TEPIAN Vol. 5 No. 2 (June 2024) 50-57 p-ISSN 2721-5350 e-ISSN 2721-5369

https://doi.org/10.51967/tepian.v5i2.2994 © 2024 Tepian Politeknik Pertanian Negeri Samarinda

Cloud Storage for Object Detection using ESP32-CAM

Imron *
Software Engineering Technology, Polytechnic Agricultural of Samarinda, Samarinda, 75242, Indonesia
imron@politanisamarinda.ac.id
**Corresponding Author*

Bagus Satria
Software Engineering Technology, Polytechnic Agricultural of Samarinda, Samarinda, 75242, Indonesia
bagussatria@politanisamarinda.ac.id

Syafei Karim
Accounting Information System, Polytechnic Agricultural of Samarinda, Samarinda, 75242, Indonesia
syfei.karim@gmail.com

Fajar Ramadhani
Accounting Information System, Polytechnic Agricultural of Samarinda, Samarinda, 75242, Indonesia
fajar.ramadhani@politanisamarinda.ac.id

Submitted: 2024-05-14; Accepted: 2024-06-10; Published: 2024-06-10

***Abstract*—** Cloud storage services can create an object storage bucket to store our pictures, among them the Cloud Storage FUSE, Scaleway, S3 bucket, Firebase, etc. intelligent IoT systems generate vast amounts of multi-source industrial data, which necessitate a large amount of storage and processing power to enable real-time data processing and analysis. Cloud computing can be intricately linked into intelligent IIoT systems due to its strong computational and storage capabilities. Cloud Storage for Object Detection using ESP32-CAM. Create a workable solution that supports distributed storage bucket and implement it in a real-world setting. Implement the entire system as an addition to the well-known IoT cloud storage and run multiple experiments to evaluate its functionality in scenarios with varying setups and system. The target objects that are used as data sets are the ESP8266, Wemos D1, and Arduino Uno. Figuring out the ideal parameters for training the FOMO (First Object, More Object) model and then putting it into practice. It was necessary to find a balance between learning rate and accuracy, on the other hand, to maintain the highest possible accuracy in the identification of the microcontroller object to minimize the number of false positive reports. Find the value learning rate effective to this object is 0.01 with F1 score 98.7% and accuracy score 89.58%.

***Keywords*—** Cloud Storage, ESP32CAM, Object Detection, FOMO.

I. Introduction

Storage for data that is either kept for short periods of Storage for data that is either kept for short periods of time or is regularly accessed ("hot" data). For machine learning applications, cloud storage is a popular option for storing training data, models, and checkpoints in cloud storage buckets. Benefits from cloud storage's scale include low cost, high throughput, and ease of use. Many cloud storage services can create an object storage bucket to store our pictures, among them the Cloud Storage FUSE, Scaleway, S3 bucket, Firebase, etc. All the while keeping apps compatible with those that need or use filesystem semantics. Additionally, storage buckets now include caching, which delivers a training throughput that is 2.9 times greater and a time to train that is up to 2.2 times faster than native ML framework data loaders. Since scala producing more data than ever before and losing it more frequently due to computer hard drive crashes, misplacing it, erasing it by accident, and other mishaps, cloud backups have become a crucial service. The application of the Internet of Things (IoT) in the industrial field is known as Industry 4.0, or the Industrial Internet of Things (IIoT) (Bagchi et al., 2022).

It has a significant impact on the advancement of manufacturing, energy, transportation, and health care and is a key component of the intelligent era. To enable the monitoring of contemporary industrial departments, the intelligent Industrial Internet of Things system of today integrates several kinds of terminal sensor equipment and virtual information systems via wireless communication networks. Because of this, intelligent IoT systems generate vast amounts of multi-source industrial data, which necessitate a large amount of storage and processing power to enable real-time data processing and analysis. Cloud computing can be intricately linked into intelligent IIoT systems due to its strong computational and storage capabilities. The industrial data kept on cloud servers helps to guarantee the continuous monitoring of the state of industrial processes. Additionally, in order to advance industrial projects, some research institutes could receive outsourced industrial data so they can perform in-depth data analysis with the aid of cloud-based industrial monitoring systems.

