AUTOMOBILE SECURITY SYSTEM BASED ON FACE RECOGNITION AND CAN BUS SECURITY IMPLEMENTATION

By

Ahmed Rahmy Mohamed

Amira Mostafa Mohamed

Eman Alaa El-din Zarif

Mohamed Ahmed Mohamed

Rokia Sayed Abdlmonem

A Graduation Project Report Submitted to

Faculty of Engineering at Cairo University

in Partial Fulfillment of the Requirements for the

Degree of Bachelor of Science

in

Electronics and Electrical Communications Engineering

Under Supervision of

Dr. Ahmed Hussein

Dr. Hassan Mostafa

Faculty of Engineering, Cairo University

Giza, Egypt

August 2020

Table of Contents

List of Tables

List of Figures

List of Abbreviations

IOT	Internet of things
V2V	Vehicle to vehicle
V2X	Vehicle to infrastructure
DAS	Driver assistance system
ECU	Electronic control unit
GPS	Global positioning system
CAN	Controller area network
OBD	On-board diagnostics
DLC	Data length code
CRC	Cyclic redundancy check
EOF	End of frame
DOS	Denial of service
MAC	Message authentication code
SHA	Secure hash algorithm
MD	Message digest algorithm
AES	Advanced encryption standard
DES	Data encryption standards
NIST	National institute for standards and technology
CNN	Convolutional neural network
ReLU	Rectified linear unit
FC	Fully connected

GPU	Graphics processing unit
ILSVRC	ImageNet Large Scale visual Recognition Challenge
MAC	Multiply and accumulate
EAR	Eye aspect ratio
AI	Artificial intelligence
Tanh	Tangent Hyperbolic function
HOG	Histogram of Oriented Gradients
CPU	Central processing unit
MMOD	Maximum-Margin Object Detector-
FLOPs	Floating point operations per second
SNN	Siamese neural network
SDK	Software development kit
BSP	Board support package
ML	Machine learning
LFW	Labeled faces in the wild

Acknowledgments

We are using this opportunity to express our gratitude to all who helped us in our graduation project and trusted us in doing great work with their guidance and advice.

First of all, we would like to thank our supervisors Dr. Ahmed Hussein and Dr. Hassan Mostafa for their following up to check our work, giving us their assistance generously all the time and their suggestions to clear up any obstacles faced us during the project work.

Secondly, we would like to thank Eng. Mohamed Abdou, Senior Algorithms Engineer and Deep Learning Researcher at Valeo Egypt, for providing us his valuable experience and his time to help us overcome any problems faced.

Thirdly, we would like to thank Eng. Abdelrahman Hussein, Research Assistant at ONE Lab for his regular following up and putting us in the right way to complete our work.

Finally, we would like to thank our families and friends for their unlimited support, faith in us and in what we are capable of doing.

Abstract

Automobile security is a demanding field of enhancement. High theft rates side by side with the new cyber threats for in-vehicle network have directed the attention towards applying new technologies to leverage car security. For this reason, a system is proposed using face recognition to identify the car driver, and also to apply security concepts to controller area network (CAN) to secure the communication between vehicle parts.

Chapter 1: Introduction

1.1 Motivation

When future and its technologies are mentioned no one can omit the fact that the two most developing and promising technologies that keeps people looking forward to what is new: are machine/deep learning and internet of things (IOT). Those technologies are being gradually added to every and each factor of our modern lives.

Creating machines that think have been a dream for a long time. Now computers are told what to do, breaking big problems up into many small, precisely defined tasks that the computer can easily perform. By contrast, we don't tell the computer how to solve our problem. Instead, it learns from observational data, figuring out its own solution to the desired problem. That's how machine learning is introduced.

Deep learning is a subset of machine learning in artificial intelligence (AI) that has networks capable of learning from data without external interfering, these networks are called neural networks. In recent years, neural networks have won numerous contests in pattern recognition, and currently provide the best solutions to many problems in computer vision, speech recognition, and natural language processing.

Among different types of deep neural networks, convolutional neural networks (CNNs) have been most extensively used with visual document tasks, specifically image-related problems such as classification or scene parsing because the convolution operation captures the 2D nature of images, also it was reported that learning process using CNN for image classification was "surprisingly fast", and one of the reasons that made usage of CNNs has increased in the last few years was that datasets of Large Scale Visual Recognition Challenge (ILSVRC) [1] have become available for training and validation.

Face recognition is one of CNNs applications that has been from the most challenging and attractive areas of computer vision. Face recognition is a method of identifying or verifying the identity of a person using their face. Hence, people in photos, video, or in real-time can be identified using face recognition systems. In the near future, face recognition technology will likely become more widely spread. It may be used to track individuals' movements out in the world.

Internet of things made the idea of having far things monitored and under control come true. Remaining at your own seat you can check on your house, your car, your business and whatever that concerns you, and even have the ability to manipulate them over the air. IOT is also employed in many industrial applications where it gives the machines in factories the ability to communicate and exchange data.

The idea is to add connectivity to all objects and also give them the ability to process data that they send and receive. This opens the door to a vast range of new applications that was only defined as imagination before. Smart city is an application in which IOT connects city objects together in order to control traffic, water distribution and various city related problems. Smart farming introduces the same concept to manage farms. IOT can also be used to monitor goods being transmitted and shipped all over the world. Trucks and buses owners can also monitor their fleets by only setting in front of a computer screen.

Implementing IOT connectivity to vehicles was indeed a great addition to automotive domain. Making the vehicle connected, it can be monitored and even can be given commands remotely. Furthermore, employing a large number of connected cars and all the sensors they have, they can be a vital source of collecting data from every part of the world. In this era of artificial intelligence and big data, this data collection network formed by connected cars can lead to a huge growth in data available for training machine learning models or data analytics purposes.

IOT does not only connect a car to the internet, it also introduces V-2-V (vehicle to vehicle communication) and V-2-X (vehicle to infrastructure). When cars on the road are interconnected they can give each other information about which is the best route to take for a certain destination, locate congestions and even avoid accidents.

