**<PROJECT 최종보고서>**

**Perfect ReCycle**

**-사물인식을 통한 분리수거 도우미**

****

**팀원 성함**

**2015104170 김효준**

**2015104180 백종현**

**2016104115 김원규**

**2016104163 정의동**

**교수님 성함**

**김재홍교수님**

Table of Contents

1. Abstract............................................................................................................. 3

2. Introduction..................................................................................................... 3

3. Background Study.......................................................................................... 3

A. 관련 접근방법/기술 장단점 분석...................................................................... 3

B. 프로젝트 개발환경.............................................................................................. 3

4. Goal/Problem & Requirements.................................................................. 3

5. Approach.......................................................................................................... 3

6. Project Architecture....................................................................................... 4

A. Architecture Diagram........................................................................................ 4

B. Architecture Description................................................................................... 4

7. Implementation Spec.................................................................................... 4

A. Input/Output Interface..................................................................................... 4

B. Inter Module Communication Interface......................................................... 4

C. Modules.............................................................................................................. 4

8. Solution............................................................................................................. 4

A. Implementations Details................................................................................... 4

B. Implementations Issues.................................................................................... 4

9. Results............................................................................................................... 4

A. Experiments........................................................................................................ 4

B. Result Analysis and Discussion........................................................................ 4

10. Division & Assignment of Work................................................................ 4

11. Conclusion........................................................................................................ 5

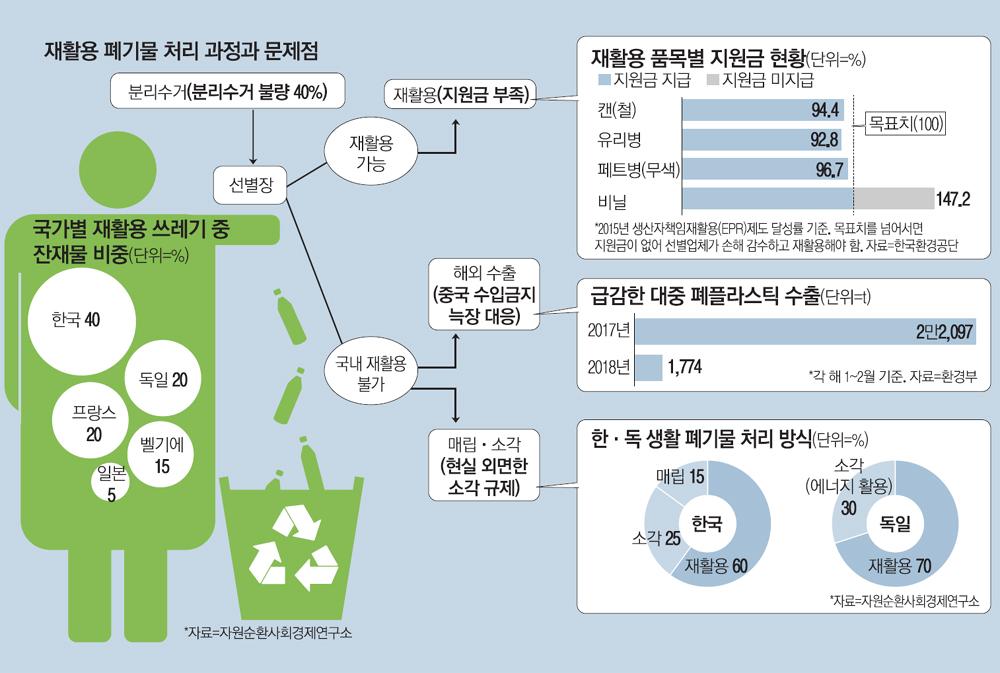
u [Appendix] User Manual.............................................................................. 5

# **1.** **Abstract**

본 프로젝트에서는 실생활의 문제 해결에 중점을 두었다. 따라서, 팀 구성원들이 실생활에서 불편함을 겪을 수 있는 문제를 찾아보기로 하였다. 그 결과, 분리 수거의 중요성에 대해 한 팀원이 언급하였다. 평소 아파트나 주택에서 분리수거를 대충하여 각 구역에 알맞지 않은 쓰레기들을 버리는 경우가 있다. 본 프로젝트에서는 이러한 잘못된 분리수거를 지양하고 올바른 분리수거를 할 수 있도록 돕는 IoT device 개발을 목표로 한다.

# **2.** **Introduction**

다음 사진은 여러 나라의 분리수거 잔재물 비율을 보여주고, 우리나라가 가진 분리수거의 문제점을 보여준다.



사람들은 최소 1주일에 2번에서 많으면 매일 매일 분리수거를 한다. 그러나 일반 가정집에서 나오는 분리수거 폐기물 중 대부분은 재활용 되지 못하고 버려진다. 왜 재활용을 하지 못하는 것일까? 일부는 이를 시민의식의 부재라고 이야기하기도 하고, 일부는 분리기준의 불명확성, 분리수거 표기가 눈에 띄지 않음 등의 문제라고도 한다. 분리수거를 제대로 할 수 있게 하는 방법에는 여러가지가 있다. 이러한 문제들을 해결할 수 있는 방법을 생각해보았다. 분리수거장에 cctv를 설치해서 제대로 하지 않는 사람들에게 벌금을 내게 하는 방법이 있을 수도 있고, 분리수거 하지 않고 모든 쓰레기를 모아 분리수거를 하는 직업을 만드는 방법도 생각해볼 수 있으며, 분리수거 캠페인을 통한 방법도 있고, 분리수거를 제대로 하지 않는 경우 알려주는 시스템을 설치할 수도 있을 것이다. 이러한 강압적인 방법을 많이 생각해보았고, 이를 통한 방법도 효율적이라고 생각했다. 하지만, 실제로 우리가 집중한 것은 분리수거가 번거롭고, 어렵다는 점이다. 좀 더 사람들이 편리하게 분리수거를 할 수 있다면, 분명히 분리수거율은 자연스럽게 올라갈 것이다. 이를 위해서 우리는 분리수거를 돕기 위한 IoT device를 개발을 시작했다.

요약하자면 다음과 같다.

분리수거를 제대로 하지 않아서 발생하는 비용이 매우 크다.  
 1) 분리 수거 현장에서 관리자의 추가적인 분리수거 관리.  
 2) 올바르지 못한 분리수거로 인한 환경오염.  
 3) 2차 분리소에서 분리하는 데 사용되는 물리적 비용.  
 4) 재활용 불가로 인한 범지구적 손해.

# **3.** **Background Study**

## **A.** **관련 접근방법/기술 장단점 분석**

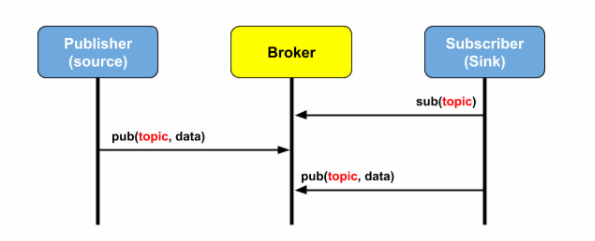
1. 통신 Protocol

(1) HTTP

HyperText Transfer Protocol의 약자인 HTTP는 www 상에서 정보를 주고받을 수 있는 프로토콜이다. 주로 html 문서를 주고받는데 사용되며, HTTP는 클라이언트 / 서버 사이에 이루어지는 응답 / 요청 프로토콜이다. 하지만 이는 모바일 기기를 위해 항상 최적화된 것은 아니다. HTTP는 MQTT 통신에 비해 비교적 무거운 프로토콜이다. HTTP가 가지는 장점은 확장성이 좋다는 것이지만 그 장점에 비해 라즈베리파이에서 사용하기에는 비교적 무거워 배터리 소모가 심하고, 3G network 측정 결과에서도 MQTT throughput이 HTTP보다 빨랐다 이러한 이유들 때문에 최근 IoT에서는 MQTT의 사용이 급증하고 있는 추세이다.