The objective this paper discusses Cloud Storage for Object Detection using ESP32-CAM. Create a workable solution that supports distributed storage bucket and implement it in a real-world setting. Implement the entire system as an addition to the well-known IoT cloud storage and run multiple experiments to evaluate its functionality in scenarios with varying setups and system.

This paper is organized into five sections as follows: Section I explains the background of the proposed method. Section II introduces the state of the art. Section III condition of problems formulation and explains the details of the problem. Section III shows the analysis and the result of simulation for planning robot people follower. Section IV concludes the final paper.

II. Literature review

The Internet of Things has made it possible to connect a variety of sensors and actuators to a production site's network, allowing for the collection and analysis of a vast amount of data for the purposes of manufacturing collaboration, operation monitoring, predictive maintenance, and other industry trends. Current IoT communication architectures are typically built on top of cloud platforms, which serve as middlemen and link IoT devices together. This approach encourages heterogeneous device disparities between application developers and device manufacturers due to the platform's complexity, but standard protocols are needed for these devices to communicate with one another (Alejandro et al., 2023; Verma et al., 2023).

One of the most important and fundamental tasks in computer vision problems is object detection. For computation on embedded and/or edge-IoT devices, real-time object detection applications need excellent accuracy at low power consumption. Conventional methods for object detection that rely on region-based and sliding window techniques have a large computing overhead and poor accuracy (Hazarika et al., 2022). Numerous academics have looked into text recognition and picture processing, as well as the best methods for extracting text only and improving system accuracy. Training and testing are the two processes required to create machine learning models. As a result, supervised learning (labelled), unsupervised learning (unlabeled), and semi-supervised learning (partially labelled) are the three distinct strategies that are defined based on whether the available data are labelled.

Based on the ESP32 chip, the ESP32-S module is a feature of the ESP32-CAM. The ESP32 is a popular and potent microcontroller featuring a dual-core processor, lots of RAM, many I/O interfaces, and Bluetooth and Wi-Fi connection. This makes it possible for the ESP32-CAM to effectively manage networking duties in addition to camera operations. A versatile development board, the ESP32-CAM combines an ESP32-S module with a camera module that records both photos and videos. Because it is made especially for projects requiring Wi-Fi and camera capabilities, it can be used for a variety of applications, including image recognition projects, surveillance systems, and Internet of Things (IoT) devices (Kaur et al., 2021).

This section will showcase projects that have employed cameras and cloud storage. Two serverless data pipeline strategies created using Apache NiFi and Message Queuing Telemetry Transport (MQTT). Tested our suggested methods using the image streaming data, doing object detection in the pictures (Mirampalli et al., 2024). Other hand Create a novel model that tracks plant growth in agricultural fields using the OV2640 camera module, ESP32-CAM, a solar panel, and a mobile application (Elhattab et al., 2023). To accomplish high accuracy license plate identification, design uses three image processing stages: pre-processing, segmentation, and character recognition(Abdellatif et al., 2023). MQTT full solution for a MQTT broker system that is distributed(Akshatha & Dilip Kumar, 2023; X. Liu et al., 2020).

This method is based on the "first object, more object" (FOMO) real-time object detection technique, which has been modified for the energy-efficient ESP32 microcontroller architecture (Novak et al., 2024). It is suggested to use privacy-preserving dynamic auditing for code-based storage regeneration in cloud fog-assisted IIoT(D. Liu et al., 2024). architecture of a storage auditing scheme that divides its computation and storage functionalities(Chen et al., 2024). The kind of data needed to train the model can be used to categorize anomaly detection techniques based on machine learning techniques. TinyML can applied to the detection(Hammad et al., 2023). Topic-based routing scheme for MQTT distributed broker. Builds several overlay networks within the distributed system, each connecting brokers whose associated clients share a common interest in certain themes (Longo & Redondi, 2023). Authentication mechanisms for IoT distributed MQTT has been reviewed (Kurdi & Thayananthan, 2021). The objective this paper discusses Cloud Storage for Object Detection using ESP32-CAM. Create a workable solution that supports distributed storage bucket and implement it in a real-world setting. Implement the entire system as an addition to the well-known IoT cloud storage and run multiple experiments to evaluate its functionality in scenarios with varying setups and system.

