Bringing intelligence to IoT Edge: Machine Learning based Smart City Image Classification using Microsoft Azure IoT and Custom Vision

Omer Ali¹, Mohamad Khairi Ishak¹

¹School of Electrical and Electronic Engineering Universiti Sains Malaysia (USM)
Nibong Tebal, 14300, Penang, Malaysia
khairishak@usm.my

Abstract. Object detection, identification and classification techniques have seen many variants and improvements over past two decades. Together with Internet of Things (IoT) devices, improved computational algorithms and cloud support, real-time classification with low-cost devices has already been achieved. This paper discusses the real-time object detection and classification using Microsoft Custom Vision multi-class Machine Learning (ML) model operating at the Edge of IoT network. This paper further examines the use of virtual dockers or containers at the IoT edge devices for better security and isolation by decoupling physical hardware as well that supports multiple applications and services on a single hardware. The experiments are performed using emulated and simulated IoT devices on Microsoft Azure IoT platform for real-time object classification using Custom Vision Machine Learning (ML) models run directly from the edge device. The experimental results are further discussed to validate the model accuracy and its implementation in a future Smart City surveillance environment.

1. Introduction

Object detection and classification has seen many practical applications from surveillance, pedestrian count, traffic pattern analysis, road signs detection for autonomous vehicles to real-time object detection and tracking for behavioral analysis(1). With recent advancements in technology and proliferation of Internet of Things (IoT), these techniques fueled the possibility of emerging application domains such as Human machine interaction, virtual reality and the usage in Smart City projects(2). In addition, more emphasis has been made on utilizing low-cost, general purpose devices such as desktop web cameras and standard surveillance cameras to perform these tasks. Currently, we have around 27 billion Internet of Things (IoT) devices deployed and the numbers are expected to reach up to 75 billion by year 2025(3) as shown in figure 1.

Recently, a host of enterprise IoT platforms offered cloud-based intelligence using their services such as Machine Learning (ML) and Neural Networks (NN) that operate in their powerful compute engines in the cloud and extend the functionality to IoT devices. This scheme enabled to utilize cloud-based efficient and advanced computational algorithms and services on the low-cost, low-power IoT devices, thus providing scalable and cost-effective future networks(4).

However, this network hierarchical model also faces its downsides, where the low-compute, network and storage IoT devices always rely on cloud-connectivity to perform tasks. Due to limited resources, the network and the overall system delays are not only significant but also vary which further makes it difficult to estimate network response time comprising of millions of devices with variable payload.
deployed in a mission critical domain(5) (6).

Another key factor that effects Machine Learning (ML) based models depends on input data streams where a pre-processed and filtered data will result in better response time and hit ratios. In an IoT environment, it is almost impossible to pre-process the data due to the compute and storage limitations. This raw input to ML models further reduces the performance as well increases the overall response time.

One of the most recent and prevailing concepts is based on Fog/Edge computing where the cloud functionality is extended to the IoT network edge, thus reducing network dependability and response times. It is also estimated that the future IoT Edge boundaries are going to be bigger than the cloud itself(7). In this paper, the same concept is utilized to bring the cloud intelligence to the edge of the network, thus making it easier for IoT devices to inference upon the network intelligence close to the device level which will reduce latencies. It is also interesting to know that by running the cloud-based ML models on the edge devices will further reduce the dependability and overall system response times.

It is interesting to introduce an intelligent edge device as a transparent gateway that enables secure and scalable network device attachment in a cost-effective manner. The cloud network is used for firmware updates, ML models and application updates that are pushed directly from the cloud to the edge gateways. The edge devices can further aggregate, process and transport the data to the cloud for further analysis and storage, thus providing the real-time face of network more resource friendly and responsive(6) (8).

In this paper, the experiments are designed to simulate a Smart City environment where the surveillance cameras can detect and identify objects in real-time using Custom Vision Machine Learning (ML) model running on IoT edge devices hosted at Microsoft Azure platform. The images are fed to IoT edge device using emulated and simulated IoT devices and the classification results can then be analyzed to distinguish rescue vehicles such as Ambulance, Fire-brigade and police from urban traffic patterns in real-time. Though the experiment involves building and deployment of the image classification model to the IoT edge; the major focus is to study the edge device provisioning that supports dockers for hardware isolation, ease of device provisioning, building and pushing of ML models to the edge device with the provision of cloud-based updates.

