International Journal of Scientific Research in Science, Engineering and Technology
Print ISSN: 2395-1990 | Online ISSN : 2394-4099 (www.ijsrset.com)
doi : https://doi.org/10.32628/IJSRSET

Object Detection System Using Yolo Algorithm

R.A. Arul Raja[1*], V.G. Anisha Gnana Vincy[2]
[1]Assistant Professor (Sr. G), Department of Mechanical Engineering, SRM Institute of Science and Technology, Vadapalani campus, Chennai, Tamil Nadu, India. Email: raarulraja@gmail.com
[2]Assistant professor, Department of Computer Science and Engineering, Dhaanish Ahmed College of Engineering, Padappai, Chennai, Tamil Nadu, India.

Article Info
Volume 9, Issue 5
Page Number : 295-300

Publication Issue :
September-October-2022

Article History
Accepted : 01 Sep 2022
Published: 20 Sep 2022

ABSTRACT
In the most recent ten years, there has been a tremendous amount of study in the field of computer vision due to the pervasive and broad applications such scene interpretation, video surveillance, robotics, and self-driving systems. Visual recognition systems, which include picture categorization, localization, and detection, have gained significant research momentum as the foundation of all these applications. These visual recognition algorithms have achieved exceptional performance as a result of substantial advancements in neural networks, particularly deep learning. One of these areas where computer vision has had considerable success is object detection. This research clarifies the function of convolutional neural network-based deep learning algorithms for object detection. A list of deep learning frameworks and object detection services is also provided. This article introduces readers to the YOLO object identification technique and explains how it works.

Keywords : Object Detection, Deep Learning, Fog computing, Cloud computing, Internet of Things

I. INTRODUCTION

The "pay-as-you-go" computation service made available to consumers by the cloud computing paradigm is an effective substitute for owning and operating private data centres for both individuals and businesses [1]. For a variety of latency-sensitive applications that demand real-time results, such as healthcare and surveillance, this Internet-based distant utility computing approach presents a challenge [2]. IoT applications and services that are resource-hungry and demand speedy responses are quickly replacing earlier IoT installations. The Fog computing paradigm, which utilises edge resources as well as remote cloud resources when necessary [3], has arisen to address this issue. Fog computing has become quite important because of its dependability and capacity to offer a variety of response characteristics depending on the intended applications. This new paradigm has made things easier.

With the widespread deployment of IoT-enabled devices like cameras everywhere, object detection and monitoring have become omnipresent in contemporary digital society [5]. In autonomous video surveillance, video sequences are analysed for multiple purposes employing object identification, segmentation, and classification. As a result, computer vision research is actively focused on object detection, which is an important component. Such object detection software is now considerably faster, more accurate, and more precise because to considerable advancements in the fields of computer vision and deep learning over the past several years. Due to rising data rates and inadequate system frameworks, such systems are still unable to operate well when they are implemented in real-world settings. Even with effective models that can analyse videos at high frame rates and resolutions in a short amount of time, network overhead and lengthy data transfers make these deployments ineffective. Fog computing appears to be a viable option for greatly reducing data transfer times, which would enhance the responsiveness and effectiveness of such systems.

However, none of them offer a framework for seamless IoT-Fog-Cloud connection that can be installed with engineering simplicity and adapt to the user and application requirements. There have been several attempts to transfer such deep learning applications to fog settings [6]. This work was driven by the dearth of models or frameworks that combine the strength of highly accurate deep learning models with the responsiveness of edge nodes. In this study, we have created EdgeLens, a deployable fog-cloud system for real-time object recognition that is based on deep learning. Our framework employs the Aneka service [7] and adjusts to user and/or application needs to give two modes in which the results are generated, one in which they are produced with high accuracy and the other in which they are produced with low latency. This article introduces readers to the YOLO object identification technique and explains how it works.

II. RELATED WORKS

A smart surveillance architecture for detecting quickly moving cars was modelled by Chen et al. [8]. Fog computing nodes, such as mobile phones, tablets, police cars, and other on-site devices with computation capability, process the footage that was obtained. However, their model can only detect one vehicle at a time, not numerous targets. Deep learning was used by Diro et al. [9] to create a distributed attack detection method.

Deep learning for smart industries using a fog computing architecture was demonstrated by Li et al. [10]. By shifting the computational work from the central server to the fog nodes, their technology is able to handle the vast amounts of data gathered from sensors used in industrial manufacturing to find product flaws. For the prediction study, they used convolutional neural networks (CNN), which also simultaneously displayed the type of error and its severity. A distributed deep learning network (DDNN) model over a distributed computational hierarchy made up of the cloud, the edge (fog), and the end devices was proposed by Teerapittayanon et al. [11]. They demonstrated that they could achieve high accuracy and lower transmission costs by processing more sensor data locally on end devices as opposed to offloading to the cloud.However, they only used binary Neural Network layers (NN) in their tests and didn't take into account the option of using mixed precision or floating-point NN layers. In order to perform data conditioning, intelligent filtering, smart analytics, and selective transmission to the cloud for long-term storage and temporal variability monitoring, Constant et al. [12] constructed a prototype of a smart gateway. Thus, they introduced end-to-end contact between the sensor devices and the cloud through smart gateways.

