BAUET JOURNAL
Published by
Bangladesh Army University of Engineering & Technology
Journal Homepage: http://journal.bauet.ac.bd
https://doi.org/10.59321/BAUETJ.V4I1.6

Violence Classification and Detection Using Regional Data of Bangladesh: An AI Approach

Partha Pratim Debnath[1, 2*], Abrar Faiaz Adnan[2], Nazmul Hussain[2], Md. Abdul Hamid[3]

[1]*Department of ICE, University of Rajshahi, Rajshahi-6205, Bangladesh*
[1, 2]*Department of ICE, Bangladesh Army University of Engineering and Technology, Natore-6431, Bangladesh*
[3]*Department of ME, Bangladesh Army University of Engineering and Technology, Natore-6431, Bangladesh*

Abstract: In this research, the focus is to utilize object detection techniques to track and classify violence in public places in Bangladesh. Based on Google's TensorFlow, the TensorFlow Object Detection API is an open-source platform that has been utilized to train and evaluate an SSD-MobileNet-v2 model on a dataset specifically created for this purpose, containing images of violent activities in public places from 2000 to 2021. The model is fine-tuned with a training dataset for 4 classes and evaluated on a test dataset. The results show that the performance of detecting armed civilians is the highest with an accuracy of 54.7%. However, the other labels such as setting fire, vandalism and law enforcement were not very accurate and were omitted from the final model to maintain a higher precision for the armed civilian class. The proposed methodology and the obtained results could potentially aid other researchers in designing custom object detectors for similar datasets.

Keywords: *Violence Detection, Object Detection, Bangladesh, TensorFlow, Regional Dataset.*

Introduction: The human visual system is quick and precise, enabling us to carry out difficult activities like driving with little conscious effort. Robotic systems, assistive technology, and vehicles without specialized sensors can all use quick and precise object detection algorithms. Classifiers are currently used in detection systems to carry out detection by assessing the classifier for the object at various locations and scales in an image. A sliding window technique is used by systems like deformable parts models (DPM) [1], where the classifier is run at evenly spaced regions throughout the entire image. The region proposal is used in more modern methods like Fast R-CNN [2]. The research topic of vision-based action recognition, specifically detecting fights from surveillance cameras in public areas, is important for quickly controlling violent incidents. Violence detection has many practical use cases and has been receiving increasing attention as a research topic. The study focuses on detecting violent activities on public places, including a broad range of activities such as vandalism, explosion, and fighting. The study also mentions the World Health Organization's definition of violence as “The intentional use of physical force or power that results in injury, death, psychological harm, mal-development, or deprivation.” [3]

As much research is done on detecting violence worldwide, many parts of the world succeed in detecting violence and taking proper steps to handle the situation. Every country has its own system that can perform smoothly in its own environment. But using their system in Bangladesh showed different types of problems that decreased the accuracy rate of this system. So, a proper system is needed to identify violence with high accuracy that can work in any environment in Bangladesh.

Our main objective was to use TensorFlow’s [4] object detection API combining with CNN to detect violence through any videos. CNN [5] based approaches has proved to be very effective to classify videos if they’re violent or not and TensorFlow’s Object detection API identifies the position of the target label (violent behaviors) using bounding boxes so we can see the location of the places where violence is taking place. Violence differs from place to place, culture to culture. Usually in different scenarios in Bangladesh civilians have different clothes and they all differ from each other, so it was very hard to distinguish them from general civilians. We’ve tried to figure out how much effective our model was to distinguish them from each other. Also, the armed forces can also be falsely identified as they can also be armed with riot shields or other weapons to neutralize the angry mob.

This research aims to determine the effectiveness of TensorFlow’s Object Detection API in identifying violent behaviors in public spaces by using armed civilians as the target label using computer vision tools and algorithms.

