Annexure | I

IOT BASED ATTENDANCE SYSTEM USING FACE RECOGNITION

Project Report

Submitted in partial fulfilment of the requirement for the award of the Degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS & COMMUNICATION ENGINEERING

Session: 2017-2021

Submitted by:

Anurag Shrivastava (1706011)

Ashwin Raj (1706013)

Gourav Kumar (1706019)

Murari Kumar (1706025)

Nilesh Kumar Tiwari, (1706029)

Project Guide:

Dr. Santosh Kumar Verma

Department of Electronics & Communication Engineering

BIT SINDRI

Department of Higher, Technical Education & Skill Development Govt. of Jharkhand.

P.O. Sindri Institute, Dhanbad-828123 (Jharkhand)

Annexure | II

Department of Electronics & Communication Engineering
BIT SINDRI
Department of Higher, Technical Education & Skill Development Govt. of Jharkhand.
P.O. Sindri Institute, Dhanbad-828123 (Jharkhand)

CERTIFICATE

This is to certify that the Project Report entitled "IoT Based Attendance System using Face Recognition" submitted by Anurag Shrivastava (1706011), Ashwin Raj (1706013), Gourav Kumar (1706019), Murari Kumar (1706025), Nilesh Kumar Tiwari (1706029) for the partial fulfilment of the requirement for the award of "BACHELOR OF TECHNOLOGY" in ELECTRONICS & COMMUNICATION ENGINEERING (Session 2017-21) has been carried out under my supervision and guidance. It is certified that the report embodies the results of the work carried out by them within the prescribed period.

APPROVED FOR SUBMISSION

Project Guide	**Head of the Department**
Dr. Santosh Kumar Verma **Department of ECE** **BIT Sindri, Dhanbad**	**Dr. Imteyaz Ahmad** **Department of ECE** **BIT Sindri, Dhanbad**

Annexure | III

ACKNOWLEDGEMENT

We express our sincere gratitude to **Dr. Santosh Kumar Verma** Sir for his valuable guidance and timely suggestions during the entire duration of our project work, without which this work would not have been possible. We would also like to convey our deep respect to HOD and all other faculty members of Electronics and Communication Engineering Department who have bestowed their great effort and help at appropriate times without which it would have been very difficult on our part to finish this work. Finally, we would also like to thank our friends for their advice and pointing out our mistakes.

Anurag Shrivastava (1706011)

Ashwin Raj (1706013)

Gourav Kumar (1706019)

Murari Kumar (1706025)

Nilesh Kumar Tiwari (1706029)

ABSTRACT

In this work, the idea of face recognition is investigated and executed to develop an attendance system. The application (here a web app) is powered with a camera-enabled device to learn and recognise the subject. The main contribution is implementing object recognition algorithm, MobileNet SSD, and connecting the program with an external camera device. These findings are the basis that inspired to restructure the attendance taking system to some extent. A remarkable development throughout the process is observed and this idea has immense possibilities to be implemented widely to develop online-proctored assessments, IoT based securities, surveillance system, etc. A range of industries/institutes is identified where it could be deployed to ease work and provide robust solution to many problems.

Keywords: Machine Learning, Face Recognition, IoT

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

"*Just as electricity transformed almost everything 100 years ago, today I actually have a hard time thinking of an industry that I don't think Artificial Intelligence will transform in the next several years*." ~ ANDREW NG

The concept of the *Artificial Intelligence* goes back to the 1950s. In the 1980s Machine Learning begins to gain popularity. Around the second decade of 21st Century, explosive development of the core computational part of AI i.e., Deep Learning drives larger progress and enables machine do much more than what was initially thought of. Deep Learning drives Machine Learning which ultimately makes AI ride high. AI is the science of making things work smartly. *A broad term for getting machines human intelligence and much beyond that.* One such wonder of giving machines the power of vision is explained by convolution neural network.

Convolution Neural Network is the development of mathematics in a very specialized manner to do the magic with images & videos. With convolution network in an application, it bestows the power of vision and brain to the machine to recognise, verify and classify things. These networks have been the most influential work of research and innovation in the field of computer vision. Today, this technology is on a surge of developments and advancements by some giant technology companies to enhance their product and business. Apple brought the feature of face recognition for unlocking their smartphones, Google uses this to power their photo-based search engines, Amazon uses this for their product recommendations, Boston Dynamics develops high-end, heavy-duty robots for industrial & military purposes and several other companies power their infrastructure with computer vision to redefine their products.

1.1 LITERATURE SURVEY

The great computer scientist, Alan Turing in 1947 predicted and thought of creating systems that can learn from their own experiences. Following to that thought in 1952, *Arthur Samuel*, a computer scientist at IBM coined the term "*machine learning*". In pursuit of developing methods to create better algorithms, work on deep neural network started in mid 1960s.

In 1986, *Hinton et. al.* published paper entitled "*Improvements in Shape Recognition and Word Prediction*". In 1989, *Yann LeCun*, one of the greatest pioneers created machines that could read handwritten digits. He combined ConvNet with backpropagation algorithm to achieve high accuracy on the algorithm. Later, he escalated the field of Deep Learning by publishing another research paper on "*Gradient Based Learning Applied to Document Recognition*". Since *Krizhevsky et al.*, 2012 introduced *AlexNet* in 2012 in the ImageNet challenge, there have been immense improvement of accuracy of convolution networks. *Szegedy et al.*, introduced *GoogLeNet* in 2014 which was an improvement of AlexNet. This network reduced the number of parameters greatly. Also, in 2014 *VGGNet* was introduced by *Simonyan & Zisserman.* VGGNet achieved great performance because of the depth of the network architecture.

Table 1: Error rates of top CNN models in ImageNet Challenge

Model	Top Error Rate
AlexNet	15.3 %
GoogLeNet	6.67 %
VGGNet	7.32 %

Our work comprises exhaustive use of MobileNet. In 2017, a research team of *Andrew G. Howard et al*. at Google presented a paper of *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.* This neural network presented it capability of high performance in image classification with much reduced calculations and complexities of the network. This convolution network performed great in various mobile and embedded vision applications.

1.2 PROBLEM STATEMENT

What we know?

We realize the involvement of technology in our lives. We understand that how business models are built around them. We heed how different fields like educational, medical, business, researches are improving by technologies. Their applications are in tremendous use lately with the development of Artificial Intelligence, the Internet of Things, platforms for software, etc.

What do we need to know?

In this work, we explore the field of the Internet of Things and Deep Learning focused on image classification to develop a face recognition application. We modulate this theory with the application of the "Attendance System."

Why do we need to know?

We understand that in the future innovation will make its place everywhere. Traditional and unsmart things would be replaced with modern and smart solutions. There would be greater demand for innovation which would ease work-handlings. To go along with this vision, we present the idea of modernizing the traditional attendance-taking system. With the integration of IoT and AI, this idea is worth many different applications which we talk about later this report.

1.3 OBJECTIVE

We aim to develop an application that would address our problem statement of revamping the existing attendance-taking system. This application would rule out any chance of getting false-positive results. This application would save time immensely when the audiences are large in number.

2. ESSENTIAL PRELIMINARIES

To begin with, we need to understand the basics of the Internet of Things, Machine Learning, and Deep Learning.

2.1 INTERNET OF THINGS

Quoting Wikipedia: The **Internet of things** (**IoT**) describes the network of physical objects—a.k.a. "things"—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the Internet.

Just the simple way to define it is: Real-time physical data can inform actions based on what is happening right now. Earlier, RFID was a key IoT technology. Unsurprisingly, the Internet of Things (IoT) is at a much different stage of development than its foundational technology. Lately, IoT has become one of the most important technologies of the 21st century. Now that we can connect everyday objects—home appliances, cars, drones—to the internet via embedded devices, seamless communication is possible between people, processes, and things. Automatic cooking of food, for the right length of time, self-driving cars whose complex sensors detect objects in path, fitness and activity tracking through body sensors which tracks heart rate, respiration rate, number of steps, speed and multiple other things have been truly signs of wonders.