III. BACKGROUND TECHNOLOGY - ANEKA

We used the Aneka platform to take advantage of the processing power of edge and cloud resources to create and implement the suggested system [7]. A platform for creating and launching applications on cloud infrastructure is called Aneka. It offers a runtime environment and APIs that permit the creation of.NET apps that utilise the computing power of public or private clouds [7]. Virtual Machines (VMs) offered by cloud service providers like Azure or Amazon Web Services might be a part of the public cloud. Enterprise cloud VMs, fog, or edge devices on the Local Area Network can all be a part of the private cloud (LAN).

The foundational elements of the Aneka framework are created and put into action in a service-oriented manner. Dynamic provisioning, or the capacity to dynamically acquire resources and integrate them into current frameworks and software solutions, is a service offered by Aneka. Resource provisioning and the Scheduler service are the two key services that Aneka uses to deliver dynamic provisioning.

IV. SYSTEM ARCHITECURE

Figure 1 depicts an IoT-based fog-cloud integration system architecture for object detection that can efficiently manage incoming photos and deliver findings in almost real-time. It enables structured communication and integrates many hardware and software components. The system has the following hardware components:

1) Input Sensors: These include cameras and video-cameras that may or may not be attached with the gateway device.
2) Gateway: Different types of gateway devices exist which include mobile phones, laptops and tablets. These act as fog devices that collect the images from cameras and forward them to the Aneka Master node for detection.
3) Aneka Master: This is the Aneka master container in fog environment, that takes job requests from the gateway device and sends it to worker containers. It contains the dynamic provisioning, load balancing and scheduling models that helps in task distribution across the Aneka containers.

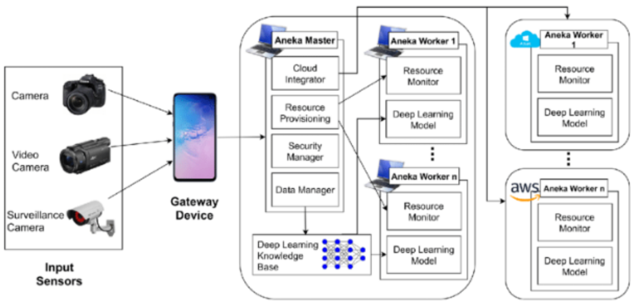


Figure 1 IoT-based fog-cloud integration system architecture for object detection

V. DESIGN AND IMPLEMENTATION

A. Preprocessing

Users of the proposed system can set up the data processing in either the High-Accuracy mode or the Low-Latency mode. The model adjusts resource distribution and input pre-processing in accordance with the mode chosen. In the first mode, the model delivers the Aneka Master container the raw input image without any compression.

B. Task Parallelism

EdgeLens distributes workload among Edge nodes and Cloud VMs using the Aneka-Task architecture. Using the.NET framework and C#, the task parallelism code was created. After pre-processing, the Aneka Master receives the input image and prepares an Aneka Task before sending it to one of the employees for processing.

C. Deep Learning Application - You Only Look Once (YOLO)

This framework makes advantage of the object detection model known as YOLO (You Only Look Once) [14]. By drawing a bounding box around the objects in the image, the aim is to locate their locations and classify them into several categories. For this task, many steps of the Recurrent Convolutional Neural Network (R-CNN) and its modifications are used. This proved to be tedious, requiring separate training for each component. However, YOLO features an end-to-end architecture that performs categorization and region suggestion simultaneously and so produces faster results. The image is divided into an SS grid, with each cell interested in predicting 5+k (k is the number of classes) quantities, including the likelihood (confidence) that this cell is actually contained in a true bounding box, the width and height of the bounding box, its centre (x, y), and the likelihood that an object in the bounding box belongs to the kth class (k-values). As a result, the output layer has SS(5+k) elements. Now, the bounding box, confidence, and object for each cell are computed [14].

