Triple Pool Net: A novel robust Convolution neural network for image/content classification

by

Shahriar Real

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2020

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

With the rise of artificial intelligence and machine learning, it is highly desired to find a more efficient neural network architecture for real-life applications. In this paper we propose a novel convolution neural network (CNN) architecture known as triple-pool network (TP-Net), to achieve light-weight training and classification processes. We will firstly provide a comprehensive review on the state-of-the-art, and give a detailed description on the proposed TP-Net.

To verify its efficiency, extensive experiments are conducted to compare its performance in terms of training time, error rate (or accuracy), and CPU load in flops, to a number of recently reported CNN architectures, where a well-known publicly available datesets, including CIFAR 10/100, German traffic signs, and SVHN.

The network is designed for a convolution neural network(CNN) and can be readily used for image classification or even light weight authentication. We have carefully designed the network to address the gradient vanishing problem that persists in several larger neural network architectures and also addressed the problem of feature loss while reducing dimensions in the pooling layer.

Acknowledgements

I would like to thank Professor Pin-Han Ho for his continuous support as a supervisor and also as a coach who ensured that I was able to perform to my full potential in every step of the way. Also would like to thank my research mates Yi Chen, Yan Jiao, Constantine, Mark who were excellent colleagues during our research at the laboratory.

Dedication

This is dedicated to my little niece Arhaa Shahriar, my parents, brother and his wife, friends, roommates who have continuously supported me even during this pandemic and helped me ensure a safe working environment during my research.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

With the rise of machine learning and artificial intelligence at the forefront, the constant need of technology and the newer ways to find an efficient network that can be deployed in a small scale system remains to be a tedious task as of date. Much of the state of the art networks focuses on the fact of accuracy when as failing to reduce the size of the parameters of the network. An ever evolving light weight neural network that can take on the tasks of the bigger challenge and to compete with the large networks while maintaining the accuracy still remains to be a great area of research.

1.2 Work Proposed

In this thesis we will explore the possibility of the domain of convolutionl neural network and how we successfully tackled the challenge of designing a light weight network that has the capability of being used in a mobile app or such system due to it using minimum memory usage which in turn can open up a new world of possibilities. In addition to that we have subsequently proved that to build a light-weight network there are several factors

that have to be taken into consideration as well such as-FLOPs(Floating Operation), Size of the network, number of parameters etc.

To make it an even comparison between the networks mentioned in our experiments we have kept the apparatus used for the simulations for all the networks to be the same.

The network mainly consists of three sub-parts and also we explore the advantages of how important pooling is in a neural network architecture. Many research have found out and mitigated the vanishing gradient problem by using a globally connected architecture. Here we will explore such a architecture but also maintaining a max and average pooling in the consecutive layers after the convolution layer to address feature loss during dimensional reduction. Our novel architecture introduces a three pooling system and afterwards we will see how it manages to reduce not only the training time but also manages to use less resources, meanwhile converging faster with better accuracy, also using less number of parameters.

The thesis consists of 5 chapters in total. The current chapter introduces us about the problem statement and the work proposed. It is followed by Chapter 2 which talks about the prior research work done in the field and Chapter 3 talks about the proposed Triple Pool Network in full details. Chapter 4 explains us about the experiments done in the various fields and the following analysis made based on the experiments. The last chapter guides us through the future scopes and conclusion.

Chapter 2

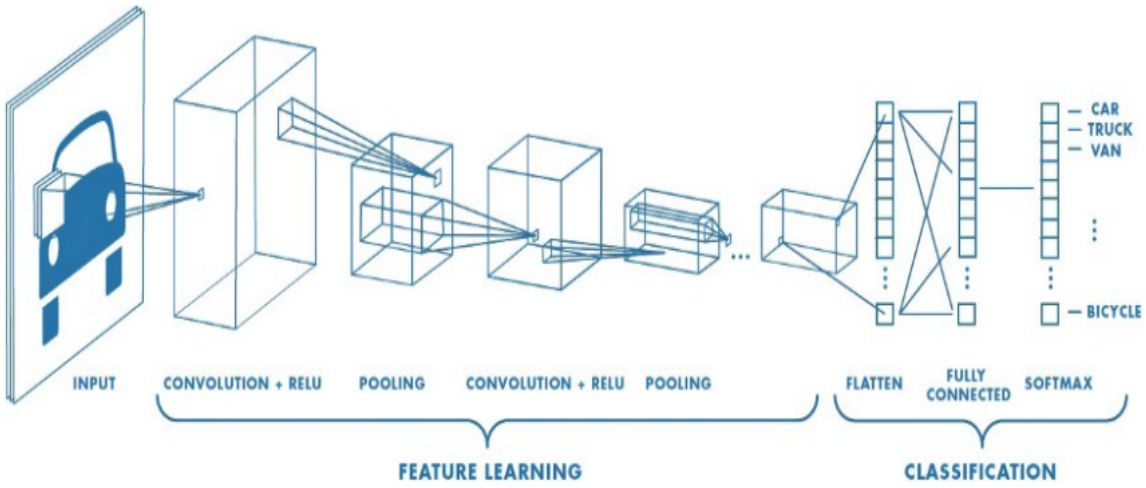
Literature Review

Since Hinton[23] proposed a deep convolution network architecture on ImageNet competition in 2012, convolution neural network has gone on to make a revolution in the last decade in the image classification and recognition sector. Deep neural networks are known to extract rich features from raw pixel image data and achieves great performance in the classification sector.

While deep networks as shown in Fig 2.1 have commendable performances in classification tasks a challenge yet remains to deploy them in embedded systems and small hand held devices. Szegedy [44] came up with a network named Inception where he conducted several experiments and reduced the computational burden of the network architecture. Howard et al.[19] came up with MobileNet which further improved on the research and developed a network that was able to be deployed for mobile and embedded vision applications. With the increased research of neural networks, more focus always remained towards building a optimized network that reduces the computational burden.

There are several methods that were studied as well in order to avoid overfitting of the model at the same time. For example dropout[41] or dropconnect[50] which turns off certain neurons by using a certain probability during the training phase. Moreover, techniques like Batch Normalization[20] and Weight Normalization[37] enables the use of higher learning rates and reduces the dependability of careful initialization in the primary stages of designing the model.

Figure 2.1: A typical CNN Structure



Our proposed novel architecture not only explores the scope of a lightweight and fast efficient network but meanwhile could prove to be a game changer in the greater prospect due to the fact that it is less computation expensive and requires less training time. We have already implemented the architecture on standard data-sets such as Cifar-10/100[24], MNIST[26], SVHN[31] and we achieved state of the art accuracy in the German Traffic Dataset[18].

The architecture achieves similar performances compared to the existing state of the art networks such as VGG[40], Inception[44] and other networks but meanwhile using less number of parameters and reduced training times to match the performance. Even with the revolution in the image classification sector some networks have a huge training time and is computationally expensive so it remains still a challenge to integrate it with mobile systems.

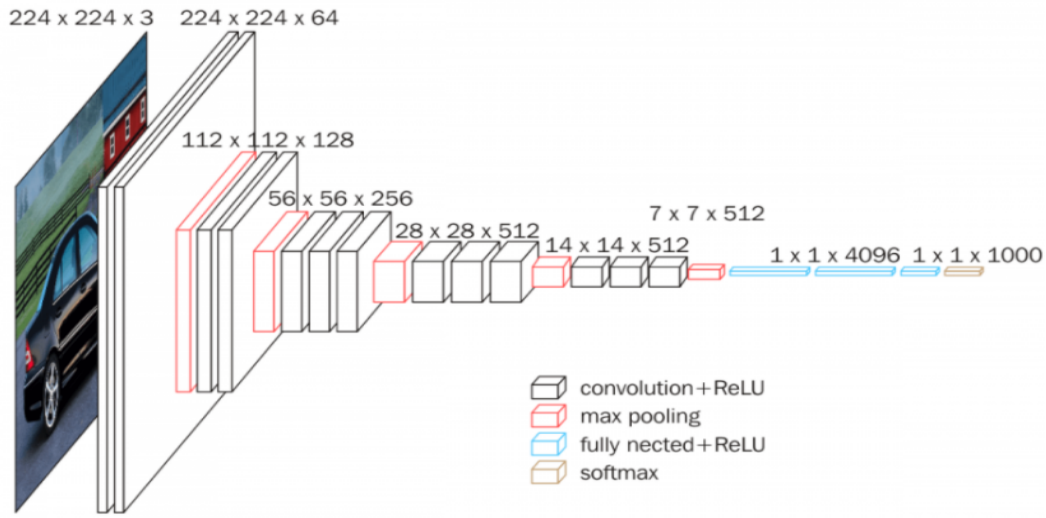
Ever since the development of the neural network architectures, activation functions were found to play an important part in terms of convergence speeds and accuracy. Linear Rectified activation Function (RELU)[1] is such a type of activation function that helps to boost CNN performances resulting in faster convergence compared to other activation functions such as tanh[25], sigmoid[48].

2.1 Exploring Different Neural Network Architectures

2.1.1 VGG

This work demonstrated by Simonyan et al. [40] focused on investigating the effect of the depth of convolutional networks on the accuracy for large scale image recognition tasks. Simonyan [40] demonstrated different experiments on depth-wise convolution and found that it has a significant effect on the increase in accuracy for image classification tasks. The proposed VGG network as shown in Fig 2.2 secured the first and second place in the ImageNet Challenge in 2014 submission across numerous teams. Same preprocessing steps were performed to maintain a fair evaluation in their experiments and a image size of fixed RGB 224x224 was used as input in the network.

Figure 2.2: VGG Architecture



The following Fig [2.1] represents the proposed VGG architecture where all the image dimensions of the depth-wise convolution are marked.

2.1.2 Inception

Although deep convolution network are successful in most of the image classification tasks, a challenge yet remained to introduce its usability in terms of mobile vision tasks and big data. Szegedy et al.[44] demonstrated in his work how to dig deeper towards a network that can be computationally less expensive and uses less resource in order to increase the efficiency. Szegedy[44] explored ways in developing neural networks that aim at utilization the added computation as efficiently as possible by using suitable factorized convolutions and aggresive regularizations.