Literature Review: Many research was done to develop a system which can detect social disturbance and violence through computer vision. In the research Violence Detection in Video Using Computer Vision Techniques which was published in Pittsburgh and robotics Institution, they tried to make this system by using STIP and MoSIFT [6]. In this research a violence can be detected faster than other model but it works for single person or small group in closed space. There was another research named Semantic context detection based on hierarchical audio models [7] where violence can be detected from video by using

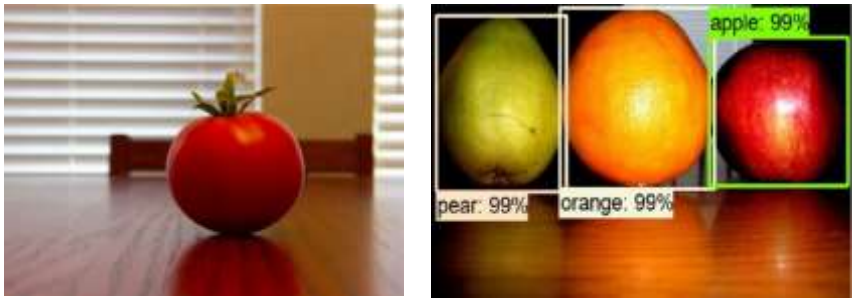
Article history:
Received 28 March 2023
Received in revised form 25 August 2023
Accepted 21 October 2023
Available online 15 November 2023

Corresponding author details: Partha Pratim Debnath
E-mail address: parthapratim.ice10@gmail.com
Tel: +880 1746-377565

audio based on gaussian mixture and hidden Markov models. These models increase the accuracy rate for small incident but also increase the computing time. All though there are many other models has already been used in this field, but none of the model and techniques are well suited for Bangladesh. In Bangladesh in any violence situation there are people who are committing crimes and there are bystanders, police, vendors, etc., so in an overcrowded place the accuracy rate to detect violence in that environment gets low by using a foreign system. So, there is a vital demand of a generalized model for violence classification and detection using regional data of Bangladesh. For this a new dataset of violent images from Bangladesh was collected and made publicly available for regional use.

Methodology: The object detection method mainly consists of three steps:

- From an input image, region recommendations are created in the first step.
- In the following stage, visual characteristics are extracted given the area proposals, and these features help to recognize objects in a picture. Region proposals, also referred to as regions of interest (RoI), are a huge set of bounding boxes formed by scanning the whole input image.



(a) (b)

Figure 1: Distinction between object detection and image categorization where image (a) shows classification on Tomato and image (b) shows Object detection on Apple, Orange, and Pear.

Non-maximum suppression's last stage involves grouping substantially overlapping boxes into a single box. We have utilized an SSD model in this instance. Single Shot Detector (SSD) is a region-based object detection technique that Liu et al. [8] introduced in 2016. It is quicker and a good option for real-time detections since it can forecast the bounding box and class of an object at the same time in a single shot. A shortened version of VGG16, an image classification architecture that has been expanded with additional feature layers, serves as the foundation of SSD. In order to create several convolutional feature maps of various sizes, an input image is passed through all of the convolution layers. Following that, the bounding boxes and class probabilities for the items in the image are predicted using these feature maps. Using numerous feature maps of various sizes allows SSD to recognize objects of various scales in a single image, which is one of its primary characteristics. Moreover, anchor boxes—pre-established bounding boxes used as a guide for anticipating the ultimate bounding boxes—help to increase the detection's precision.

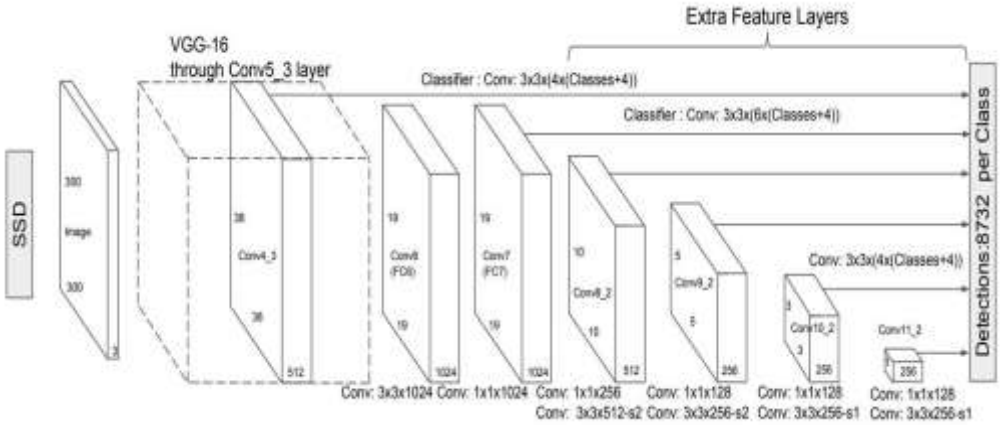


Figure 2: Single Shot MultiBox Detector Architecture (input is 300x300x3).