III. Research Methods

The primary objective of this work is to develop a workable solution that can be applied in a real-world scenario to enable distributed cloud storage. While the approach described in this paper can be used with any real time storage firebase implementation, in this case we start with the widely used firebase storage bucket and adapt it to accommodate a distributed scenario. This decision was driven by firebase extremely straightforward administration, which provides a free and open source for changing behavior through plugins. The plugin gives developers access to a number of callbacks that are readily connected to user-defined logic. The ESP32-CAM is a key component of our model. It serves as the brains of the system, allowing vital information on object detection to be collected and transmitted in real time. Its Wi-Fi connectivity allows for remote monitoring and guarantees smooth communication between different components.

OV2640 camera is an essential component for tracking plant development. Precise details in crop evolution may be captured because to its exceptional image quality and high resolution. It is especially helpful for closely monitoring a plant's growth and for identifying health problems early on. Determining the technical specifications needed to construct an all-inclusive system. This objective covers both software and hardware. The ideal hardware configuration must be determined in order to successfully apply the neural network model, taking into account the functional constraints imposed on the system's actual use. Figuring out the ideal parameters for training the FOMO (First Object, More Object) model and then putting it into practice. figuring out the best parameter combination to use this model with in devices with lesser performance. gives a thorough overview of the technical aspects of integrating the project's hardware and software components. Using sample data, the trained FOMO neural network model's quality is confirmed (Hammad et al., 2023; Novak et al., 2024).

A key component of the cloud computing infrastructure for storing data in the form of digital objects is object storage buckets. Buckets are containers that let users store and arrange data in an organized way when using cloud storage services like Microsoft Azure Blob Storage, Google Cloud Storage, and Amazon S3. Every bucket has its own name and distinct URL for access. The key benefits of employing object storage buckets are their cost-effectiveness, high durability, ease of scalability, intuitive management, and ability to integrate with other cloud services. For this reason, object storage buckets are essential to offering a safe, dependable, and effective storage infrastructure for a variety of workloads and applications.

A. Data set/Data labelling

A dataset is an entity or example-based collection of data used for analysis, machine learning, or testing algorithms. Datasets can contain a variety of data types, such as text, pictures, audio, or more numerical data. The image data collection that was gathered under these circumstances contains requirements for three different kinds of objects. The target objects that are used as data sets are the ESP8266, Wemos D1, and Arduino Uno. The process of assigning a label or annotation to every entity in the dataset is also known as data labeling. These labels offer details on each entity's traits or class, which are then utilized for further analysis needs or to train machine learning algorithms. Depending on the type of data and task difficulty, labeling can be done automatically by certain algorithms or manually by humans. In Figure 1 is a data set of images that have been labeled for each object. Figure 2 show the characteristic data set divide into three classes.

When exploring a dataset, the first step is to look at basic summary statistics to get a general idea of the data. This helps understand the distribution of values, range of variability, and extreme values that may need to be considered. After that, create a graphical visualization of the data using a bar chart, histogram, or scatter diagram. This helps in identifying patterns, trends, or correlations that may exist between variables. Next, check for the presence of missing values and make a decision about how best to handle them.

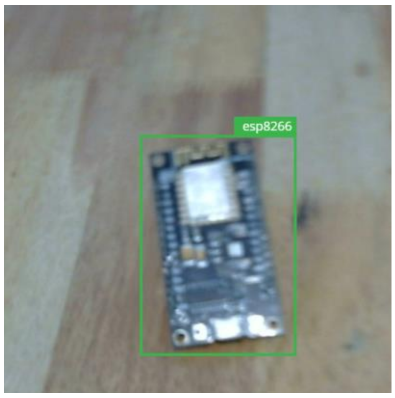


Figure.1 Data Set Labelling

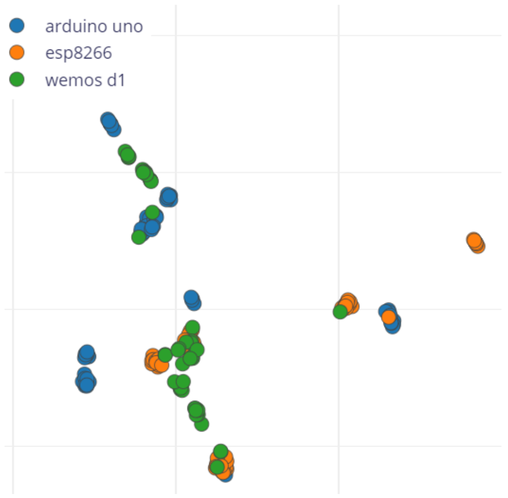


Figure.2 Visualization Data Set Classes

Then, it can look for outliers or unusual data that might influence the analysis. After that, explore the distribution of key variables in the dataset to ensure a solid understanding of the data. Correlation between variables to discover interesting relationships or hidden patterns. Figure 3 shows a visualization of the class data set. Third categories are divided into classes and exploratory data is visualized.

