Bulletin of Electrical Engineering and Informatics
Vol. 14, No. 1, February 2025, pp. 366~376
ISSN: 2302-9285, DOI: 10.11591/eei.v14i1.7627

Real-time IoT security framework for detecting a person with a weapon using Raspberry Pi, Google Vertex AI, and AWS

Storm Schutte, Jia Uddin
Department of Artificial Intelligence and Big Data, Endicott College, Woosong University, Daejeon, South Korea

Article Info

Article history:

Received Sep 28, 2023
Revised Oct 4, 2024
Accepted Oct 17, 2024

Keywords:

AI safety
Convolutional neural network
Raspberry Pi
Vertex AI
Weapon detection

ABSTRACT

Realtime crime scene detection is a vital issue for ensuring security in various environments. Building on recent advancements in machine learning algorithms, this paper presents an IoT framework for real-time weapon and face detection. By deploying a convolutional neural network (CNN) architecture in Vertex AI and utilizing the portable camera module of a Raspberry Pi, to detect whether a person is carrying a weapon. This is achieved by pre-processing, which we resize and annotate the images. Then, train and validate the CNN model with the annotated label dataset. The trained model is saved in Google Cloud's Vertex AI portal. Then we tested the model by uploading live images from a camera as well as a few video clips, to a Django application in amazon web hosting services (AWS) to Vertex AI. The model exhibited an accuracy of 97.2% along with a F1 score of 0.97. In addition, the model outperforms the other state-of-the-art models by less trainable parameters and higher accuracy.

Corresponding Author:

Jia Uddin
Department of Artificial Intelligence and Big Data, Endicott College, Woosong University
Daejeon, South Korea
Email: jia.uddin@wsu.ac.kr

1. INTRODUCTION

Given the rapidly evolving state of weapon technology globally, it is imperative to deploy counteractive technology to hold individuals accountable for weaponry-related incidents impacting individuals and societies alike. The urgency of this issue is underscored by the sobering statistic of over 250,000 gun-related incidents annually, with more than 85% of the estimated one billion firearms worldwide in civilian possession [1]. However, the challenge lies in the fact that this advanced technology is primarily accessible only to large corporations and governments. It sparks curiosity to consider the potential improvements in managing these issues if the general public were empowered with some level of access to and control over this technological world.

To address these challenges, our research focuses on leveraging deep learning, specifically convolutional neural networks (CNNs), to enhance weapon detection capabilities. We propose a novel solution that integrates a CNN-based model with a portable monitoring module, demonstrating the potential for real-time weapon detection using accessible and scalable technology. This paper outlines our methodology, experimental results, and the practical implications of our findings, showcasing how our approach can democratize access to critical safety technologies.

While previous studies have explored the use of deep learning models for weapon detection, they often rely on high-powered computing resources and lack integration with cloud-based solutions for real-time applications. Our research addresses these gaps by presenting an innovative and efficient solution that transitions from using a Raspberry Pi for local processing to leveraging Vertex AI and AWS for enhanced computational capabilities and scalability. The deep learning model, trained on an extensive dataset of 10,082 images, is deployed within this hybrid infrastructure, ensuring robust and real-time detection capabilities. This setup not only optimizes resource utilization but also demonstrates a practical pathway for deploying advanced weapon detection technologies accessible to broader audiences, thus filling a significant void in current research and applications.

Implementing this solution introduces specific challenges, notably false positives and false negatives. These often arise from objects sharing similar characteristics with weapons [1], such as instances where a person carrying a stick might be inaccurately identified as holding a knife. An additional challenge involves preventing the misinterpretation of background elements for instance advertisements or media, which could lead to false positives [1]. These obstacles can be addressed in various ways, typically by enhancing the complexity of the model. One potential solution is to compute the disparity map, using this information to optimize the selection of candidate regions from the input frame [2].

In recent years, a number of researchers have been and are working on weapon detection [3]. Nakib *et al.* [4] presented a CNN-based model to detect weapons in images, to predict whether a crime occurred. The model utilized rectified linear unit (ReLU), convolutional layer, fully connected layer, and a dropout function to enhance detection accuracy, minimizing false alerts and ensuring system efficiency. Later on, Wang *et al.* [5] applied YOLOv4 deep learning architecture for weapon detection from closed circuit televisions (CCTV) footage and it archived improved mean accuracy precision (mAP) and inference time. A scaled-YOLOv4 was applied for weapon detection by Ahmed *et al.* [6]. Raju *et al.* [7] described a hybrid deep architecture combined the RCNN and YOLOv3 was applied for weapon detection. Fathy and Saleh [2] applied software-defined network for implementing deep learning architectures such as YOLOv5, YOLOv5-lite in IoT environments. Rahman *et al.* [8] used YOLOv8 for real-time object detection from drone. The limitations of YOLO are-computationally intensive especially for high resolution images and real time scenarios. Research by Ghazal *et al.* [9], a hybrid lightweight deep architecture, MobileNetV3-SSDLite applied for handgun detection. The model detected the gun faster than the other models like mobilenetv3 and googlenet in real-time scenarios. The limitations of the model are, they did not compare the performance with computationally intensive models like YOLOv4 and EfficientNet and the model still requires a certain level of computational power to perform real-time detection. Suarez-Paez *et al.* [10] used FasterRCNN for criminal activities detection, whereas, Iqbal *et al.* [11] used a modified FasterRCNN architecture for weapon detection and compared with four CNN architectures- SqueezeNet, GoogleNet, Resnet 18 and 50. The limitations of FasterRCNN are the models struggle with detecting small objects due to their anchor sizes and aspect ratios. VGGNet was applied for weapon detection by Kaya *et al.* [12]. The model outperformed the VGG16, ResNet50, and 101 models. Although the model exhibited higher accuracy, the model has a large number of trainable parameters and required a high memory to deploy the model.