(2) MQTT

Message Queue Telemetry Transport의 약자인 MQTT는 통신장비, 모바일, 스마트폰 기기에 최적화된 가벼운 메시지 프로토콜이다. 현재 IoT에서 많이 사용되고 있다. MQTT가 IoT에 많이 사용되는 이유는 IoT와 같은 라이트 장비에 단순 메시지 교환을 위해 무거운 프로토콜을 사용하게 되면 성능 저하가 발생한다. 이를 해결할 수 있는 MQTT는 Broker라는 중계자가 존재해 중계자를 통해 Topic을 선정해서 subscribe하면 해당 Topic이 Publish 되었을 때 구독자들에게 메시지를 전달해주는 방식이다. 이를 간단하게 표현하면 아래 그림과 같다.



MQTT 브로커에는 다양한 종류가 있다. ActiveMQ, Apollo, JoramMQ, Mosquitto, RabbitMQ등 다양한 MQTT 브로커가 존재하는데, 이러한 MQTT 브로커를 통해 메시지를 주고받을 수 있고, 모바일 기기와의 통신도 가능하다. MQTT의 특징을 설명하면, 단순하고 가벼운 메시지 프로토콜이며 오버헤드를 최소화하기 위해 헤더 크기를 대폭 줄였다. 또한 클라이언트 서버간 연결이 끊어졌을 때 보정 기능을 제공한다는 장점이 있다. HTTP와 마찬가지로 TLS/SSL을 지원해 메시지 암호화도 할 수 있다.

1. Raspberry Pi의 통신 기술

(1) Bluetooth

Bluetooth는 수 미터에서 수십 미터 정도의 거리를 둔 정보기기 사이에, 전파를 이용해서 간단한 정보를 교환하는데 사용된다. bluetooth 통신은 단거리 크기가 작은 데이터를 대상으로 하는 것이 특징이다. 이로 인해 bluetooth는 단거리 기기간의 통신에 주로 하여 사용된다.

(2) Wi-Fi

Wi-Fi는 전자기기들이 [무선랜](https://ko.wikipedia.org/wiki/%EB%AC%B4%EC%84%A0%EB%9E%9C)(WLAN)에 연결할 수 있게 하는 기술이다. wifi는 대게 기기와 internet을 연결하여 web상의 데이터에 접근하는데에 사용된다.



<표1>

출처 :<http://bitly.kr/JYrQC5x>

사진이미지를 실시간으로 전송하면서 분석 결과를 얻어와야하는 해당 프로젝트의 성격상 wifi를 통해 데이터를 주고받는 것이 적절하다는 결론을 내렸다.

1. Server

Raspberry Pi와 데이터를 주고 받기 위하여 서버를 AWS IoT와 Django를 통해 개발을 진행할 것이다.

(1) AWS IoT

AWS IoT은(는) 인터넷 연결 제품(센서, 액추에이터, 내장형 마이크로 컨트롤러, 스마트 애플리케이션 등)과 AWS 클라우드 간에 안전한 양방향 통신을 제공합니다. 이를 통해 여러 디바이스에서 원격 측정 데이터를 수집하고 해당 데이터를 저장 및 분석할 수 있습니다. 또한 사용자가 휴대전화 또는 태블릿에서 이러한 디바이스를 제어할 수 있게 해주는 애플리케이션을 만들 수도 있습니다.

**Raspberry Pi와 IoT간의 동작**

① EC2 server가 MQTT publisher로 등록한다.

② Raspberry Pi에서 MQTT subscribe 하고 있는다.

③ EC2에서 machine learning result를 MQTT를 통해 Raspberry pi에게 전송한다.

(2) AWS Lambda

AWS Lambda를 사용하면 서버를 프로비저닝하거나 관리할 필요 없이 코드를 실행할 수 있습니다. 사용한 컴퓨팅 시간만큼만 비용을 지불하고, 코드가 실행되지 않을 때는 요금이 부과되지 않습니다.Lambda에서는 사실상 모든 유형의 애플리케이션이나 백엔드 서비스에 대한 코드를 별도의 관리 없이 실행할 수 있습니다. 코드를 업로드하기만 하면, Lambda에서 높은 가용성으로 코드를 실행 및 확장하는 데 필요한 모든 것을 처리합니다. 다른 AWS 서비스에서 코드를 자동으로 트리거하도록 설정하거나 웹 또는 모바일 앱에서 직접 코드를 호출할 수 있습니다.

(3) AWS ELB

AWS ELB를 이용하여 Server 부하에 대응하는 환경을 구축한다.

Elastic Load Balancing은 들어오는 애플리케이션 트래픽을 Amazon EC2 인스턴스, 컨테이너, IP 주소, Lambda 함수와 같은 여러 대상에 자동으로 분산시킵니다. Elastic Load Balancing은 단일 가용 영역 또는 여러 가용 영역에서 다양한 애플리케이션 부하를 처리할 수 있습니다. Elastic Load Balancing이 제공하는 세 가지 로드 밸런서는 모두 애플리케이션의 내결함성에 필요한 고가용성, 자동 확장/축소, 강력한 보안을 갖추고 있습니다.

1. Image Classification

tensorflow를 통해 쓰레기 이미지를 재활용 쓰레기 별로 분류한다.

- tensorflow

TensorFlow는 머신러닝을 위한 엔드 투 엔드 오픈소스 플랫폼입니다. 도구, 라이브러리, 커뮤니티 리소스로 구성된 포괄적이고 유연한 생태계를 통해 연구원들은 ML에서 첨단 기술을 구현할 수 있고 개발자들은 ML이 접목된 애플리케이션을 손쉽게 빌드 및 배포할 수 있습니다.

1. Sensor

Output으로 분리수거통을 열고 닫기 위해서 다음과 같은 sensor들이 이용된다.

(1) 초음파 센서



초음파 센서는 일정한 간격으로 높은 주파수 사운드 펄스를 통하여 짧게 소리를 발사합니다. 이러한 것들은 소리의 속도를 통하여 공기 중에 전파됩니다. 만약 그것들이 개체에 충돌한다면 센서에게 에코 신호로 반사되어 올 것이고 이러한 것은 시간-에코에 방사하고 도달하는 기간-을 통하여 목표까지의 거리를 산정하게 됩니다.

(2) 서보 모터



서보(Servo)란 시스템에서 해당 기기를 시스템이 요구하는 특정 위치로 이동하거나, 특정한 수치(속도, 토크 등)만큼 가동시킬 때, 피드백이나 에러 정정을 통해 정확하게 제어할 수 있는 구조를 갖추고 있다는 의미이다.

간단하게 말해서 서보모터라고 하면 일반적인 모터(원형으로 빙빙 돌기만 함)와는 달리 움직임을 지정하면 제어계측 회로에 의해 정확하게 움직일 수 있는 모터란 뜻이다.