2.1.1 HOW DOES IOT WORK?

Devices and objects with built in sensors are connected to an Internet of Things platform, which integrates data from the different devices and applies analytics to share the most valuable information with applications built to address specific needs. The platform can be enabled with the power of analysis to use sensors data to augment the experience. Multiple technologies like data security, artificial intelligence, data analysis can be integrated to make a robust platform.

How IoT benefits the world?

IoT is being implemented in virtually all sectors, but those that are using it most include healthcare, utilities, manufacturing, agriculture, smart cities, public safety and retail. While it's still in its nascent stages and with large scale deployment of IoT devices the interconnected sensors and technology will offer numerous benefits:

- Automation
- Safety and Security
- Improved customer experiences
- Decision Analysis

The world sees wonders happening around the Internet of Things as their applications are wide over a wide range of sectors like IoT in smart homes, IoT in Agriculture, IoT in smart cities, IoT in healthcare, IoT in industries, logistics, IoT in education, IoT in weather monitoring, IoT in Communication & Media etc. The ease and cheapness of availability of this technology have put itself in great demand across all platforms for automation and time-saving.

We believe that the concurrence of the IoT and AI is capable of creating devastating developments in this technological era.

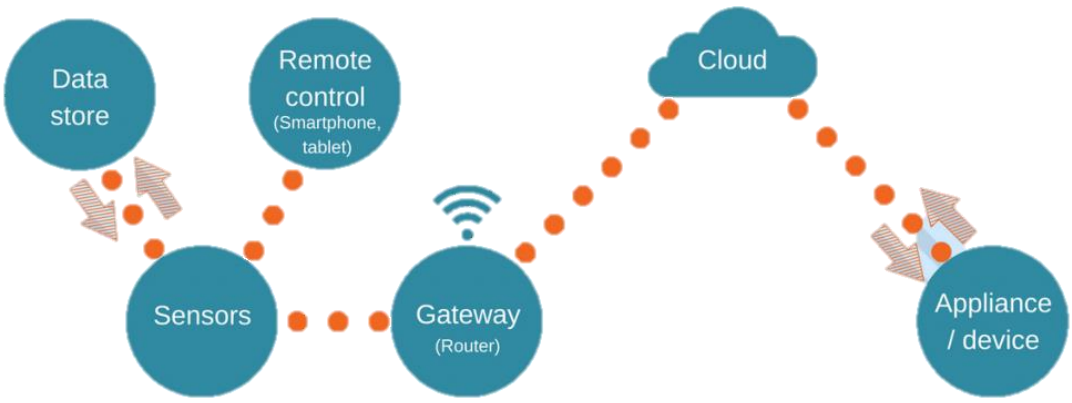


Figure 1: Architecture of the Internet of Things

2.2 MACHINE LEARNING

Machine Learning is the dominant AI technique disclosed in patents and is included in more than one-third of all identified inventions. It is a subfield of computer science that is concerned with building algorithms that, to be useful, rely on a collection of examples of some phenomenon. These examples can come from nature, be handcrafted by humans, or be generated by another algorithm. Machine learning can also be defined as the process of solving a practical problem by 1) gathering a dataset, and 2) algorithmically building a statistical model based on that dataset. That statistical model is assumed to be used somehow to solve the practical problem. Tom Mitchell, a famous computer scientist defines Machine learning as A computer program is said to learn from experience *E* with respect to some task *T* and some performance measure *P*, if its performance of *T* as measured by *P*, improves with *E*.

The discipline of machine learning employs various approaches to teach computers to accomplish tasks where no fully satisfactory algorithm is available. In cases where vast numbers of potential answers exist, one approach is to label some of the correct answers as valid. This can then be used as training data for the computer to improve the algorithm(s) it uses to determine correct answers. For example, to train a system for the task of digital character recognition, the MNIST dataset of handwritten digits has often been used. A few examples of machine learning cases:

- Database Mining:
  Large datasets from growth of automation/web. E.g., Web click data, medical records, biology, Engineering.
- Applications that can’t be programmed by hand:
  Autonomous Vehicles, Handwritten Recognition, Natural Language Programming (NLP), Computer Vision.
- Self-customizing programs:
  Amazon, YouTube, Flipkart, Netflix recommendations. - Understanding human learning (brain, real AI).

Classical Machine Learning is often divided into two categories- *Supervised Learning* and *Unsupervised Learning*. Others: *Reinforcement Learning*.

2.2.1 SUPERVISED LEARNING

Given inputs (features) to a model, model is pre-trained with the expected output label. The system then must use this data to predict the future unseen inputs. Currently this is the most studied area. The goal of a supervised learning algorithm is to use the dataset to produce a model that takes a feature vector x as input and outputs information that allows deducing the label for this feature vector. For instance, the model created using the dataset of people could take as input a feature vector describing a person and output a probability that the person has cancer.

2.2.2 UNSUPERVISED LEARNING

Unsupervised learning is a set of statistical tools for scenarios in which there is only a set of features and no targets. Therefore, we cannot make predictions, since there are no associated responses to each observation. Instead, we are interested in finding an interesting way to visualize data or in discovering subgroups of similar observations. Unsupervised learning deals with problems in which *data doesn't have labels*. That property makes it very problematic for many applications. The absence of labels representing the desired behaviour for your model means the absence of a solid reference point to judge the quality of your model. Unsupervised learning tends to be more challenging, because there is no clear objective for the analysis, and it is often subjective. Additionally, it is hard to assess if the obtained results are good, since there is no accepted mechanism for performing cross-validation or validating results on an independent dataset, because we do not know the true answer.

2.2.3 REINFORCEMENT LEARNING

Reinforcement learning is another branch of machine learning, in which an "agent" learns to maximize "rewards" in an environment. An environment can be as simple as a tic-tac-toe board in which an AI player is rewarded for lining up three Xs or Os, or as complex as an urban setting in which a self-driving car is rewarded for avoiding collisions, obeying traffic rules, and reaching its destination, similar goes with rewarding a bot when it does the task correctly and likewise punishing it. The agent starts by taking random actions. As it receives feedback from its environment, it finds sequences of actions that provide better rewards.

2.3 DEEP LEARNING

Deep learning is an AI function that mimics the workings of the human brain in processing data for use in detecting objects, recognizing speech, translating languages, and making decisions. Deep learning is a subset of machine learning in artificial intelligence that has networks capable of learning unsupervised from data that is unstructured or unlabelled. Also known as deep neural learning or deep neural network. Deep learning is often compared to the brains of humans and animals.

Deep learning, a subset of machine learning, utilizes a hierarchical level of artificial neural networks to carry out the process of machine learning. The artificial neural networks are built like the human brain, with neuron nodes connected together like a web. While traditional programs build analysis with data in a linear way, the hierarchical function of deep learning systems enables machines to process data with a nonlinear approach.

Artificial Intelligence (AI) and Machine Learning are being used, not only as personal assistants for internet activities, but also to answer phones, drive vehicles, provide insights through Predictive and Prescriptive Analytics, and so much more. This creates immense opportunities for innovation to provide solutions that ultimately drive the world into a better place to live in.

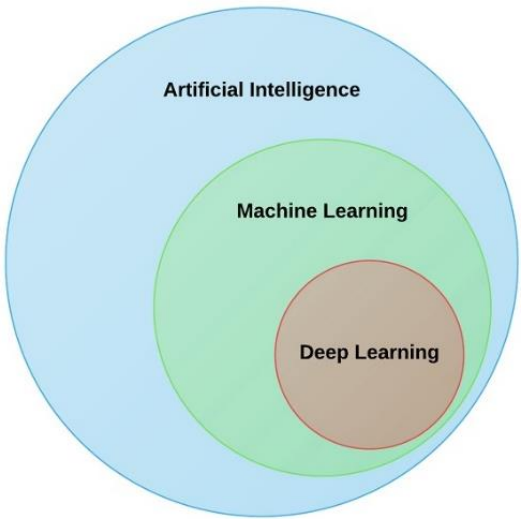


Figure 2: Relating Artificial Intelligence, Machine Learning, and Deep Learning

2.4 CONVOLUTION NEURAL NETWORK

Before we discuss the implementation of the project, let's delve into the core functionality of computer vision and understand it. It will help us understand the science behind how machines can see the world.

