화재 대응을 위한 불꽃 인식 시스템

경희대학교 컴퓨터공학과 캡스톤디자인1

2015104170 김효준 2015104195 이동찬 2015104216 장준영

요약

불의 사용은 인류를 이롭게 했으나 때때로 화마가 되어 돌아왔다. 오늘날 화재를 대응하기 위해 여러 기술들이 존재하고 있으나 여전히 부족한 것이 사실이다. 본 문서는 빠르게 발전하고 있는 정보통신기술을 통해 화재 초기 대응을 위한 시스템을 제안한다. 또 이를 기반으로, 불꽃을 인식하는 딥러닝 기반 인공지능과 화재 발생 시 사용자에게 적절한 도움을 줄 수 있는 사용자 인터페이스 및 사물인터넷 환경을 구현한다.

1. 서론

1.1. 연구 배경

통계청의 통계 결과, 최근 10년간의 통계를 보면 국내 재난사고 발생 중 화재와 산불이 약 19%으로 약 78%인 도로교통 다음으로 높았다. 전국의 화재 건수는 17년 기준 44,178건, 18년 42,338건, 19년 40,102건으로 매년 줄어들고 있다. 그러나 화재는 약 4만 건으로 지속적으로 발생하고 있으며, 이는 많은 인명 및 재산 피해로 이어지고 있다. 모든 사고와 재난은 초기 대응을 적절히 했을 때 피해를 최소화할 수 있는 것이 사실이다. 따라서 화재에 따른 빠른 대응이 가능하도록 화재의 발생을 인지하고 알리는 것은 중요하다.

화재의 발생을 인지하는 방법은 여러가지가 있다. 화재를 감지하는 방법은 크게 두가지로 연기 또는 열을 이용한다. 그러나 기존의 연기 감지기와 열 감지기의 경우, 통풍이 원활한 실내 또는 실외에서 제대로 역할을 해낼 수 없다. 또 화재로 이어지는 작은 불은 감지하기 어렵다. 다른 방법으로는 사람이 화재 환경을 직접 감시하는 방법이다. 기존 카메라 또는 열화상 카메라를 이용하여 CCTV를 통해 사람이 확인하고 판단할 수 있다. 이는 가장 정확한 방법이며 빠른 대응을 기대할 수 있다. 그러나 사람은 집중력이 떨어지면 생각이 분산되고 실수를 일으킨다. 재난의 대응에 있어 실수는 큰 피해를 가져온다. 여기서 불완전한 사람의 실수를 막기 위해 인공지능을 제안할 수 있다. 최근의 인공지능은 딥러닝 기술을 통해 사람보다 높은 인지능력을 갖는 경우가 더 많다. 이에따라 사람이 화재를 인식하고 초기 대응을 수행하는 것과 같이, 인공지능이 화재를 인식하고 화재의 발생을 알릴 수 있는 시스템 구현을 고려하게 되었다.

1.2. 연구 목표

첫째, 딥러닝 기반 인공지능을 통해 불꽃을 인식한다. 영상 속에서 불꽃의 위치를 확인할 수 있도록 object detection 모델을 구현한다. 또한, 모델이 다른 환경에 비해 비교적 자원이 한정된 raspberry pi 위에서 구동될 수 있게 모델을 압축한다.

둘째, 현재 상황에 대한 정보를 사용자에게 전달한다. 현재 감시하고 있는 환경을 사용자가 확인할 수 있도록 실시간 스트리밍과 화재 발생 시 사용자에게 보여질 정보를 제공할 수 있는 API를 구현한다.

셋째, 사용자 인터페이스를 통해 사용자의 올바른 대응을 돕는다. 제공 받은 정보를 사용자가 직관적으로 이해할 수 있도록 표현한다. 사용자에게 상황을 인지시키고 사용자의 판단에 따른 대응이 가능한 인터페이스를 구현한다.

