Web Based Home Security using CNN

Md. Nahid Hassan
Meem Mursalin Chowdhury

Supervised by

Prof. Bimal Kumar Pramanik
Prof. Subrata Pramanik

Abstract

With the passage of time technology is advancing and now we are living in an era of Machine Learning(ML). Machine Learning itself contains a lot of things. Among them Neural Networks(NN) are becoming more and more popular day by day.The usage of Neural Networks are increasing and the various fields of IT sectors are taking initiative for converting their services in these technologies. There are a lot of examples like self-driving automobiles, face and pattern recognition, natural language processing and many more things that are already implemented by many other companies. We took an initiative to use the Neural Networks for security[1] purposes, especially for home. We thought about how to create a budget friendly security solution for our home which is at the same time cost efficient, feasible, latest technology and easy to use. But Neural Networks are of many kinds and we need to use a particular kind. Here Deep Learning just saves the day. It is nothing but a technique of Machine Learning which is used for selecting a particular neural network for a particular job. So, then we came up with an idea of using the Convolutional Neural Network(CNN), which is basically a Neural Network, and used for image classification. We are developing a web application using django which takes videos from CCTV or webcam and uses a model of convolutional neural network(CNN) to predict the object and save the result on the database. CNN was found to perform better in regards to training time compared to TensorFlow with Keras as frontend. We intend to explore how a Convolutional Neural Network(CNN) can work with images and how to enhance the performance.

Table of Contents

Introduction

At present we live in the era of Machine Learning(ML). Particularly Neural Networks(NN) and Deep Learning is gaining more and more attention. From Deep Learning we come to know which Neural Network is suitable for which particular job. So many companies and organizations are becoming more interested in these and trying to convert their services in such a manner. As Machine Learning is providing more and more facilities, the usage of it is increasing. Likewise we are trying to find out the effectiveness of using the ML for *home security*.

Suppose we want to restrict the entrance of a house, say we have a number of people who can enter into the house and the others cannot, or if any unknown person enters, we want to find out who has entered the house in which particular moment. Also suppose we want to control the restriction remotely. How can we do it with minimum cost, more effectiveness, and at ease - the answer is quite simple. It lies within the use of ML. With the help of Deep Learning, we come to know that *Convolutional Neural Network* (CNN) is suitable for *image classification*.

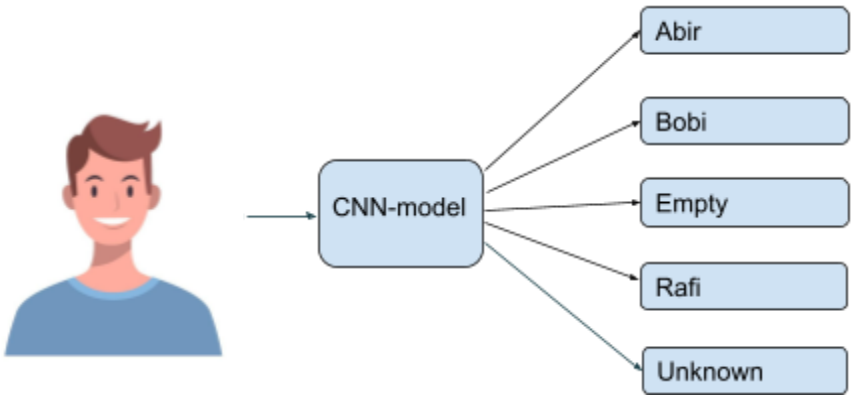


Figure 1: Image Classification using CNN model

And we can use this concept for the identification of a person and detect unwanted access. For example, the system can detect the authorized person, unauthorized persons, and empty spaces. And for any unauthorized access, the system will notify the owner or the authorized user of the inconvenience. For remote access to the system, we can just have a web app deployed to the cloud and can get access from anywhere. Now, this system is very promising, as it is both cost-friendly, easy to use, and feasible. Though the initial cost is somewhat high, the after cost and maintenance are minimal and usability is good enough.

Related Works

Our main motivation came from work done in [2] where the author introduces the idea A Study on CNN *Transfer Learning* for Image Classification. This work proposes the study and investigation of such a CNN architecture model (i.e. vgg-19) to establish whether it works best in terms of accuracy and efficiency with new image datasets via Transfer Learning. *Fine-tuning* [3] is using a convolutional neural network for modeling classification tasks.

History of Convolutional Neural Network

CNN or Convolutional Neural Networks is the backbone of many modern computer vision systems. But it was not built overnight. At first, David Hubel and Torsten Wiesel described “simple cells” and “complex cells” in the visual cortex in 1959. These two cells are used for pattern recognition. This concept builds the fundamental basis of the Convolution Neural Network models. Then in the 1980s, Dr. Kunihiko Fukushima, proposed the *“neocognitron”* model. He was inspired by Hubel and Wiesel’s work on simple and complex cells. Here “neocognitron” is a self-organizing neural network model for a mechanism of pattern recognition that cannot be affected by the shift in position. The modern convolutional neural networks working appeared in the 1990s. Inspired by the neocognitron, Yann LeCun[4] and others demonstrated in their paper how a CNN model can be used successfully for handwritten character recognition. They specially trained a CNN using the MNIST database of handwritten digits. One direct consequence of this work is automated handwritten address recognition. The researchers carried out their works on the CNN model throughout the 1990s and early 2000s. The most significant work was done in 2012. A CNN called AlexNet[5] was introduced this year which won the ImageNet Challenge. Alex Krizhevsky published a paper on this. Similar to MNIST, ImageNet is a public, freely-available data set of images with the corresponding true labels. ImageNet focuses on “natural images,” or pictures of the world i.e. labeled like “amphibian,” “furniture,” and “person.” ImageNet currently includes 14,197,122 images. At the present time, CNN models have achieved the most excellent performance in different aspects i.e.
describing natural images (ImageNet, CIFAR-10, CIFAR-100, and VisualGenome), doing facial recognition (CelebA), analyzing medical images (chest x-rays, photos of skin

lesions, and histopathology slides), etc. Many companies are developing exciting AI-based apps like describing surrounding blind people and many more things.

