Smart Home Appliance controlling by Hand Signature

Bui Le Phi Long 16619
Nguyen Duc Thang 15656

Vietnamese German University

Machine Learning Project Seminar Report

December 7, 2022

Smart Home Appliance controlling by Hand Signature

1 Introduction

In the modern day, civilized technology is growing like a whirlwind, we are progressively reaching the pinnacle of technology by developing products that make life easier for people. The family's smart home is one of the most concerning topics. Few electronic gadgets employ handwriting as their primary means of control today, with the bulk of smart homes relying instead on users operating their electronic appliances via voice commands or smartphone applications [9].
Users do not always want to use speech to manage their devices, and since everyone has a unique voice, it is easy for things to go wrong when Alexa or Google Home misinterprets a user's tone or uses the improper phrase structure, which results in inaccurate device control behavior. Therefore, for users who are uncomfortable using their voice, simple gestures may be a preferable option for managing gadgets. Our project is made to demonstrate a new Hand Signature Recognition-based method for utilizing and managing smart electronic gadgets in the house.

1.1 Problem Statement

There are not too many alternative solutions for controlling smart home. At the moment, appliance controlling by mobile application or voices with help from Home Assistant are two of the most popular approaches. However, when being used, the current system might have been facing some of these problems

1. Sequence of action of using mobile application makes user feel giving too much effort for controlling one device. To be more precisely, sometimes user just wish to turn off a light, they must go to different steps inside an application to find a button to turn off, while they can just quickly turn the physical button off in real life, which is much comfortable and faster.
2. User might have language barrier when try to use voice recognition system.[7]
3. Health issues, such as a sore throat, can prevent a user from using voice recognition properly
4. User could have disability in speaking, deaf or mute people do not have ability to use the Home Assistant.

For those kinds of problems, we decide to develop a system as an alternative solution for user to control their household appliances.
For the system to operate smoothly and robustly, accuracy and response time are the two factors that matter most in an electronic control system for the house. The real challenge is how can we detect Hand Gesture signals in real time as rapidly and precisely as feasible in order to execute these two tasks well?

1.2 Objective

The project aims to create greatly accurate hand signature recognition system, which is the most important foundation of building a good home automation system. To ensure that, the system should be able to easily learn one-hand signs from users easily, and most importantly is minimizing the latency of responding time and maximizing the accuracy of commands recognition.

2 Methodology

This section illustrates the progress of creating the system.
At the early stage, we decided to use YoloV5 as the main model in charge of hand signatures recognition. However, after spending more time on the project, we have come to the realization that YoloV5 has several drawbacks and presents us with some significant obstacles that might have an impact on the project's outcome. The section *2.1 Challenges* describes these problems in more details. We made the decision to switch from the YoloV5 recognition model to the Mediapipes Hand model.
Figure 1 demonstrates the general pipeline flow of our model:

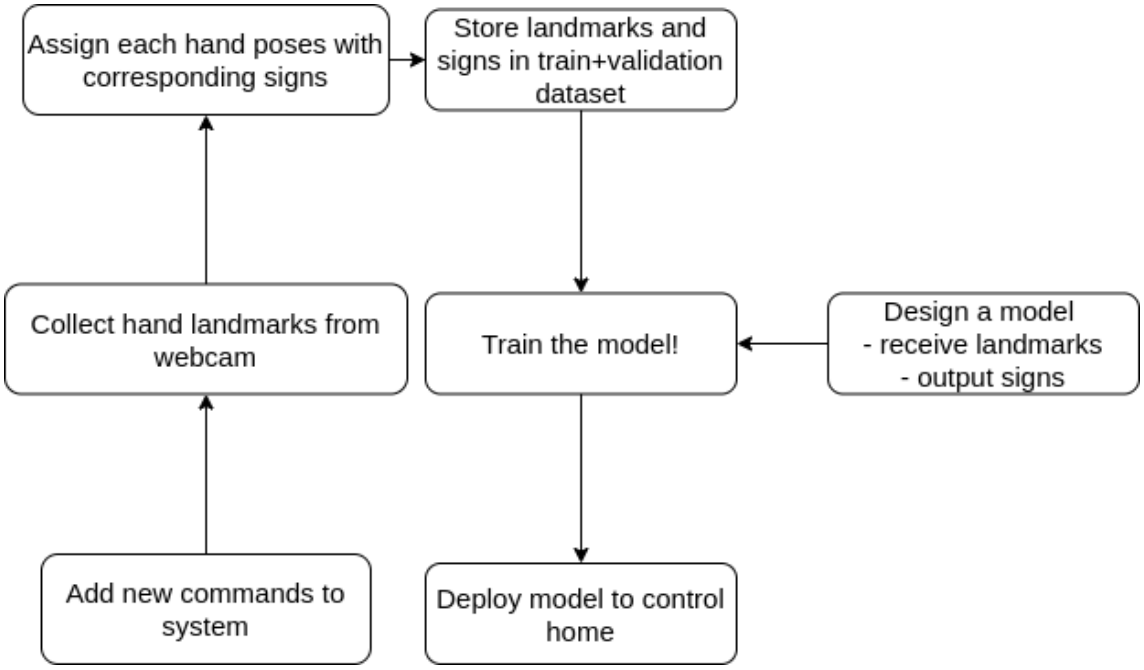


Figure 1: Pipeline of the project

The system is expected to pick up several new instructions in the first phase. To enable it to accomplish that, we must first gather hand landmarks from images taken by mobile devices' cameras. Later, Mediapipes Hand processes these images to create a collection of "hand landmarks" format. Each hand stance is given a sign that corresponds to it, and these signs are then stored in the train and validation data sets when we have collected all the hand landmarks for the "command" that we would want the system to learn. Next step is the training process, we need to design a model to receive landmarks and output the signs. We create the procedure of training that model after designing it. Finally we receive a model that recognize the learned hand signatures and ready to be deployed for the household appliances control system.

2.1 Overall system activity flow

The figure below illustrates the normally expected flow of our system:

Whenever a User want to use the system to control the household appliance, he/she should be near the System Video Recording devices, for example a wall-sticking Camera. If A user make a hand gesture, the camera will extract the images then send those raw images to the System. The system will recognize user's hand and extract landmarks point on it, then send these landmarks to a