A CNN-based model was deployed in Raspberry Pi to monitor the real-time weapon detection [13]. They evaluated the model using a Raspberry Pi camera module. They tested the model in some controlled environments and did not compare the model's performance with other models. Sultana and Wahid [14] proposed an IoT-based framework for real-time security management and tested the model using Raspberry Pi 3. They tested their model in indoor limited environments. Bhatti et al. applied a number of deep learning architectures such as VGG16, Inception-V3, Inception-ResnetV2, SSDMobileNetV1, Faster-RCNN Inception-ResnetV2(FRIRv2), YOLOv3, and YOLOv4 for weapon detection in real-time CCTV videos. They measured the performance of the deep architectures using F1 score and accuracy, however, they did not consider any hardware, IoT, and cloud platforms to deploy the models. Motivated by the papers [13], [14], where they used the Raspberry Pi model, in this paper we have utilized a portal micro-controller, called Arduino Uno with a camera module to detect the person with weapons. In addition, in this paper, we proposed a complete framework to deploy the deep CNN architecture in Vertex AI and Google Cloud S3 as a database. The rest of the paper is organized as follows: section 2 presents the detailed methodology of the model, while section 3 reports the experimental results. Finally, section 4 concludes the paper.

2. METHOD

To fully harness the potential of this model configuration, the integration of several critical components are necessary. The first component is a monitoring module endowed with image-capturing capabilities and data processing abilities, supplemented by a wireless antenna for data transmission. The second component is a data management module designed to receive, store, and perform additional preprocessing of the incoming data. Finally, a deep learning model residing within Vertex AI forms the third essential component. Together, these components create a powerful, efficient, and comprehensive system, primed to achieve our research objectives as shown in Figure 1.

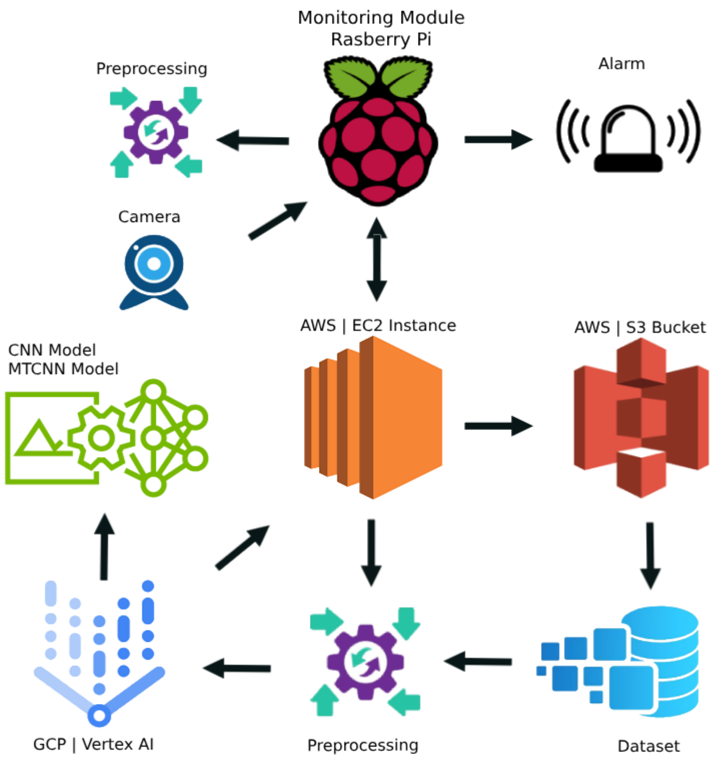


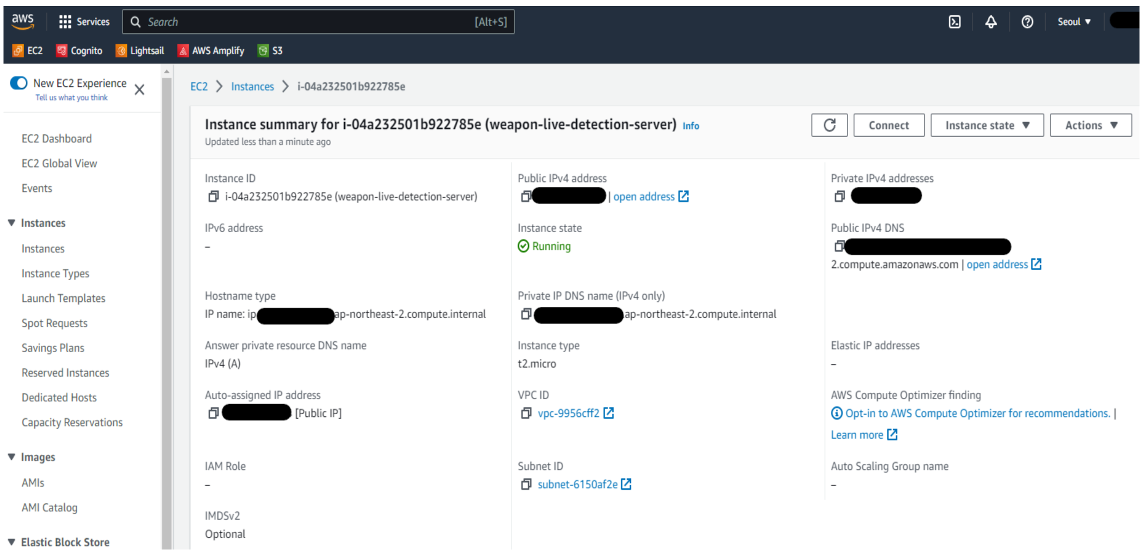
Figure 1. Detailed block diagram of the proposed system