Theory

Convolutional Neural Network

Convolutional neural network is an unsupervised deep learning algorithm. It is superior to other neural networks or we can distinguish it from other neural networks in performance with image, speech or audio signal inputs. There are three types of layers in convolutional neural networks -

- Convolutional layer
- Pooling layer
- Fully-connected (FC) layer

Convolutional Layer

The convolutional layer is the basic building block of a CNN. Here the majority of computation is done. It has a few components like - input data, a filter, and a feature map. If we want to give a color image, representation of a 3D matrix in pixels, as input, it means that there will be three dimensions - height, width, and depth which correspond to RGB in an image. For detecting features, there is a feature detector, known as kernel or filter. It moves across the receptive fields of the image for checking whether the feature is present or not. This process is known as convolution.

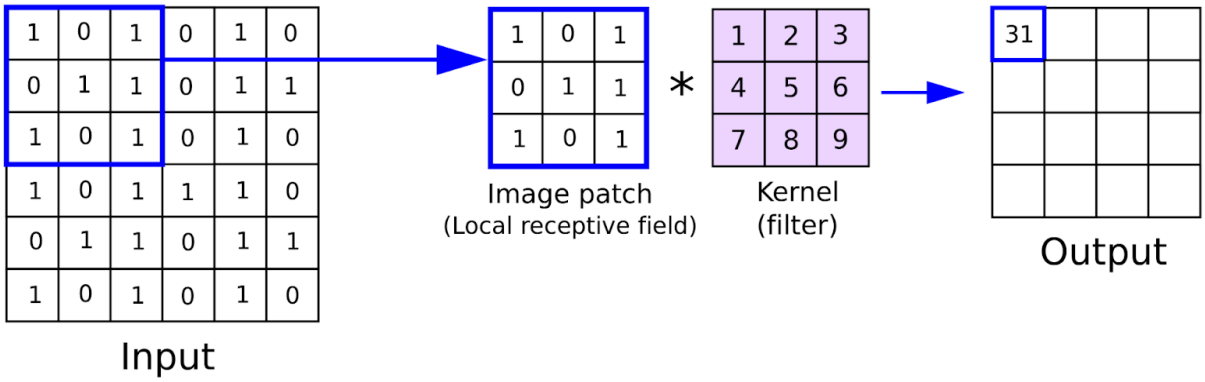


Figure 2: Convolution
Source: Adapted from [6]

Pooling Layer

Pooling layer is also called downsampling. It conducts dimensionality reduction that means reducing the number of parameters in the input. It is somewhat similar to the convolutional layer but has some differences. The pooling operation sweeps a filter across the entire input, having no weights. The kernel applies an aggregation function to the values within the receptive field and populates the output array. There are mainly two types of pooling:

- **Max pooling:** Here the filter moves across the input and selects the pixel with the maximum value to send it to the output array. This approach tends to be used more often than the average pooling in comparison.

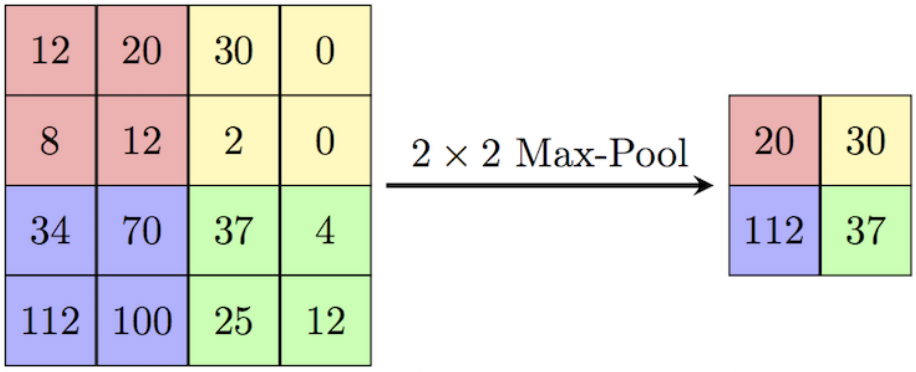


Figure 3: Max pooling

Source: Adapted from [4]

- **Average pooling**: Here, the filter moves across the input and calculates the average value within the receptive fields to send to the output array.

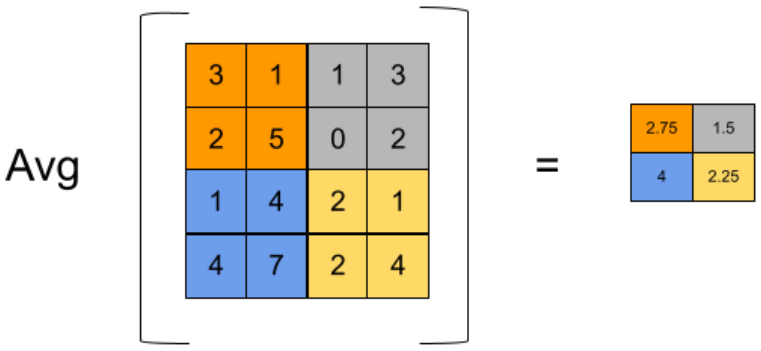


Figure 4: Average Pooling

Source: Adapted from [5]

Although a lot of information is lost in the pooling layer, it also adds some benefits to CNN. The pooling layers help to reduce complexity, to improve efficiency and to limit the risk of overfitting.

Dropout

Deep neural networks which have a large number of parameters are very powerful machine learning systems. But the problem is these kinds of networks may have serious problems like overfitting. Again large networks are also slow to use. So it becomes difficult at test time to deal with overfitting by combining the predictions of many different large neural nets. At this time, dropout comes to help. Dropout is a technique which randomly drops units (along with their connections) from the neural network at the time of training. This prevents the units from a lot of co-adapting. Dropout samples from a very large number of different networks at the time of training. At the time of

testing, it becomes easy to approximate the effect of averaging the predictions of all these thinned networks. It can be done so by simply using a single unthinned network that has smaller weights. This procedure significantly reduces the overfitting and improves the other regularization methods. We can see that dropout improves the performance of neural networks on supervised learning tasks in speech recognition, vision, document classification, and computational biology. It obtains state-of-the-art results on many benchmark data sets.