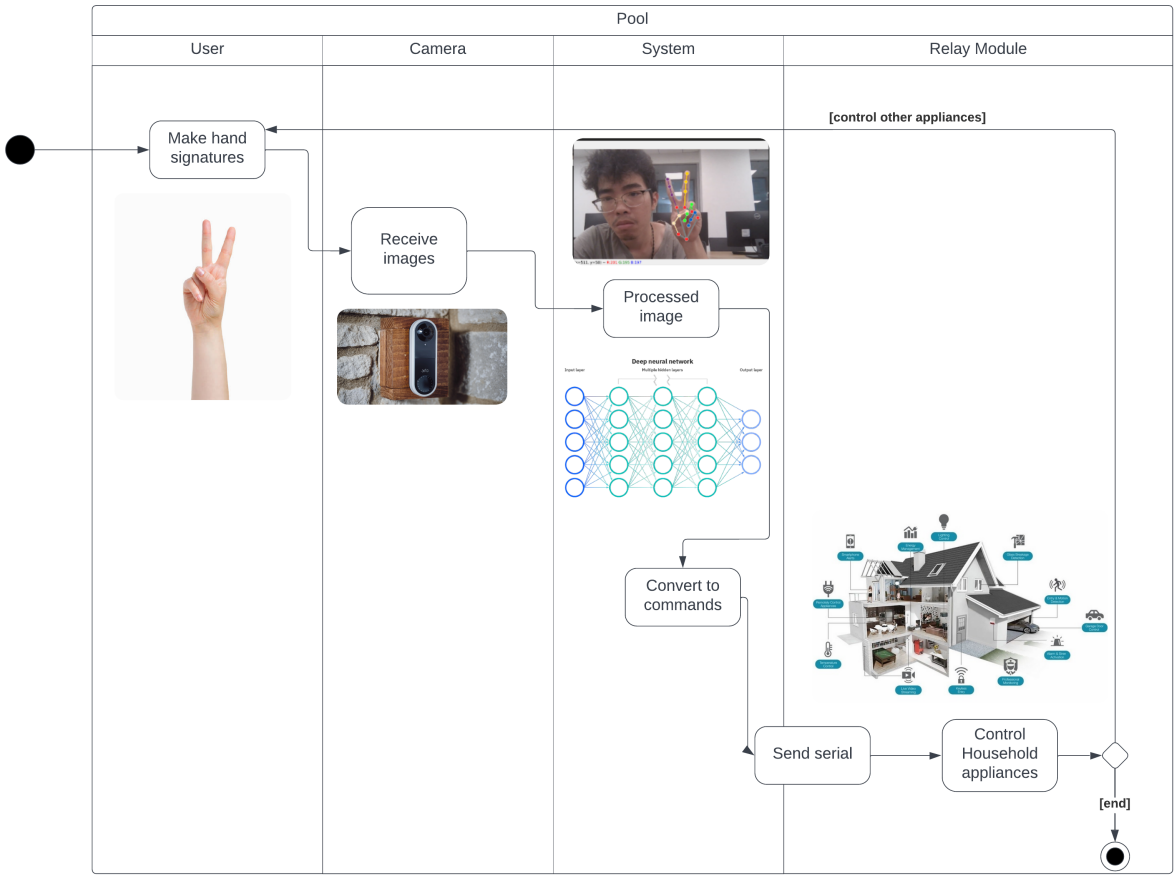


Figure 2: System flows

Neural Network system that responsible for converting these landmarks into commands. Later, the system sends these converted commands as serial signal to a relay module, which is in charge of controlling household appliances.

2.2 Challenges

2.2.1 Challenges with YoloV5

When it comes to YoloV5, there are several problems that we have met. First of all, the training time of YoloV5 takes too long, which creates pressure on our limited performance devices for training data. Also, it makes it more difficult when deploying and testing new commands or new Hand Signatures since we must waste too much time waiting for the training process result to re-validate the model quality. Furthermore, YoloV5 asks for large size datasets but the result of training model has low accuracy rate of recognition, which dramatically decreases the robustness of the system. Secondly, there are not many supporting validation sets for this type of recognition model available for YoloV5. We realize that YoloV5 at this time is not appropriate for our project and we made a decision to give Mediapipe Hand a try, which is described in the section *2.2 Analyse and Design System*.

2.2.2 Method of data collection

The difficulty in practically all computer vision projects is this. The amount and quality of images needed for data collection are very high due to the divergent nature of hand signatures and the fact that every individual's hand shape is very unique compared to those of others. For such, data collection must be accurate and automatic. The acquired datasets also need to be simple to examine and filter out any faulty data, which will improve the robustness of the model.

2.2.3 Out of Distribution Detection

This merely implies that we must figure out how to identify a command built from an unrecognized signature. We should construct a system that can filter out scenarios where it is unable recognize commands or a certain signature input from users since there are numerous potential outcomes and flawed scenarios when it comes to the issue of classification. Such feature, known as null classification, aids in boosting the system's robustness.

2.2.4 Realtime Performance

Since our project falls under the category of Internet of Things Artificial Intelligence, real-time system operation is crucial. A system must process input images and generate commands to manage home appliances in real-time while adhering to and meeting deadlines. Any strange behavior that results in serious bugs for the system and gives the user a bad experience will be created if an action does not meet the system's deadline.

2.3 Data collection and processing

Firstly, We created a small script to use **existing model**, which called hand detection from mediapipe, described later in section *2.2.1 Tools and Technology Used*, to automatically identify the hands while drawing bounding box around the hand. Processed image would be sent to Roboflow platform afterwhile. For each frame with hand signs, we will assign it to one of 3 different datasets: training, validation and test with probability 80%,10% 10% .This results in a problem called Train/Test Bleed, where the data in validation set and test set are too similar to train set) [10] Also,we attempt to create commands using two hands, but it turns out that our hands are so entwined that the mediapipe model has trouble accurately identifying our hands. Consequently, several images had inaccurate bounding boxes. Therefore, before using it as a training set in Roboflow, we must tweak many of them once more. Readjusting the label eventually became too laborious.

What would be a fantastic response to this issue?