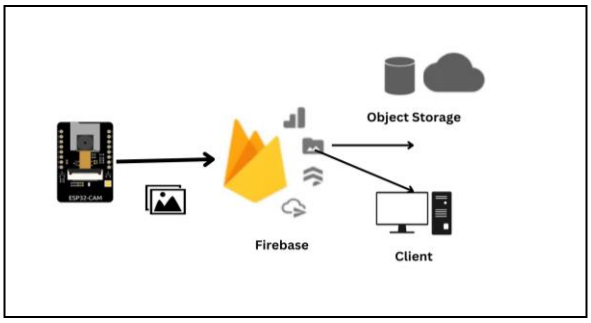


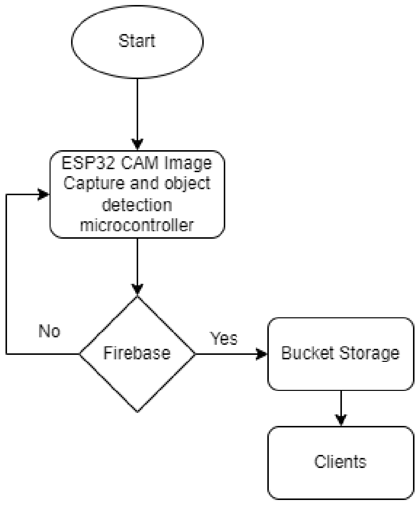
Figure.3 Object Storage bucket

Finally, it divides the dataset into subsets based on certain criteria for further analysis, such as segmentation by specific categories or value ranges. With careful exploration of the dataset, one can move to the next stage in data analysis with greater confidence.

B. Principle of operation

The process of taking pictures is carried out in several stages. The process is carried out by following the cloud storage procedural method of working. ESP32-CAM makes the object recognition part in this condition carried out in the object-type of microcontroller. The object retrieval test was chosen with a level of similarity that is not much different between Arduino Uno and Wemos D1. Meanwhile, another object was created with a different shape contrast, namely esp8266. The similar shape between Arduino Uno and Wemos D1 will be tested by looking at the accuracy of the differences in images captured by the ESP32-CAM camera. Then, it was tested with objects of different shapes and sizes, namely esp8266, which can be seen from the margin of error of significant differences.

Procedure Establishing the firebase setting up the devices. Create first Hub and with choose the free shared plan for system open and free charge. Add two distinct devices: the firebase explorer, a software that will be used to activate the camera and view the payloads passing over cloud, and the camera itself, which is ESP32 cam hardware. Allow unsecure connections so can easily connect our devices to our cloud. The IoT cloud storage allow data to adding file into destinations. In this project use the Object Storage Route. Object storage stores object images taken based on the object detection used. Figure 4 show proposed model for system.


Start
ESP32 CAM Image Capture and object detection microcontroller
Firebase
No
Yes
Bucket Storage
Clients
Figure 4. Proposed Model

C. Firebase

The tools provided by Firebase allow you to "build, improve, and grow your app." Developers would typically have to construct many of these services themselves because they would like to concentrate on creating the app experience itself. Analytics, authentication, databases, configuration, file storage, push messaging, and a long number of other items are all included in this. The services scale with little to no effort from the developer because they are hosted in the cloud. Massively scalable file storage is offered by cloud storage. Furthermore, it's not a Firebase offering; rather, it's a Google Cloud product. Using client SDKs that you can integrate into your app, Cloud Storage for Firebase allows you to upload and download files straight to and from your Cloud Storage "bucket" (Li et al., 2024; Q. Liu et al., 2023).

