Original Article

Face Mask Detection

[1] Kalpana Yadav , [2]Heena Khatun ,[3] Shivani Malik , [4] Mohd Imran , [5]Mukesh Rawat

[1-5]Department of Computer Science
Meerut Institute Of Engineering And Technology, Meerut
1 kalpana.yadav.cs.2019@miet.ac.in , 2heena.khatun.cs.2019@miet.ac.in
3shivani.malik.cs.2019@miet.ac.in , 4mohd.imran.cs.2019@miet.ac.in ,5mukesh.rawat@miet.ac.in
DOI: 10.47750/pnr.2022.13.S10.668

Abstract

This project FACE MASK DETECTION aims to create a system which identify if mask is there or not on a person face. This works on Transfer Learning. This system can easily be integrated into embedded devices as it uses MobileNetV2 architecture. It will detect Face Mask in real-time videos as well as in images. In the past few months, Covid-19 has affected each and every one. The main point of worry is of rising the graph of the total number of DEATHS. The utilization of computer vision and deep learning in this product aims to address safety concerns in the real world, making a substantial impact by implementing effective safety measures. This project can be used in public places and at high-traffic areas such as Railway stations, Airports, offices, schools, etc. To identify if mask is there or not on a person face and is following rules and regulations. The system detects as NO MASK if a person is does not have mask on his face.

1 Introduction

In this covid-19 scenario as the number of cases is rising at a very fast rate and it is required that people should follow rules and regulations to keep them safe from Covid-19 virus. For the solution of this problem, a simple approach of FACE MASK DETECTION project is created that detects identify if mask is there or not on a person face [1][2].

This system is in need of the hour as India tries to battle the novel coronavirus that has infected more than 1 crore and has caused more than 1 lakhs deaths with the figures still increasing at a rapid pace [3][4].

For the implementation, OpenCV, Keras /TensorFlow and Deep Learning concepts are used in order to identify if mask is there or not on a person face in images and real videos. Also, for making it more reliable MobileNetv2 architecture is used. By using this architecture it can be easily integrated into embedded devices [5][6].
The main objectives of this project are is to train a custom model to identify if mask is there or not on a person face and to create a unique dataset of with mask images by utilizing various resources, such as the Bing search API, relevant datasets available on Kaggle, and the RMFD dataset. Datasets will be divided into two classes: 'With mask dataset' and 'No Mask dataset'. The face mask detector model is trained on the custom dataset using Keras and TensorFlow. The trained model is implemented to identify if mask is there or not on a person face

The model is expected to give an accuracy of 90% and above.

2 Related Work

• FebriEye, a thermal camera, comes with additional analytics such as face mask and social distancing monitoring system which generates an alarm in case of any violations. It is being developed by Vehant Technologies which is to be implemented by the Telangana government. [13][10]

• Uber has confirmed to build a business system which identify if mask is there or not on a driver or passenger face

and following rules and regulations or not. [11]

• Face Mask Alert app which is in development process by LeewayHertz software solutions. It sends an alert to the users enforcing them to wear masks. [12]

• AIZOOTech face mask detection system which uses dataset composed of WIDER Face and MAFA, but lacks landmark net for the purpose of face alignment [7][8].

2.1 Literature Review :

This Face mask detection works on the concept of deep learning and uses Mobilenetv2 architecture for it's faster implementation. Previous researches based on object detection concept has made this project more successful [9][10].

Applying the concept of object detection further to detect faces and after detecting the faces it detects mask on the faces. The accuracy achieved is more than 90% while the training time of model depends on the GPU used.

Further, the concept of openCV make it possible to detect and to use the trained model for identify if mask is there or not on a person face. Google teachable machine can also be used for the same purpose but it is not accurate as compared with this model [11][12].

3 Proposed Methodology

The Face Mask detection project is divided into two main phases.

Phase 1: Training the model [13][14].