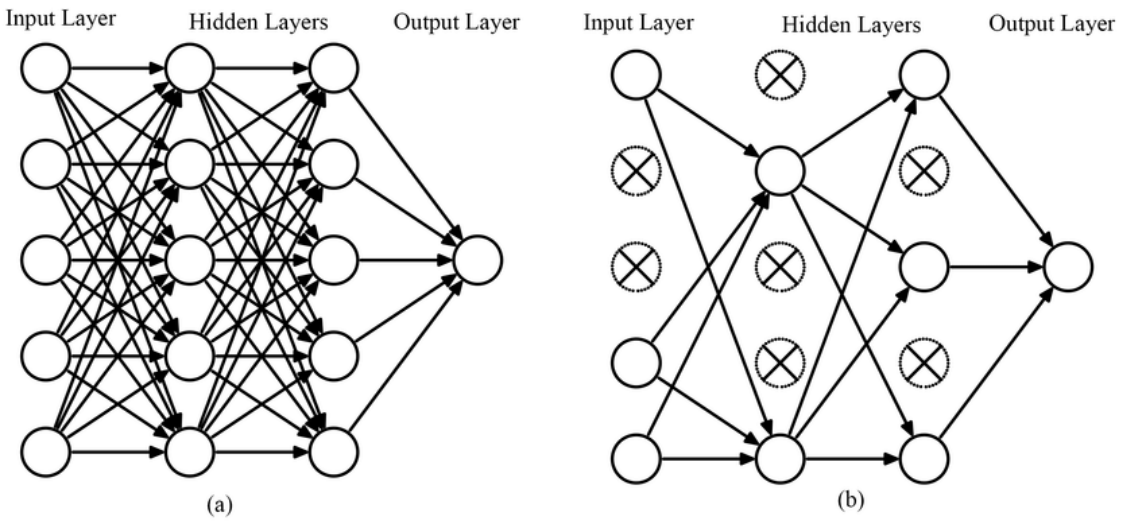


Figure: Dropout Strategy. (a) A standard neural network. (b) Applying dropout to the neural network on the left by dropping the crossed units.

Source: Adapted from [5]

https://www.researchgate.net/figure/Dropout-Strategy-a-A-standard-neural-network-b-Applying-dropout-to-the-neural_fig3_340700034

https://jmlr.org/papers/v15/srivastava14a.html

Activation Function

An activation function in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network [6]. Basically

the activation function of a node just defines the output of that node on the basis of input or set of inputs. There are some activation functions like - *Sigmoid*, *Relu*, *Softmax*, etc.

Sigmoid:
The sigmoid activation function is also known as the *logistic* function. The function is the same as the function used in the logistic regression classification algorithm. It takes any real value as input and gives an output from 0(zero) to 1(one). The bigger the input or the larger the positive number, the result is close to 1. And the smaller the input or negative number, the result is close to 0. The sigmoid activation function can be calculated as follows:

$$S(x) = \frac{1}{1+e^{-z}}$$

Where e is a mathematical constant and it is the base of the natural logarithm.

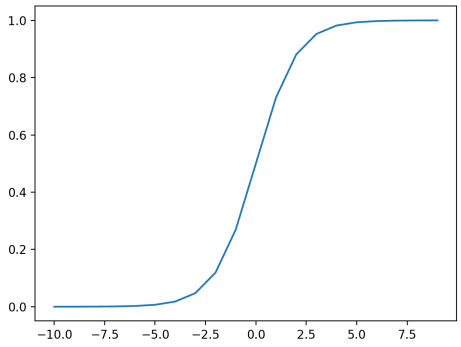


Figure : The sigmoid function

Relu
The ReLU function is known as *rectified linear* activation function. It is the most common function used for hidden layers. It is very simple to implement and effective. It can be calculated as follows:

$$ReLU(x) = max(0, x)$$

So, we can see that if any negative value is given as input, it will give output as 0(zero). Otherwise, it will give output the positive value that is given as input.

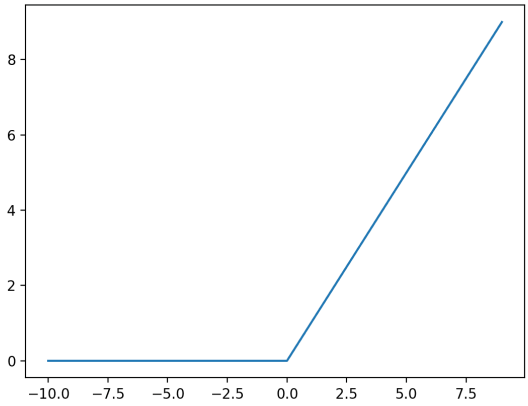


Figure : ReLU function

Softmax

The softmax function is used as the activation function in the output layer of neural network models. The Softmax activation function calculates the relative probabilities. Softmax is used as the activation function for multi-class classification problems. It can be calculated as

$$softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)}$$

Measurement Methodology

Confusion Matrix

Confusion matrix is a very popular measure used while solving classification problems. It is a summary of prediction results. It can be applied to binary classification as well as for multiclass classification problems. The key to the confusion matrix is the number of correct and incorrect predictions which are summarized with the count values and each class breaks them down. It tells the way how the classification model is confused at the time of predictions. It tells the errors of the classifier and also the types of the error.

Precision

Precision is calculated by dividing the true positives by anything that was predicted as a positive.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Recall

Recall (or True Positive Rate) is calculated by dividing the true positives by anything that should have been predicted as positive.

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Sensitivity

Sensitivity is the ability of a test to correctly identify patients with a disease.

$$Sensitivity = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

Specificity

Specificity is the ability of a test to correctly identify people without the disease.

$$Specificity = \frac{True\ Negative}{True\ Negative + False\ Positive}$$

ROC

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds.