People can submit unique avatars to Greta's games using Cloud Storage, and they can share images with each other on Shawn's social network. Since Cloud Storage can accommodate exabytes of data, neither of them has to worry about running out of room. When utilize security rules (for Realtime Database, Fire store, and Cloud Storage) to apply access control data at the source, authentication functions incredibly well with these three technologies. This helps to prevent unfortunate situations like the one with the lolrus above by guaranteeing that clients can access that data only in the manner permit.

D. Alghorithm

In this research, a low-cost smart system that can shorten the time spent looking for an object detection microcontroller is proposed. Once the system can detect the object, save the image into the object storage bucket hub. The demonstrates the suggested system model, which is made up of the image processing and capture elements. Pre-processing and object recognition are the two stages of the image processing technique, whilst the image capturing block is the initial input image that is obtained through the use of ESP32 cameras. Any image processing technique's effectiveness depends on how well the acquired images are adjusted and presented in a format that allows for future stages to operate efficiently. This occurs during the pre-processing phase, when the taken images perform a number of operations, including scaling, rotation, and resizing. After that, the altered photos are transformed from their original RGB format to grayscale. In order to process the photos more quickly and robustly in the subsequent steps, this step is required. Then, to create a smoother image to work with, the grayscale image is interpolated using the nearest neighbor method.

First go over the depthwise separable filters that form the foundation of MobileNet. After that, we go over the MobileNet network architecture before summarizing the two model shrinkage hyperparameters. multiplier for both width and resolution (Howard et al., 2017; Sandler et al., 2018). In order to create a new representation, the basic convolution operation has the effect of merging features and filtering them using the convolutional kernels. For a significant reduction in computing cost, the filtering and combination procedures can be divided into two parts by using factorized convolutions known as depthwise separable convolutions. Pointwise and depthwise convolutions are the two layers that make up a depthwise separable convolution. Both batchnorm and ReLU nonlinearities are used by MobileNets for both layers.

One filter per input channel (input depth) for depthwise convolution can be expressed from equation (1) as follows:

$$\hat{G}_{k,l,m} = \sum_{i,j} \hat{K}_{i,j,m} \cdot F_{k+i-1,\ l+j-1,\ m} \quad (1)$$

where $\hat{K}$ is the depthwise convolutional kernel of size $D_k x\ D_k\ x\ M$ where the mth filter in $\hat{K}$ is applied to the mth channel in F to produce the mth channel of the filtered output feature map G.

The computational cost of depthwise convolution from equation (2) is:

$$D_k x\ D_k\ x\ M\ x\ D_F\ x\ D_F \quad (2)$$

Comparing depthwise convolution to ordinary convolution, it is incredibly efficient. It does not, however, combine input channels to produce new features—rather, it only filters them. Therefore, to produce these new features, an extra layer that computes a linear combination of the output of depthwise convolution via 1 × 1 convolution is required.

Depthwise separable convolution, first presented in equation 3, is a combination of depthwise convolution and 1 × 1 (pointwise) convolution. which is the sum of the depthwise and 1 × 1 pointwise convolutions from equation 3

$$D_k x\ D_k\ x\ M\ x\ D_F\ x\ D_F + M\ x\ N\ D_F\ x\ D_F \quad (3)$$

By expressing convolution as a two step process of filtering and combining we get a reduction in computation from equation 4 of:

$$\frac{D_k x\ D_k\ x\ M\ x\ D_F\ x\ D_F + M\ x\ N\ D_F\ x\ D_F}{D_k x\ D_k\ x\ M\ x\ D_F\ x\ D_F} \quad (4)$$

$$= \frac{1}{N} + \frac{1}{D_K^2}$$

IV Results and Discussion

Assuming 20% of the test set's data, baseline testing was done to confirm that recognition can be implemented in ESP32-Cam for both data sets on 60 epochs. The table, Table 1, displays the results. The table shows that in a 229-frame data set with a learning rate of 0.01, the highest F1 score is 98.7%. This would mean that with the set parameters, this would be the best training and, therefore, the best model. In the data set with learning rate 0.001, the best score is 85%. However, these values are valid for the training data set. Accuracy can be misleading when used with unbalanced data sets, and therefore, there are other metrics based on the ambiguity matrix that can be useful for performance evaluation. A quick look at the quality of the result is shown by the substitute matrices. An objective result of the quality of the model with a measure of generalisation is shown by the results of computations on the test data set. At a learning rate of 0.1 the F1 score result is 74.7%. This condition explains that the learning rate used is not good enough compared to other learning rates. The accuracy obtained reached 64.5%. The accuracy (accuracy) is then best for a learning rate of 0.001.

