INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact Factor: 6.078
(Volume 7, Issue 3 - V7I3-1361)
Available online at: https://www.ijariit.com

Neural Networks based solution for Door Automation

Monalika Padma Reddy
monalika.6671@gmail.com
Visvesvaraya Technological University, Belgaum, Karnataka

ABSTRACT

Face Recognition is one of the most common biometric strategies which has gained popularity because of the accuracy and security. This paper presents the implementation of a Convolution Neural Network architecture for door automation. This model is devised to overcome the disadvantages of a traditional door system and other methods such as door automation using Bluetooth, figure prints, passwords, or retinal scans. It allows the authorized people to gain access to the house by face recognition. The proposed system makes use of convolution neural network architectures and RaspberryPi. The ResNet architecture [6] is used to implement face recognition and runs on RaspberryPi. The images of the residents of the house will be used to train the model. If the person is a resident of the house, the face will be recognized and the lock will open, else it will be recognized as a human and an alarm will ring and an email alert consisting of the image of the person in front of the door will be sent to the owner. It has numerous advantages as it is user-friendly especially for senior citizens, lesser maintenance, does not require the residents to carry the keys, and reduces the threat of robbery.

Keywords— *Convolution Neural Network, face recognition, Smart Door Automation, ResNet architecture*

1. INTRODUCTION

In the traditional door systems that are being used in our residences, the number of problems being encountered is more. Usually, it is essential for a person who is at home to leave all his work and come down to the main door to open it. The person in some cases may not be to come down to open the door due to illness or such scenarios. To overcome such cases this smart door system is being designed. The techniques used in the implementation of the Smart Door System are:

1. Convolution Neural Networks
2. Sensor and alarm technology.

The Convolutional Neural Network is a type of neural network that uses a mathematical operation called convolutions which is a specialized kind of linear operation. They can be referred to as neural networks that adopt convolution in place of the general matrix multiplication in at least one of their layers. A convolutional neural network consists of three layers- namely the input layer, the hidden layer, and the output layer. There can be multiple hidden layers in the convolutional neural network that normally consist of a series of convolutional layers.

The activation function that is commonly used here is RELU which is then followed by pooling layers, fully connected layers, and normalization layers which are the additional convolutions. The inputs and the outputs of these layers are masked by the activation function and final convolution and hence, these are referred to as the hidden layers. It must be noted that the hidden layers in a convolutional neural network must be fully connected.

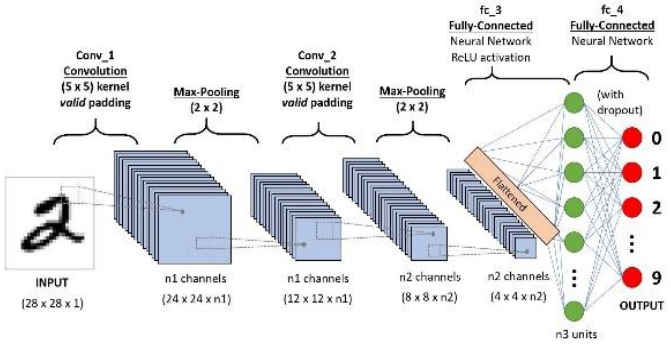

Fig. 1. Convolution Neural Network

A sensor is a device the typically responds to a stimulus and transmits a resulting impulse for operating a control. The stimulus to a sensor can be in the form of heat, magnetism, or a particular motion. The image sensors are the most popular combination of OpenCV vision software and webcam that can be used for face detection and facial recognition. An impulse will be generated by the sensor which will either open the door or sound an alarm.

Tensorflow Lite is a set of tools that are used to run TensorFlow models on IoT, embedded, and mobile devices. It enables on-device machine learning inference with a small binary size and low latency.

2. LITERATURE REVIEW

2.1 Password Based Security Lock System

In this model [1], the door automation is done using passwords. The person who wants to gain access to the house must enter a 4 digit password. If the password matches, then the person will

be granted access. However, this method is prone to hacking, the human tendency of forgetting the password, and mechanical failures. If the 4 digit password is hacked, then security will be compromised. In case the owner forgets the passwords, then there will be a huge problem.

2.2 Bluetooth Smart Radar Door Systems Via Mobile Apps

In this model [2], Bluetooth is used as a wireless connection protocol for the automation of the door lock system. The drawback of this system is it does not provide any emergency keys and this is feasible only if the person has a smartphone. In case the person does not have a smartphone or if the battery is dead or the smartphone is lost, then this model will not work.

2.3 Remote Monitoring Intelligent System Based on Fingerprint Door Lock

In this model, [3] fingerprint is usually for enforcing security. Here, the model identifies the lively fingerprint accurately and the unlock information ID, illegal burglary information to the owner over a GSM network. This method of using fingerprints for door locks has disadvantages such as software malfunctions, scanner issues, fake positives, and other major security breaches.