Receiver operating characteristics (ROC) graphs are useful for organizing classifiers and visualizing their performance. ROC graphs are commonly used in medical decision making, and in recent years have been used increasingly in machine learning and data mining research. Although ROC graphs are apparently simple, there are some common misconceptions and pitfalls when using them in practice. The purpose of this article is to serve as an introduction to ROC graphs and as a guide for using them in research.

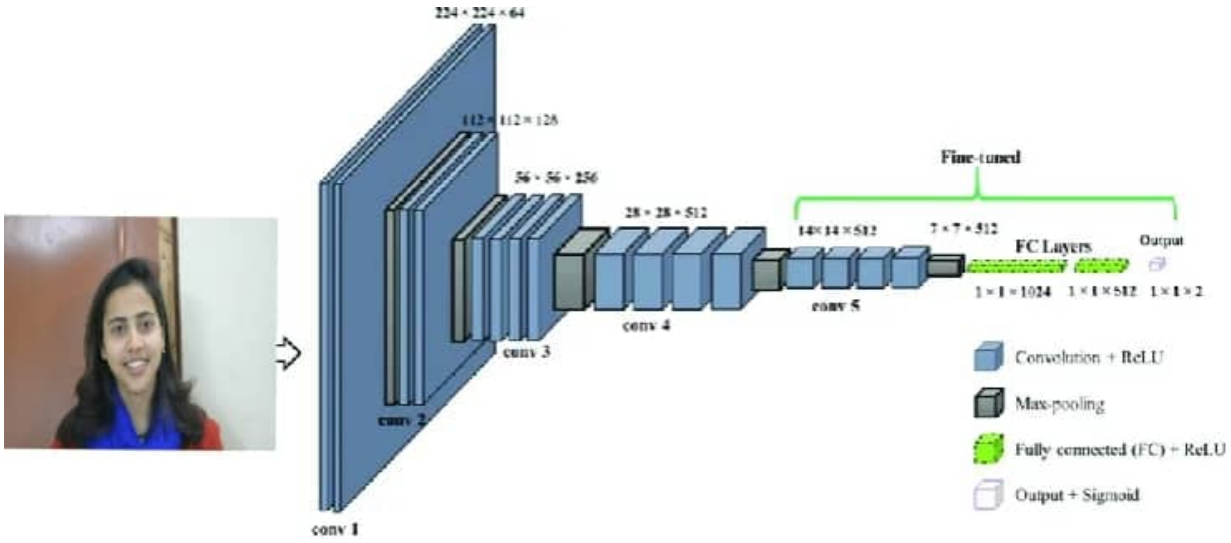
Methodology

Convnet Structure

VGG19

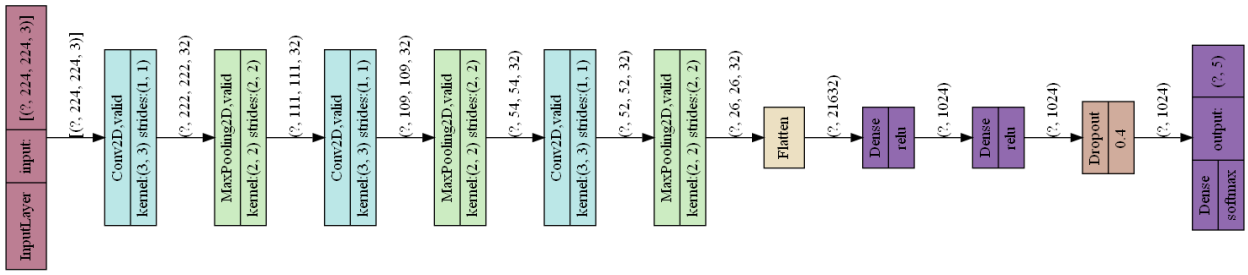
VGG-19 is an advanced CNN that uses pre-trained layers. It has a great understanding of what is actually an image in terms of shape, color, and structure. VGG-19 is very deep and millions of diverse images with many complex classification tasks are trained here. In 2014, the Visual Geometry Group of Oxford University proposed it and it showed accurate classification performance on the ImageNet Dataset.

VGG-19 is basically a model architecture that is composed of 16 convolutional layers (with 5 pooling layers) and 3 fully-connected layers. It takes an input of size (224 * 224 * 3). It has a total of 138 million parameters. There are two 64 kernels of size (3 * 3), two 128 kernels of size (3 * 3), four 256 kernels of size (3 * 3), eight 512 kernels of size (3 * 3), five max pooling was performed over a (2 * 2) pixel windows with stride 2. In the fully connected layers or dense layers, there are in total 4096 nodes with the activation function ReLU and the final layer has the nodes in total of 1000 with the activation function Softmax.



Custom CNN Model

In custom model, we use an input of (224 * 224 * 3). It has three 32 kernels of size (3 * 3) of stride (1 * 1), three max pooling performed over windows (2 * 2) with stride (2 * 2), a flatten layer, three *dense* layers where the first two have 1024 nodes each with activation function ReLU and one dropout layer, lastly one output layer with activation function Softmax.



Datasets

Here, as we are using machine learning technology, we need data to train the machine for detecting the face and recognizing it. So, for face detection and recognition, the data for training the machine is an image. For our project, we categorized the 5 classes. Three classes are for identifying particular three people, one class for an unknown person and the other one for empty space. Now for the image/data collection, we have chosen three people who were agreed to be subjected voluntarily. To have a balanced gender in subjects we have two male and one female subject. Subjects have signed a consent paper. The consent paper includes name, age, gender, and other important information. They agreed that their audio and video will be used in the analysis, report writing, and publications. They also agreed that it can further be used in research work and publication in the future. Each subject had signed two copies of the consent paper, one was for them and another for the project.