Figure 2.3: Inception v3 Architecture

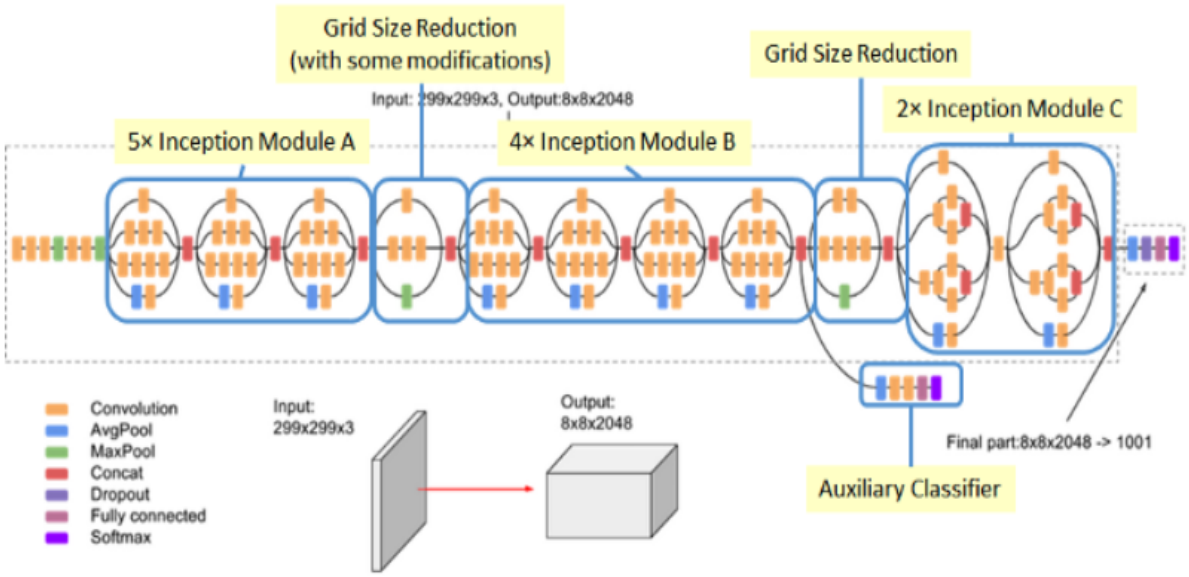
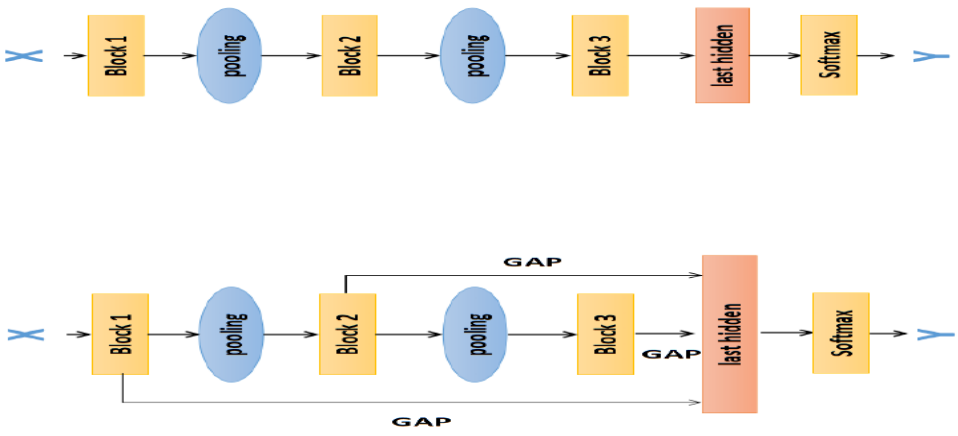


Fig 2.3 denotes a Inception architecture where skip connections are explored in comparison to depthwise convolutions as well. Hence, as a results the authors are able to bring down the number of parameters required for the network to a reasonable margin.

Figure 2.4: Globally Connected Network



2.1.3 GC NET

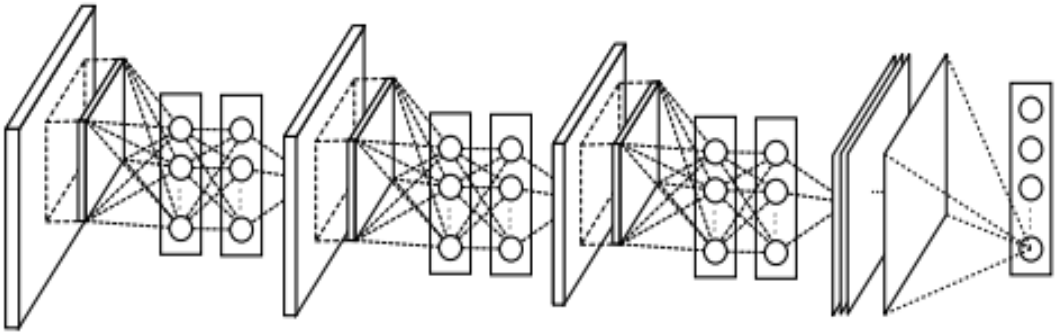
Deep neural networks although have impressive performance in terms of classification task can however, be subject to the information being lost in the last few layers as more number of layers accumulate. Zhi Chen et al.[9] demonstrated a approach that mitigated the issue by designing a globally connected network that mitigates to solve the problems of feature being lost from the intermediate layers to the last layers. Moreover, they also tried to solve the gradient vanishing problem as well.

Fig 2.4 demonstrates the architecture of a Deep-Global Connected Network. Zhi[9] also designed a activation function, namely GRELU (generalized multi-piecewise ReLU activation function) that has the capability to approximate more complex and flexible functions and has the capability to match the performance of the state of the art current activation functions.

2.1.4 Network in Networks

Min et al.[28] proposed a novel structure called Network in Network as demonstrated in Fig 2.5 which enhanced the discriminability for local patches within the receptive field. In a conventional neural network structure is always more based on linear filters and then

Figure 2.5: NIN Structure



followed by a activation function but NIN on the other hand was based on micro neural network with more complex strictures which is able to extract data from the receptive fields.

The NIN structure explored the possibility of stacking different layers as well, followed by a global average pooling layer.

Min[28] explores multi layer perceptrons because of its compatibility with convolution neural networks and also for the fact that it can be used for the purpose of re-use [4].

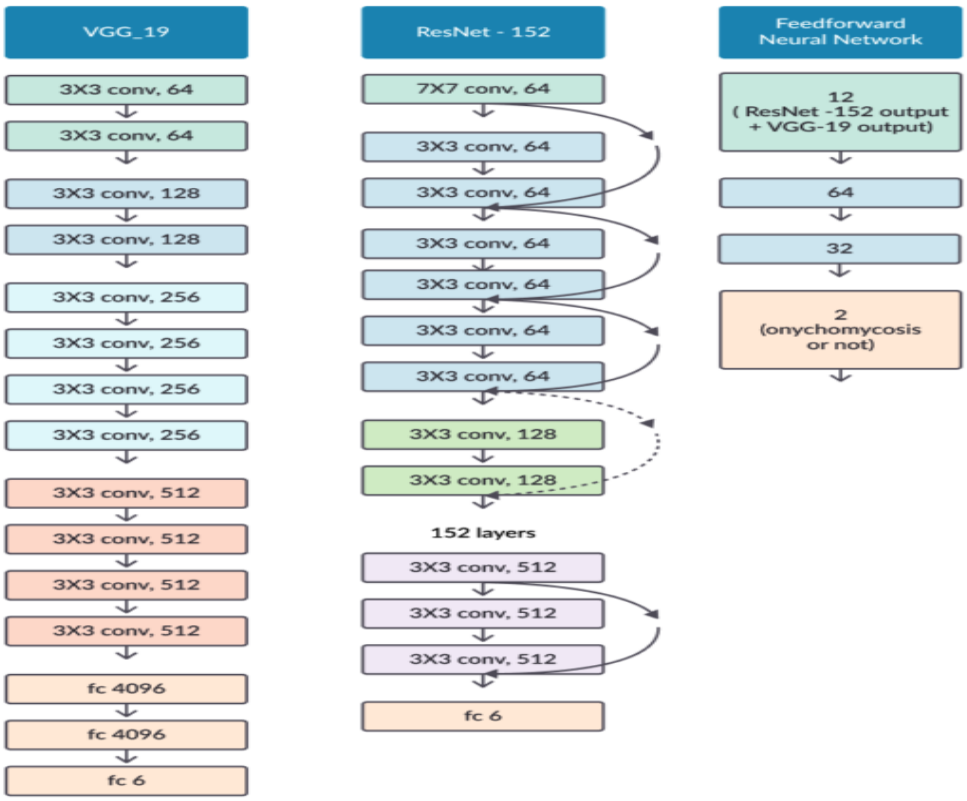
$$f^n_{i,j,k_n} = max(w^n_{k_n} f^{n-1}_{i,j} + b_{k_n}, 0) \tag{2.1}$$

Here, n represents the nuber of layers in the multi layer perceptron, w represents the weight of the neurons and x represents the input.

One of the main contributions of the paper was to employ a strategy called global average pooling which is able to replace the traditional fully connected layer in the traditional CNN structure. The last convolution layer in a traditional architecture are usually vectorized and fed into the fully connected layer then followed by a softmax logistic regression layer [15] [58]. The main idea behind using global average pooling was to take advantage of each of the average of the feature map and directly feeding the resulting vector into the softmax layer, in order to generate one feature map for each corresponding category for classification purposes.

2.1.5 Resnet 50

Figure 2.6: Architecture Comparisons of Resnet vs VGG and Feedforward Neural Networks



Kaiming et al.[17] demonstrated that a residual learning framework is able to ease the training of deep neural networks substantially than those traditional deep neural networks.