2.4 IRIS Biometric Recognition for Person Identification and Security

In this model, the identification of a person is usually done using the iris scan [4]. The hamming distance is used for calculating the distance between the input images and the images in the database and then two different classifiers are used for the identification of a person before granting access. However, it is difficult to obtain the small target from a distance. Iris tends to deform non elastically with the change in the pupil size and the illumination must be bright or visible.

3. NEURAL NETWORKS

Neural Networks are a subset of machine learning algorithms whose structure and working are inspired by the human brain. They have node layers consisting of input layers, one or more hidden layers, and output layers where each node has a weight and a threshold associated with it.

3.1 The Neuron

Neurons, popularly known as nerve cells send and receive signals from the brain. These are structurally and functionally unique that the other cells in the body. Fig.2. shows the structure of a neuron in the brain.

An axon is a specialized projection that allows the neurons to transmit electrical and chemical signals to other cells. The dendrites are root-like structures that branch out from the head of the neuron that process and receive signals from other neurons. In other words, dendrites are receivers and axons are transmitters for signals.

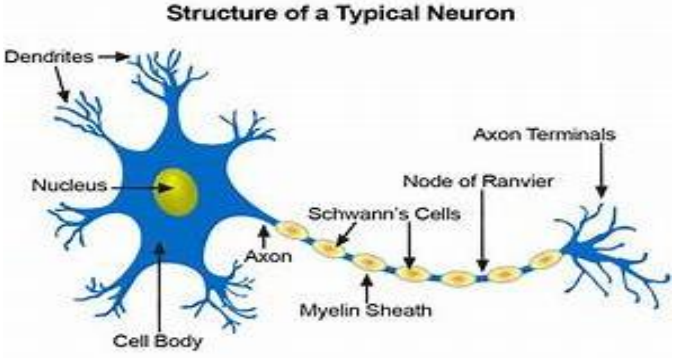

Fig. 2. Structure of a Neuron in the human brain

3.2 Representation of neurons in Artificial Intelligence

A neuron can technically be termed as a node that receives an input signal and produces a corresponding output signal. Fig.3. shows the representation of neurons

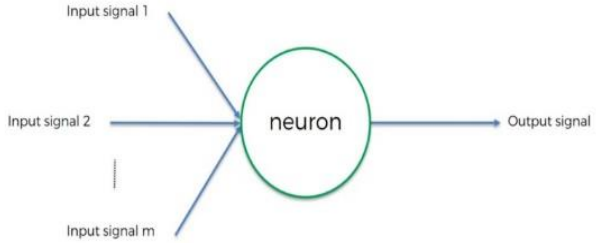

Fig. 3. Representation of Neurons

A neural network consists of an input layer, one or many hidden layers, and an output layer as shown in Fig.4.

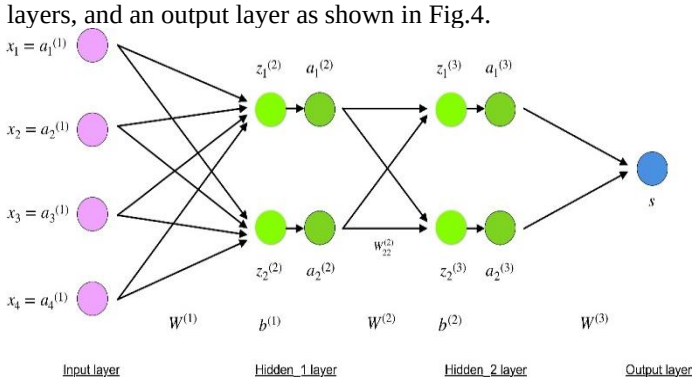

Fig.4. Illustration of a neural network

The input signals can be obtained from other neurons as well and these are called input layers. They can be vectors, multidimensional matrices, or scalars.

$$x_i = a_i^{(1)},\ i \in 1,2,3\ldots n \quad (1)$$

The equation of the input layer is as shown in Equation (1). The input signal is the independent variable of the model, such as the age of a person. Standardization and normalization need to be done on the input variables.

The hidden layer is usually located between the input layer and the output layer, which perform non-linear transformations of the obtained inputs. Here, the weights are assigned to the inputs and then directed through an activation function and an output is obtained, which is subsequently passed to the next layers.

Each hidden layer is used to produce a particular output. The final values of the neurons in the hidden layer are computed using the weighted inputs and the activations. A neural network can even have zero hidden layers. For instance, the value of the hidden neuron at layer 2 can be calculated as follows:

$$z^{(2)} = W^{(1)}x + b^{(1)} \quad (2)$$

$$a^{(2)} = f(z^{(2)}) \quad (3)$$

Where $W^{(1)}$ and $b^{(1)}$ are the weights and bias of layer 2 respectively. $a^{(2)}$ is the activation computed using an activation function.

Equation(2) represents the calculation of the neuron value at Hidden layer 1. Equation(3) represents the calculation of the activation value at Hidden layer 1 for the neural network represented in Fig. 4.

The activation function (f) is a non-linear function that allows the neural network to learn complex data patterns. Sigmoid,

ReLU, tanh are some of the well-known activation functions used.

