VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

MULTI-DISCIPLINARY PROJECT (CO3107)

Project:

Flood Warning On Urban Streets

Advisor: Băng Ngọc Bảo Tâm

Students:	Phạm Đức Anh Khoa	2053140
	Lê Đức Đồng	2052450
	Nguyễn Hữu Trùng Dương	2052929
	Nguyễn Phước Nguyên Phúc	2053342

HO CHI MINH CITY, MAY 2023

Document History:

Date	Ver	Change	Changed by
12/02/2023	1.0	- Create report template - Write introduction, requirement, list and illustrate planned device	All
20/02/2023	1.1	- Adjust requirements section - Add use case diagram and scenario	All
26/02/2023	1.1.1	- Adjust device list section	Lê Đức Đồng
12/03/2023	1.2	- Mockup of the system	All
19/03/2023	1.3	- Architecture of the system	Lê Đức Đồng
25/03/2023	1.3.1	- Add description to mock-up and architcture	N.H.Trùng Dương
09/05/2023	1.4	- Write module AI	N.H.Trùng Dương L.D.Đồng N.P.N.Phúc
10/05/2023	1.5	- Write module IOT	P.Đ.A.Khoa

Contents

List of Figures

List of Tables

1 Introduction

Flooding is a common and widespread natural disaster that can cause significant damage to infrastructure, property, and human life. One of the major challenges with floods is that they can happen quickly and without warning, making it difficult for people to take appropriate action to protect themselves and their property.

The impact of floods can be devastating. Floods can cause significant damage to roads, bridges, buildings, and other infrastructure, disrupting transportation and commerce. They can also lead to power outages, food and water shortages, and other public health emergencies.

Road flood warning systems are becoming increasingly important in many parts of the world due to the rising frequency and intensity of flooding caused by climate change. Climate change is causing more extreme weather events, including heavy rainfall, which can lead to flooding.

Even in areas where flooding is not common, unexpected heavy rainfall can cause roads to flood quickly, leading to inconvenience and safety hazards for drivers and pedestrians. When roads flood, water can cause cars to stall or slide, and pedestrians can be at risk of being swept away by fast-moving water. Stalled cars can cause traffic congestion and delays, while wet clothes and shoes can be uncomfortable and lead to health problems . As a result, road flood warning systems are becoming an essential tool for managing this risk.

Our project aims to use a camera to predict whether the street at its location is flooded or not in order to help drivers avoid flooded streets. We also plan to collect data on the street's weather conditions, such as rain and humidity, using sensors, which will be used to train our AI model. Finally, information about flooding will be transmitted to a mobile application.

2 Requirements

2.1 Functional requirements

2.1.1 AI backend:

- Identify the flood at the location of the collecting device.
- Estimate the severity of the flood, such as whether it is a minor, moderate, or major flood.
- Export a monthly report summarizing the flood conditions and other weather data.

2.1.2 IOT devices:

- Get information and send data to the server every 2 minutes, including temperature, humidity, and rainfall data.

2.1.3 Mobile app:

- Get information about the weather state at the location of IOT device.

2.2 Non-functional requirements

- Provide high accurate predictions of flood conditions in real-time
- The collecting devices should be able to operate continuously 24/7 without failure, and any faults should be automatically detected and resolved in real-time.
- The latency of data transfers and queries between edge-devices and the backend server must be at a maximum of 2 seconds for each action, to ensure that data is available in real-time.
- The mobile app should be compatible with both iOS and Android operating systems, with the latest versions of each supported.

3 Device list

3.1 Water Sensor

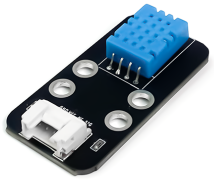
Usage: Detect if weather is raining or not then report to the server

3.2 Android phone as camera

Usage: Take picture and send to server in order that AI-backend can process and give inference on this picture

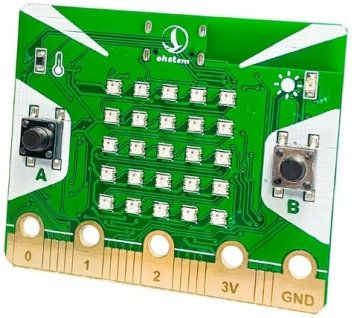
3.3 DHT11

Usage: Measure temperature, humidity then send to the server



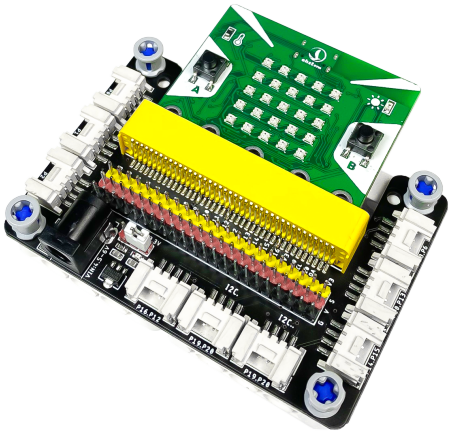
3.4 Yolobit

Usage: The hub for all sensors. It receives sensor readings through the expansion circuit board, perform preliminary processing then send to server