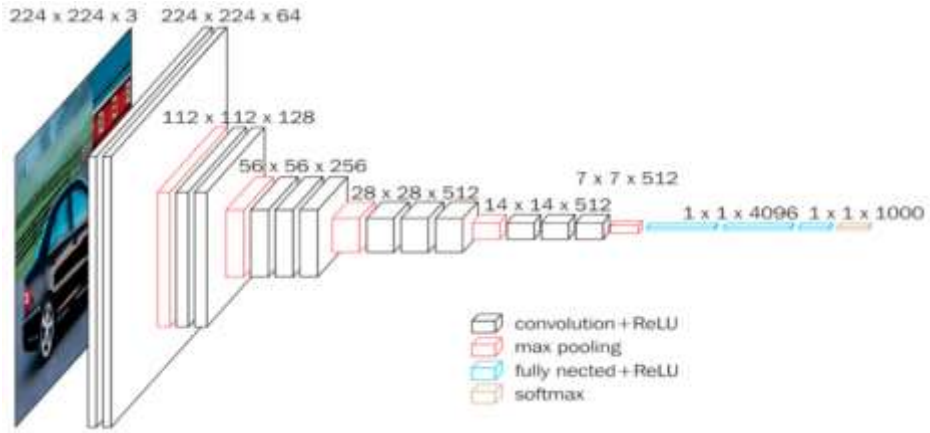


Figure 3: Architecture of VGG (input is 224x224x3).

Figure 2 shows how the architecture of SSD is based on the time-tested VGG-16 architecture but does away with the completely connected layers. VGG-16 was chosen as the basic network due to its good performance in jobs requiring the categorization of high-quality images and its widespread application in issues where transfer learning can aid to improve outcomes. A number of auxiliary convolutional layers (starting with conv6) were added in place of the initial VGG fully connected layers, allowing for the extraction of features at various scales and a gradually smaller input size for each succeeding layer.

Overall, SSD is a fast and efficient object detection algorithm that is well suited for real-time applications. Its use of a truncated VGG16 base network and multiple feature maps of varying sizes makes it a robust and accurate method for detecting suspicious objects and activities in images. Our application-specific object detectors are made feasible by open-source object detection frameworks including Google's TensorFlow Object Detection API, OpenCV's DNN module, and Microsoft Cognitive Toolkit (CNTK). The pre-trained object detection models offered by the open source frameworks can also be simply adjusted for our datasets.

The pre-trained models that these open-source frameworks provide to start the fine-tuning process are their most intriguing aspect. Transfer learning refers to the process of optimizing a model that has already been trained. Many pre-trained models for image classification are available from Microsoft Cognitive Toolkit [9], an open-source framework for constructing and training deep convolutional neural networks. These models were developed using data from the ImageNet database and the CIFAR-10 dataset [10]. Moreover, CNTK offers two Fast R-CNN pre-trained models for object detection that were trained on the PASCAL VOC 2007 and a grocery dataset7. Moreover, CNTK offers recipes for building a unique object detector by combining Fast R-CNN or Faster R-CNN with already-trained image classification models. Nevertheless, ResNet is not yet supported by CNTK as a basis model for the deployment of object detectors8. Moreover, the object detection model for mobile devices, MobileNet-v1/v2, does not even support SSD with CNTK.

Google's TensorFlow offers a number of pre-trained models for object detection that were developed using datasets like COCO, Kitti, and Open Images as well as a variety of pre-trained models for picture classification that were developed using the ImageNet database. The TensorFlow Object Detection API works with image classifiers like MobileNet-v1, MobileNet-v2, Inception-v2, and ResNet-101 to support object identification models like SSD, Faster R-CNN, and R-FCN. The key benefit of utilizing TensorFlow is that it supports the SSD model with MobileNet-v1/v2 and allows users to build object detectors that are appropriate for embedded vision applications and mobile devices.

