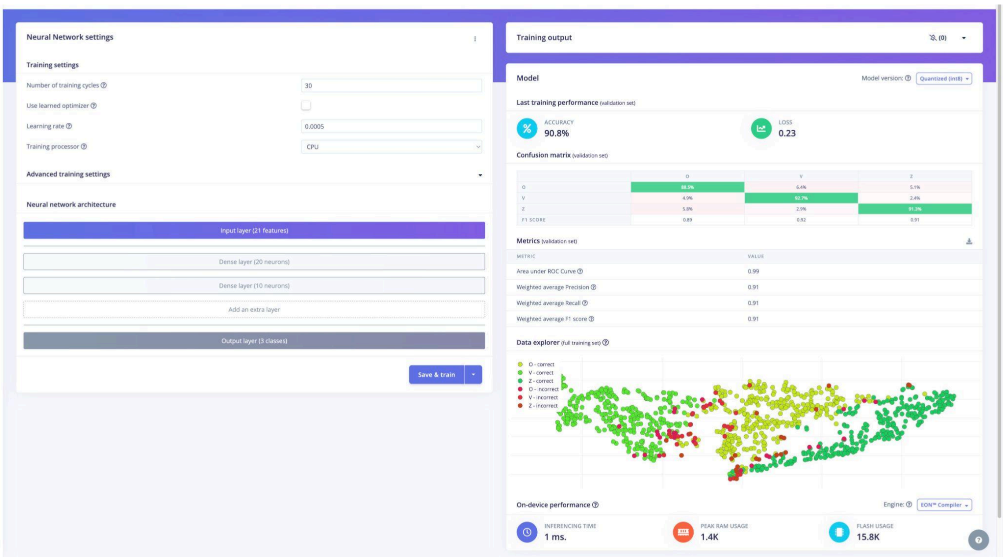


Take a screenshot of your generated features, and sketch a rough decision boundary between classes. Explain why do you believe the generated features are good enough.

I chose Flatten as the DSP module and enabled statistical features such as mean, maximum, minimum, and standard deviation. These features can effectively summarize the magnitude and trend of the gesture. From the feature distribution graph, I can see that the overall clustering of “O”, ‘V’ and “Z” gestures is clear, with good differentiation and less misclassification. The inference time is 1ms and RAM occupies 1.4KB, which is suitable for real-time deployment on ESP32. Therefore, we believe that this feature set achieves a good balance between accuracy and efficiency.

4. Choose your ML block in the side bar.

Tune the number of training epochs, learning rate, and neural network architecture until you are satisfied with the learning performance.



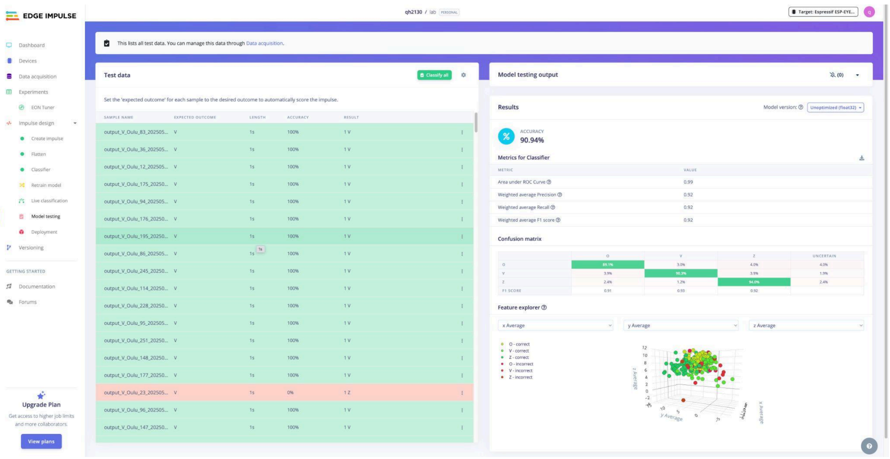
Report the learning performance, your choices of hyper-parameters, and architecture.