(3) 카메라 모듈

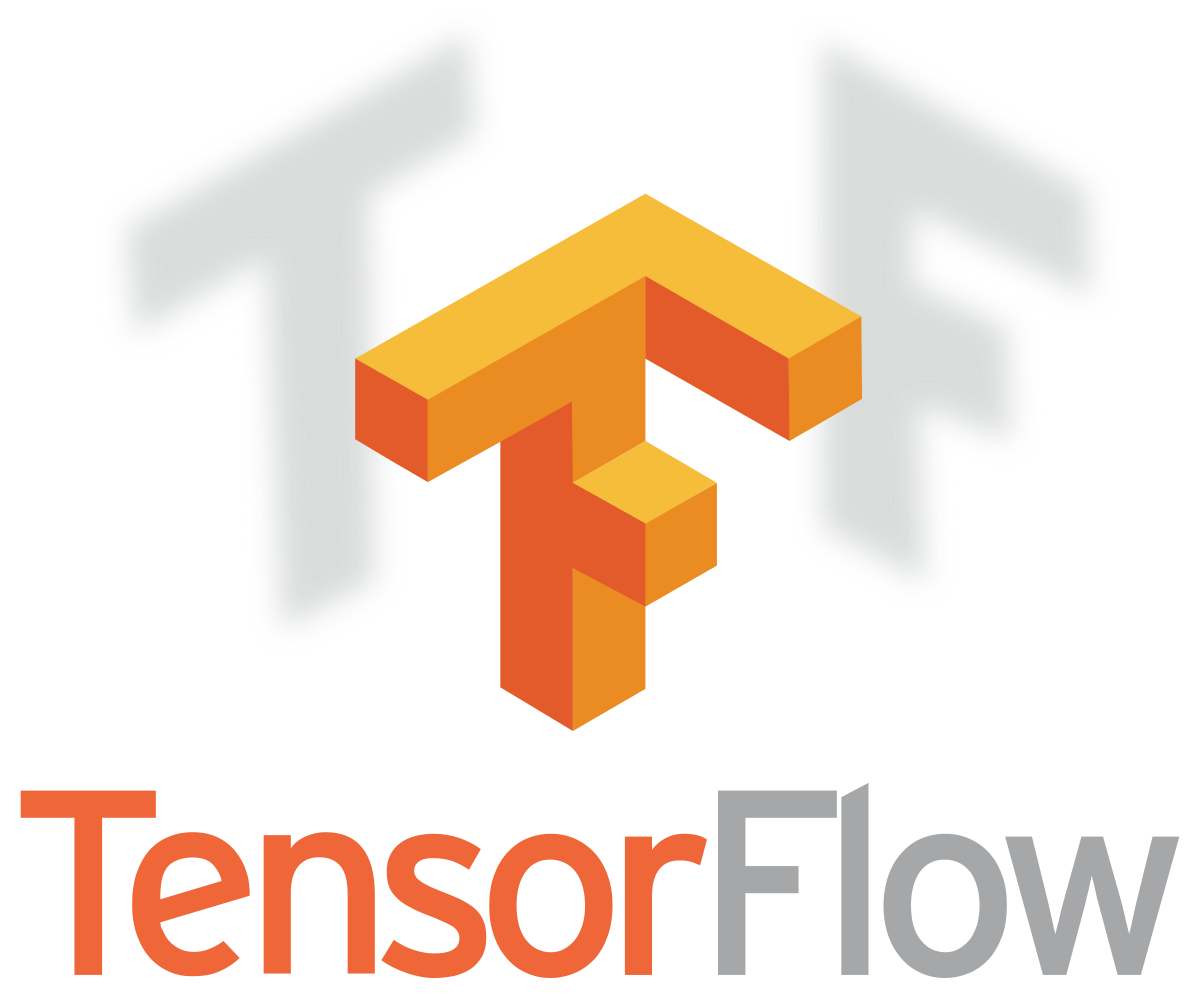
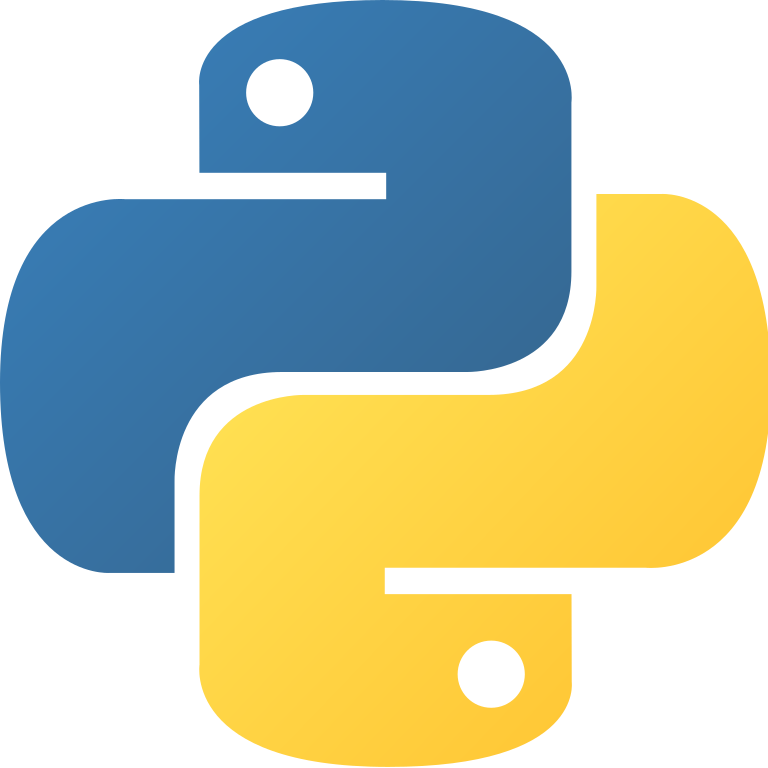


카메라 모듈을 통해서 라즈베리 파이가 영상을 저장하고 이를 서버에 보내는 것을 가능케한다. 해상도는 960 \* 640 동영상 촬영을 지원한다.

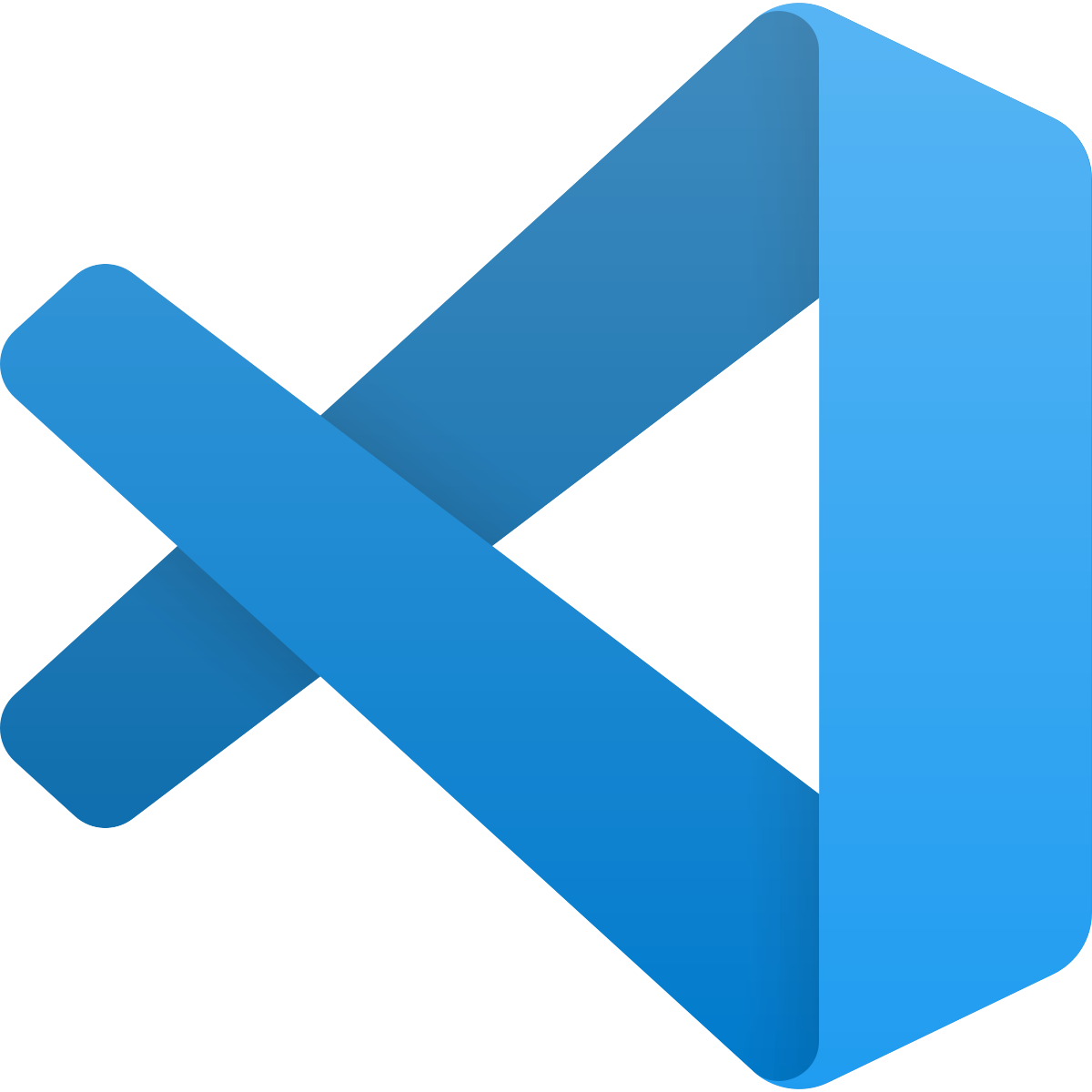
## 

## 

## **B.** **프로젝트 개발환경**

ww

Python TensorFlow



AWS VisualCode

# 

# 

# **4.** **Goal/Problem & Requirements**

* 목표

분리수거함에 잘못된 쓰레기가 들어가지 않도록 이미지 인식을 하고, 알맞은 뚜껑만을 열어주는 쓰레기통 개발.