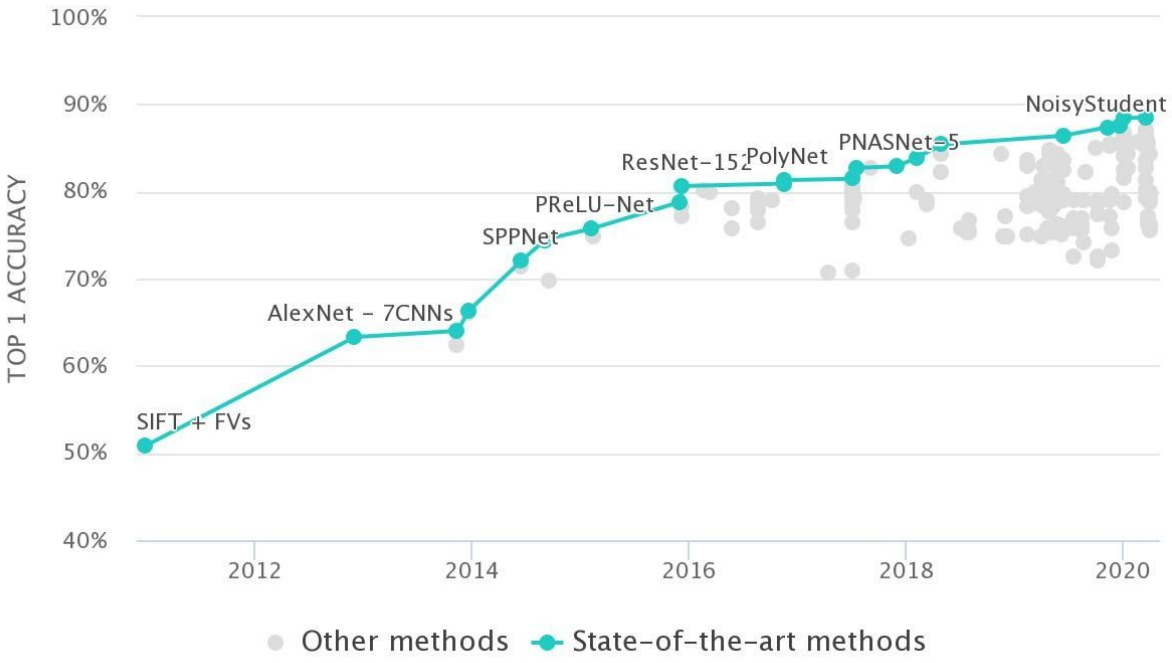
위 3개의 목표에 따라 시스템의 기능을 구현하며, 추가 기능의 필요에 따라 연구 목표를 재설정하고 화재 대응을 위한 불꽃 인식 시스템을 완성한다.

2. 관련 연구

2.1. 기존 연구

2.1.1. Image Classification and Object Detection

오늘날 컴퓨터비전을 위한 수많은 딥러닝 연구가 이어지고 있다. 그 중 image classification은 입력이미지를 정해진 하나의 카테고리로 분류하는 문제이다. 하드웨어 기술의 발전에 따라 딥러닝 모델의 정확도는 높아졌고, 현재 ImageNet에서 SOTA(State-of-the-Art)인 모델의 경우 일반적인 사람 이상의 정확도를 가진다.



[그림1] Image classification on ImageNet (top 1 accuracy)

단순히 이미지를 분류하는 것에 그치지 않고, 분류한 카테고리에 해당하는 객체의 이미지 속 위치를 파악하는 object detection 문제가 존재한다. Image classification을 훌륭히 수행하는 여러 Convolutional Neural Networks를 기반으로, 다양한 기법을 더하여 object detection 문제에서 뛰어난 성능을 보이고 있다.

2.1.2. Existing communication technology of Raspberry Pi

기존의 라즈베리파이 통신기술로는 크게 유 / 무선 통신이 있다. 이 중 우리가 중점적으로 보는 부분은 무선 통신이다. 무선 통신의 종류로는 bluetooth, wifi 등이 있다.

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

와이파이는 전자기기들이 무선랜에 연결할 수 있도록 하는 기술로, 대게 기기와 인터넷을 연결하여 web상의 데이터에 접근하도록 하게 해준다.