Table 1. Score and Accuracy

Data Set	Learning rate	F1 score – validation	Accuracy – test model
229	0.001	85	70.83
229	0.01	98.7	89.58
229	0.1	74.7	64.58

It can be seen from a thorough comparison of the test data classifications and the observed results that the network can recognize objects. A portion of all noted cases in the results are false positives, even if positive examples are marked in the results. Retraining the model and enlarging the data set should produce more accurate results. As a C++ library built in the Edge Impulse environment using the created model, the final model was implemented in ESP32-Cam. The value of the needed Flash RAM size is crucial in addition to the final F1 score component. One classifyHole() method in the device control code implements the model itself. The video is used to do picture categorization in order to evaluate this functionality. The resulting image is used to do the categorization. The image is resized to fit the model's resolution because the ESP32-Cam streams the picture at 1600 x 1200px. 320 × 320 pixels was the resolution used to train the model.

A. Testing

Tests were conducted throughout the project at two primary levels. The effectiveness and dependability of the wireless transmission, the functionality of the control software, and its capacity to interface with the ESP32-CAM were assessed during the first round of hardware testing. The outcomes demonstrated that the Wi-Fi link was reliable and able to send photos in real time to the cloud storage firebase bucket storage with no latency. Testing of extra components under control that have been shown to reliably function as a whole.

Using the FOMO model for microcontroller detection, the ESP32-CAM's performance was evaluated in the second testing phase. It was discovered that the ESP32-CAM could process image data using the FOMO algorithm with sufficient performance, especially when using MobileNet V1. At longer distances, there are certain drawbacks such as reduced resolution and image quality. Since the ESP32-Cam MCU is capable of using a variety of lens types, image quality at longer range may be enhanced by swapping out the standard OV2640 lens for a higher-resolution one like the OV3660. However, as the object is often used for detection at distances of no more than 0.5 meters, employing the normal lens is sufficient when taking into account the typical usage scenario of the camera probe. For lower distances between the object and the camera, the lens width is the most important factor to note. Upon closer inspection, the Fisheye wide-angle lens module makes this possible.

Figures 5 explain the changes in object detection at each learning rate tested. The results found that changes between correct object detection and object detection with incorrect results were carried out. At a learning rate of 0.001, there were 14 objects tested that did not match the correct detection object. Meanwhile, at an object detection learning rate of 0.01, 5 inappropriate objects were found, while at a learning rate of 0.1, 17 object detections were found that did not match the desired object. This illustrates the suitability between the training data that has been carried out on the test and validation data in the image.