* 요구사항

1. Raspberry pi 카메라 모듈을 이용해서 쓰레기 사진을 찍는다.
2. 찍은 사진을 전송한다.
3. 전송받은 사진을 구분한다.
4. 서버에서 분별결과를 라즈베리파이로 전송한다.
5. 해당 이미지의 결과에 알맞은 분리수거 통을 자동으로 연다.

* 예상되는 문제점(Challenge)

1. AWS Server 유지 비용 최소화
2. Server 부하에 대한 해결책
3. Response Time 최소화
4. Model 인식률

# **5.** **Approach**

* 요구 사항에 대한 Approach

1. 초음파 센서를 이용하여 일정거리에 들어오는지를 확인.
2. Raspberry pi 카메라 모듈을 이용해서 쓰레기 사진을 찍는다.
3. 찍은 사진을 S3로 전송한다.
4. S3에 Trigger를 걸어놓은 lambda가 ELB에 HTTP/POST를 보낸다.
5. 서버에는 쓰레기를 분별할 수 있는 모델이 있다.
6. 서버에서 분별결과를 라즈베리파이로 MQTT를 사용하여 전송한다.
7. 해당 이미지의 결과에 알맞은 분리수거 통을 자동으로 연다.

* Challege에 대한 Approach

1. AWS Server 유지 비용 최소화

고비용의 Machine Learning용 ec2 머신을 적절한 scale in/out을 적용할 수 있도록 auto scaling 수행.

1. Server 부하의 해결

* Raspberry Pi에서 1차로 쓸모없는 이미지는 보내지 않는다.

이를 위해 초음파 센서를 통해 일정거리에 들어오지 않으면 의미없는 데이터로 인식하고, 사진을 촬영하지 않는다.

* ELB를 통한 처리량 증가 시 탄력적인 Load Balancing 수행

1. Response Time 최소화

HTTP 통신보다 빠른 속도로 message 전송이 가능한 MQTT Protocol을 이용하여 가능한 빠르게 message를 전송할 수 있도록 한다.

1. 라즈베리파이와 서버와의 간단한 통신처리  
   pub/sub 방식인 MQTT를 통해 한 서버로부터 여러 라즈베리 파이에 효율적으로 쉽게 결과를 전달할 수 있다.
2. Model 인식률

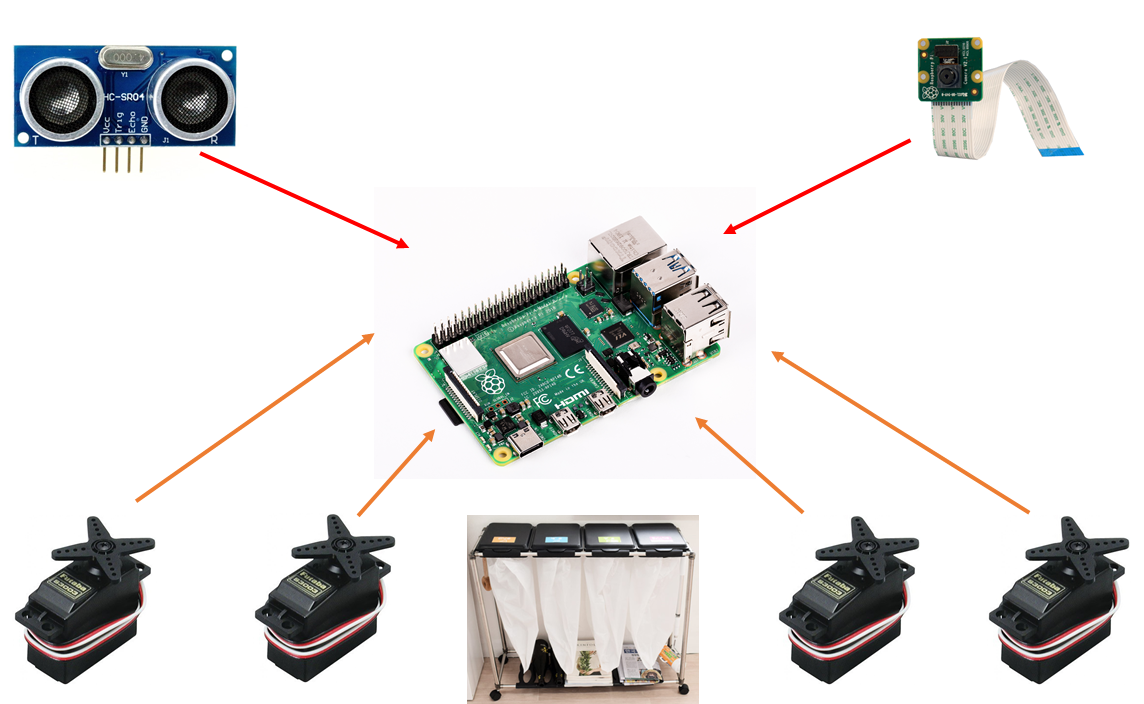
web에서 501개의 유리병, 410개의 캔, 592개의 종이, 467개의 플라스틱을 수집하였다. 추가 데이터셋으로 829장의 유리병, 883장의 캔, 840장의 종이, 840장의 플라스틱을 다양한 배경과 각도로 촬영하여 추가하였다. 각 이미지의 크기는 512X384이다. web에서 수집한 데이터셋으로 100epoch을 학습시키고

여기서 나온 weight를 pretrained weight로 사용한다. 추가로 모은 데이터셋을 추가하여 100epoch를 더 학습시킨다.

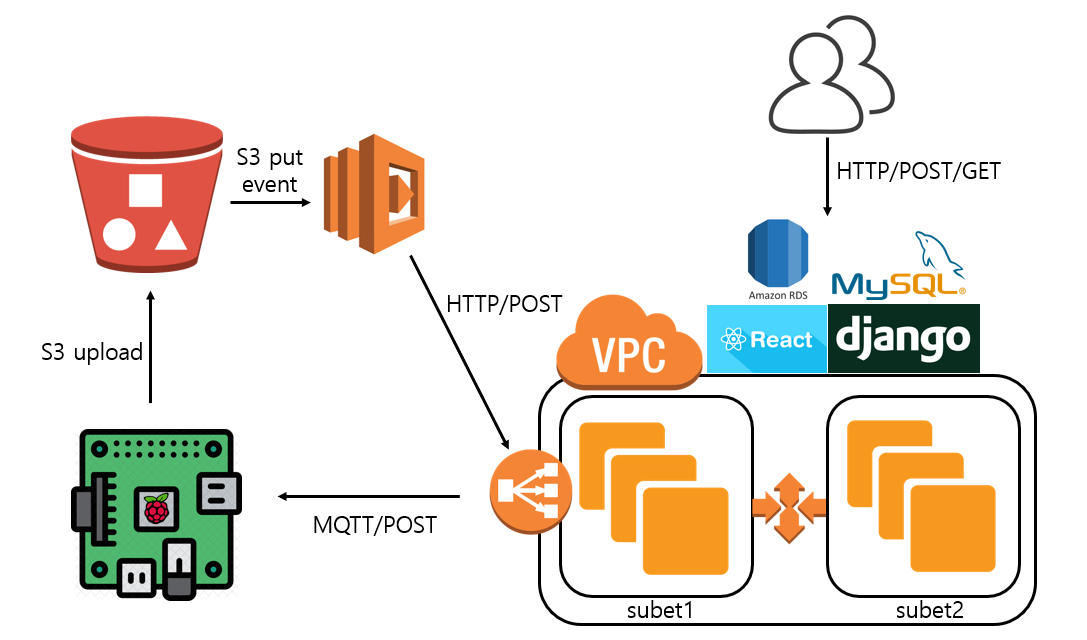
데이터셋의 20%인 1000장을 임의로 뽑아 테스트셋을 구성하고 그에 따른 mAP(mean Average Precision)가 충분히 높을 수 있도록 한다.

# **6.** **Project Architecture**

## **A.** **Architecture Diagram**



<Hardware architecture>



<AWS Network architecture>

## **B.** **Architecture Description**

1. Hardware architecture

- 하드웨어는 4개의 servo motor, 1개의 초음파 sensor, 1개의 camera module 그리고 Raspberry Pi로 이루어진다.