	와이파이(Wi-Fi)	블루투스(Bluetooth)
주파수(Frequency)	2.4GHz, 5GHz	2.4GHz
사용 목적	무선 인터넷 접속(WLAN)	휴대용 전자기기 간의 무선 연결(WPAN)
전송 속도	802.11B : 22Mbps 802.11G : 54Mbps 802.11N : 600Mbps 802.11AC : 1750Mbps	와이파이보다 느리나 버전업하면서 점차 빨라짐 25Mbps (ver.4.0)
스트리밍 데이터의 압축	압축 없이 무손실 전송 가능(AoIP)	대역폭이 좁아서 손실 압축하여야 함
오디오 신호의 지연(Latency)	VoIP : 20~30msec AoIP : 20msec 이하	SBC(Standard Bluetooth Codec): 약 250msec atpX 코덱: 약 180msec atpX LL 코덱: 약 40msec
장점	높은 전송속도 거리가 길다.	저비용 낮은 소비전력
단점	고비용 높은 소비전력	낮은 전송속도 거리가 짧다.

[그림2] Wi-Fi 와 Bluetooth

실시간을 요구하는 프로젝트에서는 wifi가 더 적절하다.

2.1.3. Webcam streaming & Application UI design(Android studio)

본 연구에서는 불꽃이 인식되면 사용자에게 알람이 가고, 불꽃이 인식된 장소의 상황을 application의 형태로 사용자에게 제공할 예정이다. 따라서 raspberry pi의 webcam을 통해 촬영되는 영상에서 불꽃이 인식되면 실제로 불꽃의 존재를 직접 확인할 수 있는 방법이

필요하다. 본 연구에서는 이를 webcam streaming을 이용해 사용자에게 실시간 영상을 제공할 것이다.

또 화재가 발생했다는 알람을 받을 수 있고, 위에서 언급한 streaming 영상을 볼 수 있는 환경을 제공하기 위해, android studio를 이용하여 간결하고 쓰기 용이한 UI로 구성된 application을 구현할 예정이다.

2.2. 기존 연구의 한계

2.2.1. 성능을 위한 높은 비용

여러 연구자들이 지적하고 있듯이 딥러닝 모델의 성능과 모델의 무게는 trade-off 관계에 놓여있다. 지금까지 Object detection의 성능에서 엄청난 발전이 있었지만 그에 따라 많은 연산량과 매개 변수를 갖는 것이 사실이다. 따라서 한정된 컴퓨팅 자원에서 높은 성능을 보이는 object detection 모델을 구동하는 것에는 한계가 존재할 수 있다.

본 문서에서 제안하는 연구의 경우, 학습된 object detection 모델을 비교적 한정된 자원인 raspberry pi 위에서 실시간으로 구동되는 것을 하나의 목표로 두고 있다. 따라서 단순히 성능이 좋은 모델을 선택하여 시스템을 구성할 수 없으며 이를 해결할 수 있는 방안이 필요하다.

2.1.2. 비교적 무거운 프로토콜

우리가 일반적으로 웹에서 자주 사용하는 HTTP 의 경우, 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. HTTP가 가지는 장점은 확장성이 좋다는 것이지만 그 장점에 비해 raspberry pi 에서 사용하기에는 비교적 무거워 배터리 소모가 심하고, 3G network 측정 결과에서도 타 프로토콜에 비해 throughput이 좋지 않았다. 확장성을 조금 줄이고, 라즈베리파이에서 사용할 수 있는 가벼운 프로토콜을 사용해야 한다.

2.1.3. 안정적인 스트리밍 영상,UI 제공의 필요성

성공적으로 연구의 목적을 이루려면 object detection까지 완료된 영상이 성공적으로 사용자에게 전송되어야 한다. 따라서 webcam streaming 영상을 실시간으로 안전하고 끊김없이 사용자에게 제공할 수 있는 환경이 필요하다. 또한 raspberry pi에서 영상의 모델링과 전송이 모두 이루어지므로 간단하고 편리하게 webcam streaming을 할 수 있어야 한다. 따라서 이 환경을 제공할 수 있는 적절한 기능을 raspberry pi에 구현해야 한다.