1.2 Problem Definition

1.2.1 Car Theft

Car theft is one of the major problems facing car owners. According to statistics a chart of the number of motor vehicles thefts reported in England and whales across time is depicted below in Figure 1-1. This chart gives an intuition of the losses sustained by car owners due to car thefts, which is actually more severe in other parts of the world rather than England [2]

Automotive industry is now revolutionized by new technologies leading to new trends e.g. autonomous driving and smart driver assistant systems (DAS). However, the security of automobiles did not experience as much development as other automotive aspects. More attention has to be directed to employing new technologies such as artificial intelligence in building more mature and reliable car security systems.

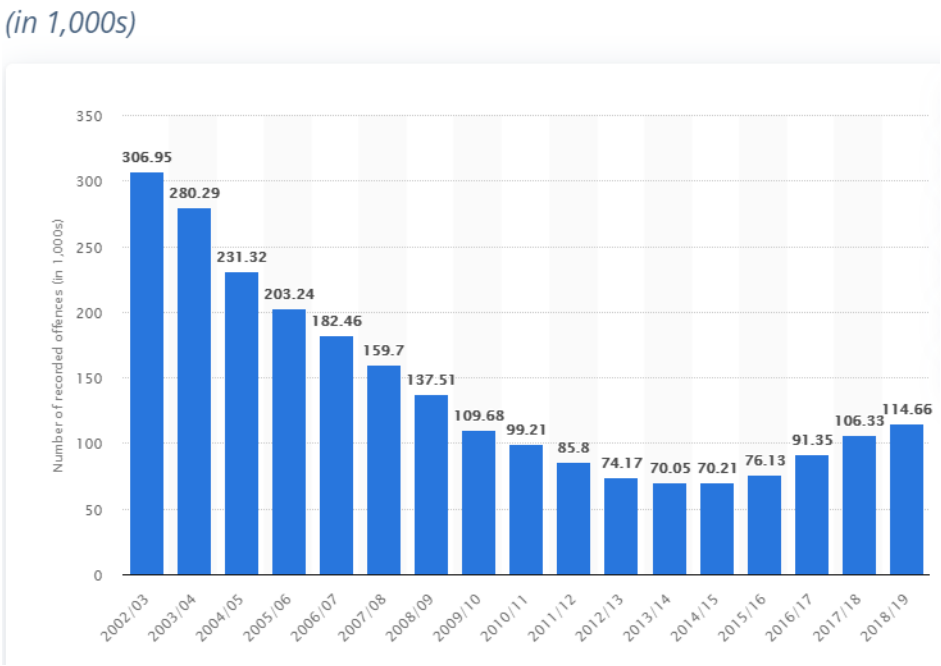


Figure 1-1: Number of motor vehicle theft offences recorded in England and Wales from 2002/03 to 2018/19

1.2.2 CAN Security

Recently, after the increasing interest in IOT, connected vehicles, V-2-V (vehicle to vehicle) and V-2-X (vehicle to infrastructure) technologies, a new car security threat came to light. This new threat gained huge attention after an incident in which two security researchers -named Charlie Miller and Chris Valasek- were able to

remotely manipulate a number of functions in a modern Jeep vehicle shown in Figure 1-2, and they were also able to stop its engine on a highway. They aimed to reveal the vulnerabilities in the car. This was not their first attempt to hack a vehicle; they were also able to manipulate a Toyota Prius and a Ford Escape after they gained wired connection to the internal CAN bus of both (through the OBD II port in the vehicles) [3].

Figure 1-2: The hacked Jeep

In vehicle controller area network (CAN) is the network dedicated for the operating electronic control units (ECU) to communicate. This network is the medium for transmitting all the critical commands among car ECUs. However, when CAN network was first designed at 1983 by Robert Bosch, the only concern was to guarantee reliable data transfer between car ECUs under the real time constrains defined by car performance specs. The fact that the cars will evolve to be communicating devices, besides being transportation means, was not then taken into consideration. The moment science thought of connecting a vehicle to internet or to other vehicles, a huge surface of vulnerabilities appeared. These vulnerabilities made CAN bus security a demanding field of enhancement.

Besides the importance of preventing CAN bus attacks to car owner's safety and security in general. It is particularly important when intending to design anti-theft car security system, because if an attacker is able to hack the network, the attacker will obviously be able to counter whatever measures the security system will take to prevent the robbery.

1.3 Solution

The proposed solution introduces a security system to stand against car theft attempts. System parts are illustrated in Figure 1-3. The system is based mainly on equipping the vehicle with face recognition module to define the person in front of the wheel, and make sure it is the authorized owner. In case it is not, the car will then have connection to the owner's cell phone to have the permission to move. In case it is not the car owner, and not someone trusted by the car owner the car may then lock itself, and contact the police station if needed. The car will also be equipped with GPS module to give the owner the ability to track his car through his mobile phone, and to define its' location in case of emergencies.

For the purpose of in-vehicle network security the proposed solution starts first at analyzing the defined threat model in CAN bus. This threat model analysis will result in a solid knowledge of CAN bus common security attacks. Then, countermeasures will be added to the communication system on CAN bus to eliminate or limit the ability of those attacks.