- 인식과정은 초음파 sensor를 통해서 일차로 데이터 입력을 확인한다.(일정 거리 안에 진입하였을 때부터 촬영 시작)

- 그 후에 camera module을 통해서 얻은 이미지를 S3 server에 업로드 한다.

- server에서 인식에 따른 결과가 나오면 알맞은 분리수거통 뚜껑을 servo motor를 이용하여 들어올린다.

1. AWS Network architecture  
   - Raspberry pi에서 S3에 이미지 file을 업로드하면, S3 PUT event를 trigger로 한 lambda가 구동한다.

- lambda는 ELB에게 HTTP/POST Request를 전송한다.

- ELB는 Load Balancing을 수행하여 요청을 분산한다.

- 만약 각 group의 EC2의 부하가 걸린다면, CPU 점유율 80%이상이 발생하면, Auto Scaling을 수행한다.

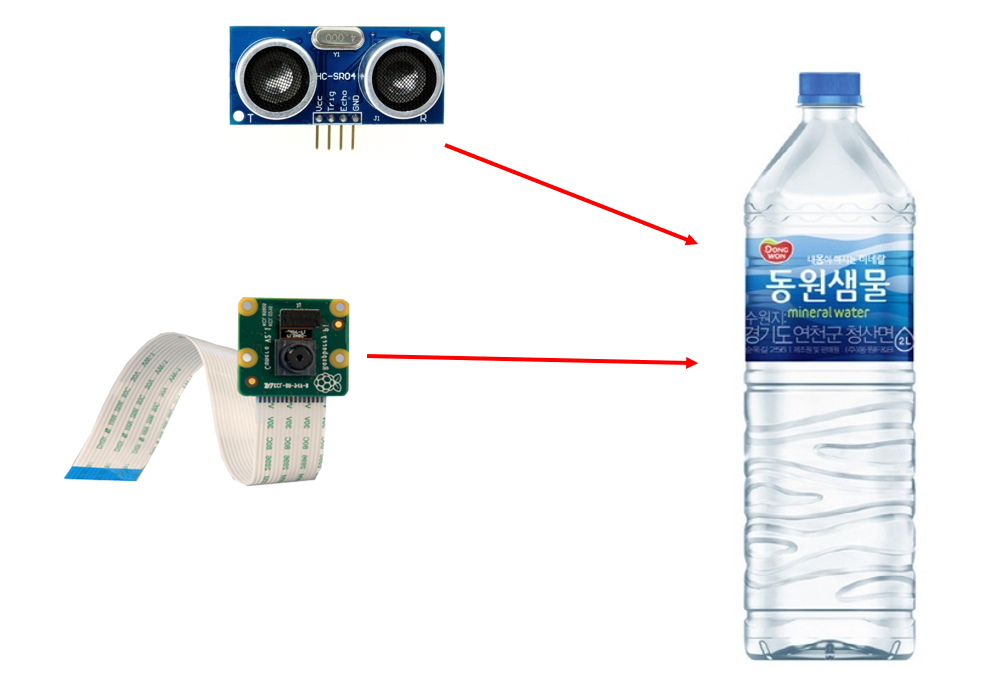
- 각 instance에서 machine learning을 수행하고 결과물을 MQTT/POST를 통해 Raspberry pi로 전송한다.

1. 기존 메인 서버를 Lambda에서 EC2로 교체한 이유  
   Lambda의 장점을 함수형으로 요청을 처리할 수 있다는 점이다. 매우 간편하고 빠르게 모듈을 만들 수 있다. 하지만 그만큼 람다가 커버할 수 있는 라이브러리가 제한적이다. python 3.7 버전 기준으로 람다 한 개당 올릴 수 있는 라이브러리 크기는 250MB가 제한이다. 따라서 본 프로젝트에서 사용하는 pytorch 1.2.1(370MB)를 탑재할 수 없을 뿐더러 Lambda에서는 GPU를 지원하지 않기 때문에 GPU를 지원하는 p2.large EC2를 메인서버로 작동 시킨다.
2. 사용자 페이지 아키텍쳐  
   사용자는 하루 단위로 배출된 쓰레기 데이터에 대해 쓰레기 양, 쓰레기 종류, 라즈베리 파이에서 찍힌 쓰레기 이미지에 대해 로그를 확인할 수 있는 웹페이지가 있다. 이 데이터들을 분석해 사용자는 필요한 작업을 진행할 수 있다.  
   RDS에 저장되는 데이터는 다음과 같다  
   TODAY(ID, GARBAGE\_TYPE, DATE, POSITIVE\_WASTEDCOUNT)  
   HISTORY(ID, GARBAGE\_TYPE, DATE, IMAGE\_PATH)  
   사용자 인터페이스는 다음과 같다.  
   

# **7.** **Implementation Spec**

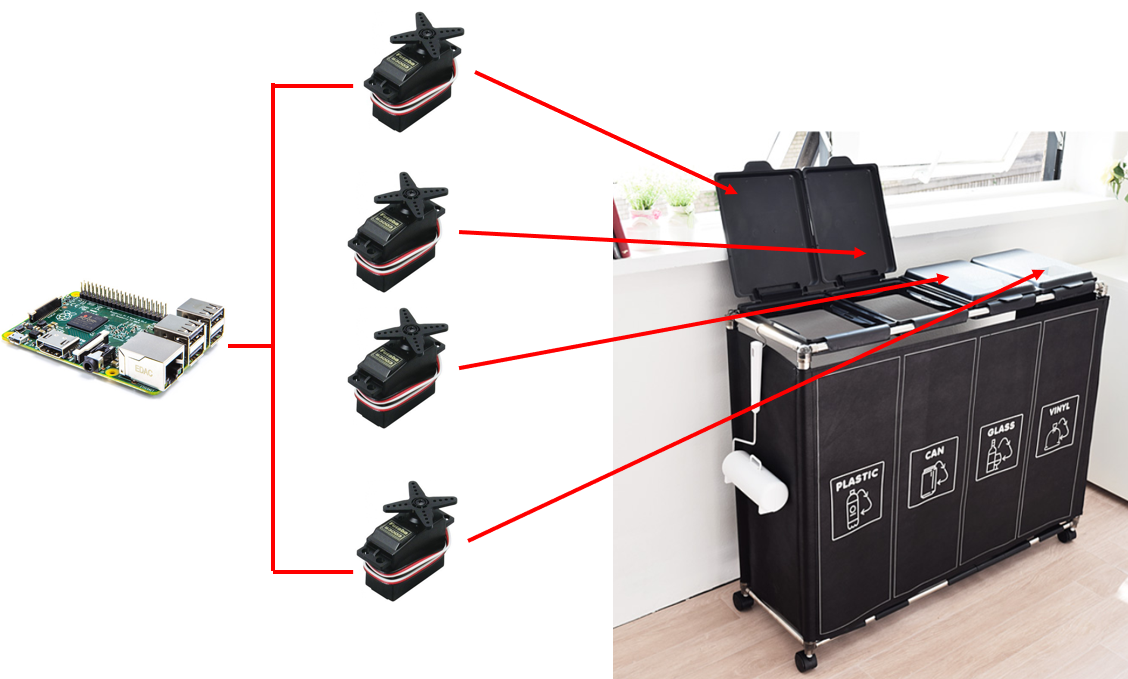
1. Input/Output Interface

-input interface



Raspberry pi는 카메라 모듈을 이용하여 이미지를 촬영하고, 이를 S3에 업로드한다. 하지만, 매순간 이 이미지를 보내는 것은 아니다. 초음파 sensor는 지속적으로 물체가 다가오는지를 탐지하고 있다. 그러던 중에, 만약 어떤 물체가 일정 거리 안에 접근했음을 감지했다면, camera module은 촬영을 시작하고 해당 이미지를 S3로 업로드한다.