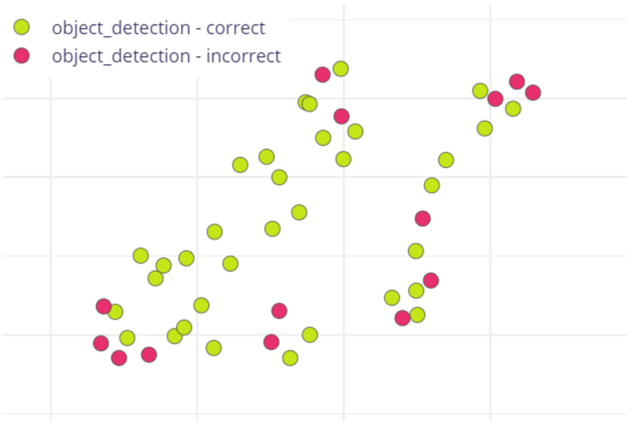


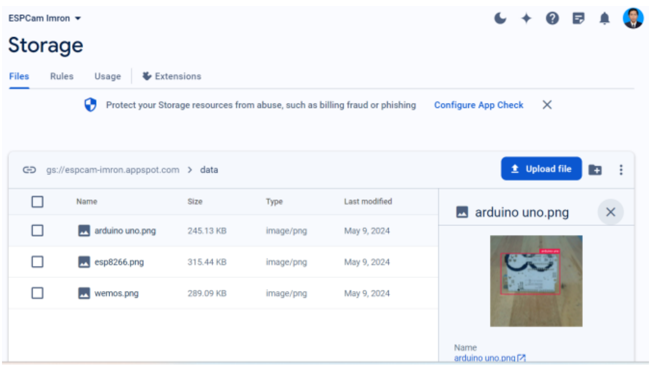
Figure 5. Learning rate 0.001

B. Bucket Storage

Google's platform for creating mobile applications, Firebase, offers a variety of services for managing data from web, iOS, and Android applications. establish a real-time database (RTDB)-equipped Firebase project. File storage in the cloud is possible with Firebase Storage. then, by accessing the Firebase console, can view those files.

The results stored in the storage bucket show that the objects that have been detected can be stored in the cloud. This data is easy to store and easy to access. The process of storing goes through several stages, namely setting authentication, setting rules, and setting storage. Figure 6 is the result of an image that has been managed and stored in the database. The data base type used is littleFS, this type will be the purpose of storing the images that have been taken. Apart from that, the process specified is the file extension used in this case img.png is set. Authentication is used to keep data accessible only by the email and password that have been set.

Specify the structure, indexing strategy, and read and write access times for your data using the declarative rules language offered by Cloud Storage for Firebase. Only authorized users are able to view or write data in Cloud Storage by default due to restrictions on read and write access.


ESPCam Imron

Storage

Files Rules Usage Extensions

Protect your Storage resources from abuse, such as billing fraud or phishing Configure App Check

gs://espcam-imron.appspot.com > data

Upload file

Name	Size	Type	Last modified
arduino uno.png	245.13 KB	image/png	May 9, 2024
esp8266.png	315.44 KB	image/png	May 9, 2024
wemos.png	289.09 KB	image/png	May 9, 2024

arduino uno.png

Name
arduino uno.png

Figure 6. Bucket storage three micro controller

V CONCLUSION

The identification of the FOMO object model, a sophisticated approach for intelligent microcontroller object detection was described in the paper. It was difficult to translate machine learning techniques based on image processing to the embedded system. On the one hand, it was necessary to find a balance between learning rate and accuracy, on the other hand, to maintain the highest possible accuracy in the identification of the microcontroller object to minimize the number of false positive reports. To attain a satisfactory success rate, this necessitated the expansion of the data collection and a thorough classification of the identified objects. Additionally, the project made full use of the ESP32-Cam board's capabilities to quickly construct a prototype for the entire devices.

It was necessary to find a balance between learning rate and accuracy condition of data, on the other hand, to maintain the highest possible accuracy in the identification of the microcontroller object to minimize the number of false positive reports. Find the value learning rate effective to this object is 0.01 with F1 score 98.7% and accuracy score 89.58%.

The result shows cloud storage bucket from firebase solution to adding picture in cloud. Cloud storage can be accessed to develop customized applications for clients. Object detection can truly detect this model and save it to the cloud. the solution is not without flaws (yet), and, for example, extended image augmentation should be performed to eliminate known identification issues. However, this requires a considerable amount of additional effort.

References

Abdellatif, M. M., Elshabasy, N. H., Elashmawy, A. E., & AbdelRaheem, M. (2023). A low cost IoT-based Arabic license plate recognition model for smart parking systems. *Ain Shams Engineering Journal*, *14*(6). https://doi.org/10.1016/j.asej.2023.102178

Akshatha, P. S., & Dilip Kumar, S. M. (2023). MQTT and blockchain sharding: An approach to user-controlled data access with improved security and efficiency. *Blockchain: Research and Applications*, *4*(4). https://doi.org/10.1016/j.bcra.2023.100158

Alejandro, L. L., Gulpric, M. M., Lanon, C. J. F., MacAlalag, F. M. A., & Placio, R. M. A. (2023). ICFY (I Care For You): An IOT Based Fall Detection and Monitoring Device using ESP32-CAM and MPU 6050 Sensors. *2023 8th International Conference on Business and Industrial Research, ICBIR 2023 - Proceedings*, 1009–1013. https://doi.org/10.1109/ICBIR57571.2023.10147586

Bagchi, T., Mahapatra, A., Yadav, D., Mishra, D., Pandey, A., Chandrasekhar, P., & Kumar, A. (2022). Intelligent security system based on face recognition and IoT. *Materials Today: Proceedings*, *62*, 2133–2137. https://doi.org/10.1016/j.matpr.2022.03.353