Image classification is the task of taking an input image and outputting a class (a cat, dog, etc) or a probability of classes that best describes the image. For humans, this task of recognition is one of the first skills we learn from the moment we are born and is one that comes naturally and effortlessly as adults. Without even thinking twice, we're able to quickly and seamlessly identify the environment we are in as well as the objects that surround us. When we see an image or just when we look at the world around us, most of the time we are able to immediately characterize the scene and give each object a label, all without even consciously noticing. These skills of being able to quickly recognize patterns, generalize from prior knowledge, and adapt to different image environments are ones that we do not share with our fellow machines.

Figure 3: (A) What a human eye perceives (B) How a computer perceives the image

We see the colours; computers see pixels values. We find shapes; computers find patterns between those numbers. One common thing about how human beings and computers learn is both of them learn through experience. We know leaves are green and have some shape because we have seen many of their kind. Similarly, when a deep learning algorithm is trained to learn shapes, colours of leaves, it automatically recognises any future instances when it looks at leaves. We are bestowed with natural power of vision and giving machines the same power of vision is no less than a godly thing to do. That's the wonder of science.

2.4.1 BIOLOGICAL ANALOGY OF HUMAN EYE

When we think about the term convolutional neural networks, it seems to be some analogical reference of neurons in human brain. Indeed, CNNs do take a biological inspiration from the visual cortex. *Dr. Hubel* and *Dr. Wiesel* worked on the area of Sensory Processing. In which, they inserted a micro-electrode into the primary *visual cortex* of a partially anesthetized cat so that she can't move. Through the micro-electrode they found that some neurons fired very rapidly by watching the lines at specific angles, while other neurons responded best to the lines at different angles. Some of these neurons responded to light and dark patterns differently, while other neurons responded to detect motion in the certain direction. This work is prime for the concept of CNN.

2.4.2 DETAILED STUDY OF CNN

Let's look at how convolution helps in detecting shapes:

Edge Detection: Using convolution operation on image with Sobel Kernel we can detect the edges. Look at the following image:

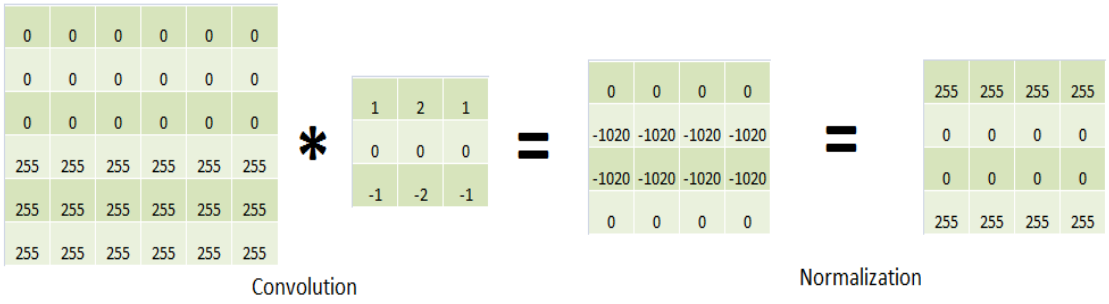
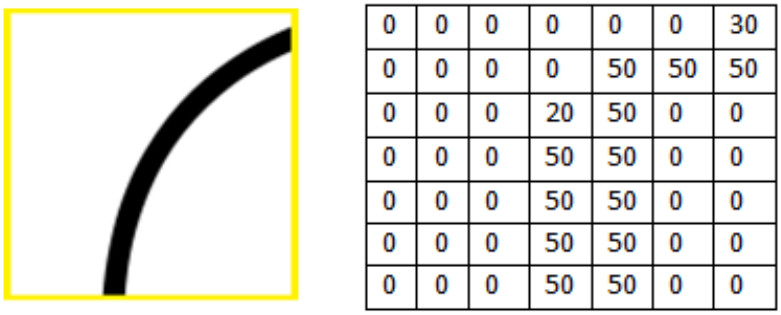


Figure 4: Detection of edge through convolution



0	0	0	0	0	0	30
0	0	0	0	50	50	50
0	0	0	20	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0
0	0	0	50	50	0	0

Figure 5: Mathematical visualization of the image shown

Max Pooling: It is used to detect, where the objects are located in the Image based on the output of each cluster of neurons in previous layer. As the face is detected where ever it is; doesn't depend on the location of face in image.

The objective of the Convolution Operation is to *extract the high-level features* such as edges, from the input image. ConvNets need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, colour, gradient orientation, etc. With added layers, the architecture adapts to the High-Level features as well, giving us a network, which has the wholesome understanding of images in the dataset, similar to how we would.

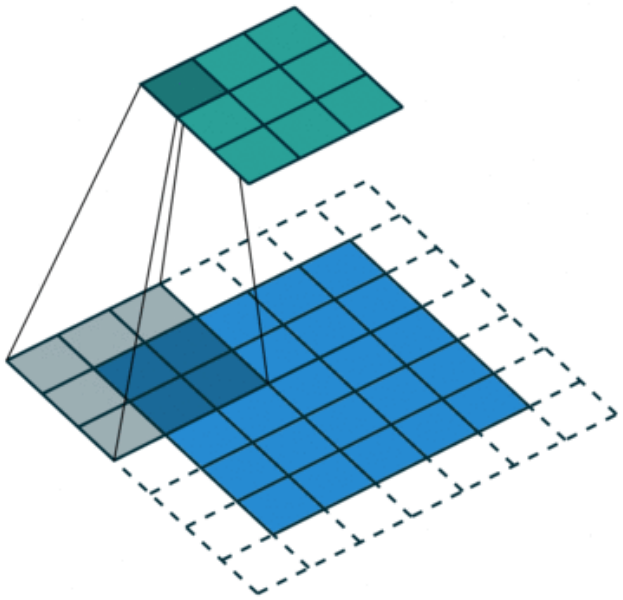


Figure 6: Convolution operation with stride

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

There are two types of Pooling: Max Pooling and Average Pooling. **Max Pooling** returns the *maximum value* from the portion of the image covered by the Kernel. On the other hand, **Average Pooling** returns the *average of all the values* from the portion of the image covered by the Kernel.

Max Pooling also performs as a *Noise Suppressant*. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. On the other hand, Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that *Max Pooling performs a lot better than Average Pooling.*

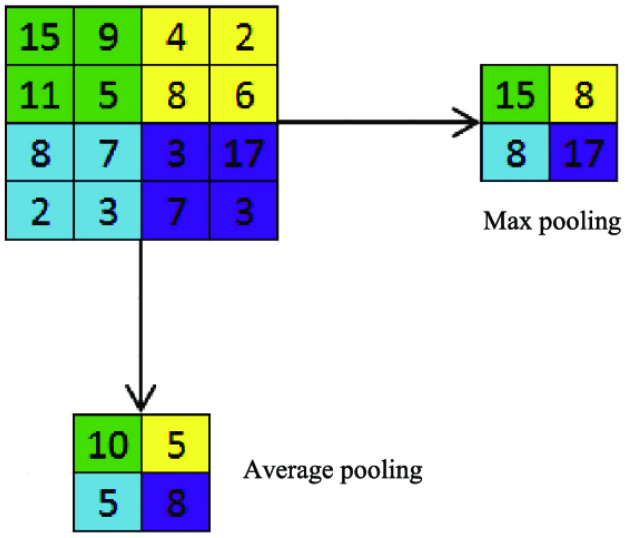


Figure 7: Operation of Max Pooling & Average Pooling

Performing Classification: Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that space.