As the neural network architecture as shown in Fig 2.6 gets more deeper the gradient vanishing/exploding problem stil persists which creates problems in the convergence of the network. One other problem that also exits is that with the network depth increasing the accuracy also starts to get saturated at the same time and also degrades rapidly. Although this problem is not a result of overfitting but simply stacking more layers cam result in an higher error rate as stated in [16][42].

Chapter 3

Triple Pool Network

The Triple Pool Network or the TP Net is a neural network architecture that is designed for image classification and recognition purposes to be used in a convolution neural network or other networks for similar tasks.

One of the most important phases of a neural network architecture is the pooling layer as it is responsible for dimensional reduction of features in a data-set. So, keeping that in mind we designed a novel architecture that consists of three types of pooling: max, average and global. We were inspired by the work of GC net and in particularly after reading this paper it motivated us to find an even better solution of addressing the same tasks in the imaging sector by using more lightweight structures. The main goal of all these architectures is to minimize the error rates in an image classification or similar tasks.

There has been significant research done on such architectures in the past such as VGG-16, Resnet, Inception but most of these architectures have several layers and consists of huge number of parameters. Even though they rank up to a greater accuracy rate in classification rates however, the problem of implementing them in real life tasks remain a challenge due to the computational inefficiency of these architectures. In 2019 the GC Net addressed the issue by obtaining similar results but using a smaller number of parameters compared to its predecessors.

Inspired by the GC net, there were some future scopes in the paper that mentioned that

Figure 3.1: TP Net Architecture

as the structure is lightweight It can potentially be used in real life mobile applications but however in the paper they did not address the issue whether the architecture is feasible to implement in real life.

So, in our research we designed the TP net, not only ranking up in terms of accuracy compared to the previous networks in the time-frame but also the we reduced the number of parameters on the architecture itself to match the performances. Fig 3.1 illustrates the design of our TP Net Model. However, it is not the only advantage of our network, our network requires less training time per epoch compared to even other lightweight networks and at the same time a smaller number of convolution layers. So, in order to prove that we did a similar experiment to calculate the number of FLOPS (Floating point operation

per second).

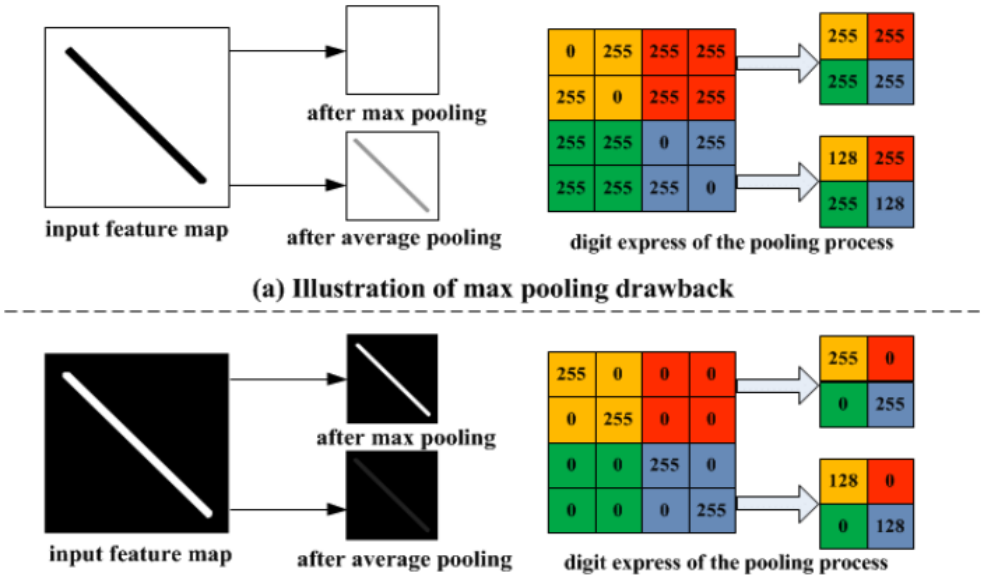
In order to conduct such an experiment, we need to keep some of the conditions the same to make sure it was a fair test. So, our apparatus or the hardware, the GPU used to conduct the experiment was same for the comparison of the networks as it plays a vital role. So, at the end we found a that not only our network ranks up to the accuracy and error rate mark but at the mean time it converges more fast compared to other conventional networks, meanwhile also taking very less time during the training of each epoch. To add to that the flop comparison made so much sense and it was really visible that how much our network outperformed the other ones in terms of computational memory usage.

Therefore, it can be implied that the network is now proven to be applied in real life mobile applications even if needed for better and faster performances. Now coming back to the theoretical aspect of the network, its triple pool meaning the three types of pooling has its advantages in their own unique ways. The max pooling network overcomes the issues if there are more variances in the data-set it takes the maximum value while doing dimensional reduction. On the other hand, the average pooling tries to make a fair comparison if there is less variance in the data-set and considers most of the values while doing dimensional reduction.

On the other hand, the global pooling decreases the dimension rapidly and turns the 3d data into a 2d shape and hence saves up on computational cost. The main reason behind using the triple pool was that the cons of the other pooling types could be overcome by the pros of another. When conducting our experiments our theory was proven to be in the right track when we found significantly important findings from our experimental analysis.

For the smaller data-sets we implied a bias within the activation function to get more proper results and for larger data-sets we deployed batch normalization within our architecture to achieve the desired performances while maintaining all the other parameters. We have reached state of the art results in the German Traffic data-set and almost a very negligible error rate on the MNIST data-set and also however for the larger data-sets such as SVHN and CIFAR-10, we managed to match the performances of the other networks in terms of accuracy while reducing the parameters and also making our architecture more computationally less expensive.

Figure 3.2: Advantages of Multiple Pooling



3.1 Theory

3.1.1 Gradient Vanishing Problem

$$Y_j = f(\sum_i W_{i,j} X_i) \tag{3.1}$$

Considering a simple neural network where F is the activation function and W is the weight and X is the input, y is the output. Moving forward from here the back-propagation equation for the network becomes:

$$\frac{\partial}{\partial W_{i,j}} = \frac{\partial E}{\partial Y_j} . \frac{\partial Y_j}{\partial W_{i,j}} \tag{3.2}$$

Since F' will be close to zero for larger and as well as small inputs, the gradient as the layer progresses and goes deep into the network might be very small. Hence, as a result the gradient vanishing problem can arise. So, to mitigate this issue, we have introduced a careful global connection between each layer to the last layer so that this problem does

not exist. Moreover, the first layer is connected to the last layer through global average pooling to ensure that rich features from the top layers are extracted into the last layers for classification tasks.

Although mixed pooling have been employed in the past[57] to reduce computational burden, but the gradient vanishing problem still persisted and it does effect the results in the long run. But, in our new designed system, we carefully extracted rich hierarchical features from every convolution layer by deploying both max pooling and average pooling but in a new strategy. While after the max pooling, the new features went onto the next convolution layer, the average pooling was deployed in a different way, where the features extracted from each layer were concatenated and fed into the last layer concurrently. As a results, the rich hierarchical features from both the types of pooling were taken into consideration and as Fig 3.2 demonstrates, disadvantages of one type of the pooling where it missed some features were overcome by the other pooling layer. Furthermore, the smooth connection between the interconnected layers ensured that gradient vanishing or exploding problem would not persist within the last layers as the layer went deep and at the same time reducing the size of the model due to the pooling layers ensured a lightweight yet powerful network was designed.

3.1.2 FLOPs

There are a few factors that determine the success of the neural network design.

- Data-set size
- Model design
- Computational cost

Due to the modern dataset size that demands to be larger in size, the demand for powerful processors and GPUs are constantly on the rise. In order to mitigate this issues scientists constantly try to design more efficient and powerful hardware systems to significantly reduce the training time. But on the other hand in order to tackle this issue,

designing a better efficient neural network architecture can prove to be pivotal as it can address this problems[14].

Convolution layer FLOPs can be calculated by the following:

$$FLOPs = S_w.S_h.X_{d-1}.X_d.k_w.k_h \tag{3.3}$$

S_w refers to spatial width of the kernel, S_h refers to the spatial height of the kernel, X_d refers to the spatial depth of the current layer, X_{d-1} refers to the previous layer depth and k_w,k_h refers to the kernel width and the kernel height. In order to calculate the FLOPs for our model and the models in comparison, we have used the built in function of FLOPs in keras to make an even comparison between our models.

In order to maintain the capabilities of the cutting edge processors, neural network models need to be significantly smaller in order to be accommodated in mobile phone applications. While VGG-16, a dense neural network requires 15.8 GFLOPs of compute per image and 552 MB of model size, mobile models on the other hand have a relatively limited FLOPs around the 600m and a model size of 5 MB [38]. And for more significant operations such as always wake on screen or facial recongnition an ideal size of the model should have a relatively smaller FLOPs of 60m or less.

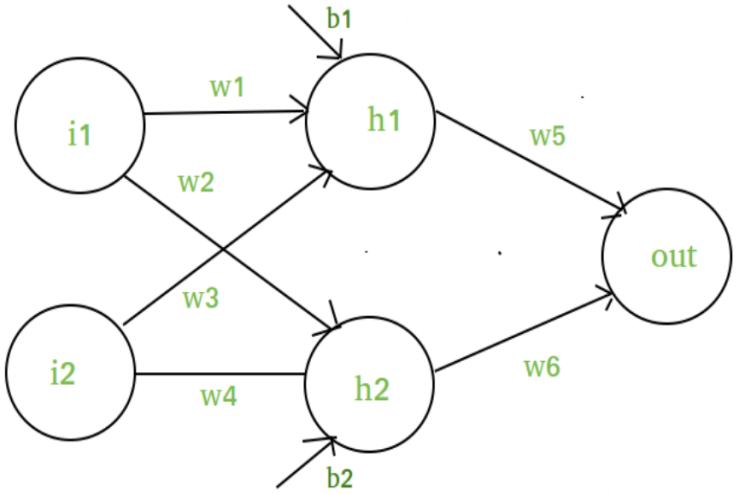
3.2 Activation Function

Inside a neural as Fig 3.3 illustrates, network inputs are fed into the neurons of the input layer and each neuron is subject to a subsequent weight and multiplying the weight an the input gives us the concurrent output for that neuron, which in turn is transferred to the next layer.

$$Out = \sum_{0}^{i,j} W_j * X_i + B_b \tag{3.4}$$

Activation function serves as a critical role in between the input for the current layer and the output that is transferring into the next layer. Activation functions can range from

Figure 3.3: Typical Neural Network Structure



being just a simple step function to a more complex non-linear function which can help tp learn complex data and compute and learn any function and helps in providing accurate predictions.