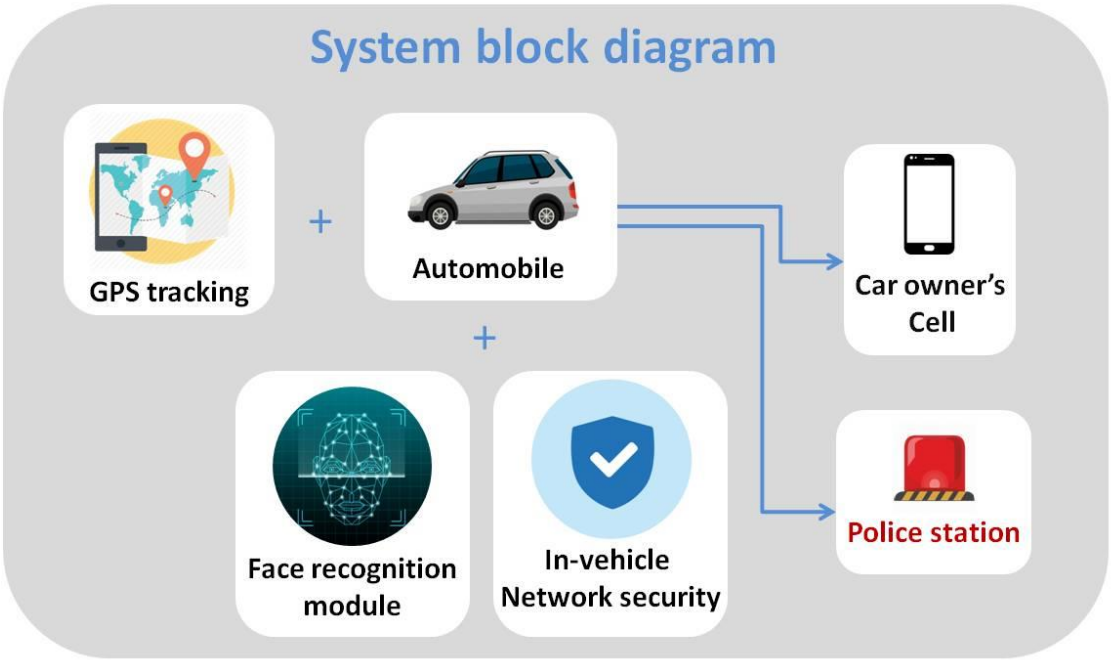


Figure 1-3: Proposed system

1.4 Organization

The following chapters will discuss the work done to achieve the proposed system for improved vehicles security. The two major points of investigation and discussion in this system will be the face recognition module and the in-vehicle network security module. Each of them will be discussed separately and in details. The flow will include the face recognition module in chapters 2, 3 and 4 starting from some basic knowledge about the deployed technology, literature survey, system details, reached results and some suggested future work to improve the system.

Then CAN security system will be discussed in chapters 5, 6 and 7 which will expose basic info about in-vehicle CAN and the communication nature in the network, study the previous work in the domain, state the proposed system elements, describe system performance and point to some future work in the way to more developed CAN bus security system.

Chapter 2: Background and Related Work

2.1 Neural Networks

2.1.1 Overview

Neural networks are multi-layer networks of neurons that we use to classify things, make predictions, etc. It's called a neural network as it works similarly to the human brain's neural network. A "neuron" is considered a mathematical function that collects and classifies information based on a specific architecture. A typical neural network consists of input layer, hidden layers and output layer as shown in Figure 2-1. The input layer collects input patterns. The output layer has classifications according to input pattern. Hidden layers tune the input weightings until the margin of error is minimal.

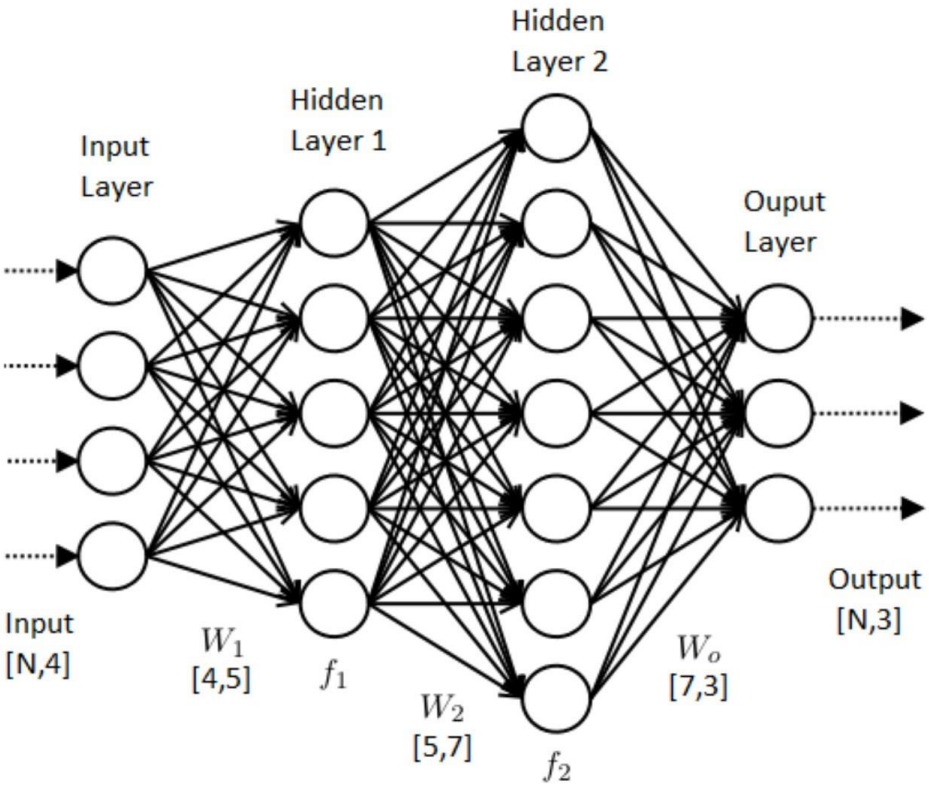


Figure 2-1: Neural Network

2.1.2 Forward and Backward Propagation

Forward propagation is the operation of the calculation and storage of intermediate variables for the neural network from the input layer to the output layer [4]. In the forward propagation, the weights and bias are initialized randomly in the beginning. Then the weighted sum of activation and bias (z) is calculated. After obtaining z, the activation function can be applied to it which will be discussed later.

$$z = Input_i \times w_i + b$$

Backpropagation is the method of calculating the gradient of neural network parameters. Therefore, the method traverses the network, from the output to the input layer. We can define a cost function that measures how good our neural network performs which will be discussed in detail later. For a certain input, and desired output, y, the cost of a specific training example can be calculated as the square of the difference between the network's output and the desired output, that is:

$$C_k = (Output - y)^2$$

The overall cost of a training set is the average of the individual cost functions of the data in the training set. For the purpose of improving the performance of the neural network on the training examples, the weights and bias are tuned, and hopefully, lower the total cost. In order to know how much the specific weights and bias affect the total cost, the partial derivatives of the total cost with respect to the weights and bias are calculated using the chain rule. Then the weights and bias are updated like this

$$w_i = w_i - \alpha \, dw_i$$

$$b = b - \alpha \, db$$

where α is the learning rate. After finishing the first iteration of the backward propagation. The forward propagation can be proceeded, calculate the cost, and then go back to the backward propagation again as shown in Figure 2-2.