Because to its ubiquity, usability, and well-documented code base, we chose TensorFlow Object Detection API to build an object detector for study.

Experimental Setup

System Environment Creation: A suitable system creation depends on proper hardware and software installation which has been described properly in this section.

Hardware Setup: To run head, face detection and random movement of body pixel detection in neural network field programs we need a computer with good processing speed. In our experiments we have used a laptop with the following specification-

Table 1. Laptop Specification.

Model	ACER E5-575G-30KT
RAM	12 GB
Processor	2.00 GHz
Hard drive	1 TB
Display	15.0 inch
GPU	NVIDIA GTX 940MX

Software Setup

We need the following software and library packages to run the program-

- Anaconda 3 version 2021.05.
- TensorFlow version 2.4.
- Opencv-python version 4.5.2.54.

For scientific computing (machine learning applications, data science, predictive analytics, large-scale data processing, etc.), Anaconda is a free and open-source [11] distribution of the Python and R programming languages that promises to streamline package management and deployment. Conda, a package management system, controls package versioning. [12] For Windows, Linux, and MacOS, the Anaconda distribution provides data-science packages.

The Google TensorFlow Object Detection API is an open-source object detection framework built on TensorFlow that enables users to create, train, and apply object detection models. Data flow graphs are used in the TensorFlow open-source software library, which was created by the Google Brain Team [13,14]. Nodes and edges are two crucial elements of data flow graphs. The mathematical processes are represented by nodes, while the multidimensional arrays that flow between the nodes are represented by edges. Applications using deep learning are frequently developed using TensorFlow [15, 16]. TensorFlow also includes a tool called TensorBoard for detailed model viewing during training, which is an intriguing feature. TensorBoard offers a web interface for computational graph visualization so users can see how a model's performance and parameters change over time [17]. As previously reported, the TensorFlow Object Detection API offers a variety of pretrained models on several datasets, including the SSD model with MobileNet, the Faster R-CNN model with ResNet, and the R-FCN model with ResNet. We have the option to commence our training with one of the pre-trained models to fine-tune our particular object detectors. The pre-trained model that is chosen depends on the goal of our application. The API also provides information on how various object identification techniques trade off speed and accuracy.

A collection of programming functions called OpenCV-python (Open-source computer vision) is primarily focused on real-time computer vision. [18,19] It was first created by Intel and afterwards sponsored by Willow Garage and Itseez (which was later acquired by Intel). Under the terms of the open-source BSD license, the library is free to use and cross-platform.

Parameter List: In an attempt to use realistic and typical values, the following parameters were used. Using these parameters and their corresponding values, we conducted all of our experiments.

Table 2. List of parameters of our model.

Parameter	**Parameter Value**
Distance from the camera	Variable
Lighting condition	200 Lux or 150 Lux
Age of the target individuals	15-45
Size of images	20KB - 2MB
Input image of the model	640x640
TFOD Model	SSD Mobilenet V2

Data Collection and Experiments: Generally, gathering labeled data to train image classifiers is one of the more difficult elements of a deep CNN. Generally speaking, supervised learning necessitates a large number of annotated training instances. There are an increasing number of publicly available datasets to suit the demand for huge training data and benchmark evaluation. The accessible datasets for computer vision comprise substantial collections of photos along with other details like annotations, segmentation masks, or other contextual information. Since there are numerous such datasets available for training and validation, choosing the one that best suits our needs is crucial. But we were focusing on Bangladeshi regional dataset but all the dataset contained data of other countries. Therefore, we made our own dataset by collecting images from different newspaper sources and we made sure that all the pictures in the dataset are of Bangladeshi people.

We first extracted the image and bounding box datasets for one class, namely "armed civilian," in order to train and test our object detection algorithms. The following are a few of the problems we ran into with the retrieved datasets:

- The datasets that were extracted include photos of various sizes. So, we must ensure that the graphics used for instruction and assessment are not too small (i.e., at least 300 pixels).
- Another issue with the generated datasets is that only the individuals in front of the camera—not everyone in the crowd—have their bounding boxes labeled for our target class. For instance, only the armed individuals at the front of a mob are identified. In spite of the obligation to annotate every image with every possible class, we decided to exclude some examples because they were noisy data and would only weaken the model.
- In addition, there aren't enough bounding box samples expected for each class. We need at least a few hundred bounding box samples for each class for training, validation, and evaluation reasons. Just 263 bounding box samples from 232 photos are present in our dataset.