- We adjusted our system to work with one hand instead of two, as the Mediapipe hand can detect one hand with higher accuracy.
- Instead of sending the whole image and the label to Roboflow, we will now extract only the hand landmarks and save it in a local file, which is much less in terms of data size. We described how to process the landmarks in Section 2.3.2.

Figure 3: Example of data with correct labels, but inaccurate bounding boxes. The boxes does not fully cover the handss

2.3.1 Tools and Technology Used

- **pytorch**:a machine learning framework [8] that is open source and speeds up the transition from research prototype to deployment in the real world. We use pytorch for creating and training our recognition model

- **mediapipe**: A cross-platform pipeline framework called MediaPipe can be used to create unique machine learning solutions [4] for live and streaming media. The framework, which Google open-sourced, is currently in alpha. We use mediapipe for using the hand detection model which mainly produces hand landmarks that is important for the system to learn.

- **roboflow**: A technology in the field of deep learning called Roboflow makes the computer vision process easier [1]. No of their level of expertise or experience, it enables developers to create their own computer vision apps. Models for object identification and classification are supported. We use Roboflow as an Online Annotation tools and as a place to store our data sets.

- **opencv**: A computer vision and machine learning software library called OpenCV is available for free use . A standard infrastructure for computer vision applications was created with OpenCV [2] in order to speed up the incorporation of artificial intelligence into products. OpenCV makes it simple for businesses to use and modify the code since it is a product with an Apache 2 license. We use opencv for auto labelling and inferencing process.

- **pytorch ood**: Out-of-Distribution (OOD) Detection research as well as related fields like Open-Set Recognition, Novelty Detection, Confidence Estimation, and Anomaly Detection based on Deep Neural Networks [3] will be accelerated by a PyTorch-based library. We use this for Out of Distribution detection task.

- **tensorboard**: The visualization and tools required for machine learning experimentation are provided by TensorBoard:tracking and displaying data like accuracy and loss, modeling graph visualization (ops and layers),observing the evolution of the histograms of the weights, biases,

or other tensors,embedding projection to a low-dimensional space and displaying text, audio, and visual data. We use tensorboard for visualizing metrics [5].

2.3.2 Other technique to increase data samples

We made a python script that can automatically collect command given a hand visible in the webcam. Figure 4 describe what this script does when run. The pipeline is as below:

1. The script chooses which data set to save in (train,validation,test)
2. The script open the webcam and detect a hand along with its landmark, also prompt user to press a character key.
3. When user press a character key, the script, from the current frame until the user cancel, will assign the landmarks to the command linked to the key, and save the data to the specified data set.
4. User press the key again to stop the assignment.

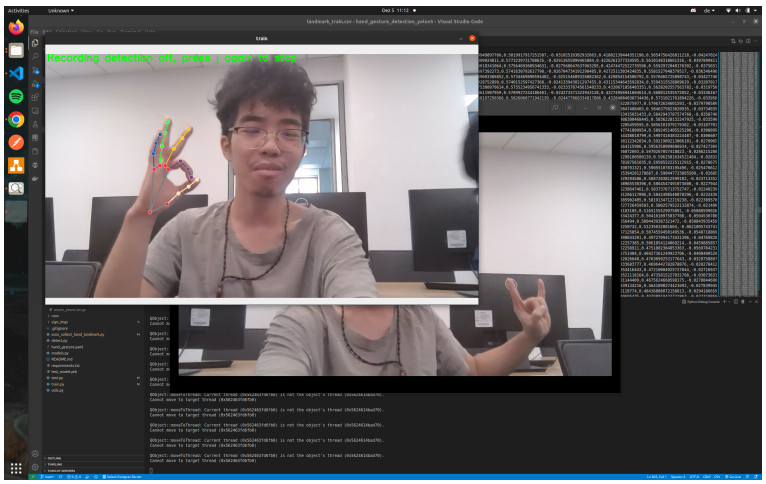


Figure 4: Interface for labelling the data set. The functionality is limited, but it was enough for our job.