In Section II, a brief insight into these technologies is presented which covers techniques from simple object detection and tracking to IoT based virtualized real-time object classification. This will help readers to understand the scope of advancement in this field and the position this paper is placed at. In Section III, the complete network hierarchy is presented together with the requirements and methodology to build the experiment environment. This section also briefly covers the key components required to
build IoT edge networks on Microsoft Azure IoT platform.

Section IV discusses the results of classifier models in detail. Model accuracy, ability of incremental updates, ease of input to the model is discussed. Finally, in Section V the results and directions are linked to conclude the research by providing future applications and scenarios where such systems can be deployed.

2. Related Work

Object detection, identification, tracking and classification and some of the very promising fields in Image Processing. Currently with the power of Machine Learning (ML), Neural Networks (NN) and Deep Learning (DL), these techniques can be deployed over scalable IoT networks to envision future Smart Cities capable of having real-time analytics on data captured from IoT devices.

**Table1**: Object Detection Algorithms & Models.

| Article | Scope          | Technique/Outcomes                                                                                                                                 |
|---------|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| (9)     | Tracking       | Edge strength for iris detection                                                                                                                   |
|         |                | Integration of eye vector with head movement                                                                                                       |
| (10)    | Detection      | Multi-resolution model that detects local deformations of the parts for improved accuracy                                                        |
|         |                | Reduced false alarms and computational cost                                                                                                         |
| (11)    | Detection      | GPU based texture mapping for background and foreground for improved detection of moving objects in real-time                                    |
| (12)    | Detection      | Background separation technique for real-time vehicle detection and counting                                                                      |
|         |                | Improved detection with accuracy in dense traffic patterns                                                                                         |
| (13)    | Classification | Zero-data learning approach by classifying objects based on semantic attributes                                                                   |
|         |                | Generation of new-sub classes without the need of re-training the model                                                                         |
| (14)    | Segmentation   | Clustering based segmentation technique to isolate area of interest for reduced communication overhead                                                |
|         | Detection      | Algorithm suitable for embedded and low-compute power devices                                                                                        |
|         |                | Focuses on distributive deployment of smart cameras for object detection and recognition                                                          |
|         | Recognition    | Formulation of real-world applications and use of geometrical constraints for modeling                                                              |
| (16)    | Detection      | Stationary image processing to detect vehicles to avoid congestion                                                                               |
|         |                | Algorithms suitable for embedded smart camera platforms                                                                                            |
| (17)    | Detection      | Time sequenced foregrounds are detected and wirelessly shared with distributed camera networks to build a semantic high-level event detection      |
|         | Tracking       | Object tracking is based on detecting consistent composite events on each camera                                                                  |
| (18)    | Detection      | Background and foreground information is separately modeled by color information                                                                  |
|         |                | Lightweight algorithm with post-spatial filtering allows point-of-interest analysis                                                                 |
|         |                | Edge Enabled IoT edge enabled virtualized container deployment of a camera system supporting dynamic configuration                                   |
| (19)    | Detection      | Image filtering and processing using low-cost component model connected over IoT network                                                            |
| (20)    | Detection      | Image filtering and processing using low-cost component model connected over IoT network                                                            |
|         |                | IoT enabled                                                                                                                                                                                                   |
Detection Automated object detection algorithm for IoT enabled Smart Cities

Algorithm utilizes high frequency image peaks to isolate point of interests

Table 1 listed some of these technological improvements that paved the way towards deploying such objection detection and classification techniques in an IoT edge-based environments.

In above discussions algorithms and cloud-based models presented a unique contrast where object detection, identification, tracking and classification varies from static to real-time, offline to real-time dense urban environment. Some of the algorithms focus on high accuracy detection by utilizing CPU/GPU power, whereas other techniques focused mostly on lightweight object detection algorithms suitable for embedded platforms.

In the next section, we present our scheme focusing on a lightweight object classification model deployed in a virtualized container deployed on Microsoft Azure IoT edge device.

3. Methodology

In order to have a lightweight Machine Learning (ML) object classification model running on the IoT network Edge, we looked into available platforms and resources that supports ease of deployment, scalability and the ability to virtualize hardware as well components to provide flexibility in running multiple models and applications on the same device. Microsoft Azure IoT portal was chosen as the target platform which supports the initial experiment setup requirements. The experiment schematic in figure 2 further explains the workflow, pre-requisites, service setup and configurations needed for this experiment.