2.1. System architecture

Monitoring module is the first critical component of our system, responsible for capturing images using a Raspberry Pi equipped with an 8 GB RAM and a Quad-core Cortex-A72 (ARM v8) 64-bit SoC operating at a clock speed of 1.8 GHz. This module also includes an integrated high-definition (HD) camera to capture video footage. The captured images undergo initial processing on the Raspberry Pi and are then transmitted wirelessly to a Django application hosted on an elastic compute cloud (EC2) instance on Amazon web services (AWS) as shown in Figure 2(a). This Django application further processes the data into JSON format, preparing it for the next stages. The processed data is subsequently stored in an Amazon S3 bucket as shown in Figure 2(b) and sent to Vertex AI for further analysis as shown in Figure 3(a). The monitoring module's design ensures that it effectively captures and preprocesses images, minimizing the data transmitted to the cloud, which is crucial for optimizing bandwidth and processing time. Figure 3(b) depicts the module used, which includes a Raspberry Pi device equipped with an HD webcam, speaker, monitor, and battery for portable access.

Data management module is designed to handle incoming data, perform additional preprocessing, and manage data storage. This module plays a pivotal role in ensuring efficient handling and transmission of data to the deep learning model. Initially, the data received from the monitoring module is stored in an S3 bucket. The data management module is responsible for organizing this data, ensuring it is correctly formatted and preprocessed before being fed into the deep learning model. This module also handles any additional data cleaning required to enhance the accuracy of weapon identification and face detection. By managing the data flow effectively, this module ensures that the deep learning model receives high-quality, preprocessed data, which is essential for maintaining the model's performance and accuracy.

Deep learning model is the core analytical component of our system, residing within Vertex AI. This model processes the data received from the data management module and sends back a response containing the results. The deep learning model employed is a CNN, structured with three layers: MaxPooling2D, Flatten, and Dense. This streamlined architecture promotes efficient computation, eliminating the need for multiple complex algorithms. After processing the data, the model generates results, which include the detection of weapons and identification of individuals. These results, along with the URLs of the processed images, are stored in a PostgreSQL database. A simplified response indicating the detection of a weapon (if any) is relayed back to the monitoring module, forming a seamless data processing pipeline. This pipeline ensures efficient image capture, processing, and interpretation, enabling real-time monitoring and detection capabilities.

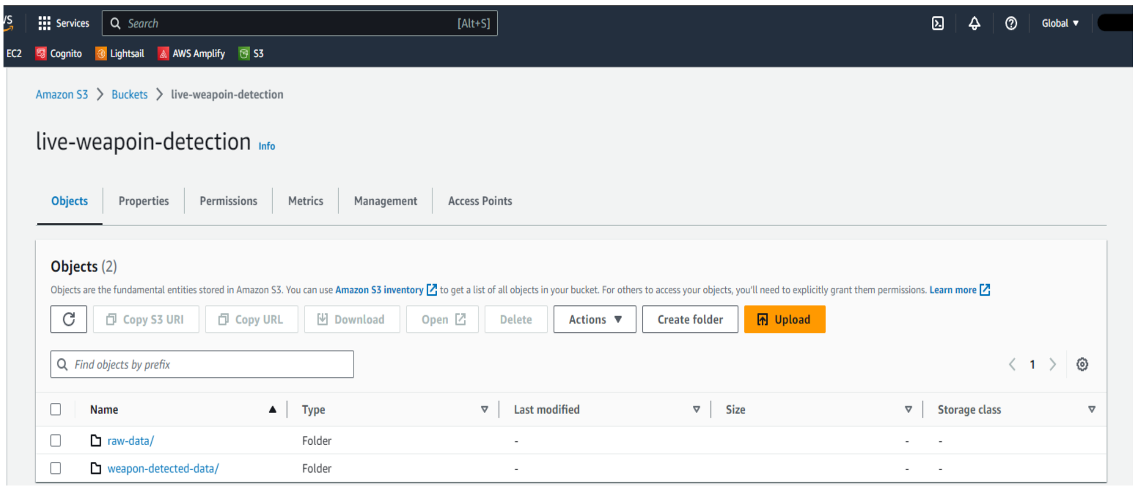


EC2 > Instances > i-04a232501b922785e

Instance summary for i-04a232501b922785e (weapon-live-detection-server) Info
Updated less than a minute ago

Instance ID i-04a232501b922785e (weapon-live-detection-server)	Public IPv4 address [redacted] | open address	Private IPv4 addresses [redacted]
IPv6 address –	Instance state Running	Public IPv4 DNS [redacted] 2.compute.amazonaws.com | open address
Hostname type IP name: ip[redacted]ap-northeast-2.compute.internal	Private IP DNS name (IPv4 only) [redacted]ap-northeast-2.compute.internal	
Answer private resource DNS name IPv4 (A)	Instance type t2.micro	Elastic IP addresses –
Auto-assigned IP address [redacted] [Public IP]	VPC ID vpc-9956cff2	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role –	Subnet ID subnet-6150af2e	Auto Scaling Group name –
IMDSv2 Optional		

(a)