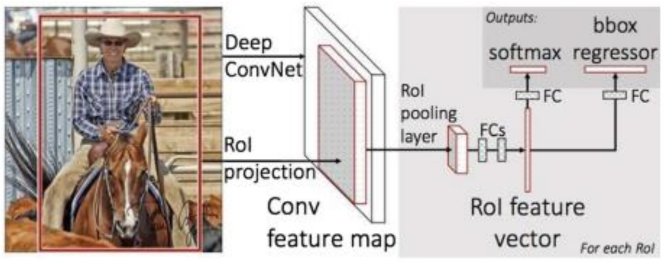


Figure 2. Fast R-CNN

The picture (Figure 1) is first separated into grids. Each grid has a S x S dimension. The graphic below demonstrates how an input image is separated into grids. There are numerous grid cells of identical size in the image above. Objects that appear within grid cells are detected by each grid cell. Despite the addition of so many layers, the design retains one of the best reaction times with the precision provided. It is particularly useful for analysing live video feeds because to its fast data unsampling and object recognition capabilities. This version has the best improvements in ML (Machine Learning) utilising neural networks.

The YOLO network is built on a systematic grid split of the provided picture. Te grids are classified into three categories, which will be discussed more below. These grids serve as a distinct picture for the algorithm and are further subdivided. YOLO employs boundaries known as bounding boxes. These are the anchors for picture analysis. These boxes are effectively accepted as results, despite the fact that hundreds upon thousands are discarded due to low likelihood ratings and are classified as false positives. DarkNet-19 was used in YOLOv2, however DarkNet-53 is now utilised in the newly modified YOLO model, where the 53 refers to the number of convolutional layers. Darknet 53 improves both speed and accuracy, making it 1.5 times faster. When compared to ResNet-152, this architecture achieves almost the same accuracy and precision while being twice as quick [37]. The YOLO model is seen in Fig. 3

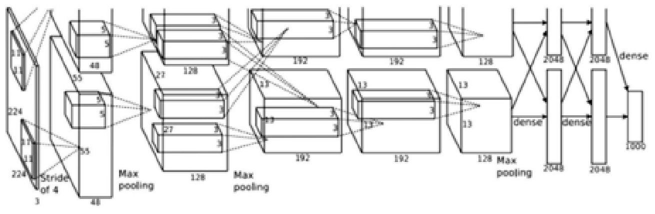
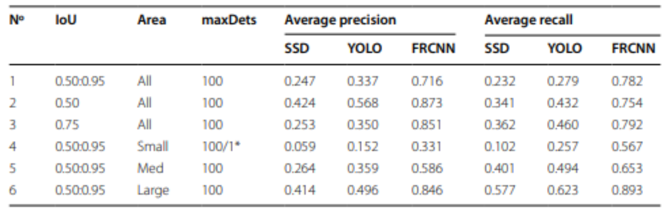


Fig. 3 CNN of the Krizhevsky model

ImageNet was created to target a broader category with a large number of different categories that were fine-grained. SUN took a more modular approach, with regions of interest chosen based on their frequency of occurrence in the data set. Finally, PASCAL VOC's approach to COCO was comparable yet distinct. It made extensive use of photographs from the surroundings and nature. Microsoft Common Things in Setting is designed to recognise and classify objects in their natural context.

Table 1 Google Colab with an integrated engine named Python 3



Nº	IoU	Area	maxDets	Average precision			Average recall		
				SSD	YOLO	FRCNN	SSD	YOLO	FRCNN
1	0.50:0.95	All	100	0.247	0.337	0.716	0.232	0.279	0.782
2	0.50	All	100	0.424	0.568	0.873	0.341	0.432	0.754
3	0.75	All	100	0.253	0.350	0.851	0.362	0.460	0.792
4	0.50:0.95	Small	100/1*	0.059	0.152	0.331	0.102	0.257	0.567
5	0.50:0.95	Med	100	0.264	0.359	0.586	0.401	0.494	0.653
6	0.50:0.95	Large	100	0.414	0.496	0.846	0.577	0.623	0.893

The software configuration used is Google Colab with an integrated engine named Python 3 Google Compute Engine Backend. It has a RAM capacity of 12.72 GB, of which 3.54 GB was utilised on average. It also has a storage space of 107.77 GB, of which 74.41 GB was used to store the training and validation datasets. The hardware accelerator utilised was a synthetic GPU provided by Google Colab. The precision-recall curves shown using the COCO metric, API, allowed us to make accurate conclusions regarding the efficiency with which these three models detect objects. Graphs were created for each model based on the size of the objects. The orange region represents the precision-recall curve with no mistakes, the violet area represents the items that were incorrectly identified, and the blue area represents the localisation errors (Loc). Finally, white areas beneath the precision-recall curve imply an IoU value larger than 0.75, whereas grey areas suggest an IoU value greater than 0.5.