Figure 4: Sample of training images from the dataset.

This is because data collection required us to rely on internet searches while the job was being done during the coronavirus shutdown. Considering the aforementioned difficulties, we pre-processed our unique dataset of armed citizens suitable for training and validation in the manner described below:

- Dataset selection and annotation. For our chosen class, we chose and annotated about 263 samples. To strengthen the model, we first intended to include other classes like fire and police, however when we trained the model with additional classes, the model's overall accuracy declined. So, we only sent armed civilians forward. The selecting procedure is quite straightforward; however, the manual annotation of the photographs took a lot of effort. We used LabelImg, a Python-based annotation tool for labeling graphical pictures, to annotate the photos. In PASCAL VOC, the bounding box coordinates and label annotations are recorded as XML (Extensible Markup Language) files (Visual Object Classes).
- Produce test and training datasets. Each class dataset was then split into a training dataset and a test dataset when the annotation procedure was complete. Each test dataset has roughly 80 labels, and the training dataset has about 255 labels.
- Produce a TFRecords. The training and validation datasets are translated into the TFRecord format in order to fine-tune a pre-trained TensorFlow object detection model. TensorFlow supports the standard format TFRecord when training object detection models. The training and validation datasets are first in XML format; they are then translated to CSV (Comma Separated Values) files and finally to a TFRecord format.

Performance Evaluation: The data are collected from internet and newspapers from year 2000 to 2021. The pictures include public protests or regional clashes between two factions in some districts and some religious clashes with the authority.

Results are obtained through the Tensor board application that comes preinstalled with TensorFlow. It shows the precision and recall of the model compared with the test images and also, we have the accuracy of the detections.

This section discusses the performance results of different pre-trained object detection models. Table 3 summaries the results of the pre-trained models and shows that our pre-trained model had a very fast speed and mAP which made it ideal for our research purpose. The accuracy measurement has been taken using the COCO [20] dataset. A sizable object detection, segmentation, and captioning dataset is called COCO. COCO offers a variety of characteristics:

- Object segmentation
- Recognition in context
- Superpixel stuff segmentation
- 330K images (>200K labeled)
- 1.5 million object instances
- 80 object categories
- 91 stuff categories
- 5 captions per image
- 250,000 persons with keypoints
- 330K images (>200K labeled)

This dataset is used as a benchmark for different models to measure their accuracy and speed.

Table 3. Speed and accuracy of different pre-trained models.

Models	Speed (ms)	COCO mAP
SSD MobileNet V1 FPN 640x640	48	29.1
SSD MobileNet V2 FPNLite 320x320	22	22.2
SSD MobileNet V2 FPNLite 640x640	39	28.2
Faster R-CNN ResNet101 V1 640x640	55	31.8
Faster R-CNN ResNet101 V1 1024x1024	72	37.1
Faster R-CNN ResNet101 V1 800x1333	77	36.6
Faster R-CNN ResNet152 V1 640x640	64	32.4
Faster R-CNN ResNet152 V1 1024x1024	85	37.6
Faster R-CNN ResNet152 V1 800x1333	101	37.4
Faster R-CNN Inception ResNet V2 640x640	236	38.7

As was already said, the SSD model is the fastest of all the models since it takes the least time to draw conclusions. Also, as predicted, the Fastest R-CNN models had the greatest mAP values, making them the most accurate of all. In comparison to the SSD model, Faster R-CNN is also considerably more accurate. Nonetheless, the SSD model is considerably faster than the Faster R-CNN model in terms of performance. In conclusion, an object detector's speed is a crucial consideration for real-time applications. Consequently, we chose SSD-MobileNet-v2 as our preferred model for our application in order to meet our goal of detecting armed citizens.