The final layer of a neural network is the output layer, which outputs the predicted values. The obtained output values can be continuous (like the price value) or binary (Y or N like a person purchasing a house or not) or categorical.

$$s = W^{(3)} a^{(3)} \quad (4)$$

Equation(4) represents the output layer for the neural network represented in Fig. 4.

Each signal consists of weights. The weights play a significant role because neural networks learn by adjusting the weights by themselves. The importance of a signal will be decided by the weights associated with that signal.

3.3 Working of Neural Network in Artificial Intelligence

Three main steps are involved in the forward propagation of a neural network.

Here, in the first step, all the weights are added and multiplied with the independent variables and a bias is added.

$$Y = w^{(1)} * x^{(1)} + w^{(2)} * x^{(2)} + \ldots + w_m * x_m + \text{Bias} \quad (5)$$

The basic formula that is used for this is represented in Equation(5), where m represents the number of input layers.

Fig.5. represents the first step of the forward propagation where weights are multiplied by independent variables and a bias is added.

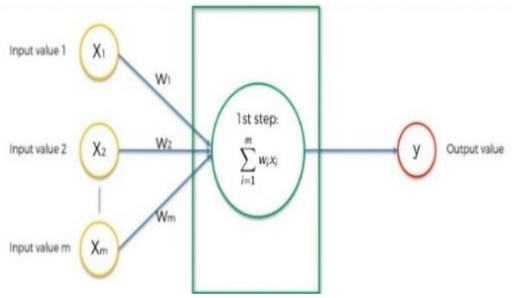

Fig. 5. Weights multiplied with independent variables and bias added

In the second step, the activation function is applied. The four popularly known activation functions are threshold function, sigmoid function, rectified function, and hyperbolic function. The activation function is used to determine which signal will pass on to the output layer.

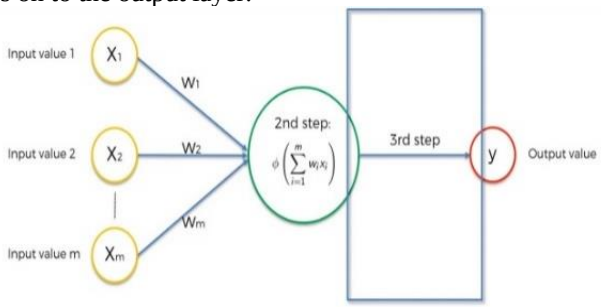

Fig. 6. Applying the activation function

Fig.6. represents the second step of forward propagation where activation function is applied

In the third step, signals are sent to the output layers. The above three steps are called forward propagation. The equations representing the operations in the forward propagation for the neural network represented in Fig 3 is as follows:

$$x = a^{(1)} \quad (6)$$

$$z^{(2)} = W^{(1)} x + b^{(1)} \quad (7)$$

$$a^{(2)} = f(z^{(2)}) \quad (8)$$

$$z^{(3)} = W^{(2)} a^{(2)} + b^{(2)} \quad (9)$$

$$a^{(3)} = f(z^{(3)}) \quad (10)$$

$$s = W^{(3)} a^{(3)} \quad (11)$$

Equation(6) represents the input layer for the neural network represented in Fig. 4. Equation(7) represents the calculation of the neuron value at Hidden layer 1 represented in Fig. 4. Equation(8) represents calculation of the activation value at Hidden layer 1 for the neural network represented in Fig. 4. Equation(9) represents the calculation of the neuron value at Hidden layer 2 represented in Fig. 4. Equation(10) represents the calculation of the activation value at Hidden layer 2 for the neural network as represented in Fig. 4. Equation(11) represents the output layer for the neural network represented in Fig. 4.

During forward propagation of the neural network, the weights are randomly assigned, which results in errors in the predicted value of the model. To improve the model accuracy, back propagation is used.

Backpropagation is the method of fine-tuning the weights of a neural network based on the loss obtained in the previous epoch, which makes the model more reliable and lowers the error rate. The output obtained is called the predicted value which is compared with the actual value of y, and the weights are updated using the chain rule.

Let 'ypred' be the value predicted by the model and the actual output be 'y'. Here, we need to observe the difference between the y and ypred values, to see how accurate the model is. For this, the loss function is used. This is called backpropagation.

The errors that are present are calculated using the loss function. The loss function uses the cross-entropy to compute the error which is then backpropagated into the neural network for it to learn. The loss function is an important component of the neural network, which defines the prediction error of the neural network. It is defined to find to improve the model accuracy by finding the minima of that particular function to optimize the model.

$$\text{Loss} = \sum (y - \text{ypred})^2 \quad (12)$$

The loss function can be calculated as shown in Equation(12). Along with the weights, feature detectors are also adjusted. It is important to note that the data goes through the whole network, the error is calculated and backpropagated.

The input weights of the neural network must be updated so that the predicted value and the actual value must be equal or close to each other. The main goal is to reduce the loss function hence, in each epoch, the weights must be updated so that the loss function will be decreased. For this, optimizers like gradient descent and stochastic gradient descent. These gradients determine much the parameters must change so that the value of the loss function will be minimized.