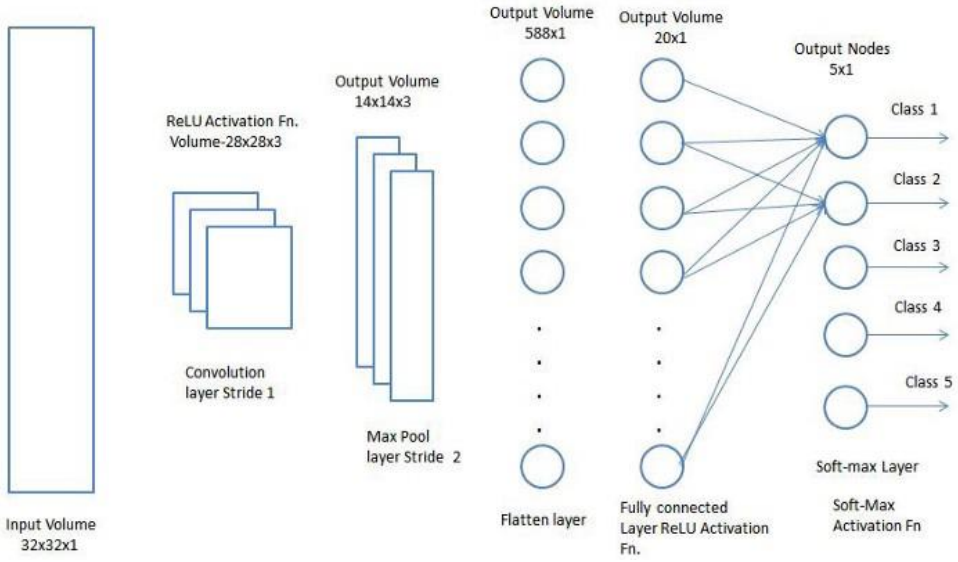


Figure 8: Architecture of a typical convolution network performing classification

Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the *SoftMax Classification* technique.

2.4.3 HAVING FUN WITH CNN

There's always a large gap between understanding technology from a researchers' point of view to a customers' point of view. To bridge this gap, and to create learning AI more fun for a user, several innovations are being done. One of the great experiments is QuickDraw by Google created to engage people in a fun activity, at the same time making them curious to learn and understand about Deep Learning or more specifically, Computer Vision. Let's understand more about it.

Figure 9: Google QuickDraw- https://quickdraw.withgoogle.com/

Over 15 million players have contributed millions of drawings playing Quick, Draw! These doodles are a unique data set that can help developers train new neural networks, help researchers see patterns in how people around the world draw, and help artists create things that haven't been begun to think of. That's why they're open-sourcing them, for anyone to play with.

Figure 10 show the steps followed by the game to recognise what a user has drawn. Let John opens the website to play the game. The website asks him to draw a pizza within 20 seconds. The algorithm behind this game starts recognising the art made in the whitespace by John. The algorithm searches its closest match from the millions of accepted answers for pizza. Had John drawn something that looked unlike pizza, the algorithm would have

given the closest match indication. For this example, The pizza drawing made by John has the first 3 closest matches: a pizza itself, a soccer ball, and a basketball.

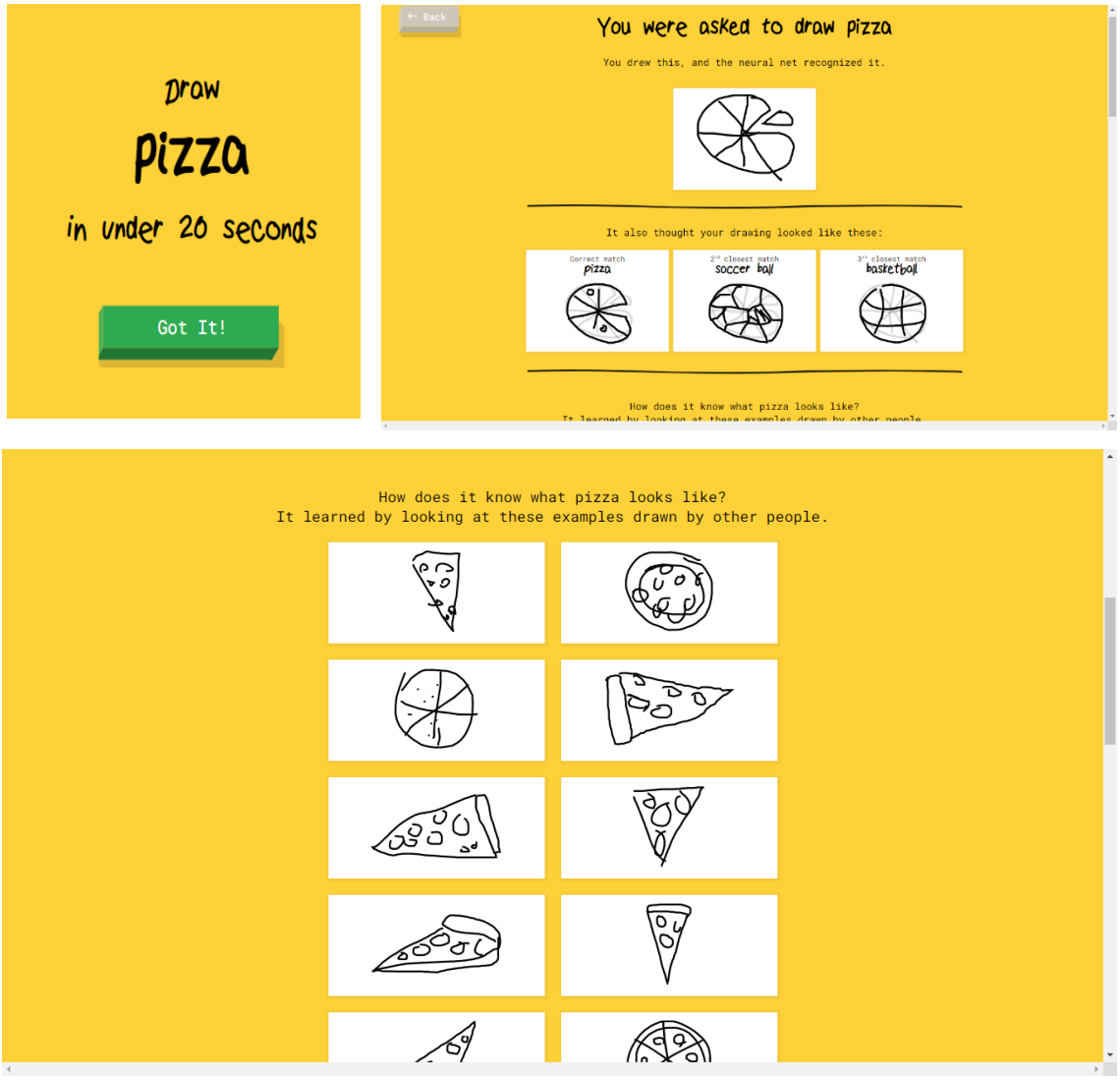


Figure 10: The workflow behind Google QuickDraw.

The understanding of our project is very similar to this game. We will talk about this later.

2.5 TRANSFER LEARNING

Transfer learning is the improvement of learning in a new task through the *transfer of knowledge from a related task that has already been learned*. While most machine learning algorithms are designed to address single tasks, the development of algorithms that facilitate transfer learning is a topic of ongoing interest in the machine-learning community.

Human learners appear to have inherent ways to transfer knowledge between tasks. That is, we recognize and apply relevant knowledge from previous learning experiences when we encounter new tasks. The more related a new task is to our previous experience, the more easily we can master it. Common machine learning algorithms, in contrast, traditionally address isolated tasks. Transfer learning attempts to change this by developing methods to transfer knowledge learned in one or more source tasks and use it to improve learning in a related target task. Techniques that enable knowledge transfer represent progress towards making machine learning as efficient as human learning.