2.4 System construction

2.4.1 YoloV5

We persisted and produced the dataset as the project developed. YoloV5 provides built-in image enhancement as described in the hyp.scratched_low.yaml fille. There, we modified the training parameters.
In order to find the ideal training parameter, we use genetic evolution—another YoloV5 feature—to change the training parameters. Unfortunately, the training period was too long. Collab disconnects after 2 hours of inactivity, and training on a PC with RTX 2060 6GB VRAM took averagely over 10 hours. Moreover, a proper validation and test set was not created,as a result of the aforementioned test bleed, therefore metrics computed on test and validation sets does not reflect the model's capability

to generalize. The inaccurate handsign bounding boxes in training set, the train/test bleed, and lack of reliable computing power culminate in failure of applying YOLOV5 for this task. During inference, while many signs can be recognized correctly, hand signs that are incorrectly automatically classified have extremely unreliable detection.

2.4.2 Custom neural network

We need to change our current solution since YOLOV5 is excessive and would take too much manual labeling to function properly in this circumstance. To recognize hand signs, we instead use an existing method for detecting hand landmarks, called mediapipe and feeding the landmarks into a tiny neural network.
The network, as shown in Figure 5 has 5 layers and include batch normalization to normalize output in hidden layers, and dropout for adding noise during training. While the network will be the common stack for all models, the loss function and how the output is used can varies, and for this case:

- **Output**: The output can go through Softmax Activation to return 10 probability elements vector corresponding to 10 commands. The number 10 is only a representative number, as adding more commands can be easily done in our system.
- **Loss**: Cross Entropy with label smoothing to avoid overconfidence. And for detecting unknown class, we added a threshold of **x** .If the largest score is below **x**, then we consider the hand sign undefined and do not do anything.