4. FACE DETECTION USING CONVOLUTION NEURAL NETWORKS

There is a staggering degree of similarity between the working of the human brain and the working of a convolution neural network.

4.1 Cerebral cortex and visual cortex

The human brain is divided into four parts out of which one is the cerebral cortex. It is a thin layer that covers the outer portion of the brain. The visual cortex is present inside the cerebral

cortex. The visual cortex is a part of the cerebral cortex that receives and processes the impulses from the optic nerves.

4.2 Similarities between the convolution neural network and human brain

Suppose we are seeing a dog, then that information that is in the form of an impulse is passed from the sensory organ - eyes into the brain with the help of optic nerves. This will then be further passed into several neurons and finally to the visual cortex in the cerebral cortex region.

Let us assume that the visual cortex has several layers say v_1, v_2, v_3, and so on. These layers play a vital role. Suppose the v_1 layer is responsible for identifying the edge of the dog like the body edge, it calculates it.

Then, the obtained information is passed on to the next layer v_2 where information of the movements of the dog or if any other object is present beside it and is responsible for distinguishing the dog and the other object.

On similar lines, the information is gathered by the different subsequent layers and passed on. It can be observed that the different operations that are being carried out here are called filters in convolution neural network

5. CONVOLUTION NEURAL NETWORKS

The convolution neural network has two important parts- namely, feature extraction and classification. A series of convolution and pooling operations will be performed by the network in feature extraction during which the features will be detected.

The fully connected layer in the convolution neural network will serve as a classifier. The probability for the object on the image that is being predicted by the algorithm will be assigned. Four important steps are involved in convolution neural networks. They are:

1. Convolution
2. Pooling
3. Flattening
4. Full Connection

5.1 Convolution

Convolution is a function that is obtained integration of two given functions which expresses how one function is modified by the other. The mathematical formula for convolution is as shown in Fig.7.

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

Fig: 7. Convolution function

Input image, feature detector, and feature maps are the three essential elements that enter into the convolution operation.

The basic idea of the convolution is Input image x Feature detector = Feature Map.

The main advantage of using feature maps is reducing the size of the input image, making it easier to read. A convolution matrix to adjust an image. It can be used to blur, edge detect, and sharpen the image. Fig.8. Represents the generation of a feature map when a convolution kernel is applied on an input image.

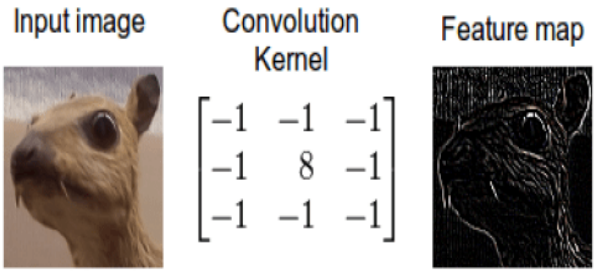

Fig. 8. Feature Map

Consider the image having a 6x6 matrix and a filter having a 3x3 matrix. As discussed earlier, layer v1 was responsible for detecting the edge of the image.

On similar lines, the vertical edge filter is represented by the filter i.e. the vertical edge of the image can be determined by applying this filter on that particular image.

Multiply the first 3x3 matrix with filter and add. The same procedure should be repeated by moving by one. This is an iterative process and once the end is reached, it should be stepped down by one, and the same procedure must be continued. A 4x4 matrix is obtained as the result of the matrix multiplication. This operation is called Convolution.

0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1

Image

*

1	0	-1
2	0	-2
1	0	-1

Filter

=

Result

0	- 4	- 4	0
0	- 4	- 4	0
0	- 4	- 4	0
0	- 4	- 4	0

Fig. 9. Convolution Operation

Fig.9. reprsents the convolution operation. The feature detector is also known as the filter or a kernel. These play a vital role. Just like the vertical filter, we have many filters that can be used for detecting the edge of the face, horizontal edge filter, and so on. From the resultant matrix, it can be concluded that the image size n=6 and filter size f = 3. The formula for the result can be devised as shown in Equation(13).

$$\text{Result} = n - f + 1 \qquad (13)$$

Hence, the result for image size n=6 and filter size f = 3 is

$$6 - 3 + 1 = 4$$

From the result, a 4x4 matrix is obtained but the size of the image is a 6x6 matrix. It can be seen that chances some of the vital information exists. To avoid this, the concept of padding is applied.

The primary purpose of convolution is to find features in an image using feature detectors, put them into a feature map, and having them in the feature map. The spatial relationship between the pixels is preserved. The images on the computer are represented in 0's and 1's.

5.2 Padding

The two main downsides of convolution are the shrinking of outputs and the loss of information, especially in the corners of the image. To overcome this, padding is done.