3.5 Groove Shield

Usage: The backbone for Yolobit and other sensors



4 Use-case scenario

4.1 Use-case diagram

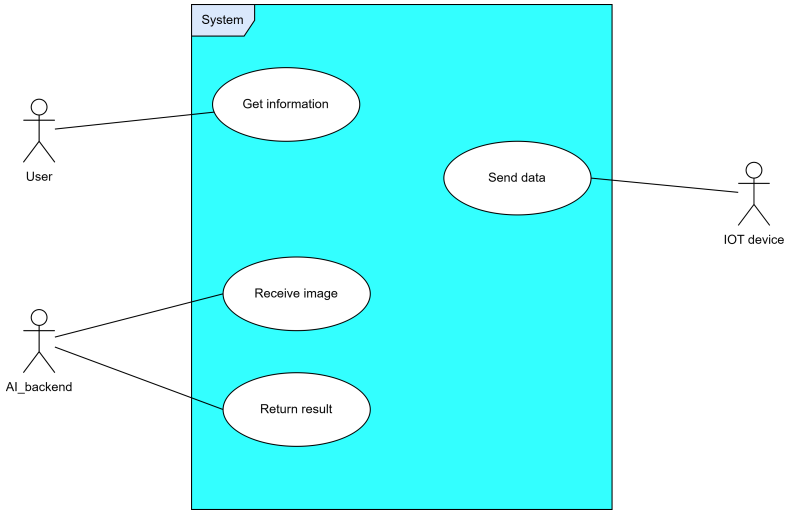


Figure 1: Use-case diagram

4.2 Use-case scenario

Table 1: Use-case Send data

Use-case	**Send data**
Actor	IoT devices
Description	This use case is for IoT devices sending data to server
Trigger	Automatically every 2 minutes, the server send images to AI backend
Precondition	1. IoT devices have proper connection to the Internet 2. IoT devices fully functioning
Postcondition	1. IoT device successfully send data without any packet

Normal Flow	1. Sensors retrieve data and send to microcontroller unit. 2. MCU break down data and send to server.
Exception	IoT device can not connect to MQTT broker server
Catch exception	1. Resend data every 10 seconds until success (maximum 5 times) 2. After 5 unsuccessful resend, disconnect and reconnect to MQTT broker server until success.

Table 2: Use-case Receives images

Use-case	**Receives images**
Actor	AI backend
Description	AI Backend receives images from server and preprocesses the images
Trigger	Automatically every 2 minutes, the server send images to AI backend
Precondition	1. Server and AI backend have proper connection 2. AI backend is connected to the server 3. AI backed is always wait for the order from the server 4. AI backend always accept the order from the server
Postcondition	1. AI backend succesfully receives the images after the sending order from server 2. The image then must be preproccessed into suitable form

Normal Flow	1. Server automatically sends images to AI backend 2. AI backend receive the images 3. Ai backend preprocess the images into suitable form

Table 3: Use-case Return result

Use-case	**Return result**
Actor	AI backend
Description	This use-case describes the steps involved in the AI backend processing an image and returning the results to save into the sever
Trigger	The AI backend receives images from the sever
Precondition	The sever is connected to the AI backend and able to receive updates.
Postcondition	The AI backend has processed the image and returned the results to the sever
Normal Flow	1. AI backend analyzes the image to detect the presence of flood. 2. AI backend predicts the level of flood (minor, moderate, or major) 3. AI backend generates a report of the flood detection and sends it to the sever. 4. Sever receives the report

Table 4: Use-case Get information

Use-case	**Get information**
Actor	User
Description	User gets information about temperature, humidity, rainfall, flood warning from database
Trigger	User wants to see this information
Precondition	1. User is connected to Internet

Postcondition	1. System returns information about temperature, humidity, rainfall, flood warning to user
Normal Flow	1. User gets access to mobile app 2. System queries the latest information in database 3. System returns results to user

5 Mock up

The images below are the page when users first open the application. After clicking the Start button, the system will recommend a list of IOT devices. User will choose one of them or click into the (+) button to get other IOT devices

Figure 2: First and second page

After choosing a IOT device, we move to below page. Above all of this page is the **location** of IOT device. It contains the district and maybe the street where IOT device is set up

Below "location part" is 4 types of **information** gathered by IOT device:

- Temperature
- Humidity
- Weather: know if there is rain in this location or not
- State: the device is working , maintaining or disconnecting

There is a **flood severity bar** which have 3 levels:

- Low : the bar reach the green color
- High : the bar reach the yellow color
- Critical : the bar reach the red color

Finally, when no color of the bar is displayed, there is no flood in this location.

After predicting the **severity of flood**, the application will attach one **picture** of this location at the moment

Figure 3: Information page

At the **Device in use** part, this is the list of IOT device at different location that users want to know. They can customize it by click into the **Add new (+)** button to choose new IOT device

In this section, user can traverse a list of devices or search a specific one by entering the information (id, location) of them in search bar.