Fig 3.4 shows the comparison between 4 non linear activation functions that are used in a common neural network setup. We can clearly see that some of the activation function is bounded between 0 and 1, where some exhibit a behaviour of not showing anything for negative values of x. On the other hand more complex and later functions such as elu can enable us in learning critical features when the value of x is tending towards more negative.

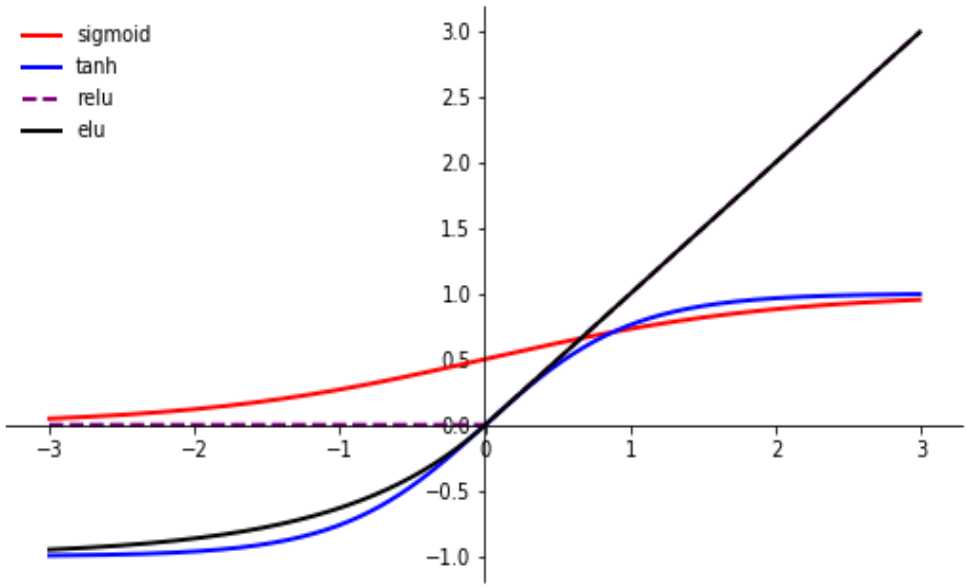
3.2.1 Sigmoid

$$\Theta(x) = \frac{1}{1 + e^{-x}} \tag{3.5}$$

Sigmoid[39] or logistic activation function is bounded between a value of 0 to 1. One of the facts is that it prevents the jumps in output values and ensures a smooth gradient and return helps in clear predictions.

But some of the limitations of sigmoid can be a vanishing gradient problem for very high or loww values of x, where it exhibits behaviours of showing no changes to the pre-

Figure 3.4: Different Activation Functions



diction for such values. On a more larger prospect it in turn can disable the network from learning further or is too slow in converging to a global minima and can turn out to be computationally more expensive.

3.2.2 Tanh

$$\frac{e^{2x} - 1}{e^{2x} + 1} \tag{3.6}$$

Compared to its predecessor, tanh[45] provides the flexibility in making it easier for model inputs to be more robust when it is near negative, neutral or for positive values. But its limitation is still very similar to the sigmoid function as well.

3.2.3 ReLu (Rectified Linear Unit)

$$f(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{otherwise} \end{cases} \tag{3.7}$$

ReLu[49] on the other hand provides the network in being computationally more efficient and allows the network to diverge more quickly. But one of the disadvantages of this widely used function is that when the inputs approach zero or negative, the gradient vanishing problem still exists and there are problems when the network is performing back-propagation.

3.2.4 eLu (Exponential Linear Unit)

$$f(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha(e^x - 1), & \text{otherwise} \end{cases} \tag{3.8}$$

eLu[12] on the other hand similar to its predecessor relu, helps in enabling the network to converge faster and is also less computationally expensive. But its main advantage lies on the fact that since it has a value in the negative region it helps the network to learn for values that are negative and in return can help to mitigate the vanishing gradient problem.

3.2.5 Design of the Network

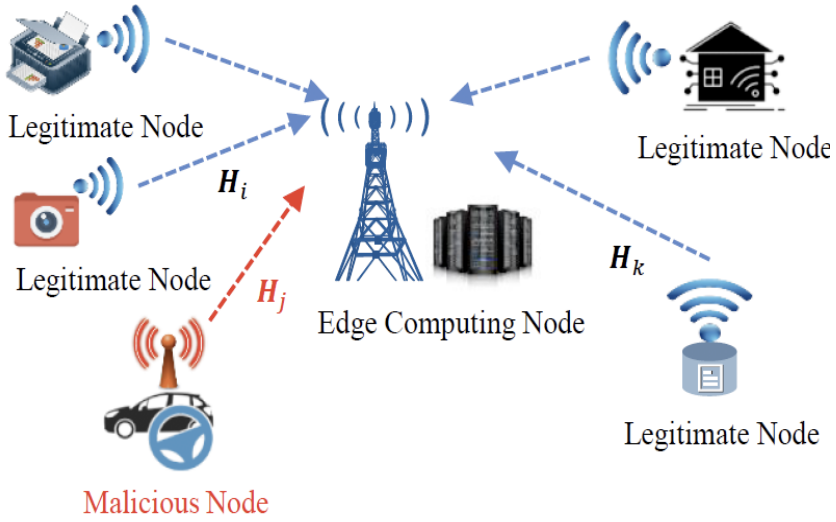
Our main design of the architecture was motivated from the design of the GC NET architecture where the author mitigated the vanishing gradient and also the interconnection between the layer issue. Our structure improves on the fact that as a triple pooling strategy is deployed, the max and the average pooling addresses the vanishing gradient problem while decreasing the dimension of the extracted features.

3.2.6 Size

The size of the network plays an important role when the model is to be deployed in a embedded system or for real time tasks. If the size of the network is too large then more computational resources will need to be allocated for performing tasks which is the scenario that should be avoided.

3.3 Channel Authentication

Figure 3.5: PHY-layer authenticationin edge computing scenarios



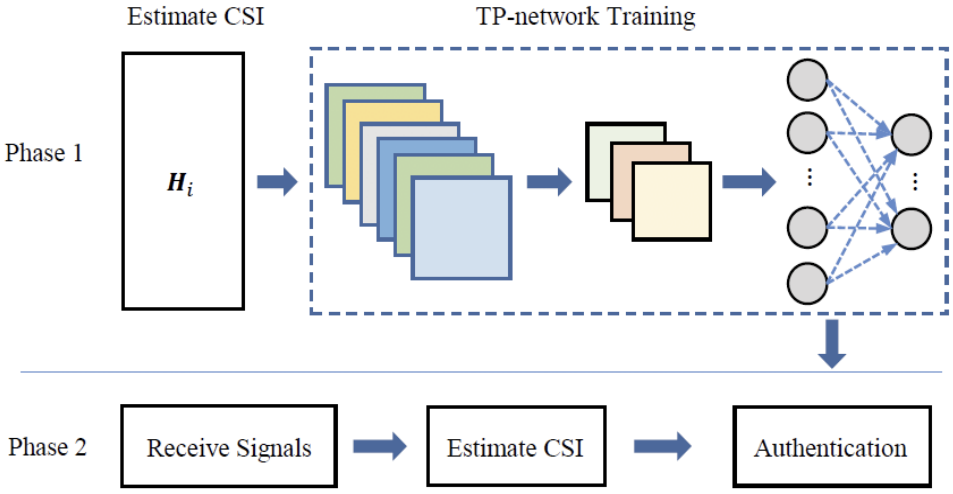
Open transmission media can be subject to various threats of attack and security is always of vital importance in the wireless communication sector. Among the security protocols, user authentication is a critical role in ensuring the legitimate use of network resources as Fig 3.5 suggests. More traditional based methods like cryptography based security can result in higher arithmetic operations from the sender and receiver sides. In such circumstances it can lead to more hardware complexity and consumption of power in the battery powered sector of Internet of things (IoT) devices where it is highly not recommendable due to the limited resources wireless terminals with limited storage and computing power.

The rich characteristics of physical-layer (PHY-layer) channel state information (CSI) of wireless links have been taken as a signature to authenticate the senders. Compared to its predecessor, the cyptography method, PHY-layer authentication is more lightweiht in nature and reduces the computational burden. Moreover, PHY-layer authentication can significantly achieve a trade-off between the level of security and latency requirements[27].

The spatial decorrelation property of the PHY-layer characteristics are exploited and

applied in a stochastic model manner. Some of these PHY-layer characteristics include received signal strength (RSS)[59], CSI [54][47], channel phase response[53], channel impulse response [29], and hardware fingerprints as well. For the user authentication purpose a binary hypothesis test(schotastic model) is used to make decisions based on the PHY characteristics and predefined threshold. Although, the accuracy of some of these approaches is highly reliant on the test threshold values that is hard to obtain in a practical environment setup.