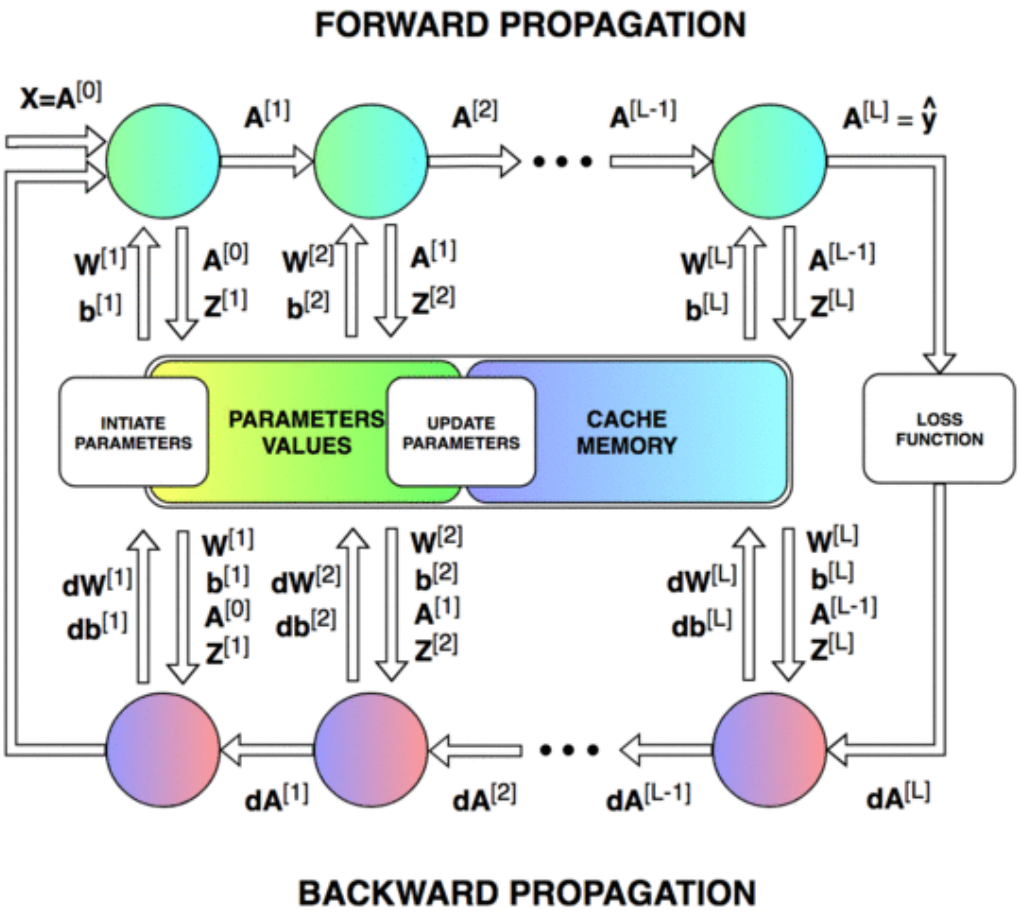


Figure 2-2: Forward and Backward Propagation

2.1.3 Activation Functions

Neural network activation functions are a significant component of deep learning. Activation functions are mathematical equations that determine the output of neural network like yes or no. They map the resulting values in between 0 to 1 or -1 to 1 etc. (depending upon the function). Activation functions introduce non-linearity into the output of a neuron and the purpose of non-linearity in a neural network is to produce a nonlinear decision boundary via non-linear combinations of the weight and inputs. Activation functions also affect greatly on the ability of neural network to converge and the convergence speed, or in some cases, they might prevent neural networks from converging in the first place [5]. They are applied after the hidden layers and the output layer. Types of activation functions as shown in Figure 2-3 are:

2.1.3.1 Sigmoid Function

It is a function which has a graph that looks like **'S'** shaped graph. It is represented by this equation:

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

where $\sigma(z)$ is between 0 and 1. It is usually used in output layer of a binary classification due to its range.

2.1.3.2 Tanh Function

This function works almost always better than sigmoid function. It is actually mathematically shifted version of the sigmoid function and also known as **Tangent Hyperbolic function**. It is represented by this equation:

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

where $\tanh(z)$ is between -1 and 1. It is usually used in hidden layers of a neural network as its values lies between -1 to 1 hence the mean for the hidden layer comes out be 0 or very close to it. This increases the capability of learning for the next layer.

2.1.3.3 ReLU Function

It stands for **Rectified linear unit**. It is the most widely used activation function. Since, it is used in almost all the convolutional neural networks. It is represented by this equation:

$$R(z) = \max(0, z)$$

where $R(z)$ is between 0 and ∞. It is less computationally expensive than tanh and sigmoid because it involves simpler mathematical operations.