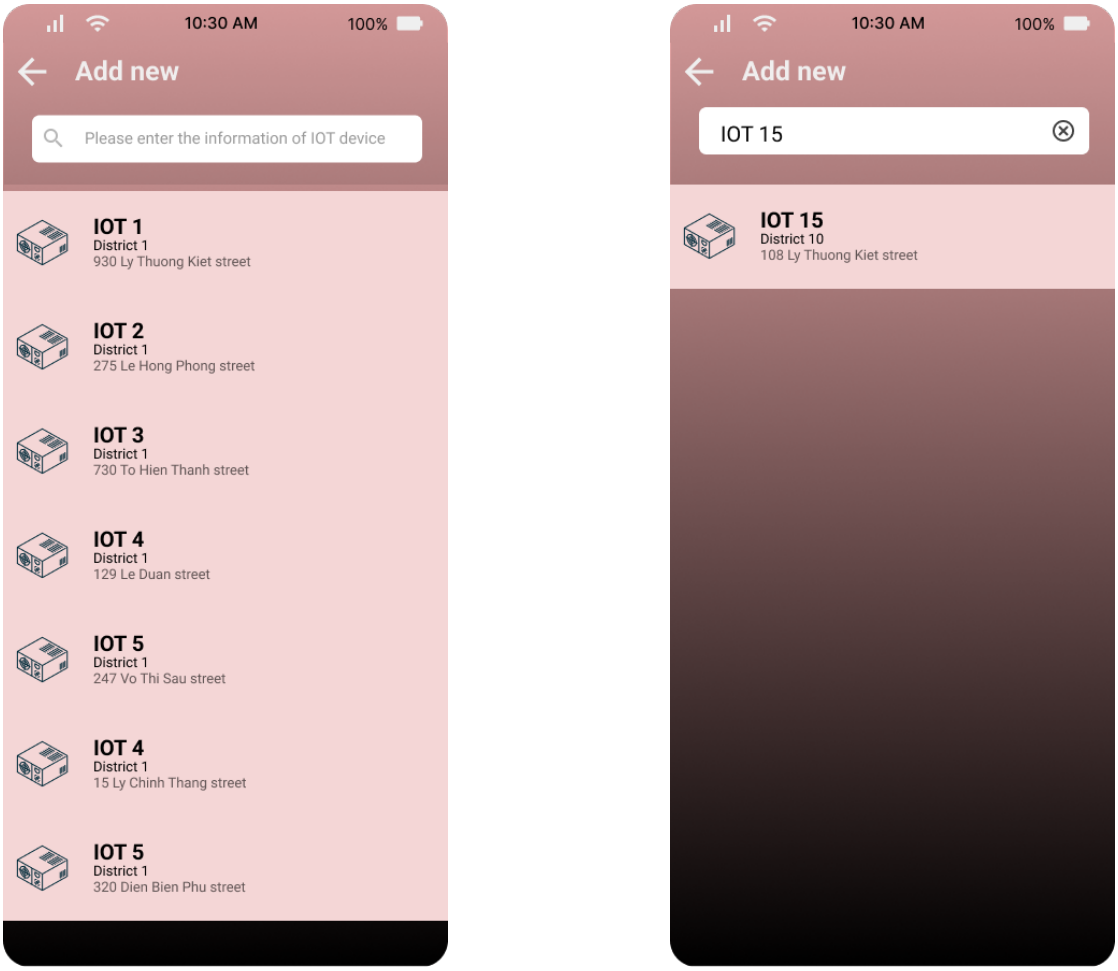
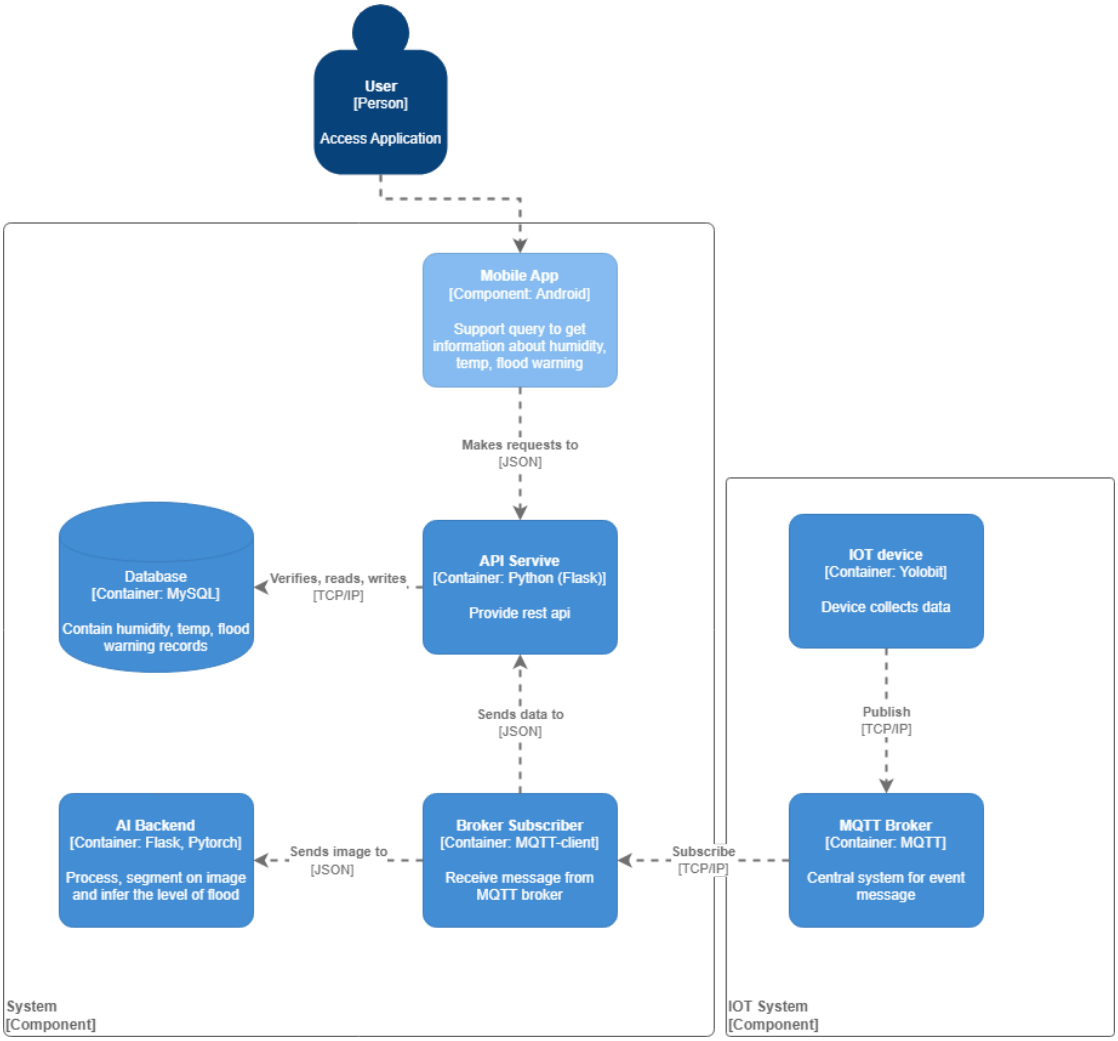


Figure 4: Add new IOT device page

6 Architecture



IOT device module:

- **Functionality and Purpose:** The IoT Device module is responsible for collecting data from sensors and devices connected to the IoT device. The collected data may include information about image, temperature, humidity, or any other environmental factor that the sensors can detect. The module processes the collected data and sends it to other modules or services within the system for further analysis, storage, or display

MQTT Broker:

- **Functionality and Purpose:** The MQTT Broker module is the center for event messaging within the MQTT container. It provides a messaging infrastructure for IoT devices and other modules to publish and subscribe to events or messages. The module receives

messages from publishers and delivers them to subscribers based on their subscriptions. It also manages the QoS (Quality of Service) levels of the messages to ensure reliable delivery.