Figure 3.6: The framework of TP-Net based PHY-layer authentication scheme



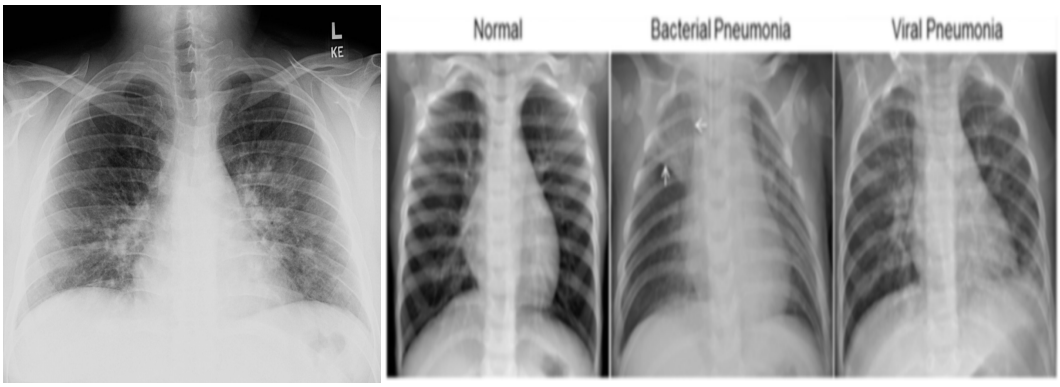
For PHY-layer authentication scenarios machine learning models have been proved to be a reliable source and employed by numerous research initiatives [29]-[51] . The study in [6] uses the k-means algorithm to detect both spoofing attack and Sybil attack based on the RSS of different transmitters. The support vector machine (SVM) classification is adopted to identify the radio frequency fingerprinting (RFF) of sensor nodes in IoT [56]. L. Xiao et al.[55] introduced a spoofing detection scheme that leverages a reinforcement learning process to accomplish the PHY-layer authentication. X. Wang et al. [52] used a deep neural network to accomplish the indoor positioning via CSI, which showed a remarkable effect compared with some other existing methods in two representative indoor environments. N. Wang et al. [51] proposed a PHY-layer authentication scheme based on ML to detect

spoofing attack. Chatterjee et al. used a ML model to enhance the IoT security [6].

The above schemes, although being claimed effective in the considered scenarios, used conventional ML models with a large number of layers and parameters, leading to significant computation time and power consumption. They are not suitable for edge computing where some nodes are subject to stringent limitation on power consumption, computation, and storage. For a ML model to be applied under the scenario of edge computing, the efficiency of such ML model can not be just measured by ML metric, we should also consider the required resources to perform model training and prediction simultaneously.

It is clear that an efficient ML model with resources consideration for the PHY-layer authentication mechanism in edge computing systems is of great importance. The paper investigates a ML-based PHY-layer authentication scheme, called TP-CNN-PHA, for lightweight message authentication in edge computing systems. The proposed TPCNN-PHA scheme incorporates with a ML model containing a novel CNN architecture, namely Triple Pool Network (TP-Net), which is uniquely featured by using three pooling schemes, i.e., max pooling, average pooling, and global average pooling (GAP). We train the TP-Net by using the channel data from a Rayleigh model as well as our testbed built on Universal Software Radio Peripheral (USRP) [5]-[7], respectively, where extensive simulation is conducted to verify the proposed TP-CNN-PHA scheme and compare with its counterparts.

3.4 COVID Chest Xray



(a) COVID-19 Chest Xray (b) Pneumonia dataset Xray

Figure 3.7: Covid-19 Analysis Dataset

One of the most interesting dataset that is open for public research is the Chest Xray image dataset as shown in Fig 3.7. Image processing and machine learning is constantly deployed in the sector of medical imaging and one of the important sectors is the classification of correctly labeled images to match performances with modern medical equipments.

The Chest Xray-image dataset [22] consists of 5,862 X-ray images (JPEG) into three categories, Viral Pneumonia, Bacterial Pneumonia and Normal patients. The Chest Xray images (anterior-posterior) were a series of images selected from cohorts of patients from Guangzhou Women and Children's Medical Center, Guangzhou.

All images that are part of the dataset is screened for quality control, where low resolution or low quality images were removed in order for efficient analysis. Before the dataset was made public it was graded by two expert physicians and cleared the data-set for quality assurance for training in AI systems.

On the other hand, given the state of the current pandemic, much research is deployed into this image processing field to enable AI systems to help find ways in image processing for COVID-19 patients. While there are many public datasets available for other chest

X-ray images, a readily available dataset was always lacking for the COVID-19 research purposes.

Joseph et. al[10] designed a dataset of collecting COVID-19 chest xray images from COVID-19 patients in order to help AI based appraoches to dig deep into the research of predicting and understanding the infection that entails the current pandemic.

Ever since, much research has been conducted in this field[13] [30] to enable image processing to play any sort of role in finding any patterns or trends in the Xray images of patients from the data.

Chapter 4

Results and Analysis

For our experiment part, we have simulated the performance of the TP net on several standard data-sets and compared its performances with powerful and modern state of the art neural network architectures.

4.1 Experimental Setup

For our experiments we used the server hph8.eng.uwaterloo.ca =. In order to make a fair comparison it is important that all our experiments are done using the same experimental setup. The specifications of the server are outlined in Table 4.1

The connection towards hph8 server was established using a local MacBook machine that runs on MacOS Catalina 10.15 consisting of a 2.3 GHz Quad-Core Intel i5 processor, 8 GB Ram DDR3 and Intel Iris Graphics 655. The design of the network is mainly built based on the programming language python[36]. Python[36] serves as the basis of numerous scientific computing, software development etc. Python is the go to language for these types of tasks mainly owing to the fact if uts rapid growth of popularity and the accessibility to many open source libraries. Within python many data manipulation libraries such as keras[21], pandas[35] serve as the basic foundation of data preprocessing tasks.

Table 4.1: hph8 Server Specifications

Parameter	Specifications
CPU Model	Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz
RAM	16 GB
Graphics Card	NVIDIA GeForce RTX 2080 Ti
Harddisk	500GB

Table 4.2: Packages Used and its concurrent versions

Package	**Version**
Python	3.7
Keras	2.3.1
Tensorflow	2.0
Pandas	0.23.4
Numpy	1.17.3
Anaconda	1.9.2
Jupyter Notebook	0.34.9

Numpy helps in the pixel level manipulation of the raw image and can help in substantial tasks like slicing, stacking etc. Keras on the other hand is a API designed for python and is considered to be one of the most used deep learning framework for designing neural network architectures. Built on top of Tensorflow 2.0[46], keras also enables the usage of large clusters of GPU to enable efficient training of the neural network model. Moreover all the versions of the packages used are outlined in Table 4.2.

For efficient coding purposes we relied on the integrated developer environment(IDE) called Anaconda[2] and used one of its package, namely Jupyter Notebook[32]. Jupyter notebook helps in the development of open source software and services for interactive

computing supporting a range of programming languages. JupyterLab is a interactive web-based development environment for Jupyter notebooks. The main purpose of using JupyterLab is that it helps to write plugins more easily and can add new components and integrate with existing ones.

4.2 Image Classification

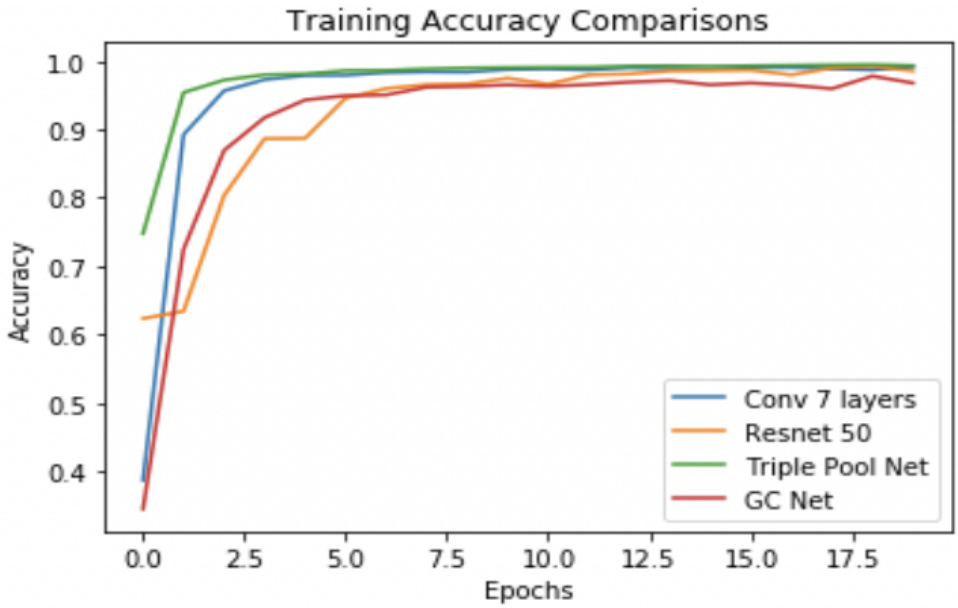
4.2.1 German Traffic Dataset

Figure 4.1: German Traffic Dataset

The German Traffic Dataset consists of 50000 30x30 colour images spanning over more than 40 classes in total, a sample is shown in Fig 4.1. It can be used for multi classification problems. A large and real lifelike database that consists of real reliable ground truth data due to semi automatic annotations.

We have conducted several experiments on this dataset keeping the GPU and all other computational resources constant in order to make a fair comparison when we are comparing metrics like training time. As training on different platforms for different networks will lead an unfair comparison.