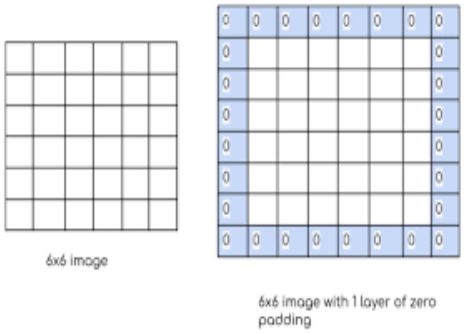

Fig.10. Padding

In simple words, it can be described as an additional layer that can be added to the border of the image. This helps in increasing the accuracy.

$$n - f + 1 = 6$$
$$n = 6 - 1 + f$$
$$n = 5 + 3 = 8$$

The value of n obtained here is 8. If a padding p=1 is applied then, we get an image of size 8x8 is obtained. Equation(14) is the result obtained on multiplying the image with the filter.

$$\text{Result} = n + 2p - f + 1 \qquad (14)$$
$$\text{Result} = 6 + 2 - 3 + 1 = 6$$

The resultant matrix, thus obtained is 6x6 and there is no loss of data. The original size of the image can be retained if padding is applied. Any number of convolution and padding can be applied to retain the size of the image. Fig.10. shows the zero-padding operation where 0's are used for padding

5.3 Pooling

The neural networks have a property called spatial invariance which means that the location of a feature in an image doe not matter. The main objective of pooling is to reduce dimensionality, down-sample an input representation, and allowing the assumptions made about the features contained in the sub-regions.

It can be described as a sample-based discretization process. Here, the max pooling feature is applied. The maximum value of the pixels is taken and placed over a separate matrix. Pooling is of many types, namely sum pooling, average pooling, and subsampling. Subsampling is the generalization of mean pooling. Max pooling as shown in Fig.11. a type of pooling where only the high pixel values are being taken and hence, we can recognize the face in the image.

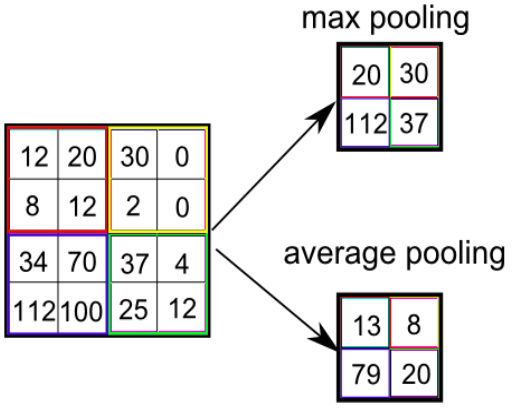

Fig. 11. Types of Pooling

5.4 Flattening

Flattening as shown in Fig.12. can be defined as the process of converting all the resultant two-dimensional arrays into a single long continuous linear vector. Flattening breaks the spatial structure of the data obtained and transforms it into a vector.

Consider a layer with N filters of size s on a WxH image. Flattening transforms this image which is in the form (W-(s - 1), H - (s - 1), N) into a vector of size (W - (s -1)) x (H - (s – 1)) x N.

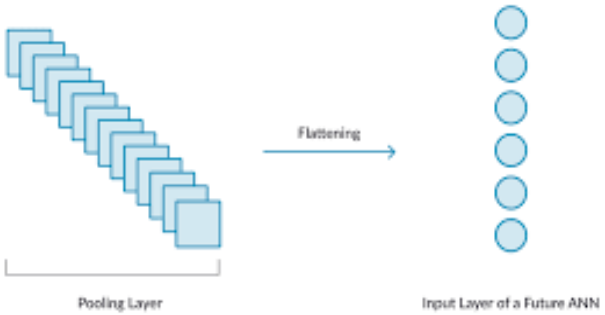

Fig.12. Flattening

Fig.13. shows the flattening of a pooled feature map. The pooled feature map is a 3x3 array that is being converted into a continuous linear vector by stacking the elements row-wise.

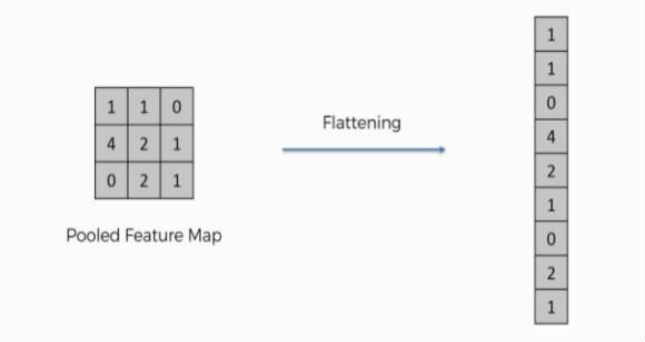

Fig.13. Pooled feature map being Flattened

5.5 Fully-Connected layer

The full connection operation is shown in Fig.14. is the last operation being applied in convolution neural networks. Here, the flattened images are given as an input to the fully connected layer.