Amazon S3 > Buckets > live-weapoin-detection

live-weapoin-detection Info

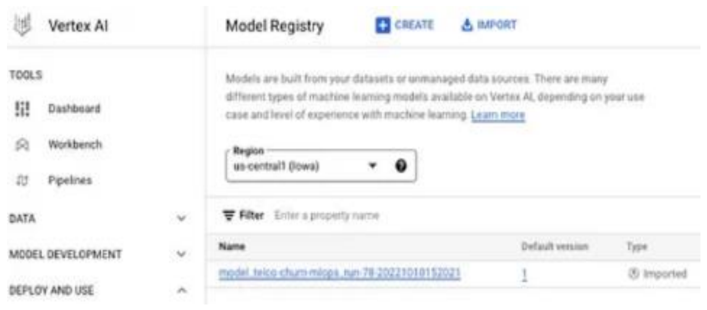
Objects (2)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
raw-data/	Folder	-	-	-
weapon-detected-data/	Folder	-	-	-

(b)

Figure 2. A demonstration of; (a) AWS EC2 and (b) AWS S3 bucket



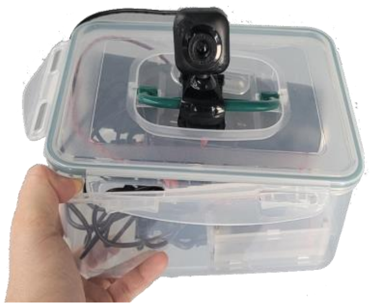
Models are built from your datasets or unmanaged data sources. There are many different types of machine learning models available on Vertex AI, depending on your use case and level of experience with machine learning. Learn more

Region
us-central1 (Iowa)

Filter Enter a property name

Name	Default version	Type
model_telco-churn-mlops_run-78-20221018152021	1	Imported

(a)



(b)

Figure 3. A demonstration of; (a) a model on Vertex AI and (b) the module used consists of a Raspberry Pi device, HD webcam, speaker, monitor, and battery for portal access

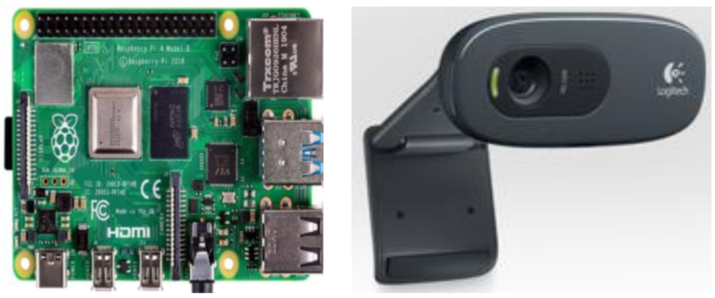
2.2. Dataset and monitoring module

In our research, we utilized two distinct datasets and constructed a compact, transportable monitoring module to refine weaponry and face recognition techniques. The first dataset emphasized weaponry, incorporating images of 32 different types of guns, along with knives, swords, and crossbows; examples can be found by looking in Figures 4 and 5. The second dataset was a collection of human faces. These datasets were further reinforced with the application of multi-task cascaded convolutional networks (MTCNN) to ensure superior accuracy.

Figure 4. A demonstration of the weapons found in the dataset of crime scenes

Figure 5. A demonstration of the weapons found in the dataset of annotated sample crime devices

A crucial element in our research process was the deployment of the Raspberry Pi our monitoring module, which was outfitted with 8 GB of RAM and a Quad-core Cortex-A72 (ARM v8) 64-bit SoC operating at a clock speed of 1.8 GHz seen in Figures 6(a) and (b). This system ran on the Ubuntu Server as its operating system, selected for its minimal computational requirements and user-friendly interface. An integrated high-definition (HD) camera enabled the Raspberry Pi to capture video footage, preprocess the data, and relay the preprocessed results to the AWS API and then to the Vertex AI endpoint.



(a) (b)

Figure 6. Sample image of; (a) Raspberry Pi and (b) the camera used in the experiment

Once the datasets were assembled, we initiated image preprocessing that involved resizing and augmenting the images to enrich the model's precision. This preprocessing was crucial to the efficacy of the compact monitoring module we developed. This module embodied our research premise and demonstrated our capability to adapt monitoring to any device equipped with our endpoint. It was engineered to capture and process images from the environment, effectively minimizing the data transmitted to the Google Cloud Portal (GCP) Vertex AI. This was achieved through data reduction techniques, such as image reduction and comparison analysis, which was facilitated by a Python 3 script running on the aforementioned Ubuntu server. The processed data was then relayed from the endpoint to the virtual private server (VPS), from where it was sent to Vertex AI.

By employing this combined methodology of diverse datasets and a transportable monitoring module, we offer a comprehensive solution for enhancing object and face recognition systems. The aim of our research was to redefine the norms in data gathering and processing, contributing to making image recognition technology more accurate and versatile.

2.3. Image pre-processing and cloud storage

In managing the data we gathered, two critical tasks were identified. The first involved cleaning the data to enhance the accuracy of weaponry identification a person might be carrying and refining face images for improved face detection. The second task aimed to provide a highly reliable endpoint for data storage, ensuring easy access from any location.

For the preprocessing of collected weapon data, we implemented two techniques. Initially, we resized the dataset to uniform dimensions of 254×254 pixels to maintain consistency across all data points. Subsequently, we performed data augmentation; for each weapon in our dataset, we increased its representation a hundredfold, elevating our data set from 1,035 to 10,082 as shown in Figure 7. This augmentation involved altering colors, adjusting image angles, and adding noise to the images as seen in Figure 7.