Broker Subscriber:

- **Functionality and Purpose:** The Broker Subscriber module within the MQTT-client container is responsible for receiving messages from the MQTT broker. The module subscribes to specific topics or channels on the broker and receives messages published on those topics. The received messages are processed by the module and sent to other modules or services within the system for further analysis or processing.

AI Backend:

- **Functionality and Purpose:** The AI Backend module is responsible for processing and segmenting images and inferring the level of flood. It is designed to handle image data as inputs and uses machine learning algorithms based on the PyTorch library to segment and classify the images. The module provides a Flask-based web API to receive image data and provide the corresponding flood level inference.

API Services:

- **Functionality and Purpose:** The API Services module within the Python (Flask) container is responsible for providing a RESTful web API to the users or other modules within the system. It provides an interface for users to interact with the system and perform various operations. The module uses the Flask web framework to implement the RESTful API.

Database:

- **Functionality and Purpose:** The Database module within the MySQL container is responsible for storing and managing the data related to humidity, temperature, and flood warning records. It provides a persistent storage solution for the data generated by various modules within the system.

Mobile App:

- **Functionality and Purpose:** The Mobile App module within the Android component is responsible for providing a user interface for users to access the system and retrieve information about humidity, temperature, and flood warnings. It provides a mobile platform for users to interact with the system on-the-go.

7 AI Module

7.1 Water Segmentation

Water segmentation is a key component of our AI-backend module that aims to detect and predict flooding events in real-time using camera images. In this section, we describe the data sources and techniques used to train our water segmentation model.

7.1.1 Dataset:

For our water segmentation module, we gathered a diverse set of datasets to train our model. These datasets include:

- **ADE20K**: a large-scale scene parsing dataset with 1812 images from diverse scenes.
- **River_Seg**: a dataset of 300 river images with pixel-level labels for water and non-water regions.
- **Flooding PixABay**: a collection of 149 images of flooding scenarios from Pixabay, a popular online platform for sharing images.
- **Urban Flood**: a dataset of 441 images of urban flooding scenarios
- **Our self-labeling** dataset: a set of 200 images we collected ourselves and manually labeled for water and non-water regions.

In total, our dataset contains 2,902 samples, each consisting of an image and a corresponding mask label. We split our dataset into two sets for training and validation, with 2,752 samples in the training set and 150 samples in the validation set.

7.1.2 Model Architecture

DeepLabV3+:

For our water segmentation module, we utilized the state-of-the-art DeepLabV3+ model. This model is an extension of the DeepLabV3 model and utilizes atrous convolution and encoder-decoder architecture to achieve high accuracy.

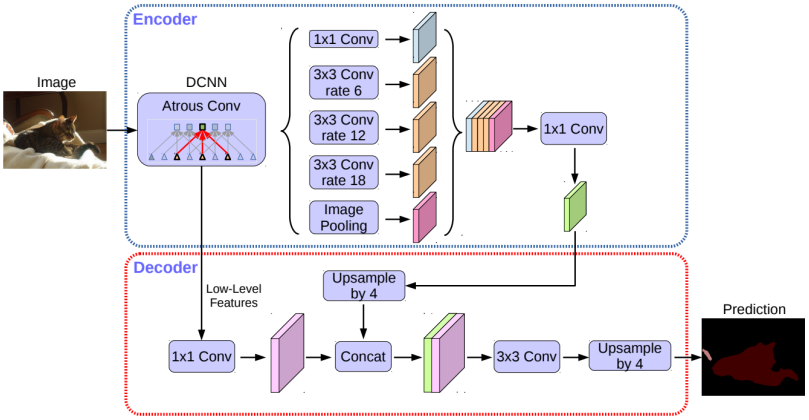
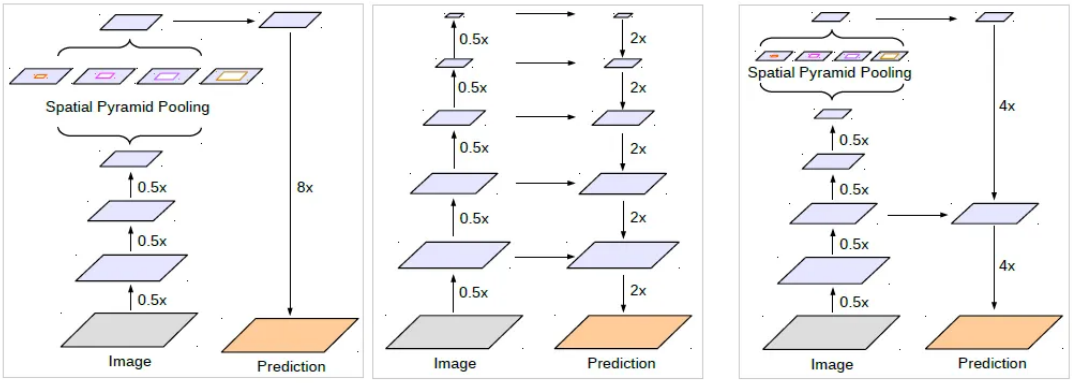


Figure 5: DeepLabV3+ Architecture

Atrous convolution, also known as dilated convolution, is a technique that allows the convolution operation to sample input pixels with larger receptive fields without increasing the number of parameters. This technique is particularly useful for semantic segmentation tasks where larger context is important. To further improve the segmentation accuracy, DeepLabV3+ also incorporates a **Spatial pyramid pooling (SPP)** module, which captures information at multiple scales and resolutions. The SPP module is added after the last convolutional layer and consists of several pooling operations with different kernel sizes.