Figure 4.2: Training Accuracy Graph comparison



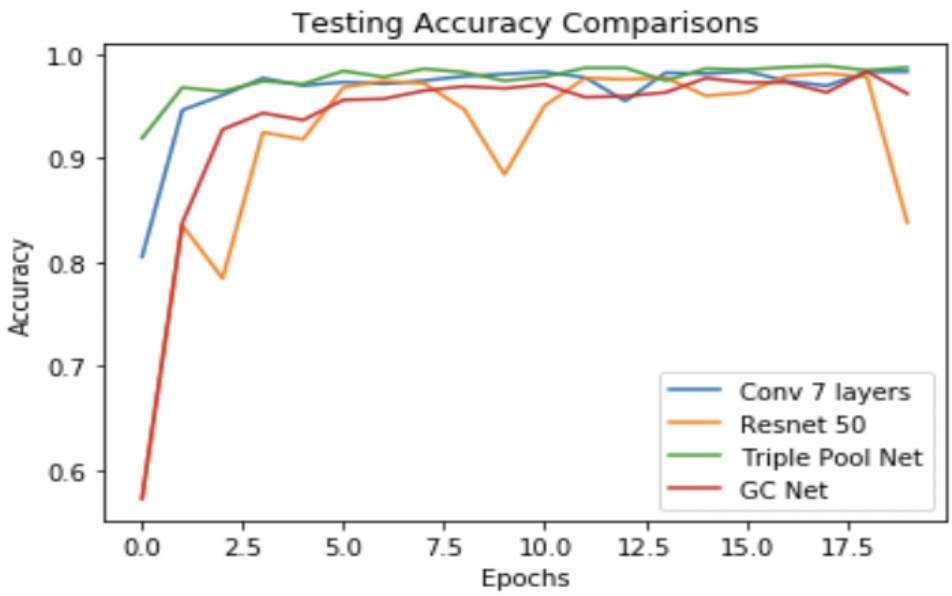
Furthermore, one of the motivation behind focusing on this dataset is that real time autonomous vehicle research is on the rise and different techniques are going under research to come to a optimum solution. When we trained our dataset we have reached exceptional results, that too in a very short span of time during the training process as shown in Fig 4.2. To add to that we have managed to significantly reduce both the training time and also the number of parameters which can have a significant impact and open possibilities of such a light weight network being deployed in real time applications.

The training and testing loss as both demonstrated in Table 4.3 outlines the reuslts from our experiments. Overall, in the span of 20 epochs the loss significantly decreased the number of epochs increased suggesting that there is no overfitting in the network. The testing accuracy comparisons can be visualized in Fig 4.3.

Table 4.3: Training and Testing Loss

Network	Training Loss	Testing Loss
Conv 7 Layers	1.6	**8.2**
Resnet 50	37	39
Triple Pool Net	**0.6**	12
GC Net	8.4	16.2

Figure 4.3: Testing Accuracy Graph comparison



We have subsequently added categorical cross-entropy [11] as the loss function in this case in order to reach a global minima. As we can see from the table, our model outperforms all the other state of the art neural network architectures in terms of training loss indicating that our model is furthermore optimized and can mitigate the issue of over-fitting as well.

Table 4.4: Training and Testing Accuracies

Network	Training Accuracy	Testing Accuracy
Conv 7 Layers	99.5	98.3
Resnet 50	85.5	83.8
Triple Pool Net	**99.8**	**98.8**
GC Net	97.9	96.2

Table 4.5: FLOPs Comparisons

Network	FLOPs in Millions
Conv 7 Layers	57.8
Resnet 50	416
Triple Pool Net	**4**
GC Net	20

Training and Testing The training Accuracy over the span of our training increased and reached a steady rate of 99.8% which indicates that the network is learning very well and reaching satisfactory results. In addition to that it also reached upto a 98.8% in the testing accuracy in our validation set which is very promising for performances in image recognition tasks in a very wide multi class setup in such a challenging image classification task as Table 4.4 states. Compared to state of the art network that we simulated in our lab most of the network either converges too slow and rankings in terms of accuracy and error rate are not superior compared to our designed networks.

FLOPs One of the most important aspects of our network is the number of floating point operation it uses. It is visible from Table 4.5 that the number of FLOPs it uses is only 4m to match the performances of the other networks indicating not only that our network is

Table 4.6: Parameter Comparisons

Network	Parameters
Conv 7 Layers	3.8
Resnet 50	25
Triple Pool Net	**0.279**
GC Net	0.291

optimized but most significantly it is computationally less expensive and can lead to small hand held mobile phone applications and in real time applications for small such devices.

Most of the state of the art network fails in this sector as training and validating a model requires huge amount of resources and increases the computational burden and makes the task of deploying it in a small handheld device or small applications a very challenging and tedious task.

Parameters One of the most crucial areas of improvement is the use of less number of parameters while designing our model. A challenging aspect is to reduce the size of the model without sacrificing any aspect of the accuracy and error rate. A Resnet which is a widely established model consists of 25 million parameters in the model. Compared to such state of the art models, our model only uses 0.279 million parameters which statistically suggests that our model only uses 1.1% of the parameters compared to a Resnet model which can be visible in Table 4.6. Such a significant improvement of about 98% more efficiency to match the same accuracy level is an astonishing improvement yet in itself. It opens the door of the possibility of a lighweight architecture that is fast yet reliable.

Moreover, one of the motivation behind focusing on this dataset is that real time autonomous vehicle research is on the rise and different techniques are going under research to come to a optimum solution. When we trained our dataset we have reached exceptional results, that too in a very short span of time during the training process. To add to that we have managed to significantly reduce both the training time and also the number of

Table 4.7: Training Time Comparisons

Network	Training Time
Conv 7 Layers	23 s
Resnet 50	84 s
Triple Pool Net	**14 s**
GC Net	23 s

parameters which can have a significant impact and open possibilities of such a light weight network being deployed in real time applications.

Training Time The training times are compared between several state of the art neural network architectures, Table 4.7, where TP net was trained in the fastest time possible and converging with seemingly impressive results. In order to determine a fair comparison of the training time we trained some networks in the same experimental setup. Our network was able to train each epoch in about 14 seconds which suggests a significant improvement compared to even some state of the art lightweight architectures such as the GC Net itself.

Table 4.8: Size of the Models

Network	Size (mb)
Conv-7	46.3
TP Net	**3.4**
GC Net	3.6

In the long run when training for huge models, it helps the researcher to train in a short span of time while achieving significantly comparable results to other networks.

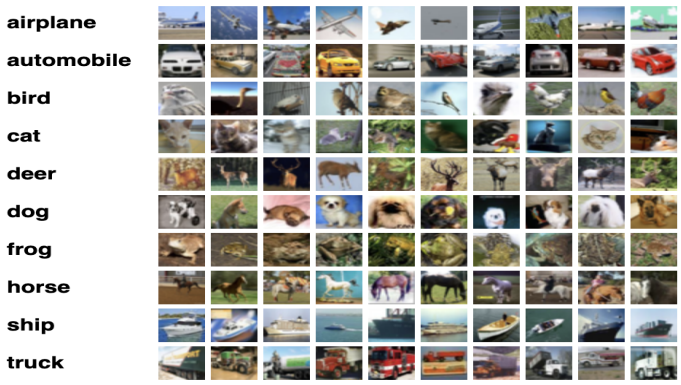
While training the model, in order to compare the size of the trained models. We saved the trained models in our hard-drive and also compare the results. Our overall model was

very compact and gives the opportunity to be deployed in small devices and embedded systems due to its minimalist size as Table 4.8 illustrates.

4.2.2 CIFAR-10

.

Figure 4.4: CIFAR-10 Dataset



CIFAR-10 is a realistic dataset containing 60000 images, 32x32 RGB images in 10 classes where there are almost 6000 images per class. A sample of the dataset is shown in Fig 4.4.

The dataset is divided into 50000 images for training and 10000 for testing.

From the experimental analysis, as shown in Table 4.9 it was possible to conclude that a typical TP Net with 5 layers can outperform even the sophisticated GC Net with 8 layers. An error rate of 9.8% proves that TP nets opens a real wide of possibilities as it is a light weight network we have also compared it with heavy network architectures like the Resnet 110 or 1001 which uses huge resources and also has more training time but at the end the accuracy remains somewhat similar.

For our choice of activation function which plays a vital role we have opted towards chosing 'elu' because it provides significant advantages over other available function as talked about before in the TP-net section.

Keeping all the improvements in mind from the previous section we directly compared the error rate on this dataset which is also very challenging. All the experiments were done without any prior transfer learning and all the networks were trained from scratch in our

Table 4.9: Error Rate and Parameter Comparison

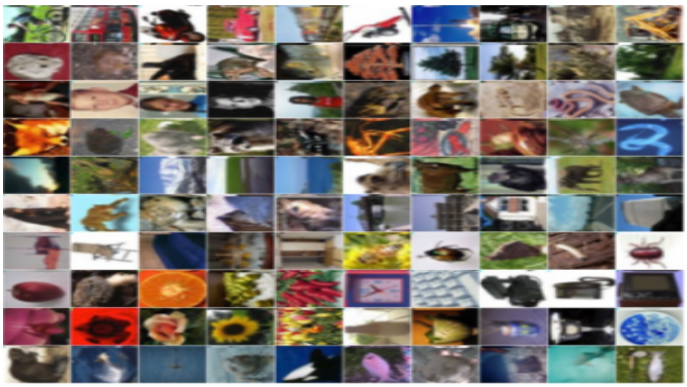
Network	**Error Rate**	**Parameters**
GC Net (6 layers)	12.55	0.11
TP Net (5 layers)	**9.20**	0.9
TP Net (5 layer non-aug)	10.6	0.6
GC Net (6 layers)	10.4	0.61
GC Net (8 layers)	9.38	0.91
Resnet 110	13.63	1.7
Resnet 1001	10.56	10.2

lab in order to determine the best results. Same preprocessing strategies were used across all the simulations and mostly normalization, batch preprocessing were used.

Moreover, the same activation function and same optimizers were used to ensure that the results we got from our simulations are fairly consistent and make it more clear about the advantages of our network.

4.2.3 CIFAR 100

Figure 4.5: CIFAR 100 Dataset across 100 classes



Spanning across 100 classes and with about 600 images per class, it still remains as one of the most hardest dataset to train a network in till date as Fig 4.5 demonstrates. The training set consists of 50000 images and the testing set consists of 10000 images. One of the most challenging aspects is the number of classes and also very few images that are there for training.