- Dataset which will be loaded from the disk
- The model will be trained using TensorFlow.
- The mask detector is to be serialized back to the disk

Phase 2: Deployment

- The trained model has to be loaded
- The requirement is to be fulfilled i.e. identify if the mask is there or not on a person's face
- Classifying With mask and without a mask with the help of trained model

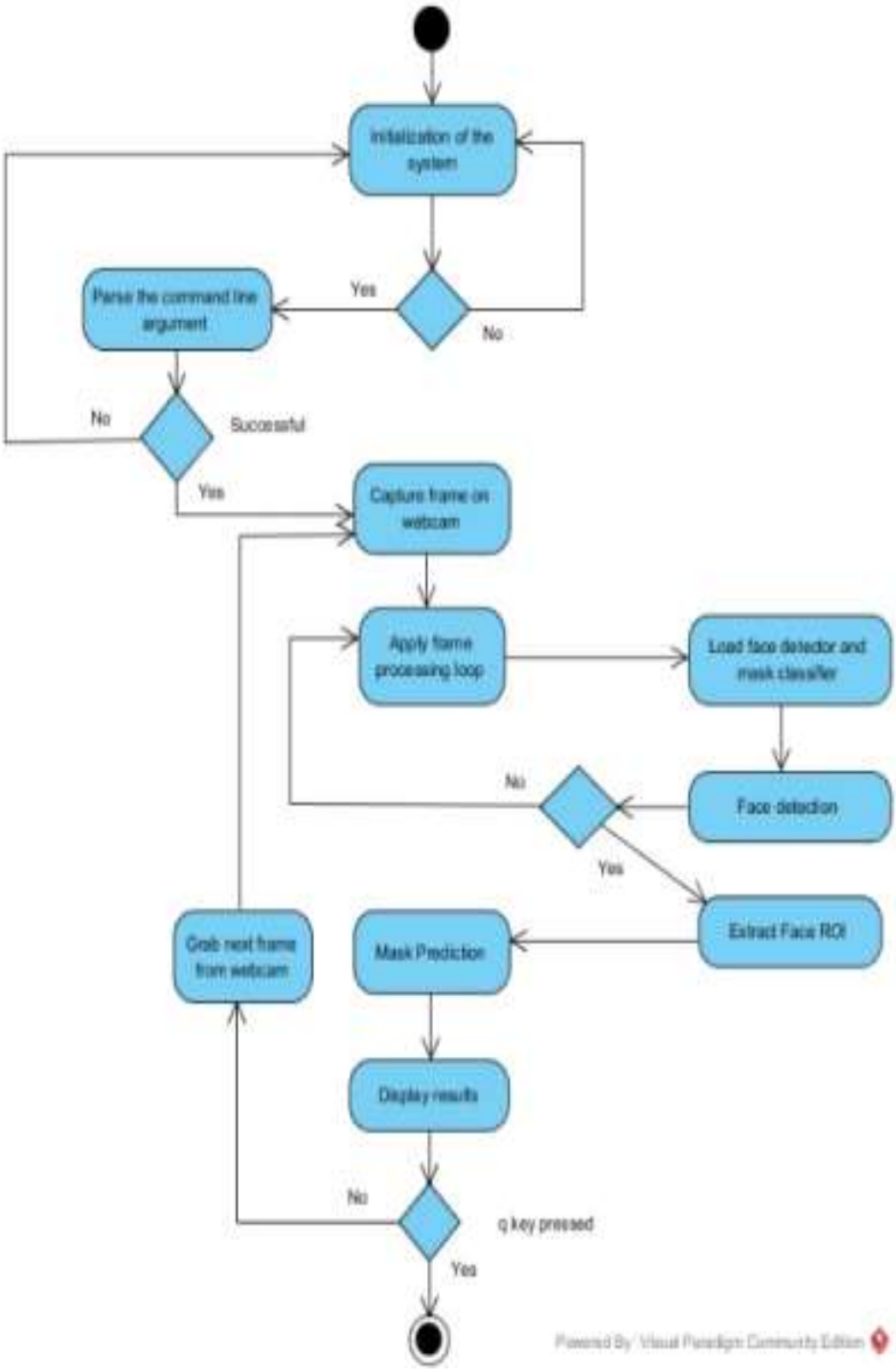

Fig1. General Architecture/DFD

3.1.1 "Deep Neural Networks" of OpenCV: OpenCV has a powerful dnn module supported by many deep learning tools such as Caffe , TensorFlow, and Torch/PyTorch.In this project, we will be training our deep learning model using Caffe, hence we need the following files:

The text file protype (text file) which gives the layers as well as architecture.

The caffe – model file which stored the model and have actual layer weights.

3.1.2 One Shot Multi-Box Detector (SSD) framework integrated with OpenCV's MobileNetV2 face detector architecture: for getting the coordinates of the box around the faces we will apply the concept of object detection [5][16]

Google introduces SSD which gives out the balance between YOLO and R-CNNs methods for the object detection. MobileNETv2 and the SSD Framework (Single shot detector) together will give a fast and efficient method for the object detection [17][18].