2.3. 해결방안

2.3.1. 모델 압축

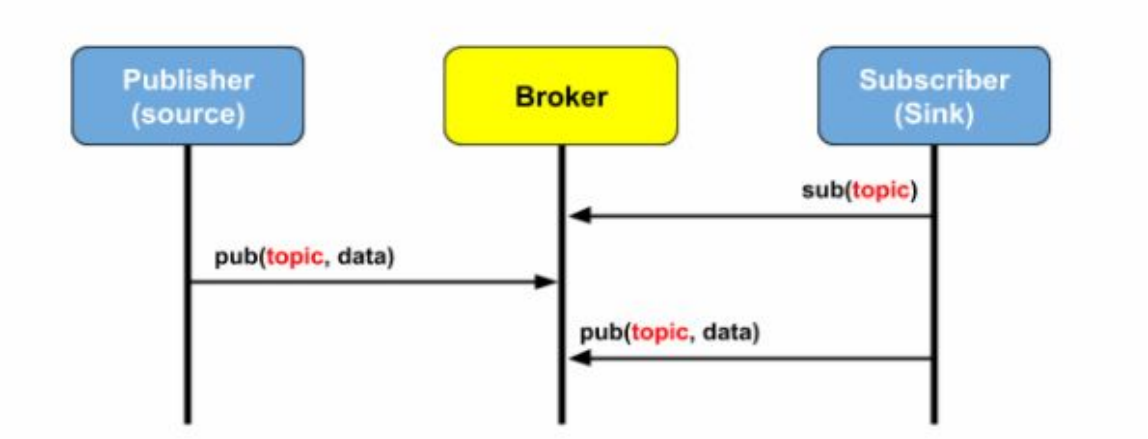
딥러닝 모델의 성능을 유지하며 모델의 크기를 줄이는 방법은 대표적으로 knowledge distillation과 quantization 방법이 있다. knowledge distillation은 잘 학습된 신경망의 softmax 분포를 이용하여 비교적 작은 크기의 신경망에서 학습시킬 때, 같은 크기의 신경망에서 일반적인 방법으로 학습시킨 것보다 성능이 좋음을 보여준다.[동찬3] 또 quantization은

모델의 매개 변수를 나타낼 때 사용되는 숫자의 정밀도를 줄이는 것으로 더 작은 모델 크기와 더 빠른 연산이 가능하게 한다.

제안하는 시스템에서와 같이, raspberry pi 위에서 불꽃을 정확하고 지연 없이 인식하기 위해서는 모델 압축이 필요하다. 위와 같은 해결 방안을 토대로 실용성 있는 방법을 통해 모델을 압축할 계획이며, 이들 외에 시스템의 완성도를 높일 수 있는 방안이 있다면 참고하여 적용한다.

2.3.2. 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.



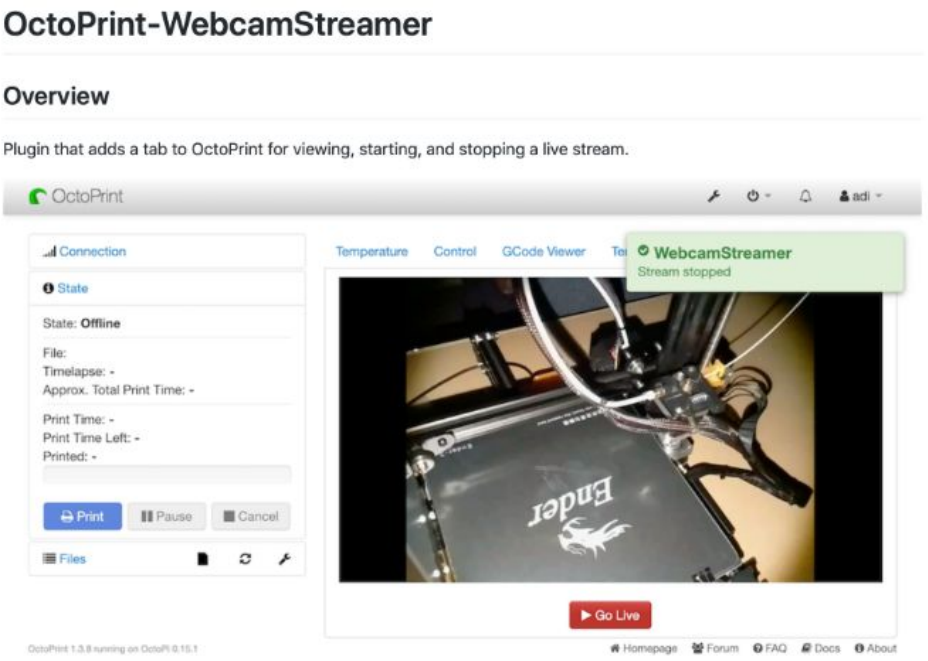
[그림3] MQTT의 동작

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.