![Figure2: Schematic workflow for ML model deployment on IoT Edge.](image)

Microsoft Azure IoT portal natively supports virtualized containers that can hold firmware to applications, thus allowing for better hardware and software isolation, flexibility as well as eliminates hardware dependency for both edge as well as associated IoT devices. The workflow required three major steps, including:

- Image classification model on Microsoft powered Custom Vision
- Azure IoT hub, container registry and associated Edge Virtual Machines that support docker containers
- Deployment application to be pushed to Edge gateway as well as IoT devices

3.1. Microsoft Custom Vision

Microsoft Custom Vision is Artificial Intelligence (AI) powered image classifier service running on its Azure platform. Custom Vision helps rapid building and prototyping of image classifiers using its...
machine learning algorithms that help to label image sets and train the model. Another distinguishing feature of Custom Vision is the ability to work with a smaller data set and at the same time optimizing the model. In this experiment, a multi-class image label model was trained and optimized to identify the emergency and rescue vehicles. Figure 3 presents an object detection test on our trained model in Custom Vision.

![Image Classification Test Results on an Image using Custom Vision.](image)

Another important consideration for selecting Microsoft Custom Vision image classifiers was the ability to export the trained model to Microsoft Azure and other platforms where it can run on virtualized containers.

3.2 Microsoft Azure IoT Platform

Azure IoT hub is the Microsoft’s enterprise IoT solution offered as Platform as a Service (Paas) with support for open source models and SDKs. One of the key distinguishing features of this enterprise platform is the ability to provision billions of devices securely with its easy deployment service. This managed service can be further integrated with current enterprise IoT solutions using IoT Central Software-as-a-Service (SaaS) solutions.

Azure IoT edge is a fully managed service that allows the integration and deployment of Artificial Intelligence (AI), Machine Learning (ML) and custom models to run on its docker containers which can be accessed and programmed through cloud. Having the flexibility of independent instances of applications running on the Edge, the platform provides negligible latencies from device to the edge gateway.

Every IoT resource in Microsoft Azure Portal is mapped to its central IoT Hub which can be thought of as centralized repository. In order to deploy edge services, the proposed virtual machines must be configured and linked to the Container Registry, which keeps a record of all the connected devices and gateway along with the security keys as well as the access management interface.

Like other cloud platforms, Azure provides flexible templates to run or import virtual machines capable to run every operating system with hardware, software as well as network virtualization features support. However, the ability to have native docker support on its virtual machine enables the seamless integration of open source model packaged into virtualized containers which can be run in parallel and completely independent. This feature not only provides increased security as hardware and applications are abstracted natively but at the same time allows to run multiple applications and services in a containerized environment that can be fully managed independently.

3.3 Building and Pushing Edge Solution

With Custom Vision Model trained and exported and Azure IoT Edge devices configured, the Edge deployment solution is then pushed which can run on Edge VM.
Microsoft Visual Studio Code, together with Azure toolkit, python and Docker container were used to import the image classification model to deploy the Edge solution. The design and operational schematic where modules push test images to the classified model on Edge devices is depicted in figure 4.

![Diagram](image.png)

**Figure 4:** Real-time operation of model on IoT Edge.

The VS code environments makes it easier to connect to Azure IoT hub and access all the available interfaces with devices attached to them, at the same time allowing modules to be programmed. In this experiment, the extracted image classifier model was fine-tuned in VS code. The modules are then programmed, and a deployment manifest was generated to push the edge solution to the devices in the cloud.

VS code deployment manifesto can then be used to adjust the parameters based on the end operating system on Edge gateways and IoT devices and the custom solution can be built and pushed to millions of devices instantly. This is also very helpful for debugging, optimizing the code offline or in the cloud which can then be pushed to the Edge devices. In our experiment, the Image classifier model was built for a Linux based amd64 distribution running on Edge device. The emulated hardware as well as simulated hardware could then send the images directly to the Edge VM using its public IP address for analysis and classification on the edge.

4. Results

Once the deployed solution was successfully pushed to the Edge gateway, a series of heart-beat tests were conducted before initiating the image classification experiments. These heart-beat tests are required to check the health and status of all the interfaces and connected devices and serve as a good starting point for further analysis and system diagnostics. The heart-beat tests included:

- Device Interface status
- Cloud to Device (C2D) messages
- Device to Cloud (D2C) messages
- Containerized module invocation

Two test case scenarios were established to test the newly deployed image classification model on the IoT Edge devices, including:
1) The use of Simulated device with pre-packaged test image as an input to classifier on Edge VM
2) The use of Raspberry Pi with camera module as an emulated device, with a rescue vehicle picture captured and fed to classifier on Edge VM

In both the test environments, the image classifier performed well with accuracy over 90%. However, it is fundamentally important to mention here that this paper focused on the deployment of containerized solution on IoT edge other than the model performance itself. However, the model can be further optimized by providing a larger data set and optimizing the associated python coding. Figure 5 (a) and (b) presents the activity stats for the experimental run on Edge VM.