The performance of detection is expressed as follows:

Accuracy= $\frac{Total\ number\ of\ detected\ images\ as\ true\ positive}{Total\ number\ of\ given\ images\ for\ detection}$ x100% …………………………………….[21]

The accuracy of different label detection has been given as follows -

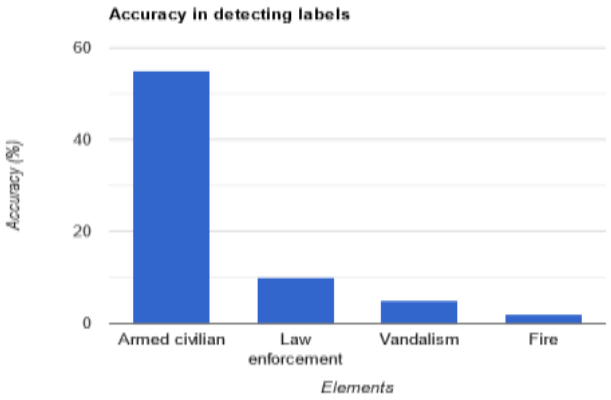
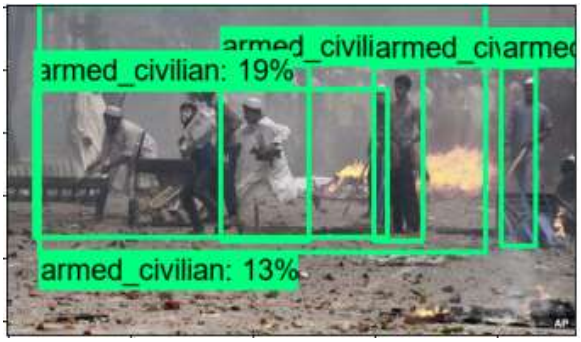


Figure 5: Accuracy of detected labels.

Below are the images that the model processed and gave outputs.



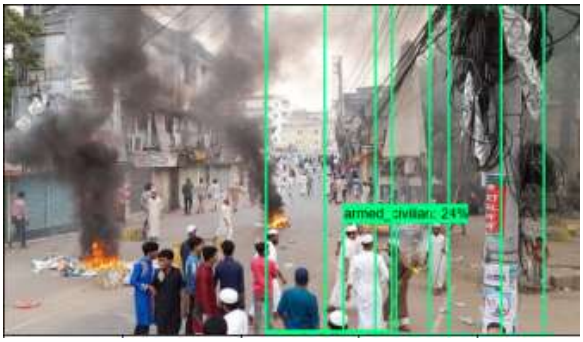


Figure 6: Output images of our model.

From these images we can see that the model gives an acceptable output compared to the small dataset we have. Sometimes it can't identify the obvious labels but sometimes it performs well. We did have some random detections and sometimes multiple detections at the same region. We can see that the confidence percentage of the detections are quite low. With some better dataset we are confident that it'll give a considerably better output. Currently our accuracy was at 54.7% for "armed_civilian", 10.7% for "law_enforcement", 5.2% "vandalism" and 2.6% for "fire". It is expected that it'll be greater when we train with a more diverse dataset with clearer quality.

Conclusion: The focus of the thesis work was to detect violent people from public places by integrating technologies of machine vision and machine learning. In the first chapter we have described the research summery extracted from our experimental results. Different experiences related to the implementation of our research works and a short description about our future work has also been included in this chapter.

The research aims to detect violent behaviors in public places in Bangladesh. A dataset of images was collected from newspapers and search engines, and the most common class selected was armed civilians, as it had the most labels. Due to Covid-19 restrictions, the data was collected online and the media in Bangladesh does not publish violent images with dead bodies or bloody images. After pre-processing, the dataset was reduced to 27 test and 89 train images. The SSD model was chosen for training due to its less training time and ability to detect labels with less data. The model was trained multiple times and tested with the labelled dataset, and it was found to have fair accuracy in detecting vandalism. The study analyzed psychological research papers related to violence to identify the crucial characteristics of suspects in social space. Overall, the research demonstrates the potential for using machine learning to detect violent behaviors in public places, although further research may be needed to improve accuracy and obtain more data.