3.1.3 OpenCV's blobFromImage and blobFromImages support image pre-sets:

The cv2 blob from image abd cv2 blob from images which are the function of Dnn module. Helps for the pre processing and for the A taxonomy of deep learning. These two functions work**:**

Mean subtraction – take out some changes which are related with the illumination. Scaling - The scaling factor aids in normalization and optionally channel swapping

[blobFromImage] helps to create 4-d blob from image and also crop and resizes the images from the center, and also swap the red and blue channels.

Blob = image , scalefactor will be set to 1.0 and swapRB will be true

Each parameter is explained as follows**:**

- **image:** the input images which is help to train out our model for further use.
- **size:** The size will be according the which the CNN expects. We will be using 224×224.
- **swapRB:** By default the swapping of RB is done by OpenCV itself also it uses mean value in RGB order.
- The cv2.dnn.blobFromImages function is exactly the same:

Blob = image , scalefactor will be set to 1.0 and swapRB will be true WE can pass multiple iamges in a batch.

3.2 Model Summary :

Model : "model"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 224, 224, 3)	0	
Conv1_pad (ZeroPadding2D)	(None, 225, 225, 3)	0	input_1[0][0]
Conv1 (Conv2D)	(None, 112, 112, 32)	864	Conv1_pad[0][0]
bn_Conv1 (BatchNormalization)	(None, 112, 112, 32)	128	Conv1[0][0]
Conv1_relu (ReLU)	(None, 112, 112, 32)	0	bn_Conv1[0][0]
expanded_conv_depthwise (Depthw	(None, 112, 112, 32)	288	Conv1_relu[0][0]
expanded_conv_depthwise_BN (Bat	(None, 112, 112, 32)	128	expanded_conv_depthwise[0][0]
expanded_conv_depthwise_relu (R	(None, 112, 112, 32)	0	expanded_conv_depthwise_BN[0][0]
expanded_conv_project (Conv2D)	(None, 112, 112, 16)	512	expanded_conv_depthwise_relu[0][0
expanded_conv_project_BN (Batch	(None, 112, 112, 16)	64	expanded_conv_project[0][0]
block_1_expand (Conv2D)	(None, 112, 112, 96)	1536	expanded_conv_project_BN[0][0]

block_1_expand (Conv2D) (None, 112, 112, 96) 1536 expanded_conv_project_BN[0][0]

block_1_expand_BN (BatchNormali (None, 112, 112, 96) 384 block_1_expand[0][0]

.

.

.

.

block_16_expand_relu (ReLU) (None, 7, 7, 960) 0 block_16_expand_BN[0][0]

block_16_depthwise (DepthwiseCo (None, 7, 7, 960) 8640 block_16_expand_relu[0][0]

block_16_depthwise_BN (BatchNor (None, 7, 7, 960) 3840 block_16_depthwise[0][0]

block_16_depthwise_relu (ReLU) (None, 7, 7, 960) 0 block_16_depthwise_BN[0][0]

block_16_project (Conv2D) (None, 7, 7, 320) 307200 block_16_depthwise_relu[0][0]

block_16_project_BN (BatchNorma (None, 7, 7, 320) 1280 block_16_project[0][0]

Conv_1 (Conv2D) (None, 7, 7, 1280) 409600 block_16_project_BN[0][0]

Conv_1_bn (BatchNormalization) (None, 7, 7, 1280) 5120 Conv_1[0][0]

out_relu (ReLU) (None, 7, 7, 1280) 0 Conv_1_bn[0][0]

average_pooling2d (AveragePooli (None, 1, 1, 1280) 0 out_relu[0][0]

flatten (Flatten) (None, 1280) 0 average_pooling2d[0][0]

dense (Dense) (None, 128) 163968 flatten[0][0]

dropout (Dropout) (None, 128) 0 dense[0][0]

dense_1 (Dense) (None, 2) 258 dropout[0][0]

==========

Total params: 2,422,210
Trainable params: 2,388,098
Non-trainable params: 34,112

Fig 2. Model Layers

This is the output of model.summary() len(model.layers) = 160

3.3 Training the model with Keras and TensorFlow :

The model will be trained with the help of keras and tensorflow with concepts of deep learning. The parameters which are needed to train out the model includes Batch size , Rate of learning and Epochs [19][20].