2.3.3. OctoPrint(Webcam Streaming)

OctoPrint는 라즈베리파이에 OctoPi를 설치해 프린터에 연결하여 사용하는 프로그램으로, 3D 프린터에 대한 원격 제어를 목적으로 만들어진 open source이다. 이를 사용하면 실시간으로 프린트되는 과정을 볼 수도 있는데, 이번 연구에서 사용할 기능이 바로 이 실시간 webcam streaming이다. OctoPrint의 source를 분석하여 streaming 및 화면을 송출하는 UI에

대한 부분을 사용하여 연구를 진행할 예정이다. 또한 실제로 Octoprint의 소스를 활용하여 개발된 여러가지 application들이 존재하기도 하므로, 이 또한 참고하여 Android app 개발에 활용할 것이다.



[그림4] OctoPrint - WebcamStreamer

3. 프로젝트

3.1. 시스템 시나리오

3.1.1. 실시간 스트리밍

[그림5] 실시간 스트리밍 영상

실시간으로 화재가 발생했을 시, raspberry pi의 webcam이 영상을 머신러닝 서버로 보내 처리한다. 그 다음, OctoPrint를 통해 처리된 streaming 영상을 API로 application으로 전달한다. [그림5]는 application에서 보여지게 될 실시간 streaming 영상이다. 이와 같이 object detection이 이루어져 불이 난 곳을 표시해주고 있고, 처리된 영상 아래에는 현재 상태를 말해주는 문구를 넣어 사용자에게 정확한 정보를 전달해 줄 것이다. 위와 같은 형태의 UI로 app에 구현될 예정이다.

3.1.2. 화재 발생 알림

[그림6] 알림 화면

raspberry pi에서 모델을 통해 object detection이 되면, MQTT 프로토콜을 통해 모바일로 알림을 보내게 된다. 사용자는 영상 확인을 통해 어디서 불꽃이 인식되었는지를 확인할 수 있다.

3.1.3. 사용자 인터페이스(default GUI)

[그림7] 메인 화면

[그림8] 알림 화면

[그림9] 실시간 영상 확인

[그림7]은 본 연구에서 제공할 application의 메인 화면이다. 불을 인식하고 찾는다는 의미에서의 불 그림과 돋보기 모양, dr.fire이라는 이름을 통해 화재를 인식하는 application임을 나타냈다. 메인화면에서는 '실시간 영상 확인' 기능을 통해 사용자가 원할 때, 카메라가 설치된 장소의 streaming 영상을 볼 수 있다.

[그림8]는 실제 화재가 인식되었을 때, 나타나는 알림 표시로써 app으로 알림 API를 통해 화재가 인식되었다는 정보를 받았을 시, 영상을 확인하겠냐는 팝업 메세지를 띄워주어 사용자가 바로 상황을 확인할 수 있도록 도와주는 알림화면이다.

[그림9]은 [그림2]에서 '영상 확인' 탭을 눌렀을 때, 나타나는 실시간 영상 확인화면으로, 실제 화재가 발생했음을 나타내고 어디 화재가 나타났는지 object detection 처리된 영상을 통해 사용자에게 실시간으로 보여주게 된다.

3.2. 요구사항

3.2.1. 스트리밍 및 알림을 위한 API

앞서 말했던 MQTT에는 여러 broker가 있다. 이 broker를 분석한 그림은 다음과 같다.

SERVER	QoS0	QoS1	QoS2	bridge	SSL	Dynamic Topics
ActiveMQ	V	V	V	X	V	V
emitter	V	X	X	X	V	V
JoramMQ	V	V	V	V	V	V
Mosquitto	V	V	V	V	V	V
MQTT.js	V	V	V	X	V	V
RabbitMQ	V	V	X	X	V	V

모든 기능이 필요한 것은 아니지만, 기본적으로 서비스의 질을 보장해주는 레벨인 QoS의 모든 레벨을 지원해주는 broker 중에서 자주 사용되는 Mosquitto를 이용할 예정이다. MQTT의 paho,mqtt.client api와 mosquitto에서 제공해주는 test 사이트를 활용해 알림을 보낼 예정이다.