In the ML block, I used a three-layer neural network architecture: containing 21 input features, two fully connected layers (20 and 10 neurons), and one output layer corresponding to the three gesture categories (O, V, Z). The training parameters were set to 30 training rounds (epochs) and a learning rate of 0.0005.

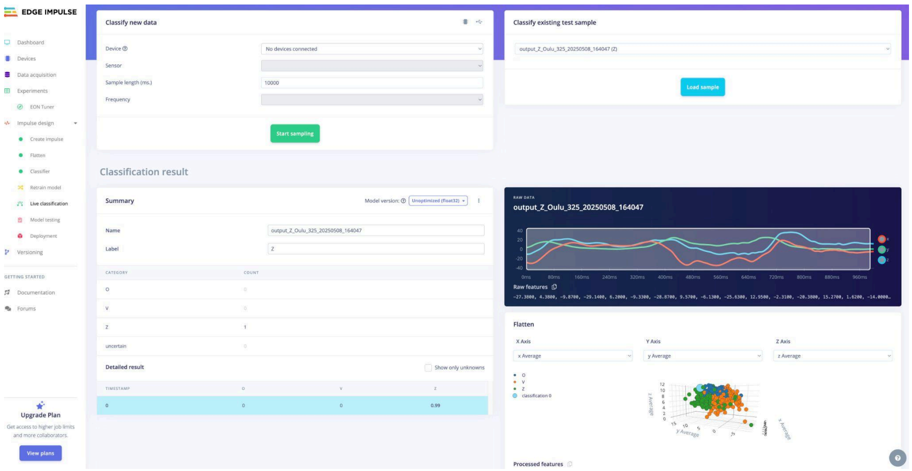
The final model achieves an accuracy of 90.8% on the validation set with an F1 score of 0.91, with good recognition rates and less confusion between categories. The model also maintains an inference time of 1ms and a RAM footprint of only 1.4KB, which is a good deployment efficiency on resource-constrained devices such as ESP32.

5. Use "Live classification" and "Model testing" in sidebar to test your model performance. Please clearly document all metrics being used, e.g., accuracy, TP, FP, F1, etc.

model testing:



live classification:



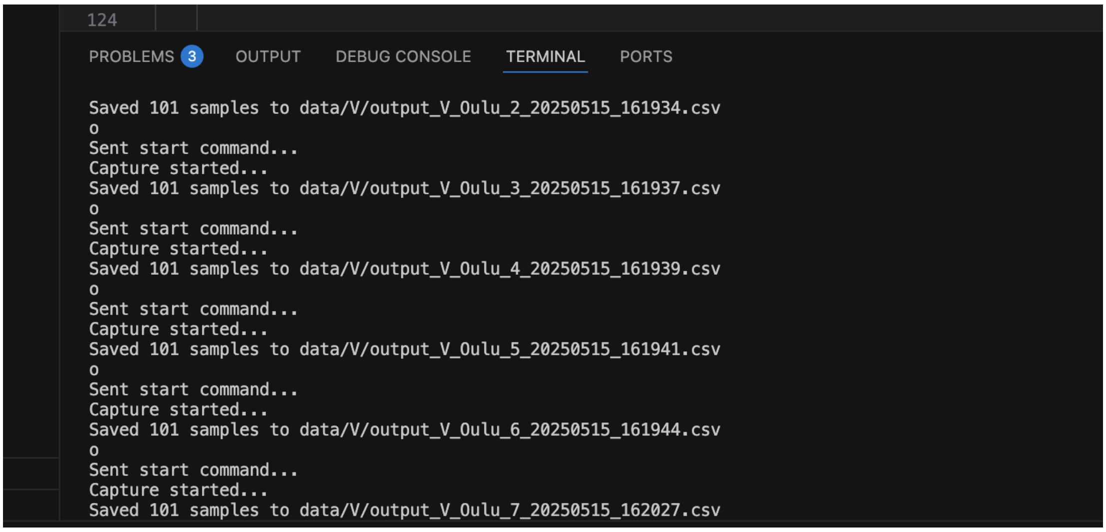
In the model testing module, after using automatic classification of all test samples, the accuracy of the model on the overall test set is **90.94%**, and the performance of each index is as follows: Precision: 0.92; Recall: 0.92; F1 score: 0.92; AUC value: 0.99.