Partitioning of the dataset: using split method (sklearn) we will use 80% for the training purpose and rest for the testing purpose. So that it can yield good accuracy [21][22].

prepare for data augmentation: for improving out the accuracy we will apply the data mutation methods on the images. The image data generator from keras accepts data and transform it randomly. It includes out the parameters such as zoom and rotatio of the image [23][24].

We employed the Adam optimizer to compile the model and implemented a time-based learning rate scheduler with the 'decay' parameter. As the classification task involved two classes, we utilized the 'binary cross entropy' loss function. To train the top layers of the neural network, we utilized an 'aug' object that supplied augmented data in batches."

Also, we need to keep in mind that we should freeze out the base layers so that they will not updated in the process of backpropagation [25].

	Precision	**Recall**	**F1-Score**	**Support**
Mask on face	**0.99**	**0.99**	**0.99**	**138**
No mask on face	**0.99**	**0.99**	**0.99**	**138**
Accuracy			**0.99**	**276**
Macro average	**0.99**	**0.99**	**0.99**	**276**
Weighted average	**0.99**	**0.99**	**0.99**	**276**

Fig 3. Demonstration of the accuracy of trained model

Fig 4. Training Loss and Accuracy Graph

Now , we have the graph that is saved in an image file. From graph we can see that there are very less chances of overfitting as curves demonstrates the high accuracy.

Now, we are ready to use our model to identify if mask is there or not on a person face and can achieve our main objective.

3.4 Implementing model for static images with OpenCV :

For the purpose to loading and using the model for the purpose we will require keras library import and for displaying out the result we will use OpenCV.

The frisr step will be getting out the dimesnsions of the image for the display purpose. When blob from image method has done the pre processing part for the image then we can perform the detection part from the model. After , this box dimension must be computed and we shloud ensure that it has to fall within the image bounadary.
Now the part comes when we need to display the results in the static images. We will first extract the Region of interest of the face with the help of slicing which can be achieved by using Numpy library. Now, it is the time to give color for a valid detection with mask and red colour with the invalid detection i.e without any mask on the face. By, using Opencv we can now draw out the boundaries around the face which will show the final result. The output of this step is shown in below images .

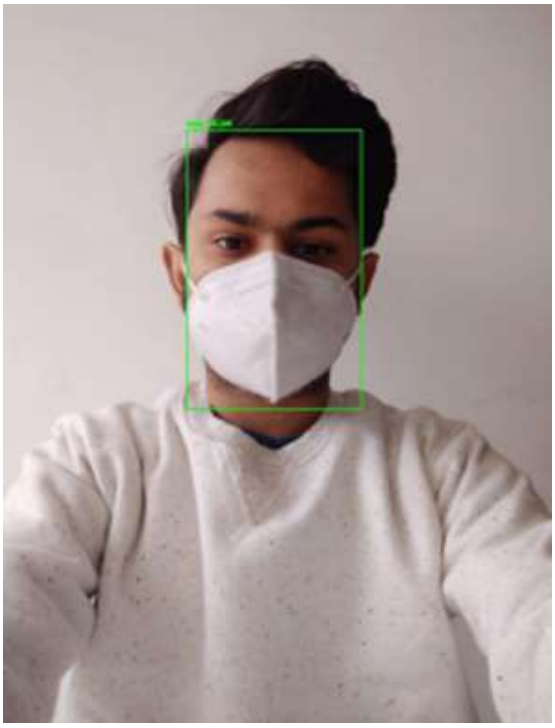

Fig 5. Face with mask

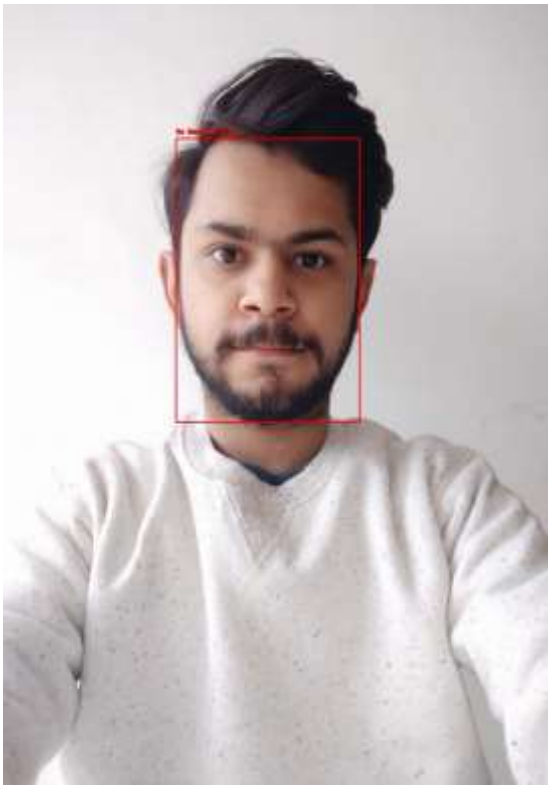

Fig 6 Face with No mask

3.5 Implementing model for Real time detection with OpenCV :

We first need to import out the video stream library for using in real time detection. Further we can imutils library in

python for the resizing purpose.