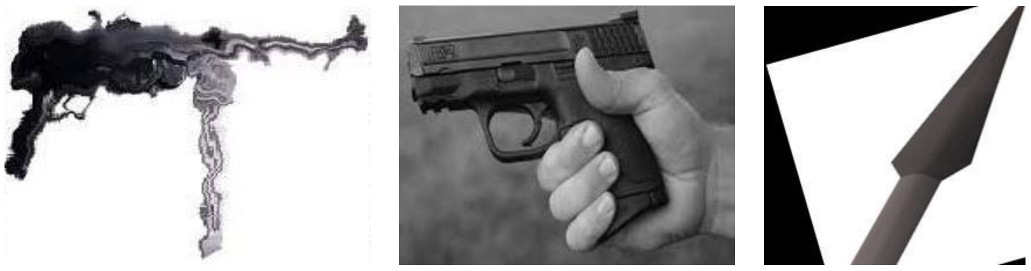


Figure 7. A demonstration of augmented images

When it came to face data preprocessing, we chose to incorporate our face dataset. This decision was driven by the aim to enhance the accuracy of identifying individuals, thereby improving the overall efficiency of face detection and allowing us to put a name to the person holding a weapon. The image preprocessing commenced with the application of the MTCNN face cropping technique. Following this, we implemented resizing and augmentation akin to the methods used with the weapon dataset.

Concerning data storage, we opted for the simple storage service (S3) on AWS. We created two distinct buckets within this service-one for storing images pending preprocessing and another for those that had been processed. The S3 service was selected for its substantial storage capacity and it is 99.9% reliability ratio, ensuring high availability. The internal address of each bucket was shared with a Django application tasked with managing the responses from the monitoring module.

2.4. Deep learning

The deep learning model we have chosen to utilize is a CNN. This model is structured with three layers, namely: MaxPooling2D, Flatten, and Dense seen in the below equations. The streamlined architecture of the CNN model promotes efficient computation within deep learning technology, thereby eliminating the need for a multitude of complex algorithms. This becomes a critical factor in our methodology, as it empowers us to transmit compact data packets to Vertex AI from the supervisory module above.
Here are the mathematical equations used in this CNN model:

Conv2D (32 filters, 3x3 kernel, ReLU activation)

$$ReLU(\sum u,v\ Input[i+u,j+v,k]\times Kernel[u,v,k]+Bias[k])$$

MaxPooling2D (2x2)

$$Output[i,j,k]=max\ u\in[0,1],v\in[0,1]Input[2i+u,2j+v,k]$$

Dense (128 units, ReLU activation)

$$Output[i]=ReLU(\sum j\ Input[j]\times Weight[j,i]+Bias[i])$$

Dense (Softmax activation)

$$Z[i]=\sum j\ Input[j]\times Weight[j,i]+Bias[i]$$

2.5. Cloud computing-Vertext AI

We leveraged the power of Vertex AI, a component of the GCP, chosen for its capacity to harness extensive computational power that would be challenging to manage manually [15]. Vertex AI, an advanced machine learning platform, is designed to facilitate the training and deployment of machine learning models and artificial intelligence applications. The platform allows for the interface with the model from any device possessing the appropriate endpoint to GCP's Vertex AI.

The first step in operationalizing Vertex AI was converting our dataset into JSON format, a prerequisite for Vertex AI to interpret the data. After this conversion, the formatted data was stored in an S3 bucket. In parallel, we completed the necessary API implementation and Google Cloud SDK setup to facilitate effective interfacing with Vertex AI.

Our existing CNN model was subsequently encapsulated within a Docker image. This Docker image was then placed in the container registry of Vertex AI and was used to train our model. Following this, the trained model was deployed within the Vertex AI environment. This operational setup allowed us to either employ a VPS or directly link our monitoring system to an endpoint on Vertex AI.

This use of Vertex AI significantly enhanced the robustness of our system, allowing us to seamlessly train and deploy complex machine learning models with high computational requirements. This streamlined process forms an integral part of our research, aiming to redefine norms in data gathering, processing, and machine learning model training and deployment.

2.6. Additional insight into similar studies

Several studies have shown the efficacy of using deep learning and cloud computing in real-time weapon detection systems. For instance, a study by Torregrosa-Domínguez *et al.* [16] demonstrated the use of YOLOv3 for weapon detection, achieving high accuracy rates and real-time performance by leveraging AWS EC2 instances for data processing and model deployment. Another research by Kumar and Jain [17] highlighted the integration of CNNs with edge devices, showing significant improvements in processing speed and data transmission efficiency when using Google Cloud's Vertex AI. These studies underscore the potential of combining advanced deep learning models with robust cloud computing platforms, aligning closely with the methodology employed in our research. By integrating these proven approaches, our system aims to enhance the accuracy and efficiency of weapon detection and face recognition, contributing to the broader field of security and surveillance technology.

3. RESULTS AND DISCUSSION

In this experimental phase, we trained the model on 100 epochs, where the final stages showcased consistent performance. The average training accuracy reached approximately 97.2%, with a negligible average training loss of 0.76%. The high training accuracy shows the model's capability to fit the data well. Furthermore, the training F1 Score was also exceedingly high at approximately 97.1%, suggesting that the precision and recall of the model during training were optimal.

If we look at the validation results, the CNN model achieved an accuracy of 79%, with an average loss of 1.21%. The validation F1 Score stood at 81 %, closely matching the validation accuracy. While there is a slight difference between training and validation metrics, which could indicate some overfitting, the model still maintains a reasonably high validation accuracy. This demonstrates its potential utility in real-world scenarios or applications.

It is important to highlight that the validation loss progressively increased over time as shown in Figure 8. Several factors could contribute to this trend. A deficiency in data quantity might be a significant factor, along with not harnessing the full capabilities of the CNN model. Over-augmenting images may also be a contributing factor, potentially introducing unnecessary variances that the model attempts to learn. Another plausible reason for the uptick in validation loss could be the excessive complexity of the model. An overly intricate model might inadvertently memorize extraneous noise, rather than discerning the essential patterns in the data. The total computational time for the entire training process spanned approximately 15,200 seconds (or about 253 minutes). This shows the computational efficiency of the training process, considering the complexities involved in neural network optimization over 100 epochs.