3.2.2. 스트리밍 및 알림을 위한 object detection 모델

Raspberry pi의 카메라 모듈을 통해 입력 받은 영상이 실시간으로 스트리밍 되기 이전에, 불꽃을 인식하는 object detection 모델을 거쳐야한다. 화재는 빠른 대응이 중요한 문제이므로 모델의 이미지 처리 속도를 50ms 내로 구현한다.

입력 이미지에 불꽃이 존재하여 알림 발생 시, 적절한 정보를 제공하기 위해 이미지의 불꽃 객체 위치에 bounding box가 그려진 이미지를 제공해야한다. 이를 위해 모델은 bounding box의 좌표들과 객체의 정확도를 최종 출력으로 한다.

3.2.3. 사용자 인터페이스

간결하고 시각적으로 보기 좋은 디자인을 선택해야 한다. 또한 화재와 관련된 application인 만큼, 붉은색 위젯 위주의 디자인으로 설계한다. 기본적인, 필수적인 디자인 외에는 불필요한 디자인을 추가하지 않는다.

스트리밍 영상을 보는데 지연이나 화질이 본래의 영상보다 최대한 나빠지지 않도록 재생 환경을 구축해야 한다. 또한 알림이 왔을 때, 알림 화면이 뜨는데 지연이 생기면 application의 목적이 사라지므로 이에 유의하여 프로그램을 설계해야 한다.

4. 향후 일정 및 역할 분담

Content	3월			4월				5월				6월	
	3w	4w	5w	1w	2w	3w	4w	1w	2w	3w	4w	1w	2w
팀구성													
아이디어 선정													
사전 조사													
중간 보고서 작성													
개발환경 구축													
프로그램 설계													
서버 개발													
데이터 수집													
데이터 학습													
프로그램 개발													
UI 구성													
테스트													
프로그램 보완 & 최종 완성													

Item	Assignee
MQTT 통신	김효준
어플리케이션 UI, Web Streaming	장준영
라즈베리파이 구축	김효준, 장준영
머신러닝 서버	이동찬
데이터셋 수집	김효준, 이동찬, 장준영
관련 논문 연구, 모델 학습	이동찬

5. 기대효과 및 결론

본 연구를 통해 raspberry pi와 같은 한정된 자원에서 구동될 수 있는 압축된 모델이 탑재된 화재 인식 프로그램을 개발할 수 있게 된다. 모델 압축을 통해서 어느정도 안정적이고 신뢰성있는 결과를 raspberry pi에서 얻을 수 있다면, 앞으로는 더 한정된 환경에서의 프로그램 개발이나, 같은 환경에서 더 뛰어난 모델의 구현도 생각해볼 수 있게 될 것이다. 또한 여러 PC를 사용하지 않고, raspberry pi 하나로 기능을 구현했으므로 집약적이고 간결하게 연구를 진행했다고 볼 수도 있을 것이다.

이렇게 안정적이고 신뢰성있는 영상을 사용자에게 실시간으로 전달이 가능하다면, 간단한 설치만으로 화재를 예방할 수 있는 유용한 application이 배포될 수 있다. CAM의 설치 및 사용 방법이 매우 간단하기 때문에 화재 방범용 CCTV와 같이 큰 규모단위 뿐 아니라, 가정집, 불이 잘 날 수 있는 여러 장소 등에서 사용이 가능할 것이다.

6. 참고문헌

Zhengxia Zou, Zhenwei Shi, Member, Yuhong Guo, and Jieping Ye, “Object Detection in 20 Years: A Survey”, arXiv preprint arXiv:1905.05055, May 2019.

Mingxing Tan Ruoming Pang Quoc V. Le, “EfficientDet: Scalable and Efficient Object Detection”, arXiv preprint arXiv:1911.09070, Nov 2019.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, “Distilling the Knowledge in a Neural Network”, arXiv preprint arXiv:1503.02531, Mar 2015.

Google. TensorFlow Lite. https://www.tensorflow.org/lite/performance/model_optimization

MQTT broker. https://github.com/mqtt/mqtt.github.io/wiki/server-support