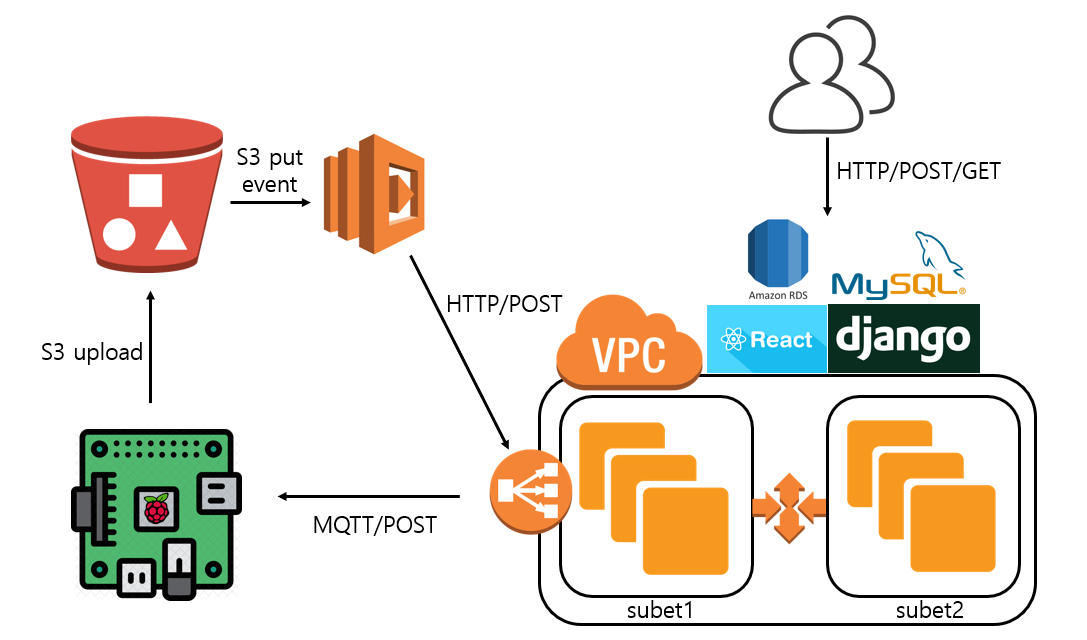
-output interface



Server에서 전송받은 결과를 바탕으로 4개의 servo motor를 이용하여 분리수거 쓰레기통의 뚜껑을 연다.

만약, 쓰레기가 유리라면, 유리를 담는 통의 뚜껑이 자동으로 열린다.

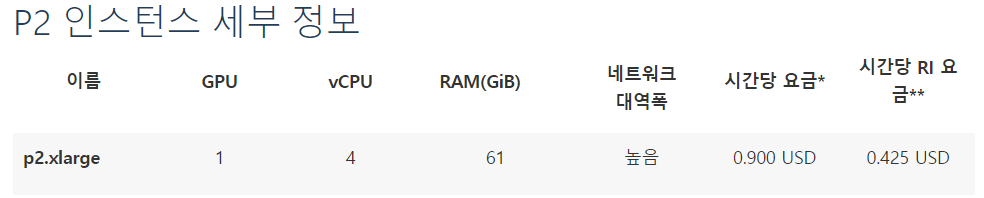
1. Inter Module Communication Interface



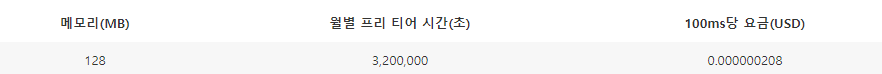
위의 그림은 본 프로젝트의 핵심적인 기능 및 커뮤니케이션을 도식화한 것이다.  
- 위의 그림의 조건을 맞추기 위해서 라즈베리파이 디바이스는 항상 wifi가 있는 곳에 설치되어야 한다.  
- 또한 서버도 분리수거가 진행되는 시간에는 항상 실행되어야 한다.  
- 디바이스가 서버에 데이터를 전송하기 위해 서버에서는 인바운드 규칙을 설정해 포트포워딩을 하도록하다.

\* 커뮤니케이션 과정 중 예상되는 문제: 본 수업에서는 크레딧 100$ 안에서 프로젝트를 완성 시켜야만 추가 점수가 주어진다.

<EC2 p2.xlarge COST>



<Lambda api COST>



<RDS COST>

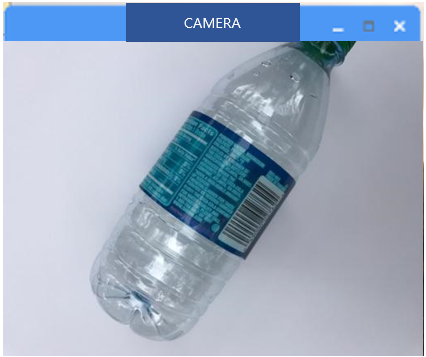


<S3 COST>

순차대로 EC2 p2.xlarge, Lambda, RDS t2.micro, S3의 코스트 비용이다.  
lambda와 S3, RDS는 작은 규모의 리소스를 이용하여 프리티어 만큼의 비용이 나올정도로 금액이 나오지만 GPU를 탑재한 EC2는 시간당 약 1달러의 비용이 청구 된다. 따라서 조심해서 사용할 요구사항이 주어지고 아래의 경우 본 프로젝트에서 사용한 총 크레딧이다.

|  |  |
| --- | --- |
| EC2 | 55.56$ |
| RDS | 4.5$ |
| Lambda | 0$ |
| S3 | 0$ |
| 총계 | 60.06$ |

1. Modules



위의 그림은 본 프로젝트의 핵심 모듈인 카메라 모듈의 예상 되는 인풋 데이터 영상이다. 카메라 모듈 해상도는 640\*480으로 4:3(가로:세로)의 형태로 서버에 전송된다. 해당 영상을 분류할 수 있는 모델은 서버에서 작동한다.  
- 모델 학습과 테스트: 공공 garbage dataset(<https://www.kaggle.com/asdasdasasdas/garbage-classification>)  
본 프로젝트에서는 여러 개의 오브젝트에 대해 분류를 하기 위해 디텍션과 분류를 동시에 실시간에 강하게 빠른 처리를 해주는 Yolov3를 사용하기로 하였다.

**8. Solution**

1. Implementations Details   
     
   1. Raspberry Pi 세팅  
     
   앞서 언급하였듯이 여러 모듈을 이용해 쓰레기를 인식하고   
     
   <Raspberry>  
   Raspberry  
   H/W Sensor : 초음파 센서  
   get\_distance(): 인식 대상과의 거리를 측정합니다.  
     
   Raspberry  
   upload\_image(): 카메라 모듈에서 촬영한 이미지를 S3로 upload합니다.  
     
   Raspberry  
   H/W Sensor : 서보 모터  
   duration : 5s  
   open\_cap(int target): 인식 결과에 따라 해당하는 쓰레기통의 뚜껑을 들어올립니다.

2. AWS lambda

S3에 이미지가 올려지면 EC2에 POST 요청을 보내게 된다. 다음 내용은 api이다.

3. AWS Lambda & EC2 & S3 & ELB

모델을 탑재하기 위해서 EC2에서 django로 웹 서버를 연다. 여러 라즈베리로 부터의 요청에 응답하기 위해서 ELB을 환경을 설정하여 요청의 분산을 수행한다.

4. YOLOv3

YOLOv3를 구성하는 각 convolution층에 배치정규화를 적용하였고 활성함수로는 leakyReLU를 사용하였다. 학습률은 0.001을 주고 훈련데이터셋을 셔플을 이용해서 랜덤으로 학습시켰다. 공공데이터셋에 존재하는 501개의 유리병, 410개의 캔, 592개의 종이, 467개의 플라스틱을 라벨링하여 학습을 진행하였다. 100epoch 학습결과 mAP(mean Average Precision)가 높지 않아서 추가 데이터셋을 수집하였다. 829개의 유리병, 883장의 캔, 840장의 종이, 840장의 플라스틱을 추가하여 100epoch 학습은 진행하였다. 각 이미지의 크기는 512X384이다.

<Machine Learning & Yolo>  
classification() : 서버에서 받은 이미지를 재활용 쓰레기 별로 구분한다.(classification)

5.MQTT 통신

<MQTT Method: POST>

content : result  
result(): 인식 결과를 대상에게 전송한다.(glass = 1, plastic = 2, paper = 3, can = 4, other = 5)

<MQTT>

MQTT Method: publish  
mqtt.publish(“topic”, message);  
메시지를 해당 topic에 발행한다.