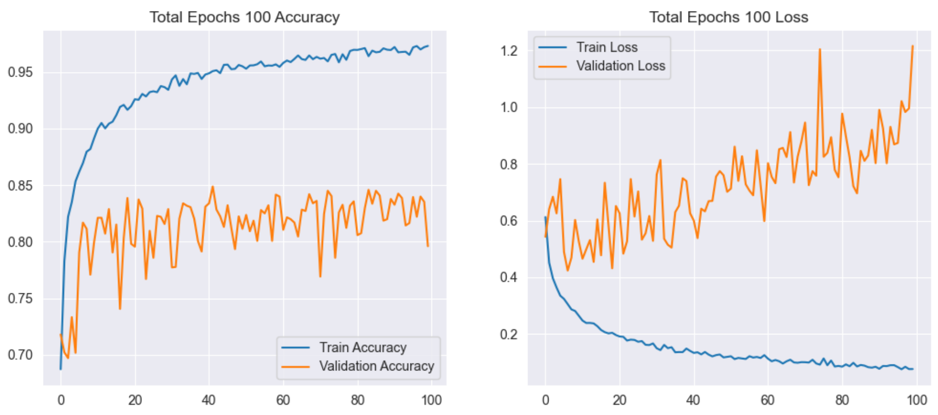


Figure 8. A graph displaying the CNN model

3.1. Experiment with real environmental dataset

In our research, we evaluated the application under realistic conditions, specifically examining it is portability, result delivery speed, and the device's battery longevity. To ascertain the product's effectiveness, we positioned it in an open field and approached the camera rapidly with a knife in hand. The goal was to see if the system could trigger an alarm before we reached the camera, signaling the successful capture, preprocessing, relay to the EC2 manager, forwarding to Vertex AI, and feedback to the module, all within the time taken to approach the device. This whole process took on average 3.2 seconds, and the models prediction took on average 0.4 ms. We further assessed the application's capabilities by placing it in a stationary position and driving past it at 60 km/h. This test aimed to gauge the application's practical utility in detecting weapons from fast-moving vehicles.

During live tests Figure 9, as depicted in the preceding images, the camera adeptly identified both an individual's face and the presence of either a knife or gun. The module was designed to capture, preprocess, and send a frame every 5 seconds. This was made possible by the monitoring module's independent battery and its connection to a mobile hotspot. Through these real-time demonstrations, we probed various use-cases for this system, including hand-held use while walking and mounting on a moving car.

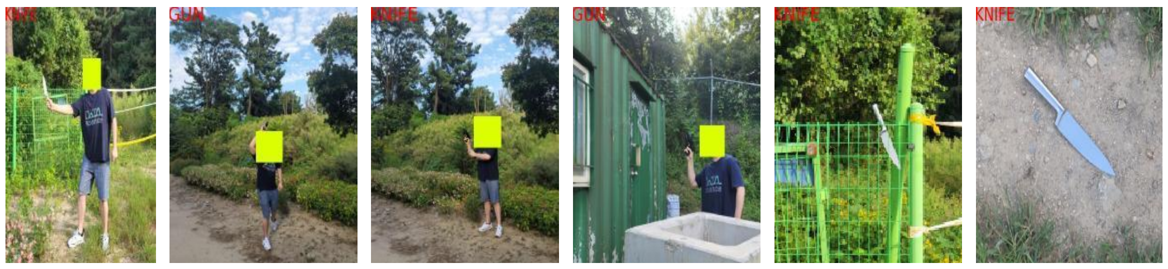


Figure 9. These pictures are illustrating the live testing

Overall, the results were promising with a high accuracy level. There were instances, however, such as shown in Figure 9, where a knife was misidentified as a gun. While this indicates a slight difference in precision, the presence of a weapon was still detected, a possible discrepancy attributable to the capture distance. All live tests employed the portable weapon and face detection module, as illustrated in Figure 3(b).

3.2. Comparing the convolutional neural network model with AlexNet

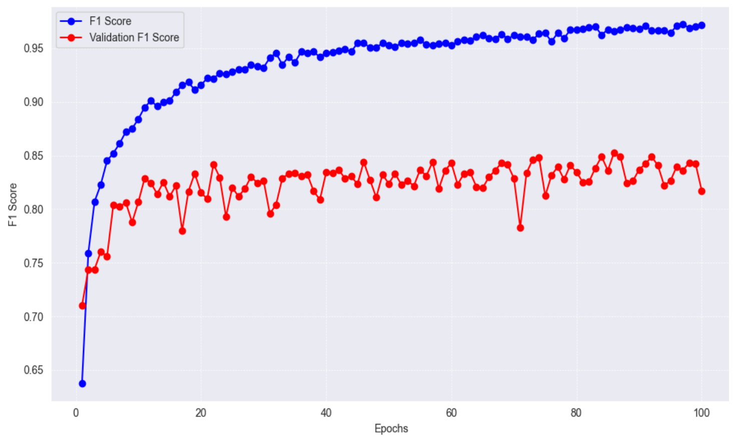
Upon comparing the two models Table 1, our CNN model outperformed the AlexNet model, achieving a 97% F1 score with 20 million fewer parameters. It also required less computation and training time. However, our model's higher training loss suggests potential overconfidence in it is predictions, which could possibly lead to incorrect predictions. In contrast, AlexNet's deeper architecture might offer better calibration in its uncertainty. As data volume increases, the performance dynamics of both models may shift, emphasizing the need for further evaluations.