The photos were the necessary data for this project. The photos of the individuals were collected using a 12-megapixel camera. Around 10,000 photos of each person were

needed. For this, some short videos were taken with the camera, and from those videos, the photos were made. The tool that was used for this purpose was VLC Media Player. The speed of capturing the photo was 5 frames per second. The videos were taken in the lab containing the face of a person from different angles. The videos were taken individually. The unknown class images were collected in the form of random capture of different people by the camera and also from the internet, especially from Kaggle. For empty class images, we have written a script for a webcam and captured images from it.

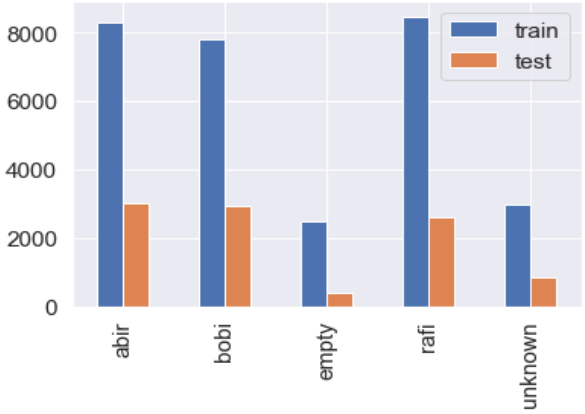


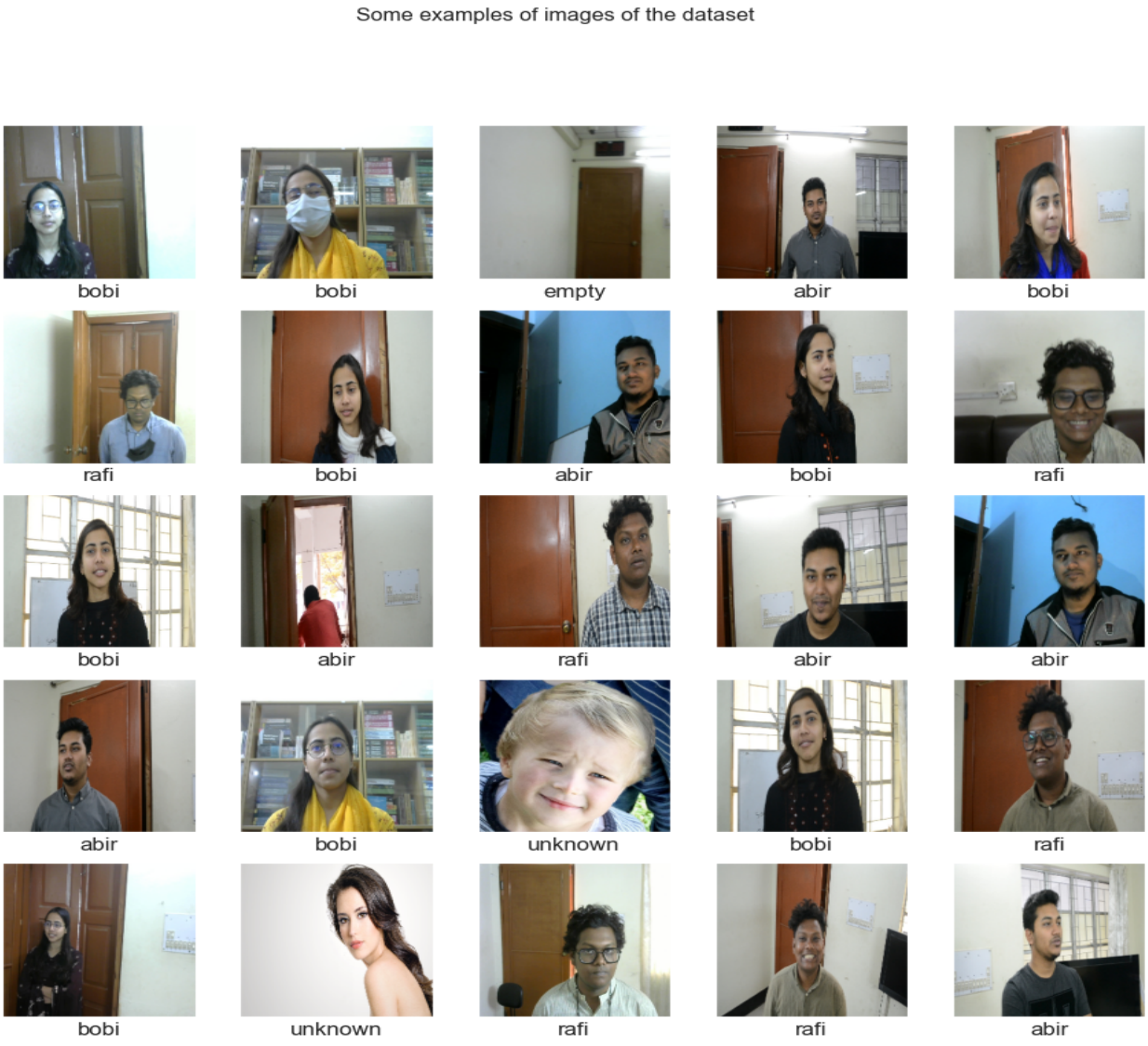
Image Preprocessing

Traditional machine learning methods like multilayer perceptron machines, support vector machines, etc mostly use shallow structures through which only a limited number of samples and computing units can be processed. When the target objects have many specifications, we face the problem of insufficient performance and generalization ability of complex classification. At present, the convolution neural network (CNN) is developed to a great extent that it is good at image classification and recognition problems and the accuracy has improved a lot.

Images in the training dataset had differing sizes, therefore images had to be resized before being used as input to the model. In our model, we reshape each image (224,224) in shape. Training a convolutional neural network on raw images will probably lead to bad classification performances. A simple algorithm for image preprocessing is used when we read images. First of all, we read images with fixed shapes. Decode jpeg format to RGB format. Convert pixel values to floating-point values and next rescale the pixel

values (between 0 to 255) to the [0,1] interval (as training a neural network with this range is efficient). In our model for image preprocessing or image augmentation, we perform rescaling, reshaping, cropping, horizontal flipping, shear, padding, rotation, shift, changing brightness, zooming and filling image.