VI. CONCLUSIONS AND FUTURE WORK

We suggested a cutting-edge deep learning method for object detection that is based on fog-clouds. Our solution offers a deployable framework for deep learning applications and offers various modes (high-accuracy and low-latency) depending on the target applications or user needs. To deploy and evaluate the performance of the suggested model, we used the Aneka platform service. In order to offer multiple modes of operation for various use cases, we compared many parameters for various fog circumstances, such as detection accuracy, reaction time, jitter, network, and power consumption.
In the coming work, we intend to expand the system to take into account a particular cost model to allocate and share resources or pre-process/resize input photographs based on the user's financial restrictions. There is still a lot of work to be done in this field in the future. Every year, new algorithms are released, as well as changes to current ones. Furthermore, each field—aviation, autonomous vehicles (aerial and terrestrial), industrial machinery, and so on—requires a different set of algorithms. These topics will be thoroughly investigated in the future.

VII. REFERENCES

[1] Islam, S.M.R., Kwak, D., Kabir M.D.H., Hossain M. and Kwak K.S.: The internet of things for health care: a comprehensive survey. IEEE Access. 3, 678-708 (2015).

[2] Afrin, Mahbuba, Md Redowan Mahmud, and Md Abdur Razzaque. "Real time detection of speed breakers and warning system for on-road drivers." 2015 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), pp. 495-498. IEEE, 2015.

[3] Tuli, Shreshth, Redowan Mahmud, Shikhar Tuli, and Rajkumar Buyya. "FogBus: A Blockchain-based Lightweight Framework for Edge and Fog Computing." Journal of Systems and Software (2019).

[4] Bonomi, Flavio, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. "Fog computing and its role in the internet of things." MCC workshop on Mobile cloud computing, pp. 13-16. ACM, 2012.

[5] Borji, Ali, Ming-Ming Cheng, Qibin Hou, Huaizu Jiang, and Jia Li. "Salient object detection: A survey." arXiv:1411.5878 (2014).

[6] Abeshu, Abebe, and Naveen Chilamkurti. "Deep learning: the frontier for distributed attack detection in fog-to-things computing." IEEE Communications Magazine 56, no. 2 (2018): 169-175.

[7] Vecchiola, Christian, Xingchen Chu, and Rajkumar Buyya. "Aneka: a software platform for .NET-based cloud computing." High Speed and Large Scale Scientific Computing 18 (2009): 267-295.

[8] Chen, Ning, Yu Chen, Sejun Song, Chin-Tser Huang, and Xinyue Ye. "Smart urban surveillance using fog computing." 2016 IEEE/ACM Symposium on Edge Computing (SEC), pp. 95-96. IEEE, 2016.

[9] Diro, Abebe Abeshu, and Naveen Chilamkurti. "Distributed attack detection scheme using deep learning approach for Internet of Things." Future Generation Computer Systems 82 (2018): 761-768.

[10] Li, Liangzhi, Kaoru Ota, and Mianxiong Dong. "Deep learning for smart industry: efficient manufacture inspection system with fog computing." IEEE Transactions on Industrial Informatics (2018): 4665-4673.

[11] Teerapittayanon, Surat, Bradley McDanel, and Hsiang-Tsung Kung. "Distributed deep neural networks over the cloud, the edge and end devices." 37th IEEE International Conference on Distributed Computing Systems (ICDCS), pp. 328-339. IEEE, 2017.

[12] Constant, Nicholas, Debanjan Borthakur, Mohammadreza Abtahi, Harishchandra Dubey, and Kunal Mankodiya. "Fog-assisted wiot: A smart fog gateway for end-to-end analytics in wearable internet of things." arXiv preprint arXiv:1701.08680 (2017).

[13] MIT App Inventor software. http://appinventor.mit.edu/appinventor-sources/. [accessed on 28-May-2019].

[14] Redmon, Joseph, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection." IEEE conference on Computer Vision and Pattern Recognition, pp. 779-788. 2016.

[15] Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement." arXiv preprint arXiv:1804.02767 (2018).

[16] COCO Dataset. http://cocodataset.org/#home. [accessed on 31-May-2019].

[17] Mircosoft Windows performance toolkit. https://docs.microsoft.com/ enus/windows-hardware/test/wpt/. [accessed on 30-May-2019].

[18] Microsoft Network Monitor 3.4. https://www.microsoft.com/enau/download/details.aspx?id=4865. [accessed on 28-May-2019].

[19] Han, Junwei, Dingwen Zhang, Gong Cheng, Nian Liu, and Dong Xu. "Advanced deep-learning techniques for salient and category-specific object detection: a survey." IEEE Signal Processing Magazine 35, no. 1 (2018): 84-100.

[20] Everingham, Mark, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. "The pascal visual object classes (voc) challenge." International journal of computer vision 88, no. 2 (2010): 303-338.

Cite this article as :

R.A. Arul Raja, V.G. Anisha Gnana Vincy, "Object Detection System Using Yolo Algorithm", International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET), Online ISSN : 2394-4099, Print ISSN : 2395-1990, Volume 9 Issue 5, pp. 295-300, September-October 2022.
Journal URL : https://ijsrset.com/IJSRSET229630