Table 4.10: Error Rate and Parameter Comparison

Network	**Error Rate**	**Parameters (million)**
Resnet 50	44.74	24
Network in Network	35.74	1
TP Net (4 layer)	**33.23**	**0.994**

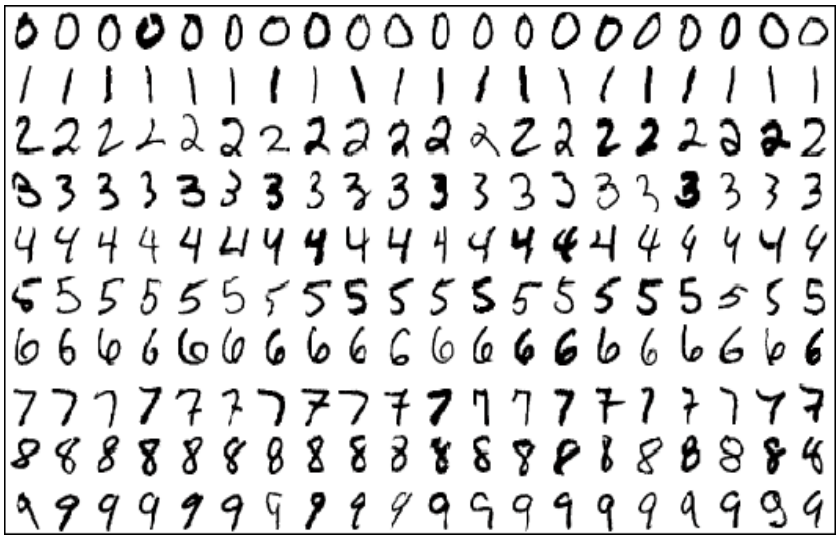
We did not employ any transfer learning to this dataset as well and trained the networks from scratch itself. Results can be improved significantly by using the transfer learning technique but we opted for training all the networks in order to setup a fair comparison.

We achieved a significant error rate of 33.23% which is a improved result compared to some of the networks we tested it against with as Table 4.10 suggests.

Again, same preprocessing technique were applied across all the network. In addition to that 'adam' was used as the optimizer and also 'elu' was our choice of activation function.

4.2.4 MNIST

Figure 4.6: MNIST DATASET Images



The MNIST database consists of handwritten digits that has a training set of 60,000 examples, and a corresponding test set of 10,000 examples. It is a database of grey-scale images of 28x28 pixels.It is one of the most standard datasets to test machine learning algorithms on. An example of the MNIST dataset is outlined in Fig 4.11.

Table 4.11: Validation Accuracy and Parameter Comparison

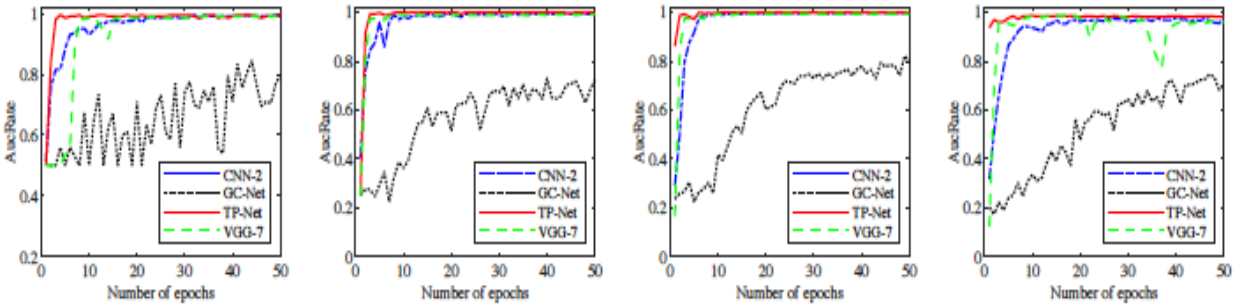
Network	**Validation Accuracy**	**Parameters**
TP Net	98.91	**62,000**
LeNet	98.87	182,000
CNN-2	98.86	180,000

MNIST is the starting data-set for any machine learning models. Is is a very compact handwritten digit challenge with 10 classes. In our experimental setup we tried to match the validation accuracy of the models and as the results suggest that even though all the

three networks ranked to the same validation accuracy but the number of parameters our network used to match the performance was significantly low compared to other models, referring to Table 4.11.

4.3 Channel Authentication

Figure 4.7: The authentication rate under different numbers of wireless nodes with channel SNR as 2dB. (a)The authentication rate of 2 wireless nodes. (b)The authentication rate of 4 wireless nodes. (c)The authentication rate of 6 wireless nodes. (d)The authentication rate of 8 wireless nodes.



(a) 2 wireless nodes. (b) 4 wireless nodes. (c) 6 wireless nodes. (d) 8 wireless nodes.

For the development of TP-CNN-PHA which we talked about in our earlier chapter rigorous experiments were conducted and compared with other methods like traditional CNN architectures,GC-NET and VGG as well. One set of experiments are conducted using the Rayleigh simulation data from the National Key Laboratory of Science and Technology China, for CNN training and testing.

The TP net is implemented with 4 convolution layers with 16 and 32 feature maps respectively. Global average pooling is applied to the initial first layer to extract a rich set of features alongside the different layers for the input to the soft-max layer for classification.

Meanwhile, the conventional CNN is modelled with 2 convolution layers which is named CNN-2 and consists of 8 and 16 feature maps and the activation function applied is relu as it provides better results in this case.

GC net on the other hand is composed of 3 convolution layers with 3x3 filters and 64 feature maps correspondingly. GAP(Global Average Pooling) is applied to the output of each convolution layer as the original GC net design and hence we also employed GRELU as the activation function as stated in the original research work of the model.

A miniature version of the VGG was designed using 7 layers stacked on bottom of each

other with 64,128,256 feature maps respectively. Here VGG-7 employs elu as the activation function which yields the best results for this model.

Furthermore, adaptive moment estimation (Adam) is used for training which is an accelerated gradient algorithm used for optimization of the network.

The performance metric that was mainly focused on this experiment is the *AucRate* which entitles the probability of correctly identifying whether a node is legitimate or not receiving an authentication request.

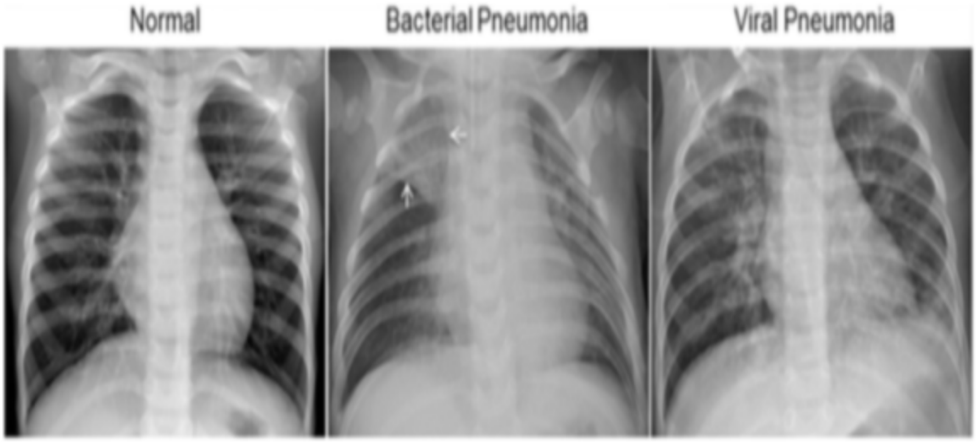
$$AucRate = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

Here TP is the number of true malicious nodes being detected and TN represennts the number of legitimate nodes being correctly detected, whereas false positive (FP) referes to the number of legitimate nodes classified as malicious and (FN) False negative is the number of malicious nodes that are not detected.

Fig 4.7 shows the results of authentication rate under different nmber of wireless nodes with the channel noise to signal ration (SNR) as 2dB. It is clear and evident that compared to its counterparts VGG-7, GC net and CNN-2 TP net yields the most promising results and even such advantage remains when the number of nodes are increased.

4.4 COVID Analysis using Machine Learning

Figure 4.8: Chest Xray Image Dataset



The Chest Xray image dataset as shown in Fig 4.8 consists of images of 224x224 size spanning across three classes. The classes are Normal, Bacterial Pneumonia, Viral Pneumonia. There are around 5863 images across all the dataset. The main challenges lie in undermining the classification task for such a dataset as the image quality sometimes is compromised and can play a vital role in terms of accuracy. However, deep learning networks play a substantial role in classification tasks.

We created a comprehensive dataset combining Chest X-ray dataset from pneumonia patients and also COVID-19 Xray images from the dataset described at [10]. This helped us to have a comparison between different classes and also at the same time helped us to have a dataset that contains much more images for training purposes. So, overfitting is a possible issue that can be overcome also by increasing the data-set size so the machine learning model can generalize well.

The small yet powerful TP Net yields much more promising results in the initial stages as Table 4.12 suggests. All the networks have been trained without any prior transfer learning. Although much of research is being conducted just a preliminary start point for the scope of research into this field yields satisfactory results.

Table 4.12: Covid-19 Chest Xray Image Results

Network	**Error Rate(%)**	**Parameters(Million)**
Resnet 50	20.95	24
TP Net	**7.7**	**0.918**
GC Net	15	1.1

In addition to that not only there was room for improvement in the error rate but as well our light-weight network also performed better in terms of less parameters which significantly reduces computational burden, which gives more scope for research into the model itself.

Chapter 5

Conclusion and Future Scopes

We have introduced TP Net which is a Triple Pooling Network, a lightweight compact CNN architecture. TP net has been tested in the image classification sector as well as fields in physical layer authentication as well which makes it more multidisciplinary. We have mitigated the issue of gradient vanishing problem with the design of our network. During our development of the model, we also ensured that our model uses less coomputational resources and has the potential to be deployed in embedded systems or small hand-held devices.