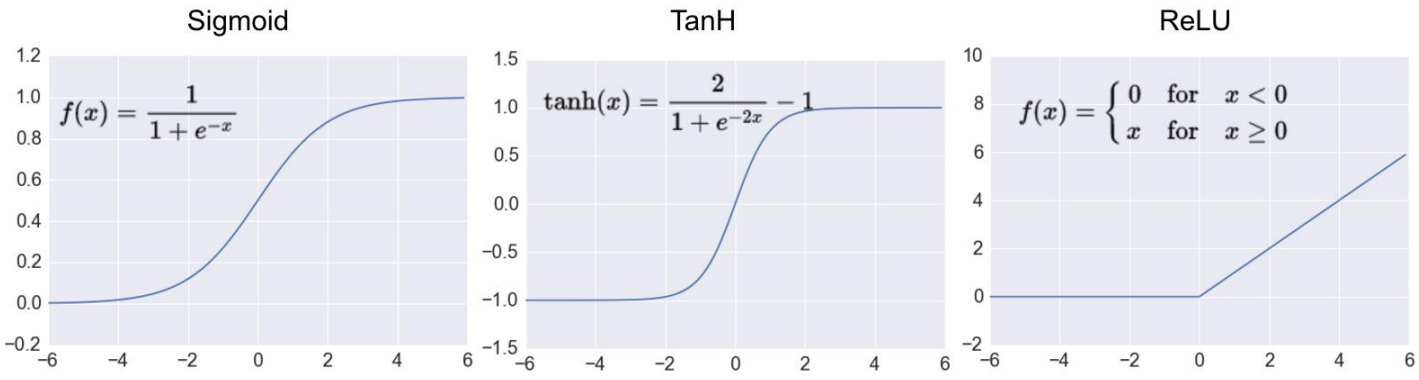


Figure 2-3: Types of Activation Functions

2.1.4 Cost Function

A cost function is a method to determine the error between the output of the network and the given target value. Cost function is expressed as the square of difference between the predicted value and the actual one. It can be estimated by running the model iteratively in order to compare the actual values against the estimated predictions.

The purpose of cost function is to be minimized as shown in Figure 2-4, then returned value is usually called cost, loss or error. The goal is to find the values of model parameters for which cost function return as small number as possible, and the way to do that is by using gradient descent which is an efficient optimization algorithm that attempts to find a local or global minima of a function. Gradient descent enables a model to learn the direction that the model should take in order to reduce errors. As the model iterates, it gradually converges towards a minimum where the parameters produce little or zero changes in the loss.

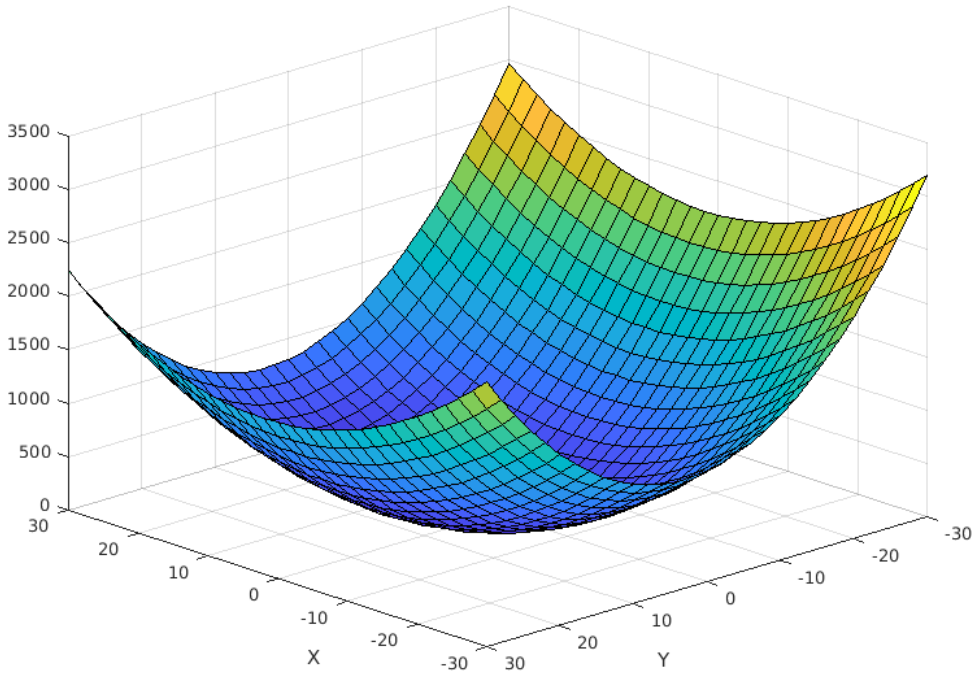


Figure 2-4: Cost Function vs. number of iterations

2.2 Convolutional Neural Networks

Convolutional Neural Networks are so similar to ordinary Neural Networks. They are made up of neurons that have learnable weights and biases. Each neuron is fed with some inputs, performs a dot product and optionally follows it with an activation function. And they are finished with a loss function (e.g. SVM/Softmax) on the last fully-connected layer [6].

So what changes? CNN architectures make the explicit assumption that the inputs are visual data such as images. A CNN is composed of two basic parts which are feature extraction and classification as shown in Figure 2-5. Feature extraction includes many convolution layers that can be followed by max-pooling and an activation function. The classifier usually consists of fully connected layers.

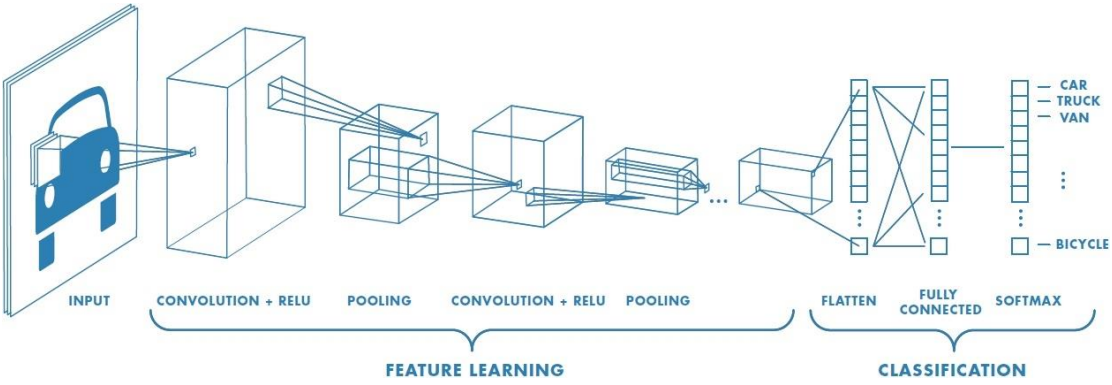


Figure 2-5: Convolutional Neural Network

2.3 Convolutional Neural Network Layers

2.3.1 Convolutional Layer

The convolutional layer is the basic building block of a Convolutional Neural Network. It is the first layer that extract features from an input image using convolution operation that utilizes small squares of input data in order to learn features of image. It is a mathematical operation that is applied on two inputs such as image matrix and a filter.