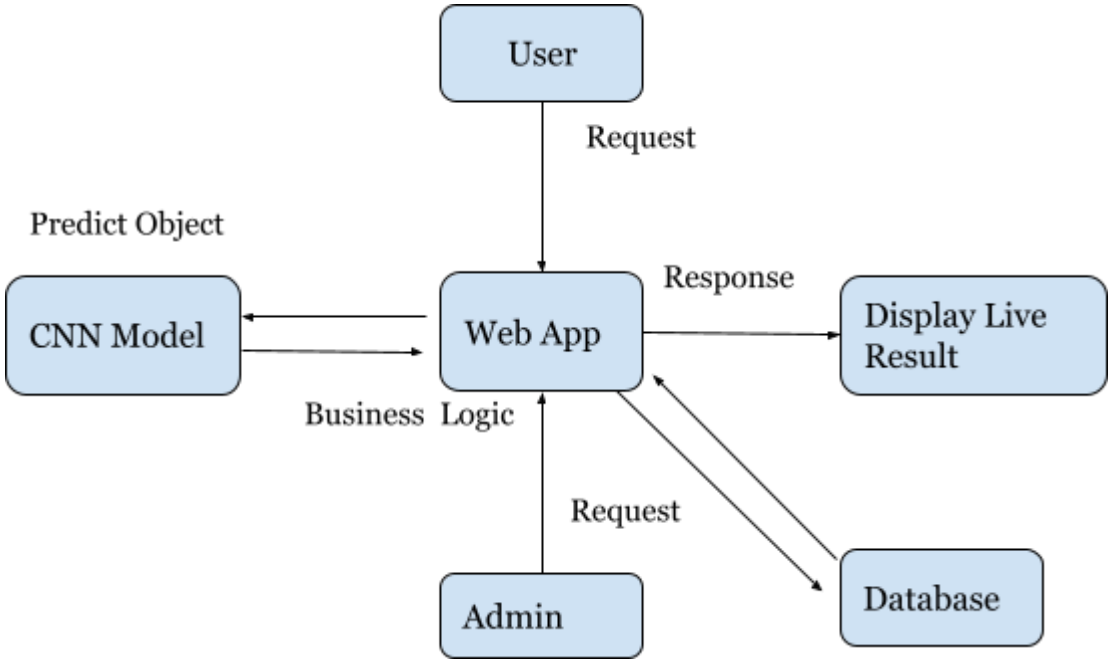
Some of the images that we use for our training with its label are displayed here. There are a total of 5 class images.



Web Application Organization

Here we are using the Python-based web framework Django for web application development. In recent years Django is extremely popular for the backend of web applications. In this application, we first need to log in to the site, and then he/she can request live camera footage with a person recognition result. Only admin users can access the admin panel of the site.