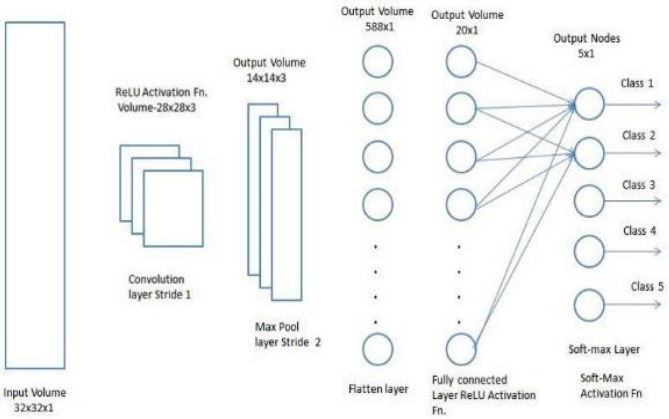

Fig. 14. A fully connected layer

A way of learning non-linear combinations of features of a higher level as represented by the output of the convolution layer can be done by adding a fully connected layer.

6. DATA AUGMENTATION

Data augmentation is one of the most important steps in Convolution Neural Networks. It helps in increasing the data

available for the training model without collecting a large number of images as shown in Fig.15. Data augmentation can transform a given image into different kinds of images. The images can be flipped, vertically shifted, horizontally shifted, inverted, zoomed, and so on. These transformed images that are obtained after data augmentation are called invariance.

This technique helps in increasing the images available for training the model ten times more than the original image. By performing data augmentation, the neural network can be prevented from learning all irrelevant patterns thereby essentially boosting the overall performance of a neural network model.

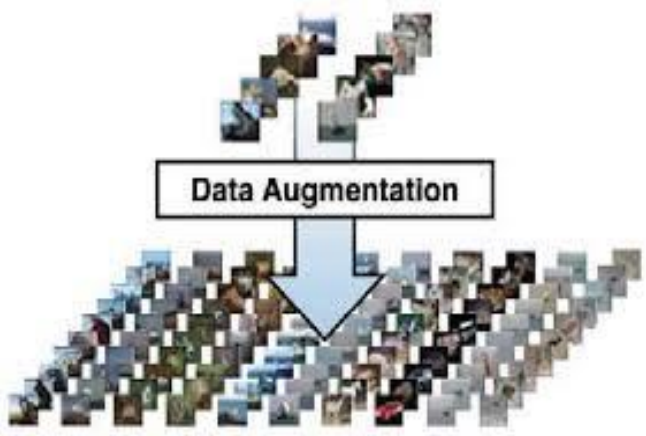

Fig.15. Data augmentation

The data augmentation techniques such as flip, rotation, varying brightness, and so on are being applied to the images and then the model is being trained on these images. By doing so, we can detect recognize the person and predict even from the side angle of the person or even in less brightness or more brightness. It will also reduce the number of images that have to be taken.

The simple data augmentation techniques that can be applied to images are :

1. Geometric transformations: Flip, rotation, scale, crop, translation, and so on are a few examples of geometric transformations. Fig.16. shows the data augmentation done using geometric transformations.

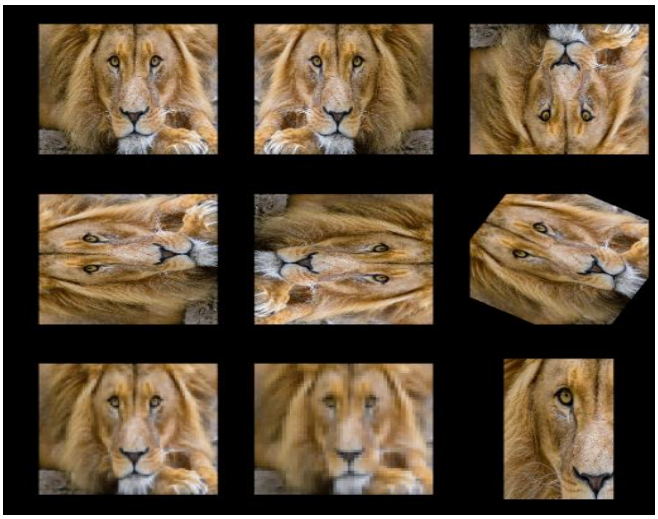

Fig.16. Data augmentation using geometric transformations

2. Colour space transformations: Noise injection, varying brightness, color casting, and so on are examples of color space transformations. Fig.17. shows the data augmentation done using geometric color space transformations.

Fig.17. Data augmentation using Colour space transformations

7. CNN ARCHITECTURES

Convolution Neural Networks are a class of neural networks used in analyzing visual imagery. They can be used for image classification, object detection, segmentation, and other image processing tasks.

Many CNN architectures were developed and can be used to improve the accuracy of predictions in custom CNN models. The CNN architectures are as follows:

1. VGG Net
2. ResNet
3. Dense Net
4. Iception Net
5. Xception Net

7.1 VGG Net [5]

It uses 3x3 convolution layers that are stacked one above the other in increasing depth. Max Pooling is used to reduce the volume size. It consists of two fully connected layers and uses a softmax classifier. VGG16 and VGG19 are VGG nets where 16 and 19 refer to the number of weighted layers in the network and are consisted very deep. It is used mainly used in deep learning image classifications.

However, VGG Net is very slow to train and the weights of the network architecture are large. VGG Net is also prone to the vanishing gradient problem.