The convolutional layer's parameters consist of a set of learnable filters. Every filter has width and height smaller than the width and the height of input volume, but has the same number of channels. For example, the first layer of CNN might have size 5x5x3 which means that the input image is RGB or has 3 channels.

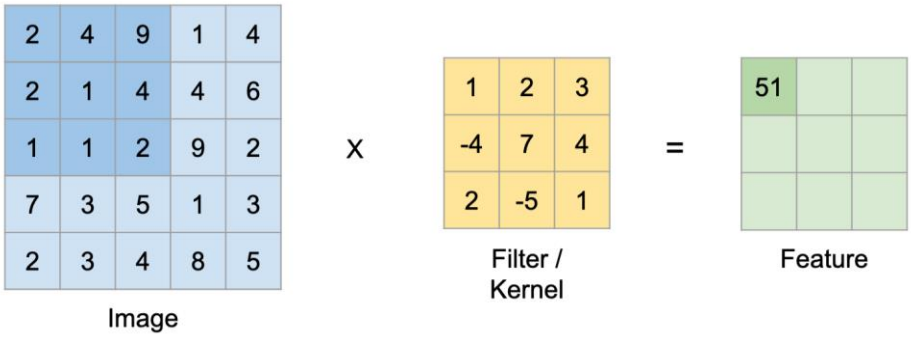


Figure 2-6: Convolution Operation

During the forward pass in Figure 2-6, we slide and convolve each filter across the width and height of the input volume and compute dot products between the entries of the filter and the input at any position. And then a 2-dimensional activation map will be produced that gives the responses of that filter at every spatial position. Every filter will produce some type of visual feature such as a blotch of some color or edge of some orientation on the first layer, or eventually wheel-like patterns or entire honeycomb on higher layers of the network.

2.3.2 Pooling layer

Section of pooling layers will lower the number of parameters when the images are too large. Spatial pooling also known as subsampling or downsampling which reduces each map's dimensionality but retains important details.

Spatial pooling can be of different types:

- Max Pooling
- Average Pooling
- Sum Pooling

Max pooling as shown in Figure 2-7 is taken from the rectified feature map as the largest dimension. The pooling units may also perform other functions, such as average pooling, or even L2-norm pooling, in addition to max pooling. Average pooling has often been used traditionally but recently has fallen out of favor in contrast with the max pooling system, which has been shown to perform better in practice.

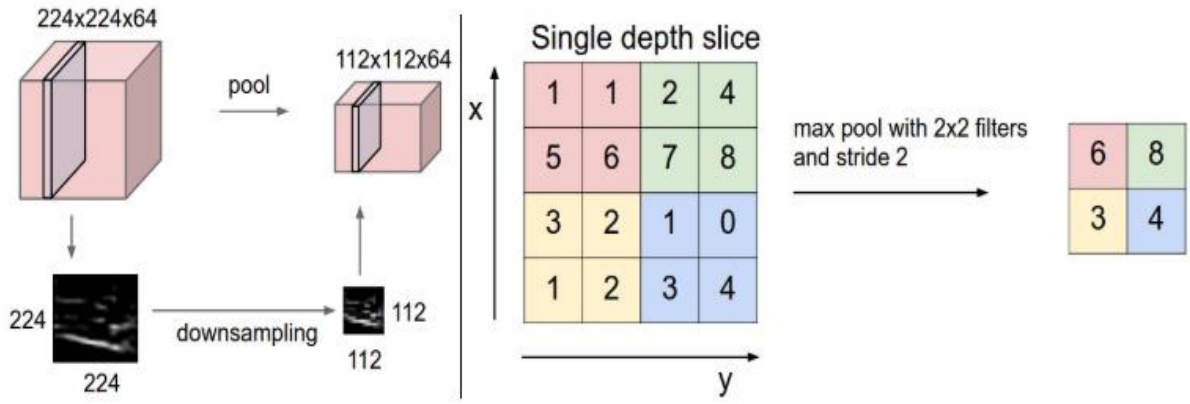


Figure 2-7: Max Pooling

2.3.3 Fully connected layer

This layer is called as fully connected layer as it connects every neuron in one layer to every neuron in another layer.