Transfer learning offers the chance for CNNs to learn with limited data samples by transferring knowledge from models pre-trained on large datasets. In this paper, the researchers proposed attentive feature distillation and selection (AFDS), which not only adjusts the strength of transfer learning regularisation but also dynamically determines the important features to transfer. According to the researchers, by deploying AFDS on ResNet-101, *a state-of-the-art computation reduction has been achieved at the same accuracy budget, outperforming all existing transfer learning methods.*

In the paper "A Study on CNN Transfer Learning for Image Classification," the authors compared accuracies check of two convolution networks, one pre-trained (A) and the other trained from scratch (B). Dataset A (CIFAR-10) is a pretrained network with 10000 images per training class (10), while Dataset B is trained from scratch with 1000 images per class.

Table 2: Comparing Transfer Learning (A) with scratch learning (B). Source credit: The paper titled "*A Study on CNN Transfer Learning for Image Classification*"

Dataset	Accuracy (%)
CIFAR-10 (A)	70.1
CIFAR-10 (B)	66.1

An observation that is underlined under this experiment is increasing the number of training steps increased the classification accuracy of the model. Also, where there are limitations such as no accessibility of GPU, unavailability of large datasets, limited research team and knowledge and lesser time, we can always adopt models pre-trained on large datasets with great performance by big organisations. We can do on-the-top training to serve our purpose, anytime.

2.5 MOBILENET

We now discuss a class of efficient model called MobileNet for mobile and embedded vision applications. MobileNets primarily focus on optimizing for latency but also yield small networks. MobileNets are built primarily from depth-wise separable convolutions initially introduced in and subsequently used in Inception models to reduce the computation in the first few layers. Flattened networks build a network out of fully factorized convolutions and showed the potential of extremely factorized networks.

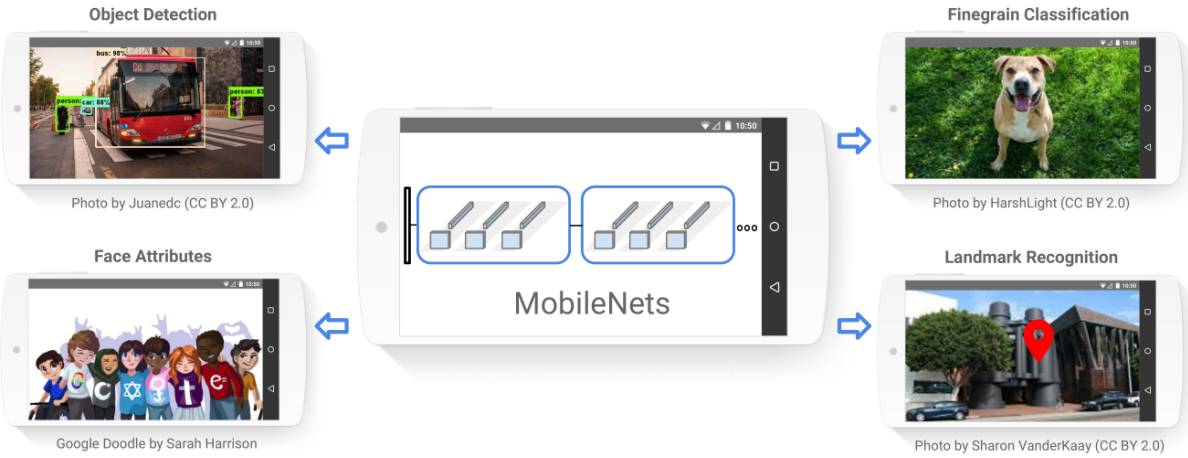


Figure 11: MobileNet can be deployed to perform various recognition like tasks like shown above.

The depth-wise separable convolution is a form of factorized convolution which factorise a standard convolution at any level into depth-wise and a 1 x 1 convolution called a point wise convolution. Using factorisation at every level reduces cost of computation drastically by forming two separate layers for filtering (depth-wise convolution) and combining the outputs (point wise convolution). MobileNet uses both batch normalization (BN) and ReLU nonlinearities for both layers. Depth-wise convolution is extremely efficient relative to standard convolution. However, it only filters input channels, it does not combine them to create new features. So, an additional layer that computes a linear combination of the output of depth-wise convolution via 1 × 1 convolution is needed in order to generate these new features.

2.5.1 NETWORK ARCHITECTURE

The MobileNet structure is built on depth-wise separable convolutions as mentioned in the previous section except for the first layer which is a full convolution. By defining the network in such simple terms, we are able to easily explore network topologies to find a good network. The table below defines network architecture of MobileNet.

Table 3: Architecture of MobileNet

Type/Stride	Filter Shape	Input Size
Conv / s2	3 x 3 x 3 x 32	224 x 224 x 3
Conv dw / s1	3 x 3 x 32 dw	112 x 112 x 32
Conv / s1	1 x 1 x 32 x 64	112 x 112 x 32
Conv dw / s2	3 x 3 x 64 dw	112 x 112 x 64
Conv / s1	1 x 1 x 64 x 128	56 x 56 x 64
Conv dw / s1	3 x 3 x 128 dw	56 x 56 x 128
Conv / s1	1 x 1 x 128 x 128	56 x 56 x 128
Conv dw / s2	3 x 3 x 128 dw	56 x 56 x 128
Conv / s1	1 x 1 x 128 x 256	28 x 28 x 128
Conv dw / s1	3 x 3 x 256 dw	28 x 28 x 256
Conv / s1	1 x 1 x 256 x 256	28 x 28 x 256
Conv dw / s2	3 x 3 x 256 dw	28 x 28 x 256
Conv / s1	1 x 1 x 256 x 512	14 x 14 x 256
5 x (Conv dw/ s1 Conv / s1)	3 x 3 x 512 dw 1 x 1 x 512 x 512	14 x 14 x 512 14 x 14 x 512
Conv dw / s2	3 x 3 x 512 dw	14 x 14 x 512
Conv / s1	1 x 1 x 512 x 1024	7 x 7 x 512
Conv dw / s2	3 x 3 x 1024 dw	7 x 7 x 1024
Conv / s1	1 x 1 x 1024 x 1024	7 x 7 x 1024
Avg Pool / s1	Pool 7 x 7	7 x 7 x 1024
FC / s1	1024 x 1000	1 x 1 x 1024
Softmax / s1	Classifier	1 x 1 x 1000

Table 4 compares full MobileNet to the original GoogleNet and VGG16. MobileNet is nearly as accurate as VGG16 while being 32 times smaller and 27 times less compute intensive. It is more accurate than GoogleNet while being smaller and more than 2.5 times less computation.

Table 4: MobileNet comparison with popular models

Model	Accuracy (%)	Number of Multiplication-Addition (in millions)	Number of Parameters (in millions)
Default MobileNet	70.6	569	4.2
GoogLeNet	69.8	1550	6.8
VGG 16	71.5	15300	138

3. EXPERIMENT

Here we elaborate working of our model through an experiment.

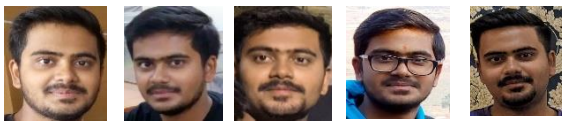
3.1 DATASETS

To train model for image classification (or face recognition) we used MobileNet as our convolutional network. We downloaded the model topology. MobileNet was pre-trained on ImageNet dataset with 1.2 million images. Each image is labelled with one of 1000 different classes.

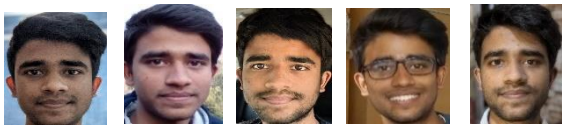
On the top of this architecture, we trained our dataset to develop it to serve the purpose of face recognition. Our dataset in this experiment consisted of faces of 5 people with 40 samples each.