7.2 ResNet [6]

It is short for Residual Networks and used quite often in computer vision tasks. The use of skip connections is to mitigate the vanishing gradient problem by providing an alternate shortcut for the gradient and adding output from the previous layers to the later layers.

ResNets have many variants. However, the basic idea is to consider the output of a Conv2D layer and add the output to the Conv2D layer and then send this as an input to the next layers. The stacked layer is added along with its input layer. The training is usually done using the batch normalization layers, which will facilitate the model to train faster by boosting the weights and minimize the vanishing gradient problem. The size of the model is smaller because the global average pooling is used rather than fully connected layers, which reduces the model size dramatically.

7.3 Dense Net [7]

The connectivity pattern is simplified in Dense Nets. They require a fewer number of parameters and the layers are quite narrow and add a small ser of feature maps. They do not add the output feature maps of a layer but concatenate them with the incoming feature maps. They usually have more intermediate connections. The vanishing gradient problem is minimized as well. Smaller filter counts can be used and are good for smaller models.

7.4 Inception Net [8]

The main goal of Inception Net is to behave as a multi-level feature extractor. It computes 1x1, 3x3, and 5x5 convolutions in the network module, and the output of these are stacked along the dimension of the channel before being fed into the subsequent layers. Inception Nets are preferred because they are wider and many such layers can be stacked and the output parameters to be trained are much less.

7.5 Xception Net [9]

Xception Net stands for extreme inception. It is an improvisation of the Inception Net in terms of computational efficiency. The main difference between the two is that in Inception Net normal convolutional operations are performed but in Xception Net, depth-wise separable convolutions are performed. By using depth-wise separable convolutions, the computational complexity can be reduced because each kernel is two-dimensional and they are convoluted over a single channel only. Xception Nets can be used in low-power devices and have fewer computations in the Conv Layer.

8. FACE DETECTION

Face detection is a computer technology that has wide areas of applications in everyday life. They are solely used for the identification of human faces in digital images. They can specifically be regarded as a case of object class detection. Face detection can be done using neural networks which is part of artificial intelligence, where a machine can learn by itself without explicitly being programmed by a programmer. The concept of face detection is used in biometrics and often as a part of the facial recognition system. Here, it is essential to train the machine with the images of the residents so that machine will learn to detect the faces of residents by itself.

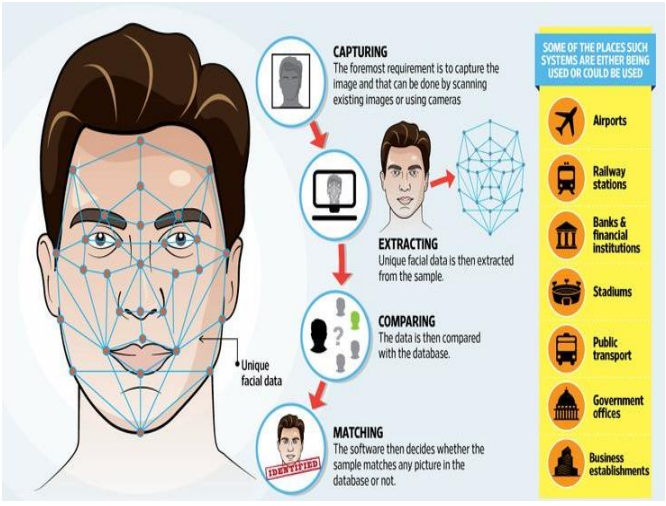

Fig. 18. Face Detection

9. PROPOSED MODEL

Locks have played an important role in the security of residential and industrial sectors. Especially at houses, they have acted as a secure foundation for centuries. The locks are usually taken for granted and eventually they have faded into the background of security, even though the significance held by them is still great. Many people don't realize how important locks are, until they are either locked out of their own houses, or they find themselves unable to lock their residential doors. It is also a tedious job especially for the senior citizens to get up and open the door. The chances of theft are also relatively high in the case of these traditional door lock systems.

To overcome the disadvantages of fingerprint door locks, Bluetooth door locks, face recognition can be used as a mechanism of door unlock. In this system, images of the people residing in the house will be fed and it will be trained using a ResNet architecture. If the person is a resident of the house, the face recognition will be performed and the door will unlock and the person is let in.

In case the person is not a member residing in that house but comes to visit the house, the person's face will be recognized as a human and an alarm will ring indicating that some human being who is not a member of that house is at the door and an email alert will be sent to the owner consisting of the image of the person in front of the door.

10. METHODOLOGY

Tensor flow (Keras) is used for image recognition.