Flowchart of the Web Application

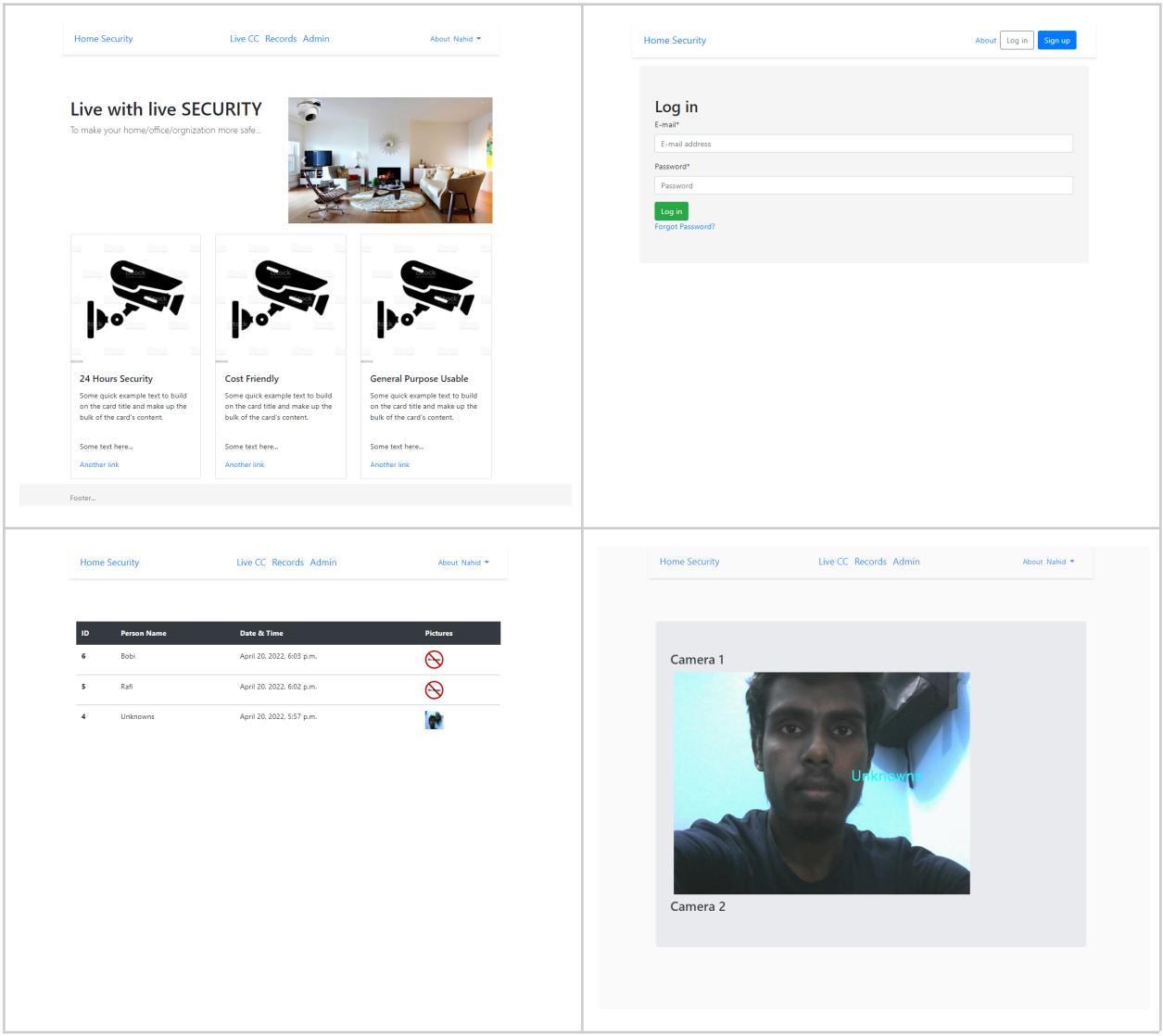


From the flow chart we can see that the web app contains the CNN model for identifying the different classes. The user can access the web app from where he or she gets the result of the CNN model classification of images. Here the main work or prediction is done in the business logic section where random images are captured at the 2-second intervals and it is given as input to the CNN model to predict the image or for image classification.

Features of the Application

First, we have to sign up for the web app to get the results. After signing up, the user can have the feature of getting the live recording option through which the image

classification or identification is done. Here when the live recording option is enabled the images are captured at an interval of 2s which are sent as input to the CNN model for image identification. At a time the prediction result and time are stored in the database for further query. In the database there are three fields (Person, record_datetime, and image). We store images only for unknown classes. After the identification, the result is shown through the web app. If the known person is already in databases on the current date, then no redundant data is stored. The user can get the result there.

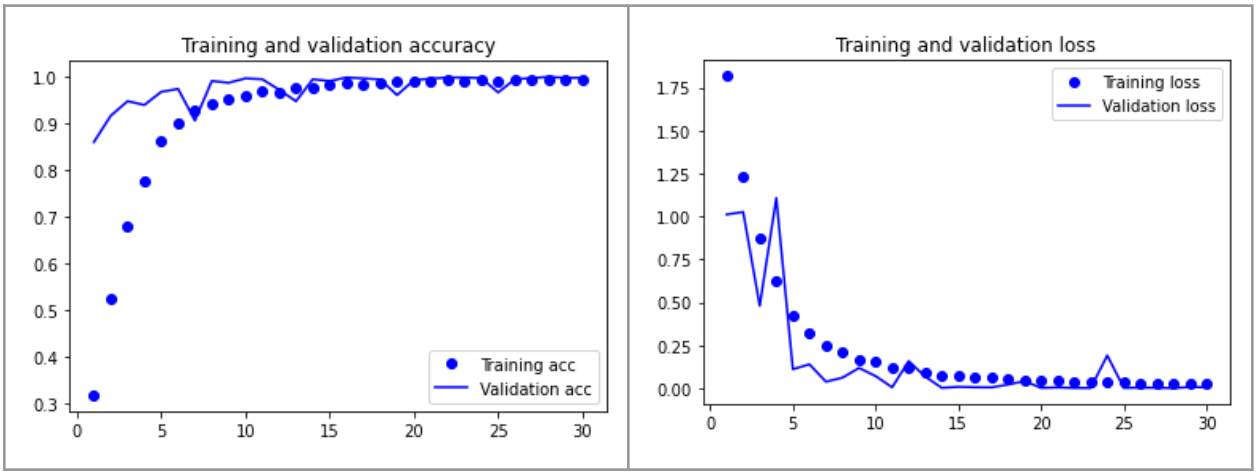


Results & Discussions

After working on the previous section, we started working with our project dataset which we collected for our project. Currently, we have worked interactively with this data. We have changed our model various times and got different results. In this particular model, we have worked with ~30,000 pictures for training and ~10,000 pictures for validation of this particular training.

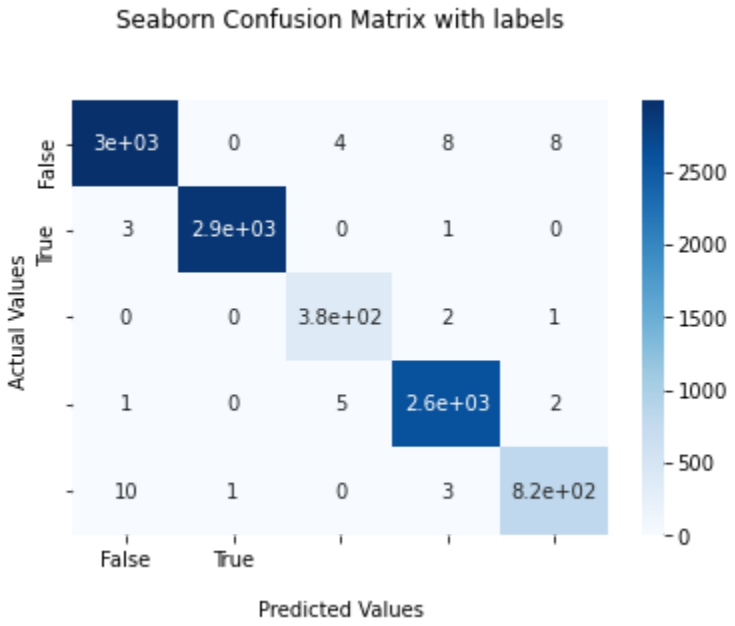
Vgg19 Architecture

These two graphs show our validation accuracy vs training accuracy, and validation loss vs training loss graph for vgg19 architecture. For every epoch, it gradually increases its accuracy.



It shows that our validation accuracy is excellent. Maximum validation accuracy was in epochs 28 where the accuracy is nearly ~99%.