Person 1: Ashwin Raj

Person 2: Anurag Shrivastava



Person 3: Gourav Kumar



Person 4: Murari Kumar

Person 5: Nilesh K Tiwari

Table 5: Dataset created using samples

Sample Name	Number of Samples
Person 1 (Ashwin)	40
Person 2 (Anurag)	40
Person 3 (Gourav)	40
Person 4 (Murari)	40
Person 5 (Nilesh)	40

3.2 EXPLAINING WITH CODES

In the following sub-sections, we try explaining the codes of the project

3.2.1 PRE-PROCESSING

We perform pre-processing with every image captured in order to reduce complexities in calculations. Since an image comes in colours with pixel values ranging from 0 to 255, we normalize them between -1 and 1 by dividing by 127 and subtracting 1. Each image has been resized to be 3 x 224 x 224, each is the expected input to this model. We didn't look for data augmentation by cropping, rotating, reversing, but these can be done for a slight better training of a model.

```
/** Captures a frame from the webcam and normalizes it between -
1 and 1.

   * Returns a batched image (1-element batch) of shape [1, w,
h, c]. */

  capture() {

    return tf.tidy(() => {

      // Reads the image as a Tensor from the webcam <video>
element.

      const webcamImage =
tf.browser.fromPixels(this.webcamElement);

      const reversedImage = webcamImage.reverse(1);

      // Crop the image so we're using the center square of the
rectangular webcam.

      const croppedImage = this.cropImage(reversedImage);
```

```
// Normalize the image between -1 and 1. The image comes in
between 0-255, so we divide by 127 and subtract 1.

      return
batchedImage.toFloat().div(tf.scalar(127)).sub(tf.scalar(1));

    });  }
```

Code Snippet for pre-processing used for normalizing an image. continued….

```
/** Crops an image tensor so we get a square image with no white
space.

   * @param {Tensor4D} img An input image Tensor to crop.

   */

  cropImage(img) {

    const size = Math.min(img.shape[0], img.shape[1]);

    const centerHeight = img.shape[0] / 2;

    const beginHeight = centerHeight - (size / 2);

    const centerWidth = img.shape[1] / 2;

    const beginWidth = centerWidth - (size / 2);

    return img.slice([beginHeight, beginWidth, 0], [224, 224, 3]);

  }
```

Code Snippet for pre-processing used for resizing an image captured

3.2.2 How did we create the dataset?

An array was constructed to hold the tensor values of the samples. This was recognized as X. The Xs were labelled Ys. The Y array was nothing but *one-hot encoding*.

One-hot encoding: Categorical data refers to variables that are made up of label values, for example, a “colour” variable could have the values “red “, “blue, and “green”. One hot encoding is one method of converting data to prepare it for an algorithm and get a better prediction. With one-hot, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns. Each integer value is represented as a binary vector. All the values are zero, and the index is marked with a 1.

Type
AA
AB
CD
AA

Onehot encoding →

Type	AA_Onehot	AB_Onehot	CD_Onehot
AA	1	0	0
AB	0	1	0
CD	0	0	1
AA	0	0	0

Figure 12: Example of One-Hot encoding

```
/* Here we populate dataset with sample images and label them
accordingly
 * xs contains tensor form of images, ys contains one-hot encoded form
as labels
 */
class Dataset {
  constructor() {
    this.labels = []
  }
/* addition of samples into the labels array
 */
  addExample(example, label) {
    // if array is empty
    if (this.xs == null) {
        // array holding tensor form  of the image captured
      this.xs = tf.keep(example);
        //push label into array
      this.labels.push(label);
    } // else if array is non-empty
      else {
      const oldX = this.xs;
      this.xs = tf.keep(oldX.concat(example, 0));
      this.labels.push(label);
      oldX.dispose();
    }
  }
```

Code snippet for populating array with dataset

```
  /*
   * mapping the tensors ofimages to one-hot encoding
   */
  encodeLabels(numClasses) {
    for (var i = 0; i < this.labels.length; i++) {
        // if array of ys is empty
      if (this.ys == null) {
          // one hot encoding as per labels
        this.ys = tf.keep(tf.tidy(
            () => {return tf.oneHot(
                tf.tensor1d([this.labels[i]]).toInt(),
numClasses)}));
      } else {
          // one-hot encoding as per labels
        const y = tf.tidy(
            () => {return tf.oneHot(
                tf.tensor1d([this.labels[i]]).toInt(),
numClasses)});
        const oldY = this.ys;
        this.ys = tf.keep(oldY.concat(y, 0));
        oldY.dispose();
        y.dispose();
      }
    }
  }
}
```

Code snippet for a mapping of samples to correct labels using one-hot encoding

3.2.3 DOWNLOADING TOPOLOGY AND WEIGHTS OF MOBILENET

We download the model of MobileNet in “json” file format containing network topology and weights of the parameters. The layer named 'conv_pw_13_relu' is picked from the network to learn on the top of that with our already created dataset.

```
// loading the architecture of MobileNet

async function loadMobilenet() {

    // downloading the mobileNet topology and weights

  const mobilenet = await
tf.loadLayersModel('https://storage.googleapis.com/tfjs-
models/tfjs/mobilenet_v1_0.25_224/model.json');

    // picking our layer from the topology over which our on-the top
training will take place

  const layer = mobilenet.getLayer('conv_pw_13_relu');

  return tf.model({inputs: mobilenet.inputs, outputs: layer.output});

}
```

Code snippet for downloading MobileNet architecture

3.2.4 TRAINING

With all samples stacked up in the dataset, now we begin to train our model with X array containing images and Y array containing one-hot encoded labels. Before explaining code, lets understand a few terms related to training of the model.

Weights: Each input in the feature vector is assigned its own relative weight (w), which decides the impact that the particular input needs in the summation function. Initially they are assigned random numbers. Further on optimizing the loss function, weights and biases are recalculated and updated in every iteration. In relatively easier terms, some inputs are made more important than others by giving them more weight so that they have a greater effect in the summation function (y). A bias (wo) is also added to the summation.

$$Y = WX + B$$, standard equation for linear regression

Activation function: The result of the summation function, that is the weighted sum, is transformed to a desired output by employing a non-linear function, also known as

activation function. Since the desired output is probability of an event in this case, a sigmoid function can be used to restrict the results (y) between 0 and 1. Other activation function used in the model is: ReLU (Rectified Linear Unit).

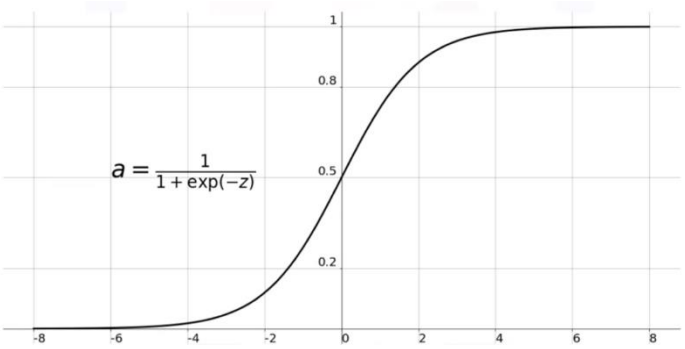


Figure 13: Expression and Graph of Sigmoid function

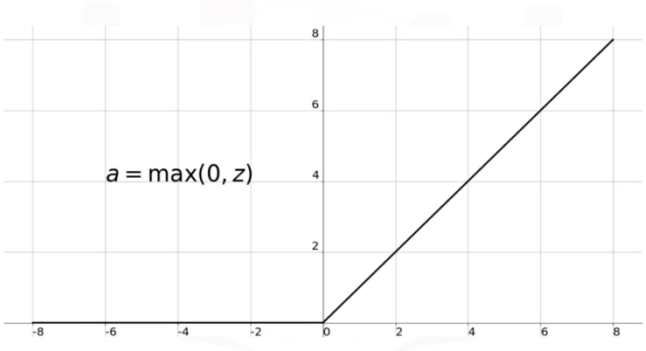


Figure 14: Expression and Graph of ReLU function

Loss Function: The function that is used to compute this error is known as Loss Function. Different loss functions will give different errors for the same prediction, and thus have a considerable effect on the performance of the model. One of the most widely used loss function is mean square error, which calculates the square of difference between actual value and predicted value. Since ours is a multi-class classification type face recognition, we used log-loss or binary cross entropy as loss function.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Expression for binary cross-entropy

Optimizer: Error J(w) is a function of internal parameters of model i.e., weights and bias. For accurate predictions, one needs to minimize the calculated error. Optimisation functions usually calculate the *gradient* i.e., the partial derivative of loss function with respect to weights, and the weights are modified in the opposite direction of the calculated gradient. This cycle is repeated until we reach the minima of loss function. We have used *Stochastic Gradient Descent* as an optimizer in our model. Stochastic Gradient Descent performs a parameter update for each training example unlike normal Gradient Descent which performs only one update. Thus, it is much faster.