7. Discussion: Give at least two potential strategies to further enhance your model performance.

1) Increase data diversity: By collecting gesture data from different users or slightly perturbing the existing data (e.g., adding noise, time offset), the generalization ability of the model can be enhanced and the risk of overfitting can be reduced.
2) Optimize model structure and parameters: Trying to adjust the number of neural network layers, neurons, or learning rate, or introducing more complex DSP features (e.g., frequency-domain features) can help the model capture gesture differences more accurately, thus improving the recognition rate.

Part 3: ESP32 Implementation

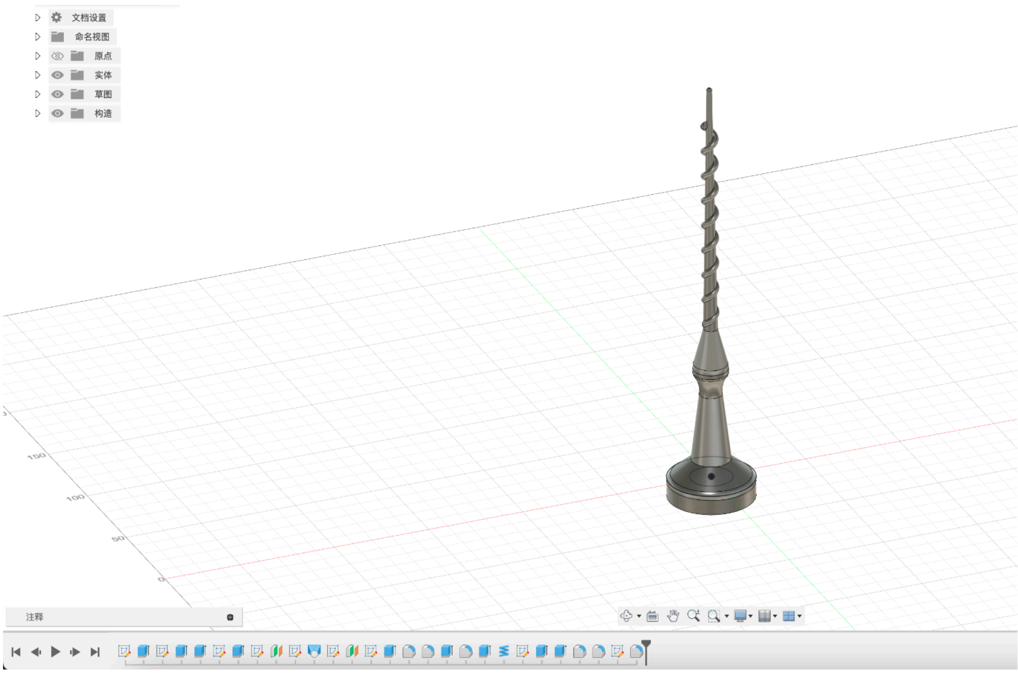
Demonstrate data collection:
https://drive.google.com/file/d/1NCtxh6_Y1nBH4ZjhpyurmRlt5EkZvOXD/view?usp=sharing



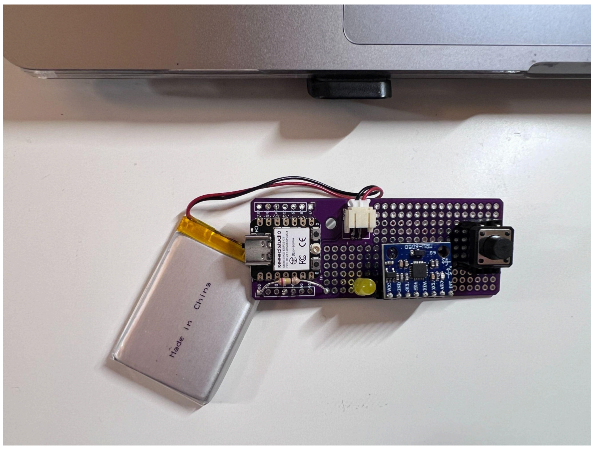
Demo video showing your wand:
https://drive.google.com/file/d/19gMnqKbXEtNmqBVV2HH_ILqrEL-0xDjh/view?usp=drive_link