Confusion Matrix on Validation Data



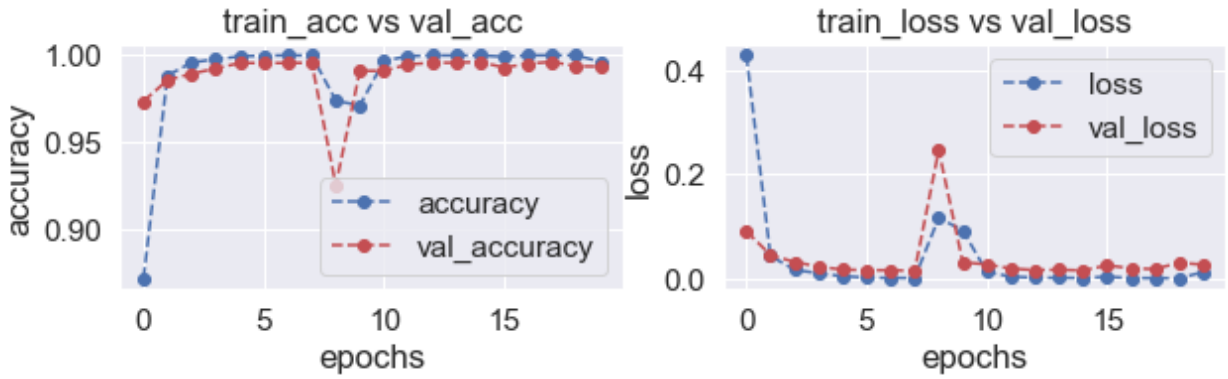
On the validation dataset our vgg19 trained model performs well. From the confusion matrix, we can see the result. Most of the classes are perfectly predicted which our desired output is! From the confusion matrix, we can see how accurately our model classifies the images.

Precision-Recall for validation data

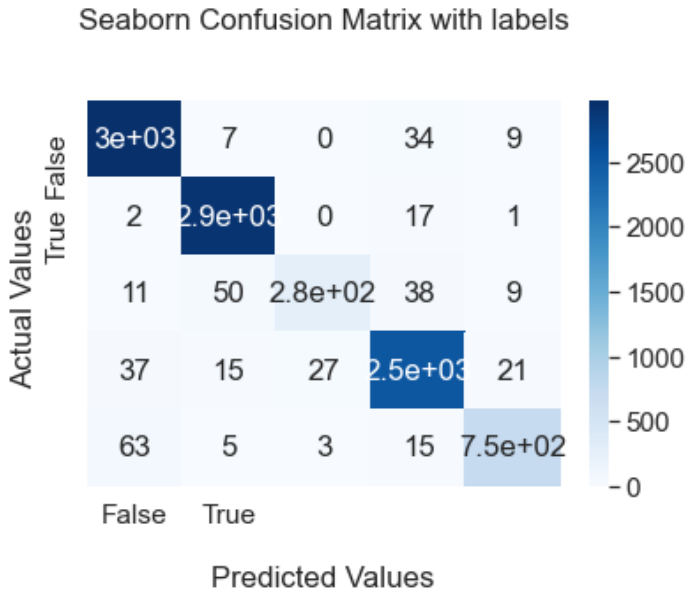
Class Name	Precision	Recall	f1-score	support
Abir	1.00	.99	.99	3016
Bobi	1.00	1.0	1.0	2929
Empty	.98	.99	.99	387
Rafi	.98	1.0	1.0	2623
Unknown	.99	.98	.98	833

Custom CNN Model

The following graphs show our validation accuracy vs training accuracy, and validation loss vs training loss graph for custom CNN model. For every epoch, the accuracy is gradually increased for a certain time. At the middle point the accuracy suddenly drops and again it continues to increase in the end.



Again for each epoch, there is a decrease in the loss for training and validation. It continues for a certain time and in the middle the loss increases suddenly and again at the end the loss continues to decrease for each epoch.



On the validation dataset our custom trained model performs well. From the confusion matrix, we can see the result which is good. Most of the classes are perfectly predicted.

Precision-Recall for validation data

Class Name	Precision	Recall	f1-score	support
Abir	0.96	0.98	0.97	3016
Bobi	0.97	0.99	0.98	2929
Empty	0.90	0.72	0.80	387
Rafi	0.96	0.96	0.96	2623
Unknown	0.95	0.90	0.92	833

After doing all this, we have used our model in a real situation. We have used Django to serve our model in order to perform real-time predictions with a webcam input.

Risk and Caution

Though our project has decent accuracy, it has some limits. For example, misclassification of images - means wrongly classified images, lighting issues for taking images, blurred or covered faces cannot be detected, the position of faces affects the recognition, loss of connection to web app make the system fall apart, any environmental or geographical issue can hamper the camera for capturing images.

References

[1] R. Zhang and E.-J. Lee, “Human Face Recognition Based on improved CNN Model with Multi-layers,” *Journal of Korea Multimedia Society*, vol. 24, no. 5, pp. 701–708, May 2021.

[2] M. Hussain, J. Bird and D. Faria, "A study on cnn transfer learning for image classification", *18th Annual UK Workshop on Computational Intelligence UKCI 2018*, June. 2018.

[3] Zongwei Zhou, Jae Shin, Lei Zhang, Suryakanth Gurudu. Fine-Tuning Convolutional Neural Networks for Biomedical Image Analysis: Actively and Incrementally. *July 2017: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[4] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998,

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton.”ImageNet classification with deep convolutional neural networks.” (June 2017)

[6] *Production-media.paperswithcode.com*, 2022. [Online]. Available: https://production-media.paperswithcode.com/methods/MaxpoolSample2.png. [Accessed: 20-Apr- 2022].

[3] https://anhreynolds.com/img/cnn.png

[4] https://production-media.paperswithcode.com/methods/MaxpoolSample2.png

[5] https://androidkt.com/wp-content/uploads/2021/06/Avg-Pooling.png?ezimgfmt=ng:webp/ngcb1

[6] First Name Initial(s) Last Name. “Page Title.” Website Title. Web Address (retrieved Date Accessed).

[6] J. Brownlee. “How to Choose an Activation Function for Deep Learning.” machinelearningmastery. https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/#:~:text=An%20activation%20function%20in%20a,a%20layer%20of%20the%20network. (accessed 17 April, 2022)