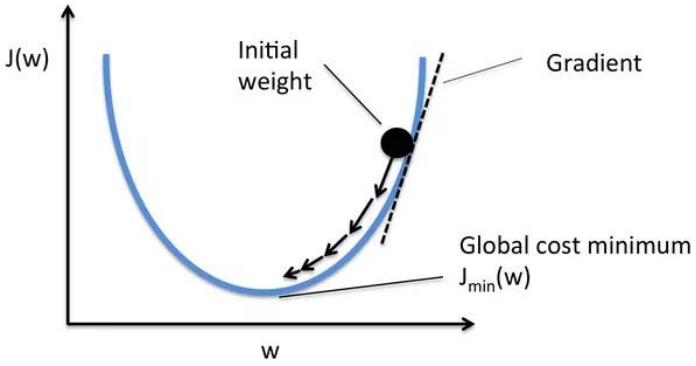


Figure 15: Optimization is used to minimise loss and train well

```
// training on the top of the topology of the MobileNet
async function train() {
  dataset.ys = null;
  dataset.encodeLabels(5);
  model = tf.sequential({
    layers: [
        // flattening the layer over which we training is to be done
      tf.layers.flatten({inputShape:
mobilenet.outputs[0].shape.slice(1)}),
        // 120 neurons are added with activation function "ReLU"
      tf.layers.dense({ units: 120, activation: 'relu'}),
        // output is governed by the multi-class classification
function "SIGMOID"
      tf.layers.dense({ units: 5, activation: 'sigmoid'})
    ]
  });
```

```
// used optimiser "Stochastic Gradient Descent" with learning-rate, alpha (α) = 0.002
  const optimizer = tf.train.sgd(0.002);
    // loss-function: measuring distances between target labels and predicted labels
    // optimizer: reduces loss and updates weights and biases regularly
  model.compile({optimizer: optimizer, loss: 'binaryCrossentropy'});
  let loss = 0;
    // training the model with above stated loss function and optimizer with epochs(no of times trained)
  model.fit(dataset.xs, dataset.ys, {
    epochs: 30,
    callbacks: {
      onBatchEnd: async (batch, logs) => {
        loss = logs.loss.toFixed(5);
        console.log('LOSS: ' + loss);
        }
      }
   });
}
```

Code Snippet used for training of model

3.2.5 PREDICTING

After training the model, we tried testing its performance. At this stage, we got the model given the aforementioned inputs to have a loss of ***0.025*** (refer figure 16) after completing all the iterations. Lower the loss, better the model fits with the dataset. The model predicted majority of the test samples correct.

```
// predicting the subject in the frame of camera
async function predict() {
  while (isPredicting) {
    const predictedClass = tf.tidy(() => {
      const img = webcam.capture();
      const activation = mobilenet.predict(img);
```

```
const predictions = model.predict(activation);

      // returning the class value with maximum probability

      return predictions.as1D().argMax();

    });

    const classId = (await predictedClass.data())[0];

    var predictionText = "";

      if(classId==0 && class1Samples>0){

                  predictionText = "Yes it's person 1  (❁´◡`❁), 
Come in!";}

      else if(classId==1 && class2Samples>0){

                  predictionText = "Yes it's person 2  (❁´◡`❁), 
Come in!";}

      else if(classId==2 && class3Samples>0){

            predictionText = "Yes it's person 3  (❁´◡`❁), Come 
in!";}

      else if(classId==3 && class4Samples>0){

            predictionText = "Yes it's person 4  (❁´◡`❁), Come 
in!";}

      else if(classId==4 && class5Samples>0){

            predictionText = "Yes it's person 5  (❁´◡`❁), Come 
in!";}

      else{

            predictionText = "I'm sorry, I see no one ¯\_(ツ)_/¯";}

      predictionText.fontcolor("green");

      predictionText.bold();

      document.getElementById("prediction").innerText = 
predictionText;

    predictedClass.dispose();

    await tf.nextFrame();

  }

}
```

Code Snippet responsible for prediction of image

3.3 HOW WE PERFORMED?

Following figures are some of the many results we tested by training above dataset.