MQTT Method: subscribe  
mqtt.subscribe(“topic”);  
해당 topic에 구독 한다.

6. 웹 프론트엔드

쓰레기 분리 과정에 있어서 분리된 데이터 개수와 그때 당시 이미지에 대한 데이터 및 로그가 남게되어 그 부분을 시각화 해준다. react js와 django를 이용해 구성한다. 아래는 api에 대한 내용이다.

<frontend>  
<http://15.164.168.205:3000/>today  
api: get  
{  
garbage\_type, date, positive\_wastedcount  
}  
  
<http://15.164.168.205:3000/>history  
api:post  
params:  
{  
garbage\_type, date  
}  
returns:  
{  
id, garbage\_type, date, image\_path   
}

1. Implementations Issues

1. Raspberry Pi 세팅

* 실제 물리적으로 라즈베리파이와 여러 모듈을 실세계의 쓰레기통에 연결하는 과정에서 단단히 고정시키기 위해 노력을 많이 하였다.

2. AWS lambda

* Lambda에서 S3에 이미지가 올라간 후 트리거 작업이 오도록 설정하는데 안되서 수고가 있었다.

3. AWS EC2 & S3 & ELB

* 모델을 돌리기 위해 GPU가 탑재된 머신을 사용해야만 했다.
* 여러 모듈과 라이브러리를 conda 채널에 맞춰서 설치하는데 있어서 환결설정에 노력을 들였다.
* EC2 머신의 이미지를 만들어 놓고 로드 밸런싱을 해야 하는데 그 부분에 무지해서 처음에 구조를 파악하는데 고생이 있었다.

4. YOLOv3

* 학습시키는 데이터셋의 이미지의 해상도가 너무 높을 경우 학습이 오래 걸렸고 CUDA run time error: out of memory가 발생하는 경우가 있었다. 데이터셋의 이미지들의 크기를 작게 만드는 전처리를 통해 해결하였다.

5. MQTT 통신

* 강의 ppt 자료, AWS에 요청해서 얻은 자료안의 코드를 이용 해도 통신이 이루어지지 않았었지만 마지막으로 AWS 측에서 답변의 코드가 작동하여서 서버로부터 라즈베리파이로 결과값 리턴이 전달되어진다.

6. 웹 프론트엔드

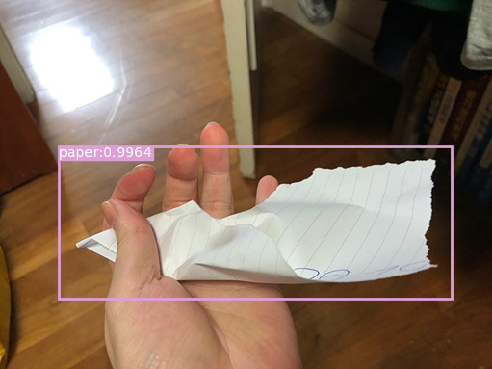
* 프론트 개발자가 아니다 보니까 React.js를 사용하는데 있어서 익숙하지 않아서, 웹페이지를 구축하는데 시간을 조금 소모하였다.

# **9.** **Result**

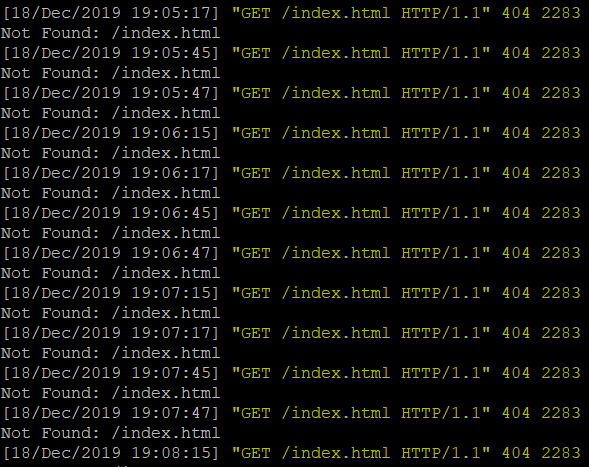
1. Experiments  
   1. YOLOv3 mAP 인식률

8-A-4에서 언급한 테스트 셋을 전체 데이터셋의 20%를 랜덤으로 뽑아 구성하고 그 결과는 다음과 같다.

|  |  |  |
| --- | --- | --- |
|  | testset1 | testset2 |
| glass AP | 0.998903 | 0.985142 |
| metal AP | 0.979075 | 0.956397 |
| paper AP | 0.995086 | 0.991159 |
| plastic AP | 0.992322 | 0.985993 |
| mAP | 0.991346 | 0.979672 |



2. ELB가 정상적으로 동작하여 해당 머신에 핑을 계속 보내는 것을 나타낸다. 404 에러가 난 이유는 index.html 파일을 리턴해주지 않기 때문이다. 리턴을 안해줘도 동작하는데 있어서 문제가 생기지 않는다.



1. Result Analysis and Discussion

input으로 들어온 재활용품을 인식 및 분류를 성공적으로 수행하였다. 서버에서 모델을 이용해서 분류를 수행할 때, Inference Time은 약 0.2초에 달했다. 하지만 입력값과 결과값을 주고받을 때, 네트워크 상태에 따라 약간의 지연이 발생하였다. 네트워크 환경이 좋다면 실시간으로 재활용품을 인식하고 분류결과에 따라 쓰레기통 뚜겅을 충분히 제어할 수 있을 것이다.

**10.** **Division & Assignment of Work**

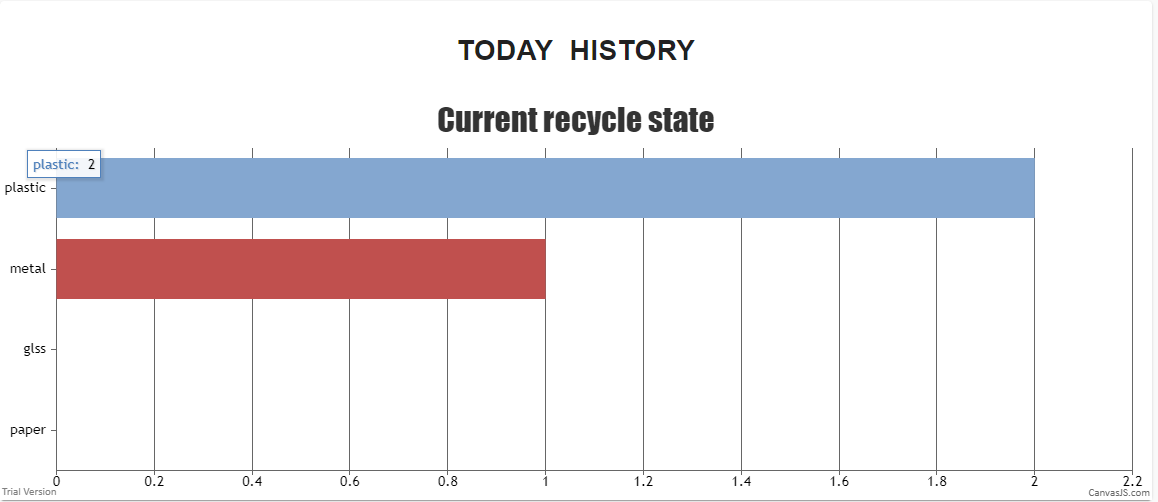
|  |  |
| --- | --- |
| 항목 | 담당자 |
| Raspberry Pi 세팅 | 정의동, 김효준 |
| AWS 환경 구축 | 정의동 |
| 쓰레기 분류 모델 생성 | 김원규, 백종현 |
| 데이터셋 수집 | 정의동, 김원규, 김효준, 백종현 |
| 관련 논문 연구 | 백종현 |
| 하드웨어 구축 | 정의동 |
| 프론트엔드 section | 김원규 |
| MQTT 통신 설계 | 김효준 |