The DeepLabV3+ model also includes an **Encoder-Decode**r architecture, where the encoder module is used to extract high-level features from the input image, and the decoder module is used to recover the spatial information lost during the encoding process. This architecture enables the model to learn more complex and fine-grained features from the input image.



(a) Spatial Pyramid Pooling (b) Encoder-Decoder (c) Encoder-Decoder with Atrous Conv

Figure 6: Deeplabv3+ makes use of (a) and (b

Loss function:

Dice loss is a commonly used loss function for image segmentation tasks, which measures the overlap between the predicted and ground truth segmentation masks. It is based on the Dice coefficient, which is a similarity measure between two sets. The **Dice coefficient** ranges from 0 to 1, where a value of 1 indicates a perfect overlap between the two sets, while a value of 0 indicates no overlap.

$$DiceCoeff = \frac{2|A \cap B|}{|A| + |B|}$$

where $|\ \mathbf{A} \cap\ \mathbf{B}\ |$ represents the common elements between sets A and B, and $|\mathbf{A}|$ represents the number of elements in set A (and likewise for set B).

The **dice loss** function is defined as the complement of the Dice coefficient. It is calculated by taking the difference between the predicted and ground truth masks.

$$DiceLoss = 1 - DiceCoeff = 1 - \frac{2|A \cup B|}{|A| + |B|}$$

The use of the dice loss function has become popular in recent years due to its ability to handle class imbalance in image segmentation tasks. It has been shown to perform well on a variety of medical image segmentation tasks, where the imbalance between the number of foreground

and background pixels is often significant.

Metric : IOU score

Intersection over Union (IoU) is a common metric used to evaluate the performance of object detection and segmentation algorithms. IoU measures the overlap between the predicted bounding box or mask and the ground truth bounding box or mask for a given object. It is calculated as the ratio between the area of intersection and the area of union between the two regions.

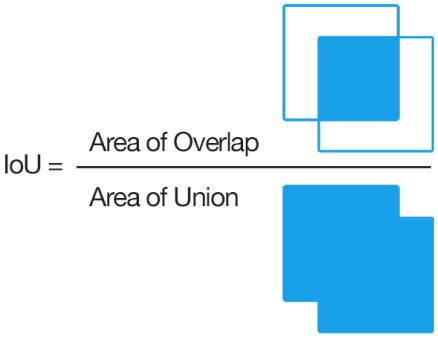


Figure 7: IOU formula

IoU ranges from 0 to 1, where a value of 0 indicates no overlap, and a value of 1 indicates perfect overlap between the predicted and ground truth regions. In general, a higher IoU score indicates a better performance of the segmentation algorithm.

7.1.3 Experiment

7.1.3.a Setting

In order to evaluate the performance of our water segmentation module, we conducted experiments with two different configurations. For each configuration, we trained the model with a set of hyperparameters and evaluated the results on a separate validation set.

The first configuration used an input size of 416x416 and a ResNet101 backbone with 42 million parameters. We utilized a pretrained encoder from ImageNet and trained the model for 110 epochs using the Adam optimizer.

The second configuration also used an input size of 416x416, but with an Efficient B5 backbone with 28 million parameters. We also utilized a pretrained encoder from ImageNet and trained the model for 75 epochs using the Adam optimizer.

Configuration	Backbone	Input size	Epoch	Optimizer
1	ResNet128 (42M parameter)	416 x 416	110	Adam
2	EfficientB5 (28M parameter)	416 x 416	75	Adam

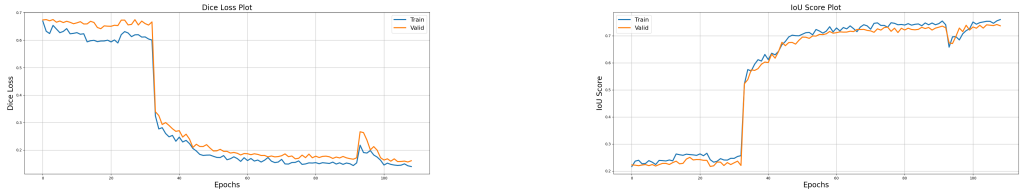
Table 5: Two Configurations

7.1.3.b Benchmarks

After training, we compare 2 model based on the final result of Mean Dice Loss, Mean IOU score from development set, and then we get the following Table 6

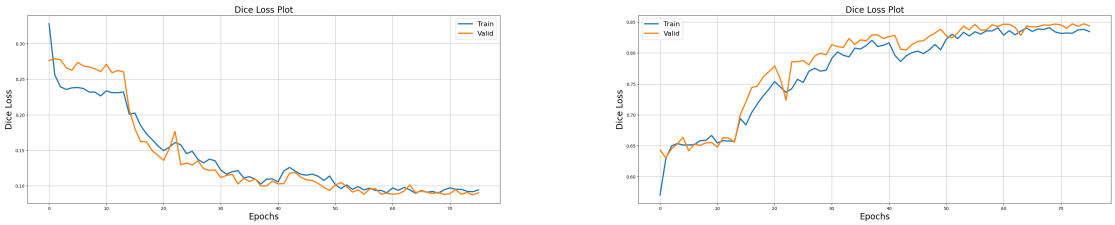
Model	Mean IOU score	Mean Dice loss
ResNet128 Backbone	72	0.1845
EfficientB5 Backbone	84.35	0.0903

Table 6: Evaluation From Two Configurations



(a) Mean Dice Loss (b) Mean IOU Score

Figure 8: Dice Loss and IOU Score From Resnet101 Backbone



(a) Mean Dice Loss (b) Mean IOU Score

Figure 9: Dice Loss and IOU Score From Efficient-B5 Backbone

After comparing the performance of the two configurations based on the Mean Dice Loss and Mean IOU score from the development set, we decided to select the **second configuration** as our primary model for predicting water segmentation.