![Figure 5: (a): Disk Read/Write activity on Edge VM (b): Network activity on Edge VM.](image)

As shown in the above figures, the Edge VM activity response was observed during the experimental run where the emulated as well as the simulated devices pushed the test images to the image classifier models running on Edge VM in Azure IoT Hub. The accuracy and performance metrics of the model was observed and one such records from the runs is presented in figure 6.

![Figure 6: Image Classifier Model Performance Metrics.](image)

For all the inputs, images of rescue vehicles such as ambulances and fire-brigades were fed to the classifier model. The image sets were carefully chosen where in one iteration urban dense traffic images were fed to the model whereas the other iterations involved isolated rescue vehicles images. The model predicted well with an average precision of above 90% whereas average precision remained on higher orders at or above 95% in all the cases.

It is however very important to consider that the Azure IoT hub configuration, device initialization and solution deployment using Visual Studio code is a cumbersome process and requires a detailed understanding of the development environment as well as familiarization with Microsoft Azure Portal.
The configuration of IoT edge resources on the Azure IoT portal is a step-by-step process which involves:

1. IoT Edge resource group creation
2. Initialization of Edge Virtual Machine
3. Request a permanent IP Address
4. IoT device creation in IoT Hub
5. Configuration of IoT Edge Runtime environment on Edge Devices

The added complexity may add up to the one-time deployment delays but ensures system level security, which is required to provision millions of devices. IoT Edge Resource group creation provides a further extension by which IoT devices and assets can be easily provisioned at large scale and can be monitored under respective resource groups.

The image classifier model once exported can be directly optimized within the development environment, during real-time operations with larger datasets or can be optimized on Custom Vision platform and later imported to docker containers.

5. Future Directions and Conclusion
In this paper, we envisioned a Smart City surveillance scenario where IoT devices such as surveillance cameras can be made intelligent by provisioning them with Edge based machine learning models. Introducing Edge based computing helps to reduce overall system latencies and improved response times. It is also noticeable that the by reducing the cloud dependability and bringing the intelligence on edge, the network scalability is further enhanced allowing more IoT devices to connect to a transparent Edge gateway that is natively scalable, supports hardware and network virtualization and container based multi-application models for underlying networks.

An IoT Edge scalable device can further help to segment the underlying networks based on application requirements at the same time making it possible to be centrally monitored and administered, which is going to build bigger scalable edge-enable IoT networks.

In this study, it was also observed that a native virtualized container environment helps to isolate hardware, system firmware, application modules and attached interfaces thereby providing fine-grained control over attached devices, their interfaces and processes. The ability to create, optimize and push the machine learning models to the IoT edge provides further opportunity and flexibility to control over the air update, while at the same time performing offline processing at edge nodes. This will improve the system response time and reduce the cloud-based dependency, which is greatly needed in future IoT applications where a great emphasis is given on improving overall system response time.

Another important characteristic of running virtualized containers on IoT edge provides the ability to run multiple virtualized models from different vendors, which can coherently work together to improve the overall system efficiency. Previously, cloud-based virtualization allowed for application-specific as well as resource-specific deployments by providing an interface to end devices. The ability to utilize the same approach and to be able to run it on edge devices will prove to be a game changer in future mission critical applications.

In the near future, our Smart cities will be guarded by low-cost distributed surveillance equipment that can detect, identify traffic patterns, count vehicles and pedestrians and classify rescue vehicles in real-time for better traffic control and congestion avoidance. With the ability to detect rescue vehicles, machine learning algorithms can automate traffic route, send alert notifications to connected smart vehicles and activate emergency alerts at hospitals in the vicinity.

This paper studied one such application scenario where we could see the potential of cloud and edge-enabled future IoT networks in action.

Acknowledgment
The authors would like to thank Universiti Sains Malaysia, for providing the financial support under Research Grant RUI (1001/PELECT/8014049) to carry out this research.

References
[1] Verschae R, Ruiz-del-Solar J. Object Detection: Current and Future Directions. Frontiers in Robotics and AI. 2015;2:29.