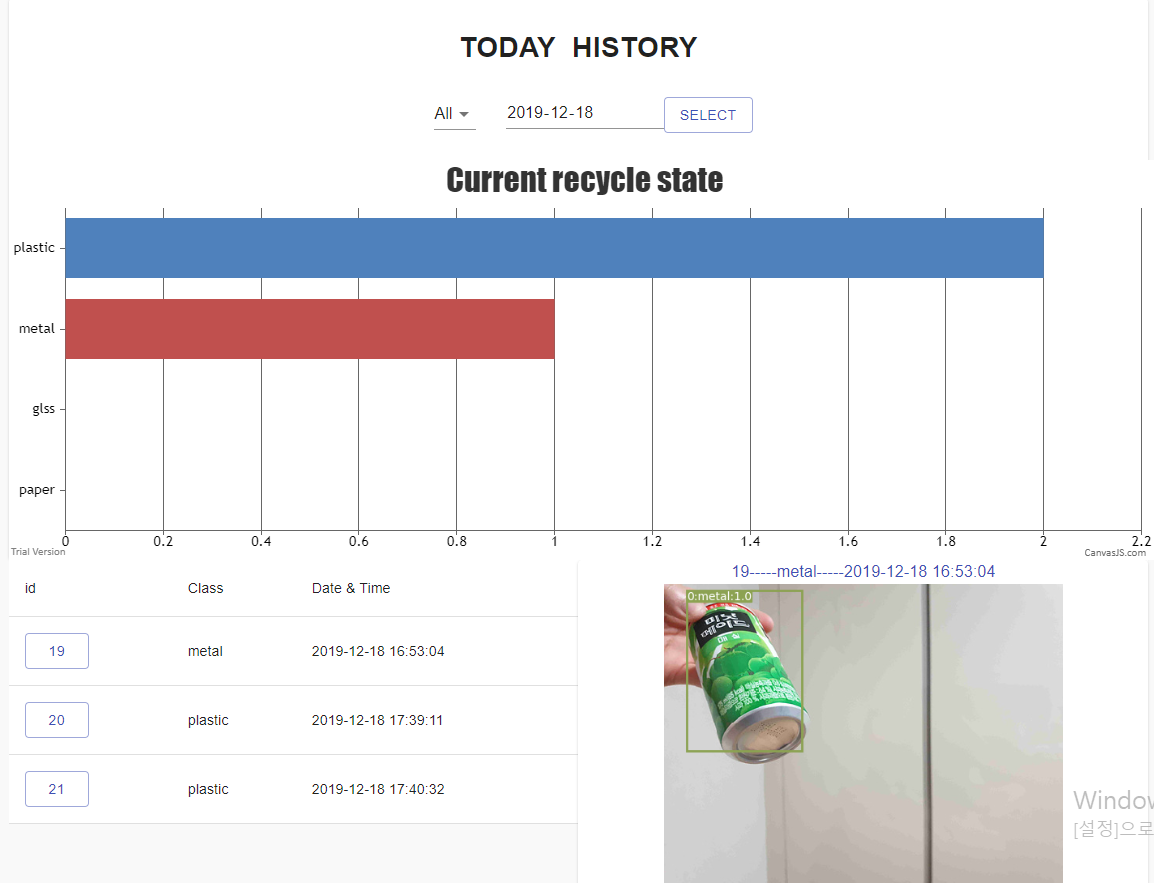
**11.**  **Conclusion**

* 실생활에서 분리수거의 수행이 잘 이루어지지 않아 그 문제를 해결할 수 있도록 가정용 쓰레기 도우미를 만든다.
* 가정용 쓰레기 도우미란 카메라로 쓰레기를 인식한 후 해당 종류에 해당하는 뚜껑 부분을 열어주어 분리수거를 확실히 할 수 있도록 동작하는 IoT 시스템이다.
* 실세계의 쓰레기통에 카메라 모듈, 모터 모듈, 초음파 모듈을 이용해 서버 단의 모델에 쓰레기 이미지를 보내준다. 이때 S3에 이미지를 올리면 Lambda에서 그 트리거를 인식해 EC2 서버단에 있는 모델에 이미지를 보낸다. 모델은 이미지를 인식하여 어떤 종류의 쓰레기인지 라즈베리파이에 결과를 보낸다.
* 모델의 결과는 언급하였듯이 높은 인식률을 보이고 있다.
* 데이터 시각화를 위해 웹 페이지를 작성하였다.

◇ **[Appendix] User Manual 이거 8번으로 올라가야함**

<abstract user manual>  
1. 유저는 쓰레기를 쓰레기통 앞으로 들고 온다.  
2. 쓰레기를 카메라 모듈 앞에 인식하게 한다.  
3. 쓰레기통은 해당 쓰레기에 맞는 뚜껑을 열어준다.  
4. 쓰레기를 버린다.  
5. 웹페이지에 들어가서 버려진 쓰레기에 대해 날짜별로 확인할 수 있다.  
<http://15.164.168.205:3000/>

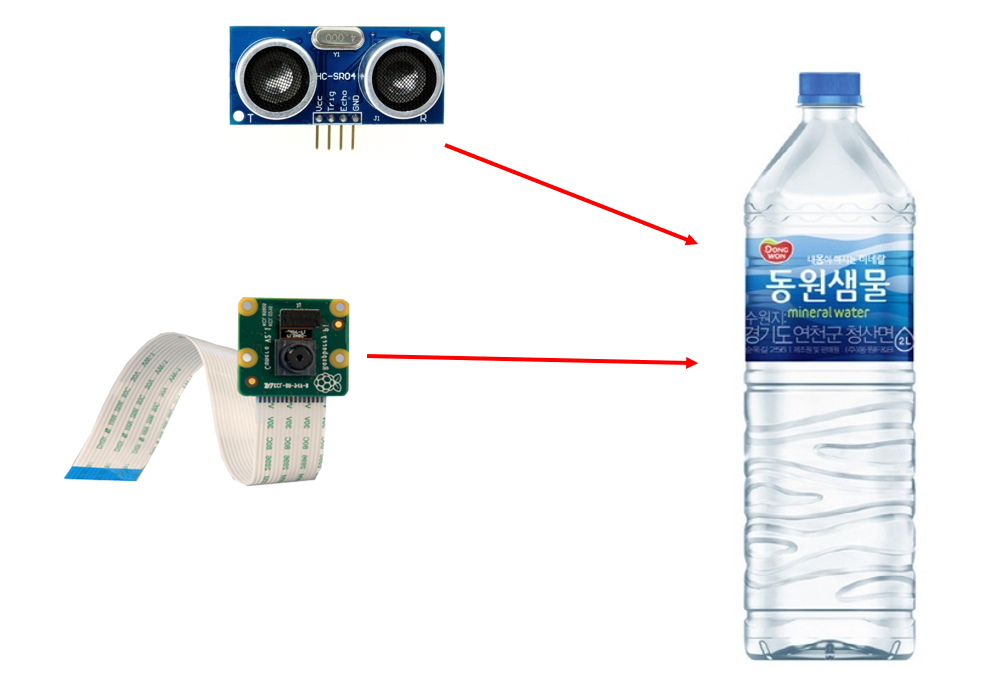
6. 오늘 버려진 쓰레기에 대해 확인한다.  


7. 이전 날짜의 쓰레기에 대해 확인한다.  


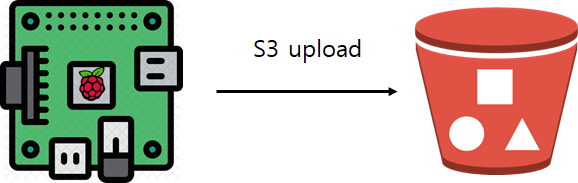
<detail user manual>

7-A,B의 인풋, 아웃풋 인터페이스에서 언급하였듯이 사물들을 이용해 실제 가정에서의 분리수거 환경처럼 데모 환경을 조성해 시연을 할 계획이다.

1. 라즈베리파이의 초음파센서를 통해 물체가 근처에 존재하는지 판단한다.
2. 물체가 존재한다면, 카메라 모듈에서 찍은 이미지를 라즈베리파이에 저장한다.



1. Raspberry pi에서 AWS S3에 이미지를 업로드한다.



1. S3 Put event를 trigger로한 lambda가 ELB에 HTTP/POST를 요청한다.



1. ELB에서 EC2에게 분산 요청을 수행한다.
2. 각 EC2 Instance는 machine learning을 통해 classification을 수행한다.



1. 결과를 MQTT/POST를 Raspberry pi에 전송한다.



1. 라즈베리파이는 결과에 맞게 분리수거 통의 뚜껑을 연다.