Table 1. A summary of the models used in the application

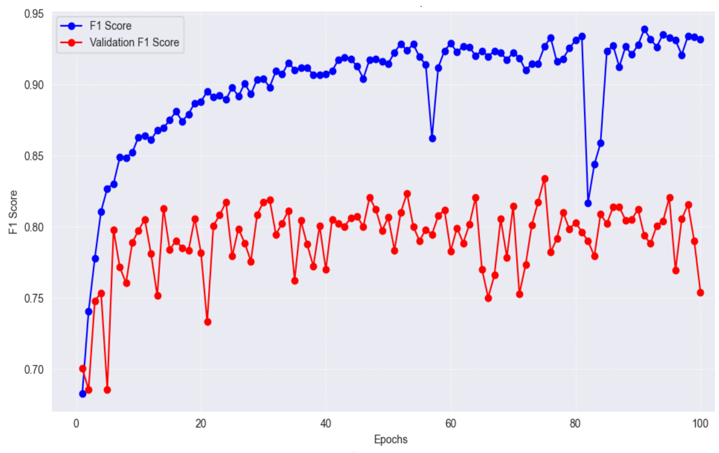
Table head	CNN model result (100 epochs)			
	Train accuracy	Train loss	Trainable parameters	F1 score
Units	97.2%	0.76	10.6 M	0.971
Units	0.93	0.17	29.97 M	0.81

However, it is essential to note that despite having a higher accuracy, the model’s training loss is significantly higher than AlexNet’s. This might indicate that the model is more confident in it is predictions even when they’re incorrect, while AlexNet, despite being less accurate, might be more “calibrated” in its uncertainty as shown in Figures 10(a) and (b).

Simpler architecture, has 10.6 M trainable parameters and achieves a 97.2% training accuracy and an F1 score of 0.971. In contrast, the more complex AlexNet boasts 29.97 M parameters but secures a lower training accuracy of 93% and an F1 score of 0.81 as shown in Figures 10(a) and (b).



(a)



(b)

Figure 10. F1 graph; (a) of the model and (b) of AlexNet

4. IMPLICATIONS FOR THE RESEARCH FIELD AND COMMUNITY

Scalability and accessibility: by integrating cloud-based solutions such as Vertex AI and AWS, our framework can be scaled to cover larger areas and more complex scenarios, making advanced weapon detection accessible to smaller organizations and the general public [18], [19].

Future research directions: future work can focus on refining the model to reduce false positives and negatives further, possibly by incorporating additional data types or leveraging more sophisticated architectures [20], [21]. Research can also explore the integration of this framework with other IoT devices to create a comprehensive security system [22], or even leveraging security predictions in 3D detection environments [23].

Potential applications: beyond weapon detection, this framework can be adapted for other security-related applications, such as detecting unauthorized access, monitoring restricted areas, and even in smart city implementations for enhanced public safety [24].

Community impact: empowering communities with such technology can significantly improve local security measures, allowing for quicker response times and potentially deterring criminal activities [25]. Moreover, the involvement of the general public in security measures fosters a collaborative approach to safety [26].

5. CONCLUSION

The proposed IoT security framework for real-time weapon detection using a Raspberry Pi, Google Vertex AI, and AWS has demonstrated significant potential in enhancing public safety through advanced technological means. Our CNN-based model exhibited a high training accuracy of 97.2% and a validation accuracy of 79%, indicating robust performance in detecting weapons in real-time scenarios.

Recent observations indicate that the increased efficacy of real-time weapon detection systems is linked to the optimization of deep learning algorithms and cloud-based processing capabilities. Our findings offer definitive proof that this phenomenon is linked to the integration of CNN architectures with cloud computing resources, rather than being caused by increased quantities of data alone. This highlights the importance of sophisticated model design and resource management in achieving high detection accuracy and efficiency.

In summary, our research presents a practical, scalable, and efficient solution for real-time weapon detection, leveraging cutting-edge deep learning techniques and cloud computing resources. The broader implications of this work suggest that similar approaches can be adapted for various other applications, contributing to a safer and more secure society.

ACKNOWLEDGEMENTS

This research is funded by Woosong University Academic Research 2024.

REFERENCES

[1] A. H. Ashraf *et al.*, “Weapons detection for security and video surveillance using CNN and YOLO-V5s,” *Computers, Materials and Continua*, vol. 70, no. 2, pp. 2761–2775, 2022, doi: 10.32604/cmc.2022.018785.

[2] C. Fathy and S. N. Saleh, “Integrating deep learning-based IoT and fog computing with software-defined networking for detecting weapons in video surveillance systems,” *Sensors*, vol. 22, no. 14, Jul. 2022, doi: 10.3390/s22145075.

[3] N. Vallez, A. Velasco-Mata, J. J. Corroto, and O. Deniz, “Weapon detection for particular scenarios using deep learning,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 371–382, 2019, doi: 10.1007/978-3-030-31321-0_32.

[4] M. Nakib, R. T. Khan, M. S. Hasan, and J. Uddin, “Crime scene prediction by detecting threatening objects using convolutional neural network,” in *International Conference on Computer, Communication, Chemical, Material and Electronic Engineering, IC4ME2 2018*, 2018, doi: 10.1109/IC4ME2.2018.8465583.

[5] G. Wang, H. Ding, M. Duan, Y. Pu, Z. Yang, and H. Li, “Fighting against terrorism: A real-time CCTV autonomous weapons detection based on improved YOLO v4,” *Digital Signal Processing*, vol. 132, Jan. 2023, doi: 10.1016/j.dsp.2022.103790.

[6] S. Ahmed, M. T. Bhatti, M. G. Khan, B. Lövström, and M. Shahid, “Development and optimization of deep learning models for weapon detection in surveillance videos,” *Applied Sciences*, vol. 12, no. 12, Jun. 2022, doi: 10.3390/app12125772.