[2] Le J. The 5 Trends Dominating Computer Vision HeartBeat Website: Medium; 2019 [cited 2019]. Available from: https://heartbeat.fritz.ai/the-5-trends-that-dominated-computer-vision-in-2018- de38fb9bd86.

[3] Research S. Internet of Things (IoT) connected devices installed base worldwide from 2015 to 2025 (in billions). Statista Website: Statista; 2016 27 Nov 2016.

[4] Jeong H, Lee H, Moon S, editors. Work-in-progress: cloud-based machine learning for IoT devices with better privacy. 2017 International Conference on Embedded Software (EMSOFT); 2017 15-20 Oct. 2017.

[5] Kadarla K, Sharma SC, Bhardwaj T, Chaudhary A, editors. A Simulation Study of Response Times in Cloud Environment for IoT-Based Healthcare Workloads. 2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS); 2017 22-25 Oct. 2017.

[6] Pan J, McElhannon J. Future Edge Cloud and Edge Computing for Internet of Things Applications. IEEE Internet of Things Journal. 2018;5(1):439-49.

[7] Farrel N. The Edge will be bigger than the cloud Fudzilla: Fudzilla; 2019 [Available from: https://www.fudzilla.com/news/49455-the-edge-will-be-bigger-than-the-cloud.

[8] El-Sayed H, Sankar S, Prasad M, Puthal D, Gupta A, Mohanty M, et al. Edge of Things: The Big Picture on the Integration of Edge, IoT and the Cloud in a Distributed Computing Environment. IEEE Access. 2018;6:1706-17.

[9] Cheung Y-m, Peng Q. Eye Gaze Tracking With a Web Camera in a Desktop Environment. IEEE Transactions on Human-Machine Systems. 2015;45(4):419-30.

[10] Pedersoli M, Gonzalez J, Hu X, Roca X. Toward Real-Time Pedestrian Detection Based on a Deformable Template Model. IEEE Transactions on Intelligent Transportation Systems. 2014;15(1):355-64.

[11] Berjon D, Cuevas C, Moran F, Garcia N. GPU-based implementation of an optimized nonparametric background modeling for real-time moving object detection. IEEE Transactions on Consumer Electronics. 2013;59(2):361-9.

[12] Yang H, Qu S. Real-time vehicle detection and counting in complex traffic scenes using background subtraction model with low-rank decomposition. IET Intelligent Transport Systems. 2018;12(1):75-85.

[13] Lampert CH, Nickisch H, Harmeling S. Attribute-based classification for zero-shot visual object categorization. IEEE Trans Pattern Anal Mach Intell. 2014;36(3):453-65.

[14] Singh S, Dunga SM, Mandal AS, Shekhar C, Vohra A, editors. Real-Time Object Segmentation in Smart Camera for Remote Surveillance Scenario. 2010 International Conference on Advances in Computer Engineering; 2010 20-21 June 2010.

[15] Sankaranarayanan AC, Veeraraghavan A, Chellappa R. Object Detection, Tracking and Recognition for Multiple Smart Cameras. Proceedings of the IEEE. 2008;96(10):1606-24.

[16] Alpatov BA, Babayan PV, Ershov MD, editors. Vehicle detection and counting system for real-time traffic surveillance. 2018 7th Mediterranean Conference on Embedded Computing (MECO); 2018 10-14 June 2018.

[17] Wang Y, Velipasalar S, Casares M. Cooperative object tracking and composite event detection with wireless embedded smart cameras. IEEE Trans Image Process. 2010;19(10):2614-33.

[18] Cuevas C, García N, editors. Moving object detection for real-time high-quality lightweight applications on smart cameras. 2011 IEEE International Conference on Consumer Electronics (ICCE); 2011 9-12 Jan. 2011.

[19] Jang SY, Lee Y, Shin B, Lee D, editors. Application-Aware IoT Camera Virtualization for Video Analytics Edge Computing. 2018 IEEE/ACM Symposium on Edge Computing (SEC); 2018 25-27 Oct. 2018.

[20] Rupani A, Whig P, Sujediya G, Vyas P, editors. A robust technique for image processing
based on interfacing of Raspberry-Pi and FPGA using IoT. 2017 International Conference on Computer, Communications and Electronics (Comptelix); 2017 1-2 July 2017.

[21] Hu L, Ni Q. IoT-Driven Automated Object Detection Algorithm for Urban Surveillance Systems in Smart Cities. IEEE Internet of Things Journal. 2018;5(2):747-54.