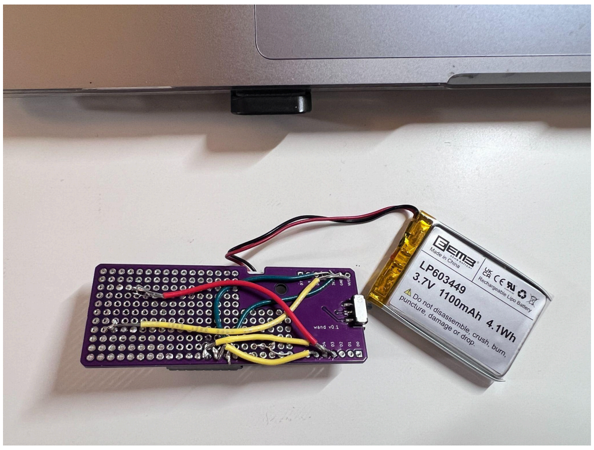
Part 4: Battery and Enclosure

enclosure:



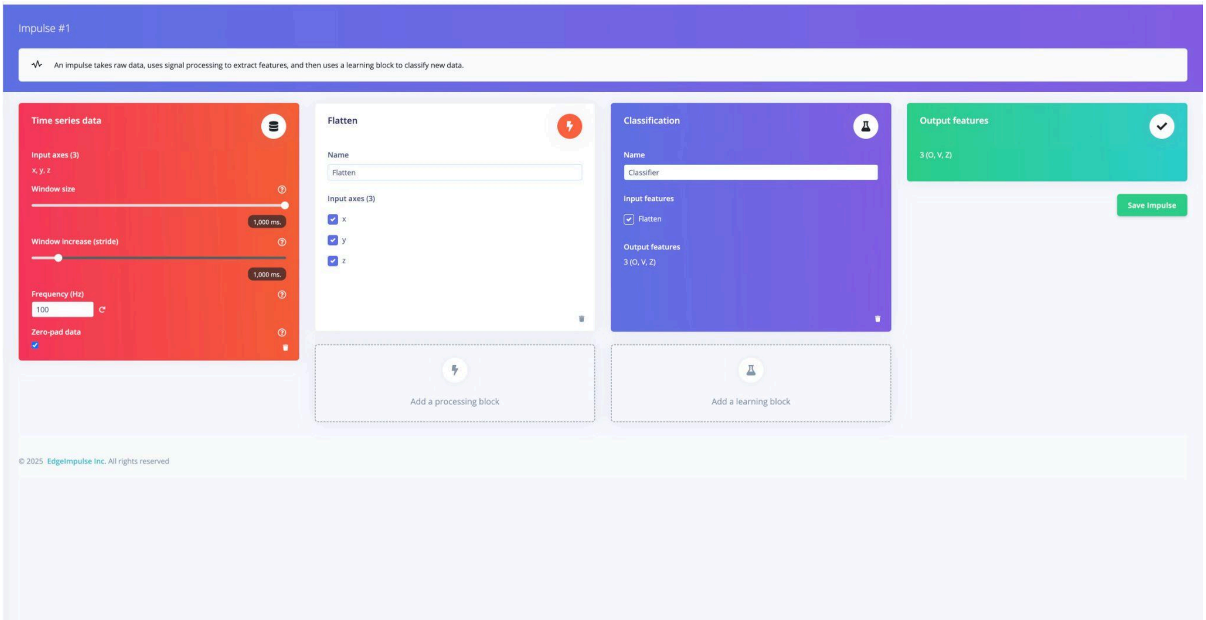
hardware setup:





Summary:

1. Edge Impulse model architecture and optimization



2. Performance analysis and metrics (see Part 2 Q5)
3. Challenges faced and solutions

   In the initial data collection, there were large differences between different samples, which led to unstable performance of the model after training. I improved the generalization ability and accuracy of the model by re-standardizing the gesture movements, controlling the speed and unifying the direction, as well as combining the training with other students' data.