As evidence of our model's success, we have provided some samples of its output in Figure 11 below.

(a) Image (b) Mask (c) Overlay

Figure 10: Our Model Tested On Some Sample Images

7.2 Depth Estimation

In this section, we will explain the method that we use to estimate the water depth.

Our method:

- In short, we will calculate water depth by using reference objects. We choose objects that we already know their size, then by calculating the real size and the submerged size, we can know the depth of water. However, we are not going to detect the objects, we use a different approach called reference points.
- To be more detail, firstly, we will choose a list of pair of point - first point is the top of the object and second point is the bottom. For example, i choose the roadside to be my object, i can choose any part of the roadside, just to make sure that the distance of these tow points is also the height of the roadside. Because we choose by hand so it will have little margin of error, so we choose many pairs of point of that roadside and then take average to lower this error. Of course we must know the real size of the chosen object by direct measurement.
- For each pair, begin from the top point, we use a vector that moves down until it touch the water mask, so the gap between the touched point and the bottom point is the height of water
- we also have the second list of pair of point. This list is activated in case the water submerge at least one of a top point of the first list. The object chosen for the second list should be above 1 meter, just to make sure that the water won't submerge this list.

Figure 11: The bottom red line is the connection between bottom points in the first list. The top yellow line contains top points of second list. The middle line has top points of first list and bottom points of second list

- Finally, we set flood level based on the water depth. For instance:
  - Level 0: 0 - 10 cm
  - Level 1: 10 - 30 cm
  - Level 2: 30 - 50 cm
  - Level 3: $>$ 50 cm

8 IoT system

8.1 MQTT broker

In this section, we will explain how our MQTT broker work and how we utilize it to manage IoT devices. We use NodeJs to host a MQTT broker server.

Functionality of MQTT broker:

- MQTT Broker is a central server for receiving and publishing messages by subscribing to topics.
- Each IoT devices or servers subscribe to various topic on the MQTT broker server, whenever messages get published on topics, each devices/servers that subscribe the topic will receive those messages.

How we utilize the MQTT Broker:

- For communication between central server and IoT devices:
  We specify some topics for various purposes as describes below:

Topic	Send device	Data	Describe
iot/capture	Central server	any	To request data from IoT devices and Cameras.
iot/active	IoT device	any	To notify which IoT device is active.
iot/deactive	IoT device	any	To notify which IoT device will deactive.
iot/server	Central server	any	To notify that the server is connected to broker, the broker will then send the active devices list.
weather_data	IoT device	{humidity, temperature, rain, device_id}	Contain weather data.
image	Camera	{image_data, device_id}	Contain image data.

Figure 12: Topic list

- For managing active devices, we implement a slave subscriber that subscribe to broker these topic: iot/active, iot/deactive, iot/server. This way, when a device connect or disconnect from broker, the slave can keep track of changes and publish current active devices.

8.2 Yolobit and sensors

In this section, we will describe how our yolobit control and receive data from sensors and send it to MQTT broker. We use MicroPython for the implementation.

Our Yolobit controlling flow:

- **Setup:**
  - Initiate sensors
  - Establish Connection to MQTT broker
  - Publish a message containing device id to broker to notify which device is active.
  - Subscribe device to topics that used to establish the data sending request.

- **While running:**
  - Yolobit will constantly check message to subscribed topic every 1 seconds.
  - Whenever the message come, Yolobit will observe data from sensors and publish to the desired topic.
- **Deinit:**
  - Yolobit will catch Exception in case the device gone wrong.
  - Whenever exceptions catches, it will publish the disconnect message to the desired topic.

8.3 Camera

As the hardware limitation in term of quality, we do not use the camera sensors, instead we use an Android phone as a camera. React-native framework is used to control the phone.
Camera controlling flow:

- **Setup:**
  - Initiate camera sensors on the phone
  - Establish Connection to MQTT broker
  - Subscribe device to topics that used to establish the data sending request.
- **While running:**
  - The camera will constantly check message to subscribed topic.
  - Whenever the message come, Camera will take a picture.
  - Before sending the image data, it encode the data to Base64 form as this is the safe way to generalize image data form.
  - The Base64 encoded image will be publish to desired topic.

8.4 Advantages and disadvantages:

During the implementation for this system, we realize some advantages as well as the disadvantages:

Advantages:

- Simple architect and easy to understand.
- Has the feasibility to manage and control many IoT devices at a time.
- Easy to maintain.
- Easy to upgrade.

Disadvantages:

- Does not have many alternative ways to handle connection lost.
- IoT devices monitoring is not real time in case the device get power off unexpected or error coming from the firmware, not the software.
- Base64 image is quite heavy to send and takes long time to arrive broker and servers.

9 Conclusions and Development Strategies

9.1 Summary

9.1.1 Advantages:

Successfully build an application can help everyone know the flood level. The application is easy to use, and it provides a visual representation of flood level which makes it easy for people to understand the flood situation.

One of the key advantages of our project is the integration of IoT devices. This allowed us to work with hardware devices and incorporate real-time data into our system, which is crucial for detecting and predicting flooding events. Working with IoT devices presented several challenges, such as configuring the devices or establishing a reliable connection between them and our software. However, by overcoming these challenges, we were able to gather data from sensors in real-time and use it to make accurate flood predictions.