One of the most promising fields we have went into is the COVID-19 Chest Xray using image classification of patients. As the data set is now limited, and is growing everyday, we plan in the future to test the system using a wide database as well. As well as implementing heatmap visualizations that can help the user to identify potential patterns in the images. Moreover, to add to that we are currently conducting research in the adversarial network field where CNN architecture seems to suffer from sticker attacks and also some gradient attacks in the architecture.

We aim to in the future make the neural network architecture more robust in open world challenges by introducing some defense mechanisms against these attacks so that our network is able to generalize well under challenging circumstances. And in the future we also plan to deploy and test its peformance in a real time embedded system for measuring performance metrics. With so much scope and possibility of this new designed network

we aim to continue our research and further develop the network so that it can have some implications in the real world once deployed.

References

[1] Abien Fred Agarap. *Deep Learning using Rectified Linear Units (ReLU).* eprint arXiv:1803.08375, 2018.

[2] Anaconda. *https://www.anaconda.com/products/individual*.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey Hinton. *Layer Normalization.* arXiv:1607.06450, 2016.

[4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. *Representation learning: A review and new perspectives.* IEEE transactions on pattern analysis and machine intelligence, 2013.

[5] Kapil M. Borle, Biao Chen, and Wenliang Kevin Du. *"Physical layer spectrum usage authentication in cognitive radio: Analysis and implementation.* IEEE Transactions on Information Forensics and Security, 2015.

[6] Baibhab Chatterjee, Debayan Das, and Shreyas Sen. *Rf-puf: Iot security enhancement through authentication of wireless nodes using in-situ machine learning.* IEEE International Symposium on Hardware Oriented Security and Trust (HOST), 2018.

[7] Yi Chen, Hong Wen, Jinsong Wu, Huanhuan Song, Aidong Xu, Yixin Jiang, Tengyue Zhang, and Zhen Wang. *Clustering based physical-layer authentication in edge computing systems with asymmetric resources.* Sensors, vol. 19, no. 8, p. 1926 edition, 2019.

[8] Yingying Chen, Jie Yang, Wade Trappe, and Richard P. Martin. *Detecting and localizing identity-based attacks in wireless and sensor networks*. IEEE Transactions on Vehicular Technology, 2010.

[9] Zhi Chen and Pin han Ho. *Deep Global-Connected Net With The Generalized Multi-Piecewise ReLU Activation in Deep Learning*. Computer Vision and Pattern Recognition, 2019.

[10] Joseph Paul Cohen, Paul Morrison, and Lan Dao. *COVID-19 Image Data Collection*. Computer Vision and Pattern Recognition https://arxiv.org/abs/2006.11988, 2020.

[11] Categorical Cross-Entropy. *https://www.tensorflow.org/api_docs/python/tf/keras/losses/CategoricalCrossentropy*.

[12] Clevert Djork-Arné, Thomas Unterthiner, and Hochreiter. *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. https://arxiv.org/abs/1511.07289.

[13] Terry Gao. *Chest X-ray image analysis and classification for COVID-19 pneumonia detection using Deep CNN*. https://doi.org/10.1101/2020.08.20.20178913, 2020.

[14] Xavier Glorot and Yoshua Bengio. *Understanding the difficulty of training deep feedforward neural networks*. Proceedings of Machine Learning Research, 2010.

[15] Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. *Maxout networks*. arXiv preprint arXiv:1302.4389, 2013.

[16] Kaiming He and Jian Sun. *Convolutional neural networks at constrained time cost*. Conference on Computer Vision and Pattern Recognition (CVPR), 2015.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[18] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. *Detection of Traffic Signs in Real-World Images: The German Traffic Sign Detection Benchmark*. International Joint Conference on Neural Networks, 2013.

[19] Andrew G Howard. *Mobilenets: Efficient convolutional neural networks for mobile vision applications*. Addison-Wesley, Reading, Massachusetts, 2017.

[20] Sergey Ioffe and Christian Szegedy. *Batch normalization: Accelerating deep network training by reducing internal covariate shift*. https://arxiv.org/abs/1502.03167, 2015.

[21] keras. *https://keras.io/*.

[22] Daniel S. Kermany, Michael Goldbaum, Wenjia Cai, M. Anthony Lewis, Huimin Xia, and Kang Zhang. *Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning*. Cell Press Journal https://doi.org/10.1016/j.cell.2018.02.010, 2018.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. *ImageNet classification with deep convolutional neural networks*. Advances in Neural Information Processing Systems, 2012.

[24] Andrew Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. 2009.

[25] Yann LeCun, Yoshua Bengio, , and Geoffrey Hinton. *Deep learning*. Nature, vol. 521, no. 7553, pp. 436–444 edition, 2015.

[26] Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. *Gradient-based learning applied to document recognition*. Proc. of the IEEE, vol. 86, no.11, pp.2278-2324 edition, 1998.

[27] Run-Fa Liao, Hong Wen, Songlin Chen, Feiyi Xie, Fei Pan, Jie Tang, and Huanhuan Song. *"Multiuser physical layer authentication in internet of things with data augmentation*. IEEE Internet of Things Journal, vol. 7, no. 3, pp. 2077– 2088, edition, 2020.

[28] Min Lin, Qiang Chen, and Shuicheng Yan. *Network in Network.* International Conference on Learning Representation, 2014.

[29] Jiazi Liu and Xianbin Wang. *Physical layer authentication enhancement using two-dimensional channel quantization.* IEEE Transactions on Wireless Communications, 2016.

[30] Shervin Minaee, Rahele Kafieh, Milan Sonka, Shakib Yazdani, and Ghazaleh Jamalipour Soufi. *Deep-COVID: Predicting COVID-19 from chest X-ray images using deep transfer learning.* Elsevier Public Health Emergency Collection 10.1016/j.media.2020.101794, 2020.

[31] Yuval Netzer, Tap Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. *Reading Digits in Natural Images with Unsupervised Feature Learning.* NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2011.

[32] Jupyter Notebook. *https://jupyter.org/*.

[33] Numpy. *https://numpy.org/*.

[34] Muhammad Shahmeer Omar, Syed Ahsan Raza Naqvi, Shahroze Humayun Kabir, and Syed Ali Hassan. *An experimental evaluation of a cooperative communication-based smart metering data acquisition system.* IEEE Transactions on Industrial Informatics, 2017.

[35] Pandas. *https://pandas.pydata.org/*.

[36] Python. *https://www.python.org/*.

[37] Tim Salimans and Deiderik P. Kingma. *Weight normalization: A simple reparameterization to accelerate training of deep neural networks.* Neural Information Processing Systems, 2016.

[38] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. *MobileNetV2: Inverted Residuals and Linear Bottlenecks.* Computer Vision and Pattern Recognition, 2018.

[39] Sigmoid. https://en.wikipedia.org/wiki/Sigmoid_function.

[40] Karen Simonyan and Andrew Zisserman. *Very Deep convolutional networks for large-scale image recognition.* International Conference on Learning Representations (ICLR), San Diego, California, 2015.

[41] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov R. *Dropout: a simple way to prevent neural networks from overfitting.* J. Machine Learning Res, vol. 15, 1929-1958 edition, 2014.

[42] Rupesh Kumar Srivastava, Klaus Greff, and Jurgen Schmidhuber. *Highway networks.* arXiv:1505.00387, 2015.

[43] CNN Structure. https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148.

[44] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, and Jon Shlens. *Rethinking the inception architecture for computer vision.* Computer Vision Pattern Recogniton, 2016.

[45] Tanh. https://en.wikipedia.org/wiki/Hyperbolic_functions.

[46] Tensorflow. https://www.tensorflow.org/.

[47] Jitendra K. Tugnait. *Wireless user authentication via comparison of power spectral densities,.* IEEE Journal on Selected Areas in Communications, 2013.

[48] Joseph Turian, James Bergstra, and Yoshua Bengio. *Quadratic features and deep architectures for chunking.* The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, vol. companion volume:, 2009, pp. 245–248 edition, 2009.

[49] Nair Vinod and Geoffrey Hinton. *Rectified linear units improve restricted boltzmann machines.* ICML, 2010.

[50] Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. *Regularization of neural networks using dropconnect*. Int. Conf. Mach. Learn. (ICML'13), second edition, 2013.

[51] Ning Wang, Ting Jiang, Shichao Lv, and Liang Xiao. *Physical-layer authentication based on extreme learning machine*. IEEE Communications Letters, 2017.

[52] Xuyu Wang, Lingjun Gao, Shiwen Mao, and Santosh Pandey. *Csi-based fingerprinting for indoor localization: A deep learning approach*. IEEE Transactions on Vehicular Technology, 2017.

[53] Xiaofu Wu and Zhen Yang. *Physical-layer authentication for multi-carrier transmission*. IEEE Communications Letters, 2015.

[54] Liang Xiao, Larry J. Greenstein, Narayan B. Mandayam, and Wade Trappe. *Channel based spoofing detection in frequency-selective Rayleigh channels*. IEEE Transactions on Wireless Communications, 2009.

[55] Liang Xiao, Yan Li, Guoan Han, Guolong Liu, and Weihua Zhuang. *Phy-layer spoofing detection with reinforcement learning in wireless networks*. IEEE Transactions on Vehicular Technology, 2016.

[56] Feiyi Xie, Hong Wen, Yushan Li, Songlin Chen, Lin Hu, Yi Chen, and Huanhuan Song. *Optimized coherent integration-based radio frequency fingerprinting in internet of things*. IEEE Internet of Things Journal, 2018.

[57] Dingjun Yu, Hanli Wang, Peiqiu Chen, and Zhihua Wei. *Mixed Pooling for Convolutional Neural Networks*. International Conference on Rough Sets and Knowledge Technology(Springer), 2014.

[58] Matthew Zeiler and Rob Fergus. *Stochastic pooling for regularization of deep convolutional neural networks*. arXiv preprint arXiv:1301.3557, 2013.

[59] Yulong Zou, Xianbin Wang, and Weiming Shen. *Optimal relay selection for physical layer security in cooperative wireless networks*. IEEE Journal on Selected Areas in Communications, 2013.