[7] A. R. Raju, T. Maddileti, S. J, R. Srinivas, and K. Saikumar, “Pseudo trained YOLO R_CNN model for weapon detection with a real-time kaggle dataset,” *International Journal of Integrated Engineering*, vol. 14, no. 7, Dec. 2022, doi: 10.30880/ijie.2022.14.07.011.

[8] S. Rahman, J. H. Rony, J. Uddin, and M. A. Samad, “Real-time obstacle detection with YOLOv8 in a WSN Using UAV aerial photography,” *Journal of Imaging*, vol. 9, no. 10, 2023, doi: 10.3390/jimaging9100216.

[9] M. Ghazal, N. Waisi, and N. Abdullah, “The detection of handguns from live-video in real-time based on deep learning,” *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 18, no. 6, Dec. 2020, doi: 10.12928/telkomnika.v18i6.16174.

[10] J. Suarez-Paez *et al.*, “A novel low processing time system for criminal activities detection applied to command and control citizen security centers,” *Information*, vol. 10, no. 12, Nov. 2019, doi: 10.3390/info10120365.

[11] M. J. Iqbal, M. M. Iqbal, I. Ahmad, M. O. Alassafi, A. S. Alfakeeh, and A. Alhomoud, “Real-time surveillance using deep learning,” *Security and Communication Networks*, pp. 1–17, Sep. 2021, doi: 10.1155/2021/6184756.

[12] V. Kaya, S. Tuncer, and A. Baran, “Detection and classification of different weapon types using deep learning,” *Applied Sciences*, vol. 11, no. 16, Aug. 2021, doi: 10.3390/app11167535.

[13] A. Al-Mousa, O. Alzaibaq, and Y. Abu-Hashyeh, “Deep learning-based real-time weapon detection system,” *International Journal of Computing and Digital Systems*, vol. 14, no. 1, pp. 531–540, Aug. 2023, doi: 10.12785/ijcds/140141.

[14] T. Sultana and K. A. Wahid, “IoT-guard: event-driven fog-based video surveillance system for real-time security management,” *IEEE Access*, vol. 7, pp. 134881–134894, 2019, doi: 10.1109/ACCESS.2019.2941978.

[15] “Introduction to Vertex AI,” Google. [Online]. Available: https://cloud.google.com/vertex-ai/docs/start/introduction-unified-platform. (Accessed: Feb. 21, 2023).
[16] Á. Torregrosa-Domínguez, J. A. Álvarez-García, J. L. Salazar-González, and L. M. Soria-Morillo, “Effective strategies for enhancing real-time weapons detection in industry,” Applied Sciences, vol. 14, no. 18, Sep. 2024, doi: 10.3390/app14188198.
[17] S. Kumar and M. Jain, “Enhancing edge device capabilities with CNNs and Vertex AI,” *International Journal of Machine Learning Applications*, vol. 18, no. 2, pp. 137–149, 2023.
[18] “Vertex AI,” Google Cloud. [Online]. Available: https://cloud.google.com/vertex-ai?hl=en. (Accessed: May 28, 2024).
[19] “AWS Cloud,” Amazon Web Services. [Online]. Available: https://aws.amazon.com. (Accessed: May 28, 2024).
[20] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *arXiv*, Jul. 2012, doi: 10.48550/arXiv.1207.0580.
[21] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[22] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of things (IoT): A vision, architectural elements, and future directions,” *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.
[23] P. Ananthachari and S. Schutte, “Big Data Tools, Deep Learning & 3D Objects in the Metaverse,” *Frontiers in Artificial Intelligence and Applications*, vol. 376, pp. 236–245, Oct. 2023, doi: 10.3233/FAIA230736.
[24] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of things for smart cities,” *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014, doi: 10.1109/JIOT.2014.2306328.
[25] V. Bauer, “Predictive Policing in Germany. Opportunities and challenges of data-analytical technology in order to prevent crime,” *Revue Internationale de Droit Penal*, pp. 117–148, 2020.
[26] P. M. Cozens, G. Saville, and D. Hillier, “Crime prevention through environmental design (CPTED): A review and modern bibliography,” *Property Management*, vol. 23, no. 5, pp. 328–356, 2005, doi: 10.1108/02637470510631483.

BIOGRAPHIES OF AUTHORS

Storm Schutte with 16 years in IT and a decade in programming, began at SwitchAir (Pty) Ltd in 2007, advancing to a directorial role in IT management. He further honed his skills internationally before leading at SableApps (Pty) Ltd 2017, developing a notable healthcare system for Korean hospitals. In 2022, he joined SLVRCLD, enhancing content claims processes through visual computing, NLP, and machine learning. concurrently, he is pursuing a Masters in AI at Woosong University, South Korea, marrying academic pursuits with practical tech applications. He can be contacted at email: stormschutte8@gmail.com.

Jia Uddin is as an Assistant Professor, Department of AI and Big Data, Endicott College, Woosong University, Daejeon, South Korea. He received Ph.D. in Computer Engineering from University of Ulsan, South Korea, M.Sc. in Telecommunications, Blekinge Institute of Technology, Sweden, and B.Sc. in Computer and Communication Engineering, International Islamic University Chittagong, Bangladesh. He was a visiting Faculty at School of Computing, Staffordshire University, United Kingdom. He was an Associate Professor in Department of Computer Science and Engineering, Brac University, Dhaka, Bangladesh. His research interests are industrial fault diagnosis, machine learning/deep learning-based prediction and detection using multimedia signals. He can be contacted at email: jia.uddin@wsu.ac.kr.