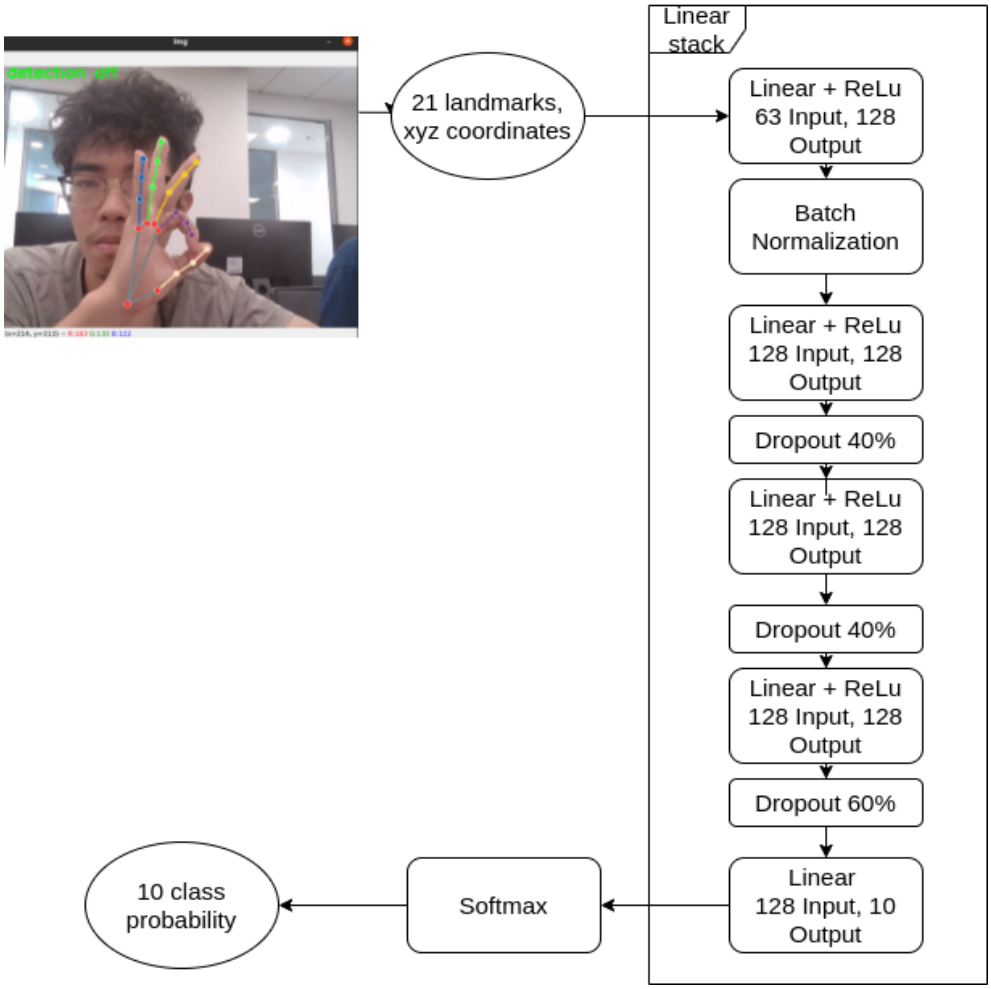


Figure 5: Pipeline of our model from the raw input to desired output. Softmax layer can be replaced by other layers that help in classification

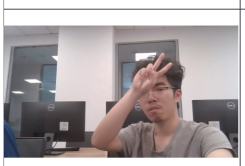
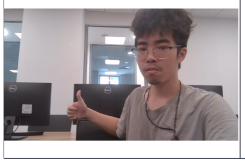
However, the model does not do well enough on Out Of Distribution (OOD) Detection , which, in our case, is recognizing that the hand sign does not belong to any of the classes we specified. There are several approaches for this problem, which are:

- Confidence threshold for Softmax, which is used by the model.
- Expose the model to unknown data, which will not scale well in our case. An example is when we want to add another new command, which is previously tagged unknown in the training set.
- Instead of cross entropy loss, we use CAC loss to perform distance based OOD detection. As stated in [6], this loss make the learnt logits space embedding form tight, class-specific clusters, and from that we can use distance to class centers to spot outliers and recognize the class the data belong to.

$$\mathcal{L}_{CAC}(\mathbf{x}, y) = \mathcal{L}_T(\mathbf{x}, y) + \lambda \mathcal{L}_A(\mathbf{x}, y)$$

2.4.3 System Classes and Commands

The figures below illustrates all of the classes and corresponding commands that the system use to operate:

Class	Commands
	1. all start
	2. toggle light 1
	3. toggle light 2
	4. toggle light 3
	5. toggle air conditioner
	6. turn on computer

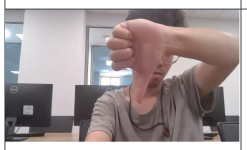
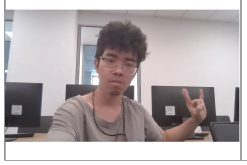
Class	Commands
	7. all stop
	8. disable command recognition
	9. enable command recognition

Figure 6: System classes and commands

Each hand gesture stands for a function of controlling the household appliances. For the sake of simplicity, our project consider the system works in a house that has three lights, one computer, and one computer.

- Command 1: this will start all the devices in the house system included 3 lights, the air conditiioner and the computer.
- Command 2,3,4,5,6: each of this would only toggle the light 1,2,3, or start the computer, or air conditioner correspondingly.

- Command 7: this will stop all the working devices in the system.
- Command 8: this will disable the recognition function of the camera. If a user is in a scenario that he/she does not want the camera to recognize actions from him/her. An example for this scenario is when a user greetings or talking to someone else, a case when user want to pause the controlling system function.
- Command 9: this will re-enable the recognition function of the camera, which allows the system to continue on working properly.

3 Experiments setup

3.1 Dataset

We test both model on our own data set of 800 hand poses, with around 200 of the data are unknown hand poses.

3.2 Evaluation Metric

We used the following metric to assess the performance of both model in terms of both known sign detection and unknown sign detection:

- **Area Under the ROC Curve (AUROC)** is a calibration free measure of the open set performance of a classifier.The Receiver Operating Characteristic (ROC) curve represents the trade-off between true positive rate (known inputs correctly retained as 'known') and the false positive rate (unknown inputs incorrectly retained as 'known') when applying varying thresholds to a given score.
- **Area Under the Precision-Recall Curve (AUPR)** is similar to AUROC, but become a more useful metric for imbalanced data in a problem setting where we focus on detecting positive data.The Precision Recall curve represents the trade-off between Recall and Precision for different threshold. As 25 per of test dataset is outlier, the baseline of this metric will be 0.25.
- **Classification Accuracy** measures the classifier's accuracy when applied to only the known classes in the dataset, equivalent to closed set classification. An open set classifier should maintain the classification accuracy of a standard closed set classifier.