Despite not having extensive experience, we were able to work together effectively, communicating our ideas and progress regularly to ensure that everyone was on the same page. This allowed us to divide the workload efficiently and ensure that everyone's strengths were utilized to their fullest potential. Additionally, by working together, we were able to provide constructive feedback and support to each other, leading to a stronger final product

This project give us the opportunity to work with various frameworks and technologies, including PyTorch, Flask, React Native, and MQTT. It allowed for the development of a robust and functional system, while also providing valuable learning experiences and skill development opportunities. The team gained valuable experience in working with different frameworks and technologies, which will be useful for future projects and career opportunities.

9.1.2 Drawbacks

The main drawbacks of our project is the limitation of resources for training our model. Due to the limited computing power and storage capacity of our hardware, we were only able to use a small input image size. As a result, this limitation can affect the accuracy of our model, as it may not be able to capture all the necessary details and nuances of the image.

9.2 Development trend

The development trend for this project includes several potential directions that can improve the system's performance and expand its functionalities.

- One possible direction is to integrate the flood level information into a map application, allowing people to plan their routes based on the current flood level. This would be especially helpful for people living in flood areas, as they can avoid streets with high water levels and find alternative routes to reach their destinations safely.
- Another direction is to incorporate crowdsourced data into the system in case of streets that do not have IoT devices installed. Crowdsourced data can come from various sources, such as social media, facebook, ... By analyzing this data and cross-referencing it with the existing data from IoT devices, the system can provide more accurate and comprehensive information about flood levels in a given area.

References

[1] Liang-Chieh Chen et al. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: *CoRR* abs/1802.02611 (2018). arXiv: 1802.02611. URL: http://arxiv.org/abs/1802.02611.

[2] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully Convolutional Networks for Semantic Segmentation". In: *CoRR* abs/1411.4038 (2014). arXiv: 1411.4038. URL: http://arxiv.org/abs/1411.4038.

[3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597 (2015). arXiv: 1505.04597. URL: http://arxiv.org/abs/1505.04597.

[4] Carole H. Sudre et al. "Generalised Dice overlap as a deep learning loss function for highly unbalanced segmentations". In: *CoRR* abs/1707.03237 (2017). arXiv: 1707.03237. URL: http://arxiv.org/abs/1707.03237.

MEETING MINUTE 1ST WEEK

MEETING NAME	First Weekly Practice on Multi-disciplinary Project Assignment Meeting			
MEETING DATE	Feb 19th, 2023	TIME	Start	20:00
			Finish	21:00
MEETING PURPOSE	- Discuss use-cases - Assign tasks for next week			
LEADER	Lê Đức Đồng			

1. ATTENDANCE AT MEETING	
Name	Student's ID
Phạm Đức Anh Khoa	2053104
Lê Đức Đồng	2052450
Nguyễn Hữu Trùng Dương	2052929
Nguyễn Phước Nguyên Phúc	2053342

2. MEETING DISCUSSION	
Topic	Content
1. Report Adjustment	Discuss aboutt conflict in requirements and try to resolve it
2. Use-case	Discuss use-case of project
3. Task Discussion	List of all necessary tasks in this project and propose tasks next week

2. ACTION TERMS		
Research U-net	Nguyễn Hữu Trùng Dương	Mar 3rd, 2023
Research DeepLab net	Nguyễn Phước Nguyên Phúc	Mar 3rd, 2023
Research Yolov8 instance segmentation	Lê Đức Đồng	Mar 3rd, 2023
Design Circuit diagram and testing out IoT devices	Phạm Đức Anh Khoa	Mar 3rd, 2023

MEETING MINUTE 2ND WEEK

MEETING NAME	2nd Weekly Practice on Multi-disciplinary Project Assignment Meeting			
MEETING DATE	Mar 4th, 2023	TIME	Start	20:30
			Finish	21:30
MEETING PURPOSE	- Discuss mock-up - Select AI model - Assign tasks for next week			
LEADER	Lê Đức Đồng			

1. ATTENDANCE AT MEETING	
Name	Student's ID
Phạm Đức Anh Khoa	2053104
Lê Đức Đồng	2052450
Nguyễn Hữu Trùng Dương	2052929
Nguyễn Phước Nguyên Phúc	2053342

2. MEETING DISCUSSION	
Topic	Content
1. Mock-up	Discuss about UI and adjust it on figma
2. AI selection	Compare among segmenation models U-net, Deeplab, YOLOv8 instance segmentation. Then, we decides to use DeepLabV3+
3. Meeting	Make an appointment at university in order to try to connect channel on broker mqtt between server and iot devices

2. ACTION TERMS		
Lableling image, train AI	Nguyễn Hữu Trùng Dương	Mar 31st, 2023
Labeling image, train AI	Nguyễn Phước Nguyên Phúc	Mar 31st, 2023
Labeling image, train AI, make a subscribed server to connect iot device, build api service	Lê Đức Đồng	Mar 31st, 2023
Set up yolobit with mqtt	Phạm Đức Anh Khoa	Mar 31st, 2023

MEETING MINUTE 3RD WEEK

MEETING NAME	3RD Weekly Practice on Multi-disciplinary Project Assignment Meeting			
MEETING DATE	Mar 11th, 2023	TIME	Start	13:00
			Finish	15:00
MEETING PURPOSE	- Connect backend server with yolobit via mqtt - Adjust Mockup			
LEADER	Lê Đức Đồng			

1. ATTENDANCE AT MEETING	
Name	Student's ID
Phạm Đức Anh Khoa	2053104
Lê Đức Đồng	2052450
Nguyễn Hữu Trùng Dương	2052929
Nguyễn Phước Nguyên Phúc	2053342

2. MEETING DISCUSSION	
Topic	Content
1. Connect MQTT	Connect server with mqtt, and test
2. Mock-up	Based on comment from lecturer, try to add some components

MEETING MINUTE 4TH WEEK

MEETING NAME	4th Weekly Practice on Multi-disciplinary Project Assignment Meeting			
MEETING DATE	Mar 24th, 2023	TIME	Start	20:30
			Finish	21:30
MEETING PURPOSE	- Fix mock-up - Test restful api - Ai training report			
LEADER	Lê Đức Đồng			