We now have to go for the construction of the blob that will be used for the purpose of detection purpose including the grace location , region of interest and the predictions of whether the mask is on face or not.

We need to compute out the boundary of the boxes and need to take the notice that it should fit in the viewing frame.

We need to apply the same process on the batches of the faces that lies in the frame so that we can make the pipeline more faster and accurate. After that we need to return out the boundaries of the box locations and the prediction which have been done.

Finally, the frame is displayed and the goal has been achieved.

The complete steps are displayed by using the below flow diagram :

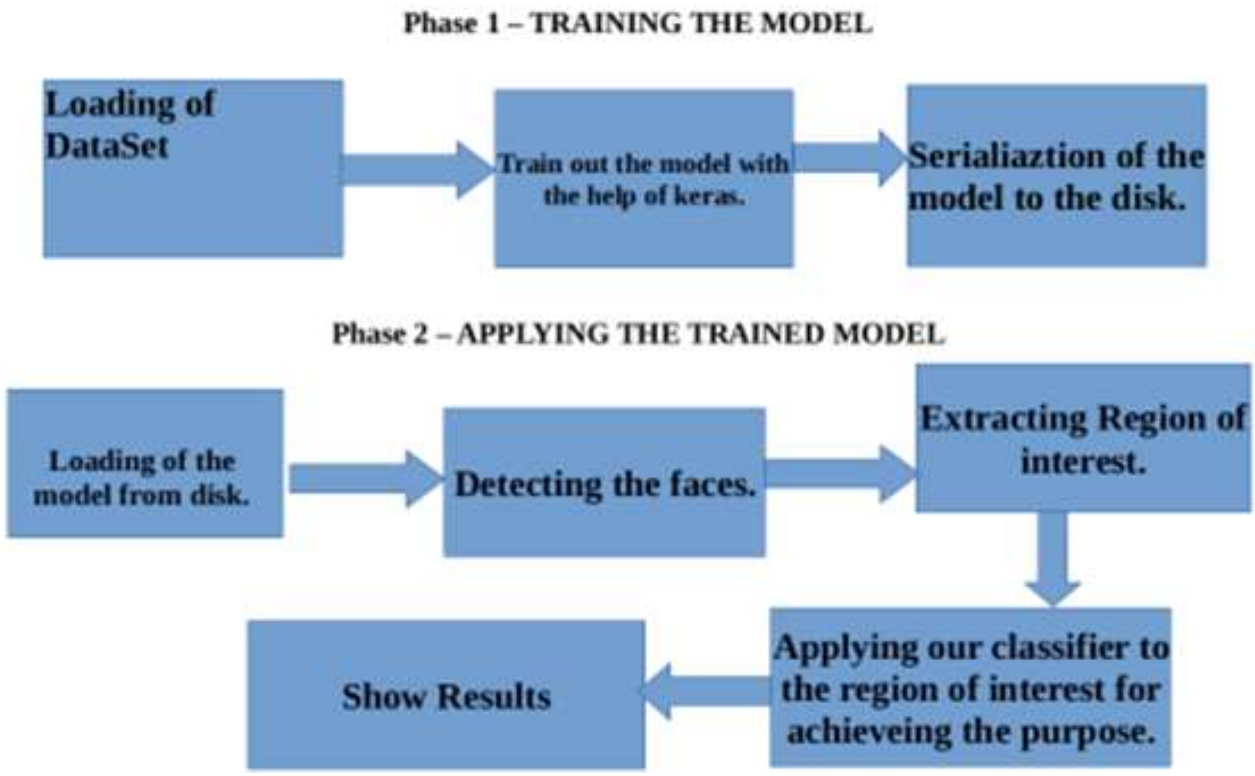

Fig 7. Flow Diagram of complete Implementation

4 Conclusion :

It is built on the MobileNetV2 architecture using accurate face detection, adaptive and deep learning. Create custom data using Kaggle, Bing Search API, and the RMFD dataset. After training the
Model, it achieves approximately 94% accuracy in detecting facial expressions in both static and live streams.
The classifier identify if mask is there or not on a person face :

- Detecting if mask is there or not on a person face
- Region of interest extraction.