In this section, for calculating AUPR and AUROC, we assume positive example is the unknown hand sign and the negative example is the hand signs the models knows.

3.3 Implementation detail

We trained both model with an ADAM Optimizer with a learning rate of 0.0001 until the validation loss stop decreasing, or for 300 epochs, whichever comes first. For Model with Cross Entropy Loss,we added 0.04 label smoothing, and for model with CAC Loss, we set the anchor magnitude α of 1 and try multiple anchor loss weights λ.

3.4 Result

From the result of our evaluation show in Table 1, we can infer two main facts: that models with CAC loss did not outperform models Cross Entropy loss in terms of recognizing unknown hand sign, however they did maintain the same classification accuracy, and secondly, the choice of λ in CAC loss does affect the OOD detection rate significantly.

3.5 Controlling the appliances

We deployed the model into production after training.So we made a script in which the model detect hand signs and switch on the corresponding appliance. The appliances are connected to a relay

	AUPR	AUROC	Accuracy
Cross entropy loss (label smoothing=0.04)	**0.835**	**0.878**	**0.98**
Cross entropy loss (no label smoothing)	0.406	0.671	0.958
CAC loss $(\alpha = 1, \lambda = 1)$	**0.573**	**0.824**	**0.972**
CAC loss $(\alpha = 1, \lambda = 2.5)$	0.765	0.878	0.982
Random classifier	**0.244**	**0.493**	**0.104**

Table 1: The model with CAC loss provide worse AUROC and AUPR than the model with Cross entropy loss. The random classifier acts as a baseline for all 3 metrics

module with 4 channels, each channel is connected to an appliance. When a hand sign is detected, our computer will send data to the relay, which will close or open the circuit powering the corresponding appliance. The data is sent over Modbus protocol via RS485 standard.

4 Conclusion and Future Work

While both model achieve real-time inference speed and high accuracy, the model with CAC loss perform worse than the traditional model.One may then question our action of incorporating CAC loss in the model. Perhaps, it is true,and further research is needed to make an OOD detection model which actually have any considerable advantage over traditional one.

Regarding the final product, we successfully deployed the model for practical use i.e: controlling appliance. However this system for the time being can only detect one-hand signs, and in the near future, we can adjust the models to accept two hands commands.

5 Appendix

5.1 Work Distribution

Project workload distribution		
Task name	Bui Le Phi Long	Nguyen Duc Thang
Write script for auto labeling	x	x
Write script for training, validation and test	x	x
Write script for benchmark model	x	x
Design model	x	x
Code for prediction		x
Code for interface, hardware, relay module		x
Final documentation	x	x
Project Slides	x	x

Figure 7: Project Workload Distribution

References

[1] S. Alexandrova, Z. Tatlock, and M. Cakmak. Roboflow: A flow-based visual programming language for mobile manipulation tasks. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5537–5544. IEEE, 2015.

[2] G. Bradski and A. Kaehler. Opencv. *Dr. Dobb's journal of software tools*, 3:120, 2000.

[3] K. Kirchheim, M. Filax, and F. Ortmeier. Pytorch-ood: A library for out-of-distribution detection based on pytorch. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4351–4360, 2022.

[4] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, 2019.

[5] F. Luus, N. Khan, and I. Akhalwaya. Active learning with tensorboard projector. *arXiv preprint arXiv:1901.00675*, 2019.

[6] D. Miller, N. Sünderhauf, M. Milford, and F. Dayoub. Class Anchor Clustering: a Loss for Distance-based Open Set Recognition, Mar. 2021. URL `http://arxiv.org/abs/2004.02434`. arXiv:2004.02434 [cs].

[7] A. Patil, S. Samant, M. Ramtekkar, S. Ragaji, and J. Khanapuri. Intelligent voice assistant. In *Proceedings of the 3rd International Conference on Advances in Science & Technology (ICAST)*, 2020.

[8] A. D. I. Pytorch. Pytorch, 2018.

[9] N. K. Suryadevara and S. C. Mukhopadhyay. *Smart Homes*. Springer, 2015.

[10] R. Team. Train test split. 2022. URL `https://blog.roboflow.com/train-test-split/`.