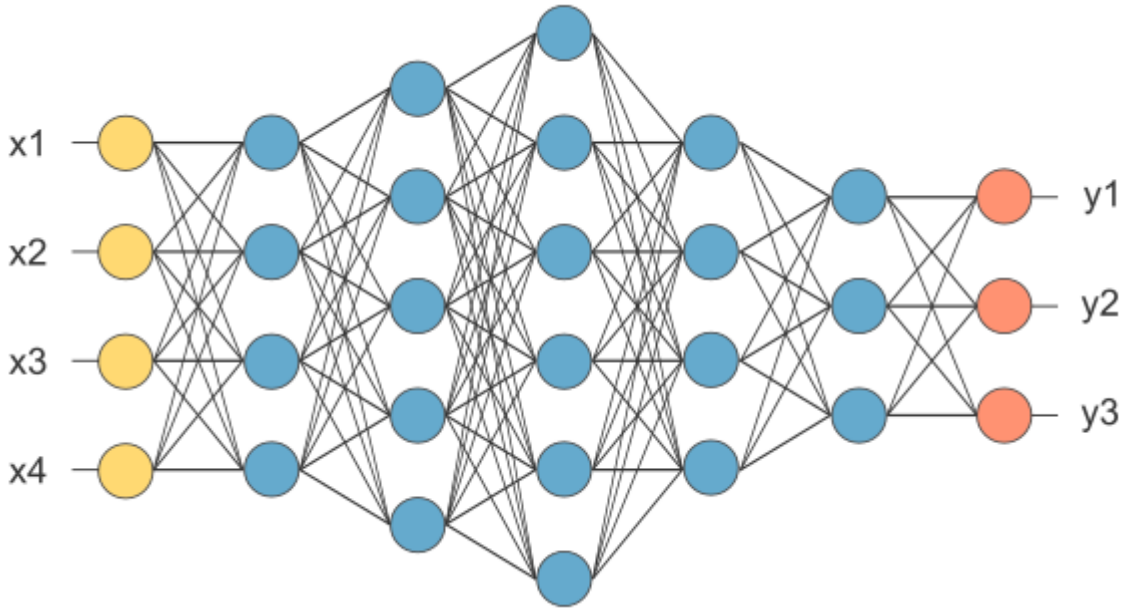


Figure 2-8: Fully Connected Layers

In Figure 2-8, the feature map matrix will be converted as vector (x1, x2, x3 and x4). With the fully connected layers, these features are combined together to create a model. Finally, an activation function such as softmax or sigmoid is applied to classify the outputs.

2.4 Classic Architectures

2.4.1 LeNet-5

LeNet-5, a 7-layer convolutional network by LeCun et al in 1998 that classifies digits. Several banks used it to recognize hand-written numbers on checks digitized in 32x32 pixel greyscale input images. The LeNet-5 architecture is constrained by the availability of computing resources. Since, the ability to process higher resolution images requires larger and more convolutional layers [7]. The architecture consists of two sets of convolutional and average pooling layers, followed by a fully-connected convolutional layer, then a fully-connected layer and finally a softmax classifier as shown in Figure 2-9.

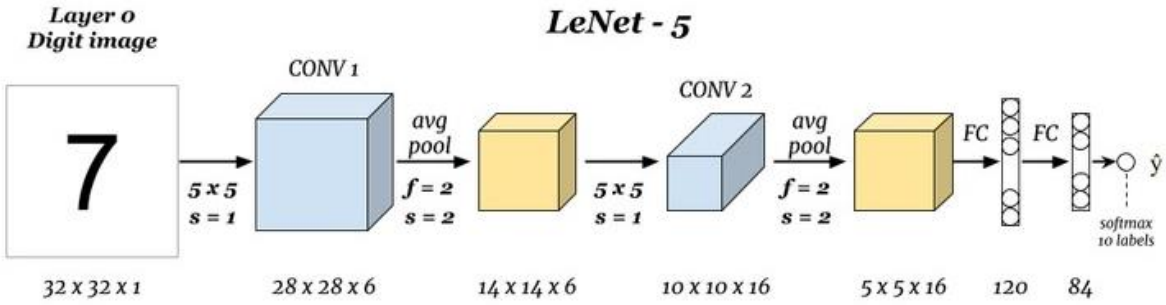


Figure 2-9: LeNet-5 Architecture

2.4.2 AlexNet

AlexNet is a Deep Convolutional Neural Network (CNN) for image classification that won the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry, named after Alex Krizhevsky, who was the first author of the paper describing this work. The network had a very similar architecture as LeNet-5 but was deeper, with more filters per layer, and with stacked convolutional layers.

AlexNet has 5 convolutional layers and 3 fully connected layers. And between them, there are some pooling and activation layers. AlexNet input starts with 227×227×3 images. And then the first layer applies a set of 96 of 11×11 filters with a stride of 4. And because it uses a large stride of 4, the dimensions shrinks to 55×55. And then it applies max pooling with a 3×3 filter and a stride of 2. So this reduces the volume to 27×27×96, and then it performs a 5×5 same convolution, same padding, so it ends up

with 27×27×276. Followed by max pooling again, this reduces the height and width to 13. And then another same convolution, so same padding. So it's now 13×13×384. And then 3×3, same convolution again. Then 3×3, same convolution. Then max pool, brings it down to 6×6×256. If all these numbers are multiplied, 6×6×256, that's 9216. And then finally, it has a few fully connected layers that uses a softmax to output which one of classes the object could be as shown in Figure 2-10. AlexNet had about 60 million parameters.

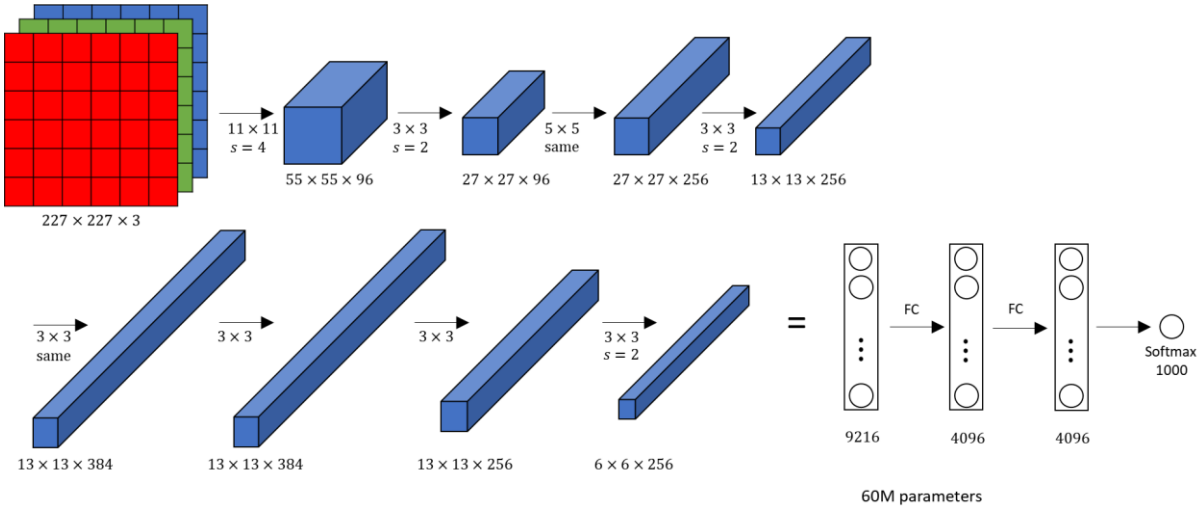


Figure 2-10: AlexNet Architecture

AlexNet made a number of changes that helped in developing networks that came after such as using ReLU instead of Tanh to add non-linearity which accelerates the speed by 6 times at the same accuracy, using dropout instead of regularization to handle overfitting, and using spatial pooling to reduce the size of network which reduces the top-1 and top-5 error rates by 0.4% and 0.3%, respectively.