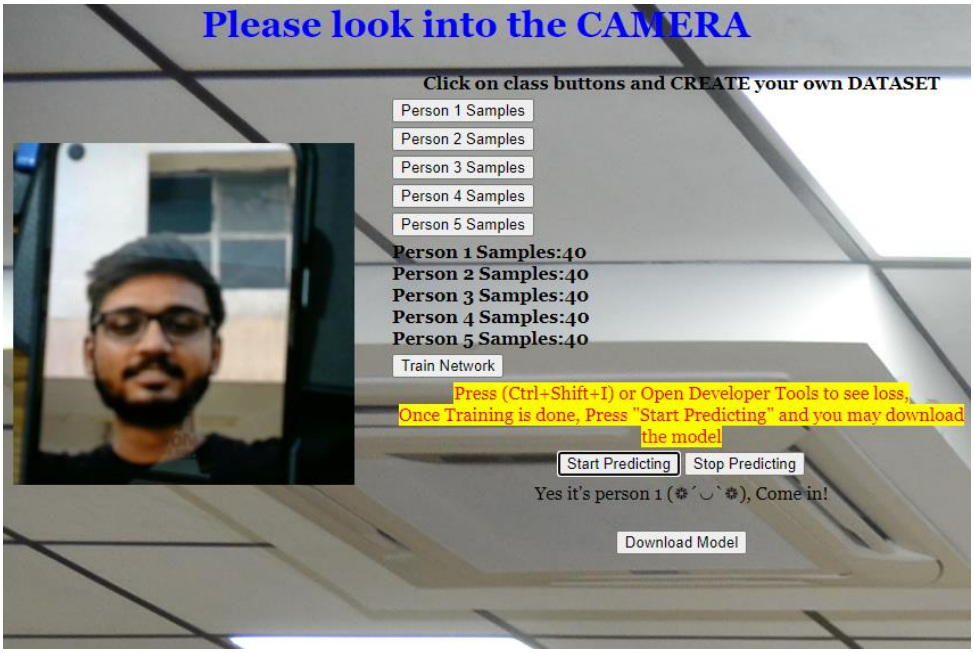


Figure 16: Testing person 1

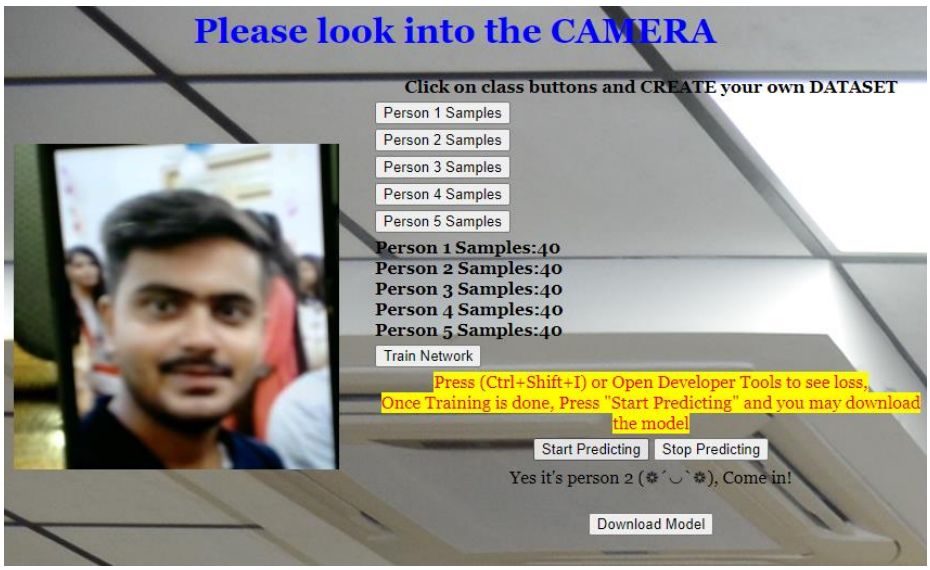


Figure 17: Testing person 2

Figure 18: Testing person 3

Figure 19: Testing person 4

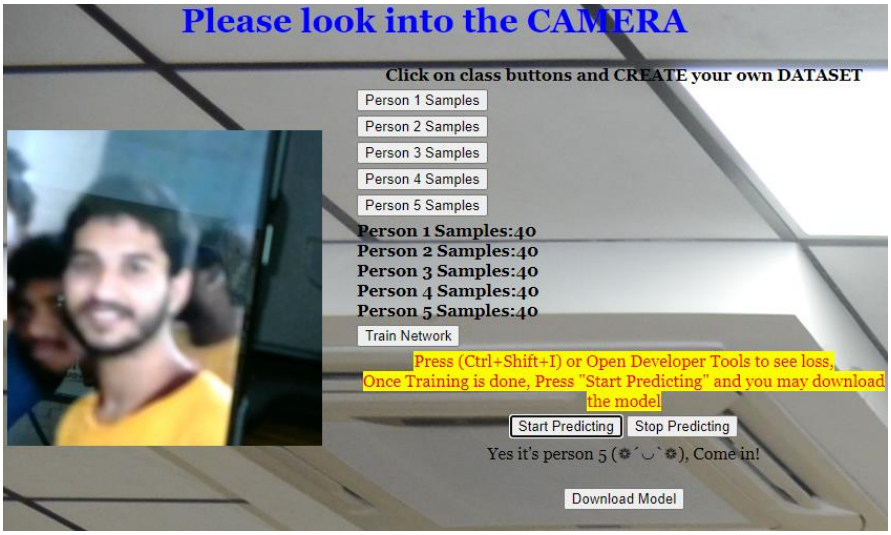


Figure 20: Testing person 5

In the above figures, we used the images on which our dataset was not trained on. This kind of dataset is also called *test-data*. We can see how fairly-accurate our algorithm has been trained to predict someone with some pose not seen previously. Though, the prediction stability was not very concrete and it fluctuated between different persons, yet the algorithm was able to recognize the person.

We suffered limitations with hardware upgradation. Therefore, a small dataset was built (40 samples each) to go along with the computation power of the system we developed this upon. Despite these limitations, we could see how fairly good our algorithm performed in the above images. Had there been no bars over computation power, this face recognition system would have been excellent. We feel happy to share that we could meet our expectations. Though not a robust one, yet a great application for "IoT Based Attendance System using Face Recognition."

The figure below gives the quick understanding about the workflow of the project we did till now.

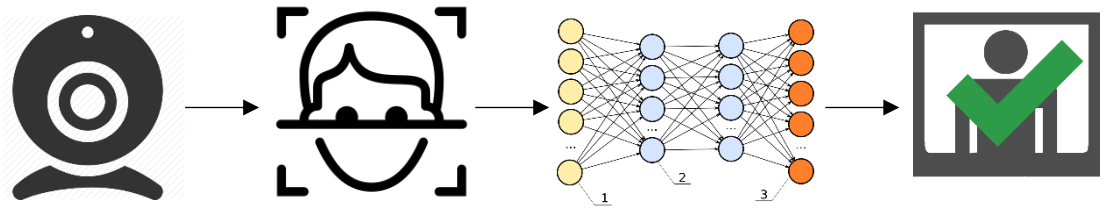


Figure 21: Workflow of the project

4. CONCLUSION

That was an end to our discussion where we understood fundamentals in detail, unrolled the idea and, explained about the working of the project. We carefully customised a classification model to behave as a face recognition system, which served the purpose of the idea. Further, we have discussed about how we integrated camera, training, and prediction in a web browser to make the project fully-functional. Our understanding to this field which is heavy-research oriented is limited hence we implemented the idea using the concept of transfer learning. To build and deploy this entire model, we require high-end flexible infrastructure under Internet of Things. We proposed our idea with limited resource yet we found a great achievement in the performance of our model. At such initial level, we believe this to be a great example. To make a robust system, we certainly require greater research, larger accessibility to resources and more learnings.

This project remains an open-ended solution to our idea of "IoT based Attendance System using Face Recognition". We believe that computer vision is the future for any technological advancements and its integration with IoT would be incredible.

5. FUTURE SCOPE

Finally, all that we learnt was quite an interesting topic. The world today is totally technology oriented. People, Society, Business advance because of developments in science and technology. Today world's economy is dominated by companies from technology sector. That's a big sign of why we should always update ourselves with the recent technological developments or else we'll be left behind economically & socially.

IoT and AI are no different from recent developments in the tech-world. They govern significant changes in the things we experience on day-to-day basis.

We see our project into production in future as the next milestone to look up to. As for the idea proposed has unlimited applications, in the future we may see a greater involvement of such technologies in our lives. This idea of ours is not limited just for taking attendance, rather this can be modified wherever required to make security systems, face ID based entries in various areas, maintaining records of the entrants, etc. The future seems very promising with AI as the world has already shifted to the era of using robots and machine intelligence to boost activities.

6. REFERENCES

[1] Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. Gradient-based learning applied to document recognition. In Proceedings of the *IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791*

[2] Krizhevsky, K., Sutskever, I., Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Advances in neural information processing systems 25, 1097-1105*

[3] Kingma, D.P., Ba, J. Adam: A Method for Stochastic Optimization. *arXiv: 1412.6980v9 [cs.LG]*

[4] Ioffe, S, Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv:1502.03167 [cs.LG]*

[5] Padmanabhan, Sachin. Convolutional Neural Networks for Image Classification and Captioning. Stanford University

[6] Torrey, Lisa and Shavlik (2010), J. Transfer Learning. Handbook of research on machine learning applications and trends: algorithms, methods, and techniques. Page no.- 242-264

[7] Prince, Simon J.D. (2012). Computer vision: models, learning and inference. *Cambridge University Press 2012.*

[8] Andrew NG (2018). Machine Learning Yearning. Deeplearning.ai

[9] [Online] https://www.coursera.org/professional-certificates/tensorflow-in-practice

[10] [Online] https://www.coursera.org/learn/machine-learning

[11] [Online] https://blog.ecosystm360.com/guides/iot-internet-of-things/

[12] [Online] https://www.investopedia.com/terms/d/deep-learning.asp

[13] [Online] https://cacm.acm.org/magazines/2021/7/253464-deep-learning-for-ai/fulltext

[14] [Online] https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-convolutional-Neural-Networks/

[15] [Online] https://teachablemachine.withgoogle.com/train

[16] [Online] https://quickdraw.withgoogle.com/

[17] [Online] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[18] [Online] https://www.aitrends.com/machine-learning/artificial-intelligence-vs-machine-learning/

[19] [Online] https://analyticsindiamag.com/top-10-papers-on-transfer-learning-one-must-read-in-2020/

[20] [Online] https://quickdraw.google.com

[21] [Online] https://teachablemachine.withgoogle.com/train

[22] [Online] https://www.educative.io/blog/one-hot-encoding

[23] [Online] https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a

[24] [Online] https://news.mit.edu/2015/optimizing-optimization-algorithms-0121

[25] [Online] https://medium.com/data-science-group-iitr/loss-functions-and-optimization-algorithms-demystified-bb92daff331c