References:

[1] T. Mordan, N. Thome, M. Cord, G. Henaff, Deformable Part-based Fully Convolutional Network for Object Detection, Computer Vision and Pattern Recognition, 15 (2017) 1200-1210.
[2] X. Xie, G. Cheng, J. Wang, X. Yao, Xiwen, J. Han, Oriented R-CNN for Object Detection, In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), October, 2021.3520-3529.
[3] Wikipedia contributors. "Violence.": Wikipedia, The Free Encyclopedia, June, 2021. Web. 4 Jul. 2021.
[4] I. Zafar, G. Tzanidou, R. Burton, N. Patel, L. Araujo, Hands-On Convolutional Neural Networks with TensorFlow: Solve computer, first ed, Packt Publishing Ltd, UK, 2018.
[5] R. Chauhan, K. K. Ghanshala and R. C. Joshi, Convolutional Neural Network (CNN) for Image Detection and Recognition, In: Proceedings of First International Conference on Secure Cyber Computing and Communication (ICSCCC), Jalandhar, India, 2018, p. 278-282.
[6] E.B Nievas, O.D Suarez, G.B Garcia, & R Sukthankar, Violence detection in video using computer vision techniques, In: Proceedings of International Conference on Computer Analysis of Images and Patterns, USA, (2011)332–339.
[7] W.H Cheng, W.T Chu, J.L Wu., Semantic context detection based on hierarchical audio models, In: Proceedings of the 5th ACM SIGMM international workshop on Multimedia information retrieval, November, (2003) 109-115.
[8] A. Kumar, S. Srivastava, Object Detection System Based on Convolution Neural Networks Using Single Shot Multi-Box Detector, Procedia Computer Science, 171 (2020) 2610-2617.
[9] F. Seide and A. Agarwal. CNTK: Microsoft's open-source deep-learning toolkit. In: Proceedings of 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM,(2016). 2135–2135.
[10] K. Wang, B. Zhao, X. Peng, Z. Zhu, S. Yang, G. Huang, H. Bilen, CAFE: Learning To Condense Dataset by Aligning Features, In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), (2022), 12196-12205.
[11] R.M. May, K.H. Goebbert, J.E. Thielen, J.R. Leeman, M.D. Camron, Z. Bruick, E.C. Bruning, R.P. Manser, S.C. Arms, P.T. Marsh, MetPy: A Meteorological Python Library for Data Analysis and Visualization, Bulletin of the American Meteorological Society 103.10 (2022), 2273-2284.
[12] O. Snihovyi, O. Ivanov, & V. Kobets. Cryptocurrencies prices forecasting with anaconda tool using machine learning technique,. In: Proceedings of CEUR Workshop, (2018), 453-456.

[13] L. Bai, T. Zhao and X. Xiu, Exploration of computer vision and image processing technology based on OpenCV, In: Proceedings of International Seminar on Computer Science and Engineering Technology (SCSET), Indianapolis, IN, USA, (2022),145-147.
[14] C. Le, T.K. Mohd, Tauheed, Facial Detection in Low Light Environments Using OpenCV, (2022), 624-628.
[15] G. Bradski, & A. Kaehler, Learning OpenCV 3: Computer vision in C++ with the OpenCV library. fourth ed., O'Reilly Media, Inc, 2016.
[16] D. Paper. State-of-the-Art Deep Learning Models in TensorFlow, First ed, Apress Berkeley, CA, 2021.
[17] A.S. Ali, M.E. Abdulmunem, Image classification with Deep Convolutional Neural Network Using TensorFlow and Transfer of Learning, Journal of the College of Education for Women, 31 (2020) 156-171.
[18] S.A. Sanchez, H.J. Romero, A.D. Morales, A review: Comparison of performance metrics of pretrained models for object detection using the TensorFlow framework, In: Proceedings of IOP Conference Series: Materials Science and Engineering, May, (2020), 012024.
[19] C. Contoli, Lattanzi, A Study on the Application of TensorFlow Compression Techniques to Human Activity Recognition, IEEE Access, 11 (2023) 48046-48058.
[20] S. Jain, S. Dash, R. Deorari, Kavita, Object Detection Using Coco Dataset, In: Proceedings of International Conference on Cyber Resilience (ICCR), Dubai, United Arab Emirates, (2022),1-4.
[21] A.F Gad, Accuracy, precision, and recall in Deep learning | Paperspace blog. Paperspace Blog, (2021). https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/