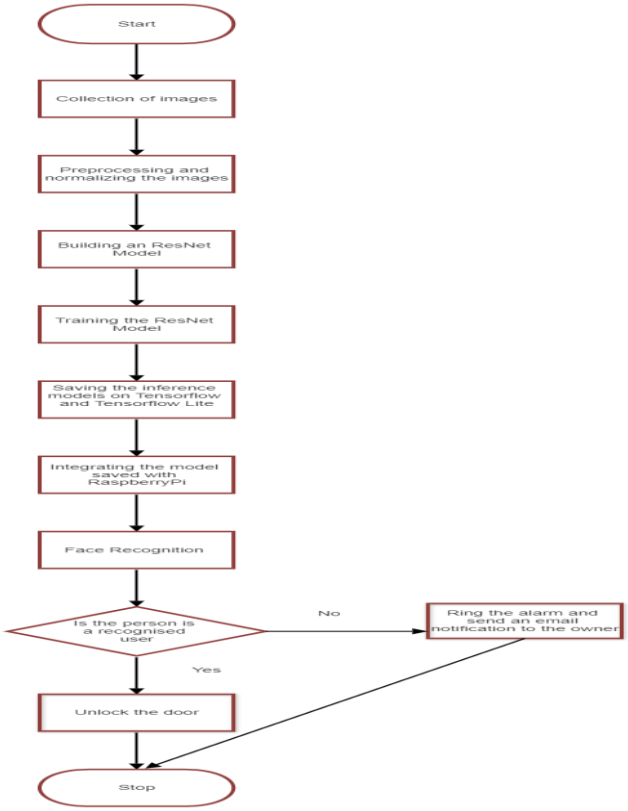

Fig. 19. Flow Chart

The techniques that are being implemented are:

(a) Collection of images
(b) Pre-processing and normalization of images
(c) Building the ResNet model
(d) Training the ResNet model
(e) Saving the inference models in both Tensorflow and Tensorflow Lite
(f) Integrating the model saved in Tensorflow Lite with RaspberryPi
(g) Face recognition
(h) Unlocking the door or generating an alarm and sending an email notification to the owner.

Fig.19. shows the flowchart of the methodology being followed to implement the door automation system using neural networks.

11. OBSERVATIONS

When a person is detected, the model will recognize the human being by using face detection. If it is a resident the door will automatically unlock. Otherwise, an alarm will ring indicating that the person is an outsider and an email notification with the person's image will be sent to the owner. Thus be used to prevent any kind of drawbacks that arise in the case of traditional door lock systems.

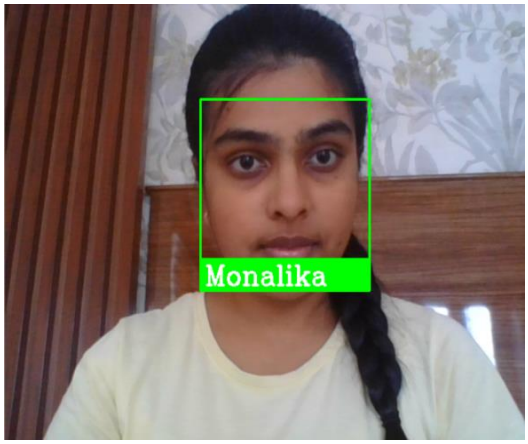

Fig. 20. Face detection and recognition

Fig.20. shows the implementation of face recognition. On integration with Raspberry Pi, the door will be unlocked for the authorized user. In case, the person is not a recognized user, the alarm will ring and an email notification will be sent to the owner.

12. CONCLUSION

The proposed model can be used to provide a safe and secure residential home there, solving all the drawbacks of the traditional door lock systems. By using the ResNet architecture for face recognition, it improves the accuracy to a great extent. The size of the model is also less and hence, it is easier to integrate it with RaspberryPi. The accuracy of the model is over 98%.

The requirement to train the system is also quite less and hence is optimal and feasible. It also enables a full automation system with minimal human effort or interaction, such as opening the door and it eliminates a majority of the security threats.

Door automation using face recognition is also no contact and is easy to use even for age groups. It also makes it easier to track down thieves and trespassers because if the person is not recognized as a resident, then the email alert with the person's image will be sent to the owner.

13. REFERENCES

[1] Arpita Mishra, Siddharth Sharma, Sachin Dubey, S.K. Dubey, "Password Based Security Lock System", International Journal of Advanced Technology in Engineering and Science, 2011.

[2] Muhammad Yusry Bin Ishak, Samsiah Binti Ahmad, Zalikha Zulkifli, " Iot Based Bluetooth Smart Radar Door Systems Via Mobile Apps", 1st International Conference on Artificial Intelligence and Data Sciences (AiDAS), 2019

[3] Wu Ping, Wu Guichu, Xie Wenbin, Lu Jianguo, Li Peng, "Remote Monitoring Intelligent System Based on Fingerprint Door Lock", International Conference on Intelligent Computation Technology and Automation, 2010

[4] S.Vinitha, R. Karthiyan, "IRIS Biometric Recognition for Person Identification and Security", International Journal of Scientific Research in Computer Science, Engineering and Information Technology, Volume 2, 2017

[5] Karen Simonyan, Andrew Zisserman, "Very Deep Convolutional Networks for Large Scale Image Recognition" Conference paper, ICLR, 2015

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, "Deep Residual Learning for Image Recognition", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778

[7] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, "Densely Connected Convolutional Networks", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700-4708

[8] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, "Going Deeper With Convolutions", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1-9

[9] François Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