Chen, F., Meng, F., Li, Z., Li, L., & Xiang, T. (2024). Public cloud object storage auditing: Design, implementation, and analysis. *Journal of Parallel and Distributed Computing*, *189*. https://doi.org/10.1016/j.jpdc.2024.104870

Elhattab, K., Abouelmehdi, K., & Elatar, S. (2023). New Model to Monitor Plant Growth Remotely using ESP32-CAM and Mobile Application. *Proceedings - 10th International Conference on Wireless Networks and Mobile Communications, WINCOM 2023*. https://doi.org/10.1109/WINCOM59760.2023.10322939

Hammad, S. S., Iskandaryan, D., & Trilles, S. (2023). An unsupervised TinyML approach applied to the detection of urban noise anomalies under the smart cities environment. *Internet of Things (Netherlands)*, *23*. https://doi.org/10.1016/j.iot.2023.100848

Hazarika, A., Poddar, S., Nasralla, M. M., & Rahaman, H. (2022). Area and energy efficient shift and accumulator unit for object detection in IoT applications. *Alexandria Engineering Journal*, *61*(1), 795–809. https://doi.org/10.1016/j.aej.2021.04.099

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. http://arxiv.org/abs/1704.04861

Kaur, A., Jadli, A., Sadhu, A., Goyal, S., Mehra, A., & Rahul. (2021). Cloud Based Surveillance using ESP32 CAM. *International Conference on Intelligent Technology, System and Service for Internet of Everything, ITSS-IoE 2021*. https://doi.org/10.1109/ITSS-IoE53029.2021.9615334

Kurdi, H., & Thayananthan, V. (2021). Authentication mechanisms for IoT system based on distributed MQTT brokers: Review and challenges. *Procedia Computer Science*, *194*, 132–139. https://doi.org/10.1016/j.procs.2021.10.066

Li, J., Wu, J., Jiang, L., & Li, J. (2024). Blockchain-based public auditing with deep reinforcement learning for cloud storage. *Expert Systems with Applications*, *242*. https://doi.org/10.1016/j.eswa.2023.122764

Liu, D., Ding, Y., Yu, G., Zhong, Z., & Song, Y. (2024). Privacy-preserving dynamic auditing for regenerating code-based storage in cloud-fog-assisted IIoT. *Internet of Things (Netherlands)*, *25*. https://doi.org/10.1016/j.iot.2024.101084

Liu, Q., Zhang, X., Xue, J., Zhou, R., Wang, X., & Tang, W. (2023). Enabling blockchain-assisted certificateless public integrity checking for industrial cloud storage systems. *Journal of Systems Architecture*, *140*. https://doi.org/10.1016/j.sysarc.2023.102898

Liu, X., Zhang, T., Hu, N., Zhang, P., & Zhang, Y. (2020). The method of Internet of Things access and network communication based on MQTT. *Computer Communications*, *153*, 169–176. https://doi.org/10.1016/j.comcom.2020.01.044

Longo, E., & Redondi, A. E. C. (2023). Design and implementation of an advanced MQTT broker for distributed pub/sub scenarios. *Computer Networks*, *224*. https://doi.org/10.1016/j.comnet.2023.109601

Mirampalli, S., Wankar, R., & Srirama, S. N. (2024). Evaluating NiFi and MQTT based serverless data pipelines in fog computing environments. *Future*

Generation Computer Systems, *150*, 341–353. https://doi.org/10.1016/j.future.2023.09.014

Novak, M., Doležal, P., Budík, O., Ptáček, L., Geyer, J., Davídková, M., & Prokýšek, M. (2024). Intelligent inspection probe for monitoring bark beetle activities using embedded IoT real-time object detection. In *Engineering Science and Technology, an International Journal* (Vol. 51). Elsevier B.V. https://doi.org/10.1016/j.jestch.2024.101637

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. http://arxiv.org/abs/1801.04381

Verma, K., Charan, G. S., Pande, A., Abdalla, Y. A., Marshiana, D., & Choubey, C. K. (2023). Internet Regulated ESP32 Cam Robot. *2023 7th International Conference On Computing, Communication, Control And Automation, ICCUBEA*