- Model apply.

This system can be integrated in embedded devices such as Raspberry P , google coral etc. Therefore we can use this in real-time applications such as on railway stations, airports, schools, banks where there are more people and we need to ensure everyone safety by following rules and regulations.
The similar approach of this can be made by using Goggle Teachable Machine and that can be integrated into web and any application running on mobile. Google teachable machine is already implemented and it is developed by me. But it does not provide good accuracy. In future, with the help of large dataset it can become more accurate.

5 References :

1. A.K , I. Sut. and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", Advances in Neural Information Processing Systems 25, pp. 1097-1105.
2. P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in 2008 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2008, pp. 1–8.
3. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.
4. R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2014, pp. 580–587.
5. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05), vol. 1. IEEE, 2005, pp. 886–893.
6. R. Girshick, "Fast r-cnn," in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.
7. S. Yang, P. Luo, C.-C. Loy, and X. Tang, "Wider face: A face detection benchmark," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 5525–5533.
8. A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 761–769.
9. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga et al., "Pytorch: An imperative style, high-performance deep learning library," in Advances in Neural Information Processing Systems, 2019, pp. 8024–8035.
10. K. He, X. Zhang, S. Ren and J. Sun, "Deep residual learning for image recognition",2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, 2016.
11. A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks",Advances in Neural Information Processing Systems 25, pp. 1097-1105
12. Sawhney, Rahul, et al. "A comparative assessment of artificial intelligence models used for early prediction and evaluation of chronic kidney disease." Decision Analytics Journal 6 (2023): 100169.
13. Srivastava, Swapnita, et al. "An Ensemble Learning Approach For Chronic Kidney Disease Classification." Journal of Pharmaceutical Negative Results (2022): 2401-2409.
14. Irfan, Daniyal, et al. "Prediction of Quality Food Sale in Mart Using the AI-Based TOR Method." Journal of Food Quality 2022 (2022
15. Pramanik, Sabyasachi, et al. "A novel approach using steganography and cryptography in business intelligence." Integration Challenges for Analytics, Business Intelligence, and Data Mining. IGI Global, 2021. 192-217.
16. Mohseni, Sina, et al. "Machine learning explanations to prevent overtrust in fake news detection." Proceedings of the International AAAI Conference on Web and Social Media. Vol. 15. 2021.
17. Narayan, Vipul, et al. "To Implement a Web Page using Thread in Java." (2017).
18. Paricherla, Mutyalaiah, et al. "Towards Development of Machine Learning Framework for Enhancing Security in Internet of Things." Security and Communication Networks 2022 (2022).
19. Tyagi, Lalit Kumar, et al. "Energy Efficient Routing Protocol Using Next Cluster Head Selection Process In Two-Level Hierarchy For Wireless Sensor Network." Journal of Pharmaceutical Negative Results (2023): 665-676.
20. Faiz, Mohammad, et al. "Improved Homomorphic Encryption for Security in Cloud using Particle Swarm Optimization." Journal of Pharmaceutical Negative Results (2022): 4761-4771.
21. Narayan, Vipul, A. K. Daniel, and Pooja Chaturvedi. "E-FEERP: Enhanced Fuzzy based Energy Efficient Routing Protocol for Wireless Sensor Network." Wireless Personal Communications (2023): 1-28.
22. Babu, S. Z., et al. "Abridgement of Business Data Drilling with the Natural Selection and Recasting Breakthrough: Drill Data With GA." Authors Profile Tarun Danti Dey is doing Bachelor in LAW from Chittagong Independent University, Bangladesh. Her research discipline is business intelligence, LAW, and Computational thinking. She has done 3 (2020).
23. Narayan, Vipul, et al. "Enhance-Net: An Approach to Boost the Performance of Deep Learning Model Based on Real-Time Medical Images." Journal of Sensors 2023 (2023).
24. NARAYAN, VIPUL, A. K. Daniel, and Pooja Chaturvedi. "FGWOA: An Efficient Heuristic for Cluster Head Selection in WSN using Fuzzy based Grey Wolf Optimization Algorithm." (2022).
25. Mohseni, Sina, et al. "Machine learning explanations to prevent overtrust in fake news detection." Proceedings of the International AAAI Conference on Web and Social Media. Vol. 15. 2021.