1. ATTENDANCE AT MEETING	
Name	Student's ID
Phạm Đức Anh Khoa	2053104
Lê Đức Đồng	2052450
Nguyễn Hữu Trùng Dương	2052929
Nguyễn Phước Nguyên Phúc	2053342

2. MEETING DISCUSSION	
Topic	Content
1. Mock-up	Adjust UI
2. RESTful api in module api service	Test and evaluation of each api
3. AI traininig report	Report the progress of AI training

2. ACTION TERMS		
Prepare ReactNative mobile app	Nguyễn Hữu Trùng Dương	Mar 31st, 2023
Investigate dept estimation algorithm	Nguyễn Phước Nguyên Phúc	Mar 31st, 2023
Continue train AI segmentation, update api service components	Lê Đức Đồng	Mar 31st, 2023
Implement MQTT broker server and add exception when lost connect on yolobit, design and implement stable connection check and reconnection flow to handle unstable network	Phạm Đức Anh Khoa	Mar 31st, 2023

MEETING MINUTE 5TH WEEK

MEETING NAME	5th Weekly Practice on Multi-disciplinary Project Assignment Meeting			
MEETING DATE	April 2nd, 2023	TIME	Start	20:30
			Finish	21:30
MEETING PURPOSE	- Review progress each member			
LEADER	Lê Đức Đồng			

1. ATTENDANCE AT MEETING	
Name	Student's ID
Phạm Đức Anh Khoa	2053104
Lê Đức Đồng	2052450
Nguyễn Hữu Trùng Dương	2052929
Nguyễn Phước Nguyên Phúc	2053342

2. MEETING DISCUSSION	
Topic	Content
1. Review	Evaluating progress

2. ACTION TERMS		
Continue build other pages	Nguyễn Hữu Trùng Dương	April 15th, 2023
Propose a function for the idea	Nguyễn Phước Nguyên Phúc	April 15th, 2023
Continue train AI on Efficient backbone, Build RESTApi service for AI-backend module	Lê Đức Đồng	April 15th, 2023
Start implement camera controlling on Android phone	Phạm Đức Anh Khoa	April 15th, 2023

MEETING MINUTE 6TH WEEK

MEETING NAME	6th Weekly Practice on Multi-disciplinary Project Assignment Meeting			
MEETING DATE	April 16th, 2023	TIME	Start	15:00
			Finish	17:30
MEETING PURPOSE	- Review progress each member - Combine AI model, depth estimation in AI module service			
LEADER	Lê Đức Đồng			

1. ATTENDANCE AT MEETING	
Name	Student's ID
Phạm Đức Anh Khoa	2053104
Lê Đức Đồng	2052450
Nguyễn Hữu Trùng Dương	2052929
Nguyễn Phước Nguyên Phúc	2053342

2. MEETING DISCUSSION	
Topic	Content
1. Review	Evaluating progress
2. Combine	Combine depth estimation with ai model, then integrate in AI service

2. ACTION TERMS		
Connect API service with frontend	Nguyễn Hữu Trùng Dương	April 22th, 2023
Fix bug in water depth estimation function	Nguyễn Phước Nguyên Phúc	April 22th, 2023
Connect subscriber with AI-backend module	Lê Đức Đồng	April 22th, 2023
Implement MQTT handle for camera	Phạm Đức Anh Khoa	April 22th, 2023

MEETING MINUTE 7TH WEEK

MEETING NAME	7th Weekly Practice on Multi-disciplinary Project Assignment Meeting			
MEETING DATE	April 23th, 2023	TIME	Start	15:00
			Finish	17:30
MEETING PURPOSE	- Review progress each member - Testing			
LEADER	Lê Đức Đồng			

1. ATTENDANCE AT MEETING	
Name	Student's ID
Phạm Đức Anh Khoa	2053104
Lê Đức Đồng	2052450
Nguyễn Hữu Trùng Dương	2052929
Nguyễn Phước Nguyên Phúc	2053342

2. MEETING DISCUSSION	
Topic	Content
1. Review	Evaluating progress
2. Testing	Testing flow from front-end to api service, subscriber to api service, subscriber to AI module
3. Test IOT connection	Try to send data from hardware to subscriber

2. ACTION TERMS		
–	Nguyễn Hữu Trùng Dương	–
–	Nguyễn Phước Nguyên Phúc	–
Fix bug in subscriber	Lê Đức Đồng	May 2nd, 2023
Implement devices management mechanism using slave subscriber, optimize and run full flow for IoT system.	Phạm Đức Anh Khoa	May 2nd, 2023

MEETING MINUTE 8TH WEEK

MEETING NAME	8th Weekly Practice on Multi-disciplinary Project Assignment Meeting			
MEETING DATE	May 7th, 2023	TIME	Start	13:00
			Finish	17:30
MEETING PURPOSE	- Completely connect subscriber to IOT modules - Testing all flow			
LEADER	Lê Đức Đồng			

1. ATTENDANCE AT MEETING	
Name	Student's ID
Phạm Đức Anh Khoa	2053104
Lê Đức Đồng	2052450
Nguyễn Hữu Trùng Dương	2052929
Nguyễn Phước Nguyên Phúc	2053342

2. MEETING DISCUSSION	
Topic	Content
1. Connect Subscriber with IOT Module	- Successfully receive data and send message between subscriber and iot modules
2. Testing	Testing all flow from all modules

2. ACTION TERMS		
Write report	Nguyễn Hữu Trùng Dương	May 9th, 2023
Write report	Nguyễn Phước Nguyên Phúc	May 9th, 2023
Write report	Lê Đức Đồng	May 9th, 2023
Write report	Phạm Đức Anh Khoa	May 9th, 2023