2.4.3 VGG-16

VGG-16 is a convolutional neural network by K. Simonyan and A. Zisserman from the University of Oxford. VGG-16 achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images. VGG-16 model was one of the famous models submitted to ILSVRC-2014. It makes improvement over AlexNet by replacing large kernel-sized filters with multiple 3×3 kernel-sized filters.

The input to conv1 layer is of fixed size 224×224 RGB image. The image is passed through a stack of convolutional layers, where the filters were used with a very small receptive field 3×3. Convolution filters of 1×1 is also utilized in one of the configurations, which can be considered as a linear transformation of the input channels. The convolution stride is fixed to 1 pixel. Spatial pooling is done by five max-pooling layers, which follow some of the convolutional layers. Max-pooling is performed over a 2×2 window, with stride 2. Three Fully-Connected layers follow a stack of convolutional layers followed by a softmax layer. All hidden layers are equipped with the rectification non-linearity as shown in Figure 2-11. The network contains almost 140 million parameters.

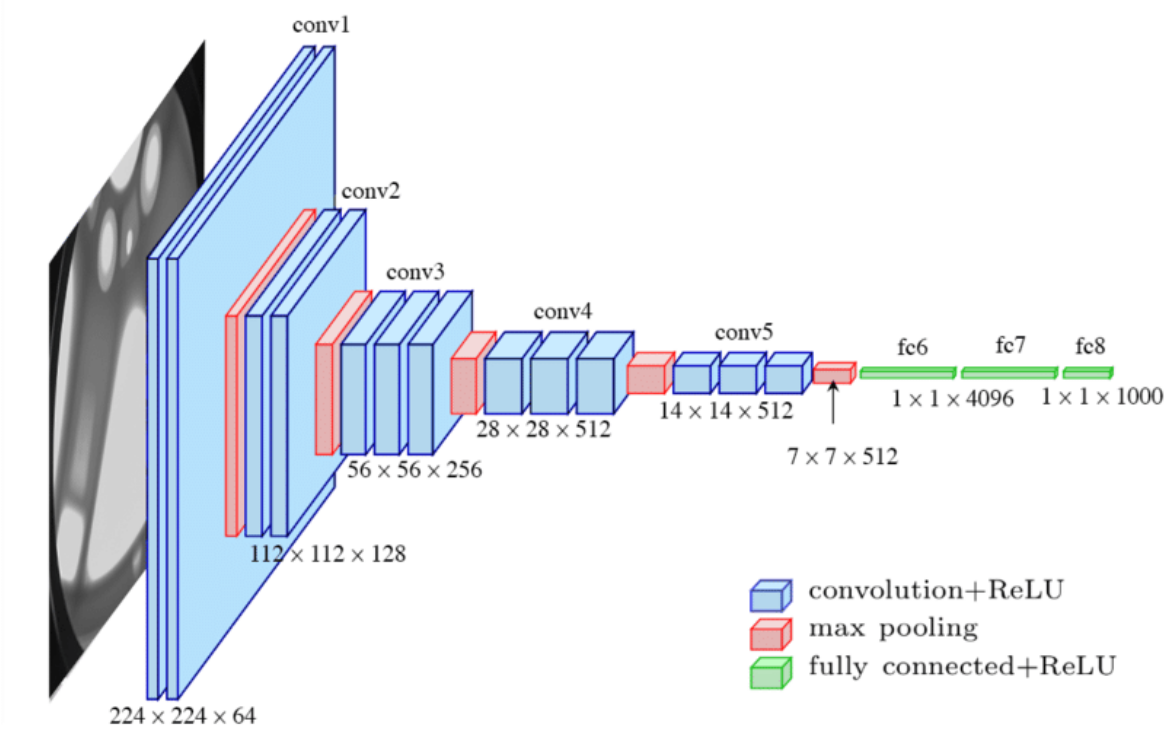


Figure 2-11: VGG-16 Architecture

Chapter 3: System Design

Face recognition is the task of making a positive identification of face in a photo or video against a pre-existing database of faces. It begins with detecting and distinguishing human faces from other objects in the image and then works on identification of those detected faces.

Face recognition technology based on CNN has become the main method adapted in the field of face recognition, so we use convolutional Siamese network for face recognition which is one of the most popularly one-shot learning algorithms.

In our project we first tried a pre-trained face recognition model, then we build our model to achieve the objective of identification before sending the required result over communication protocol.

In this chapter, we will discuss the models we used in face recognition, how they work and our contribution to make them fit the objective of face identification. We also will discuss the liveness detection and our chosen technique to make sure there is a live person in front of the camera. So the flow of work will be as follow:

- First, a pre-trained model for face recognition have been applied
- We started to build our model from scratch and we did our best to achieve the best accuracy, so we can summarize our work in some points:
    - Choosing the CNN architecture which achieve high accuracy.
    - Choosing the dataset which the model can be trained on.
    - Dividing the dataset into pairs (positive and negative pairs)
    - Using one shot learning to train the pairs of images.
    - Choosing similarity function to compute similarity between input images.
    - Choosing activation function that turns the output to 0 or 1.
- Implementing the liveness detection using eye blink detection technique.

All previous points will be discussed in this chapter.

3.1 Pre-trained Model

This pre-trained model uses dlib liberary to perform face recognition. It uses ResNet network with 29 conv layers which is essentially a version of the ResNet-34 network. And it has accuracy of 99.38% on the standard Labeled Faces in the Wild benchmark.

To build face recognition system, first face detection should be performed, and then face embedding would be extracted from each face using deep learning. The next step is to train face recognition model on the embedding and finally recognize faces in images or in a video stream.

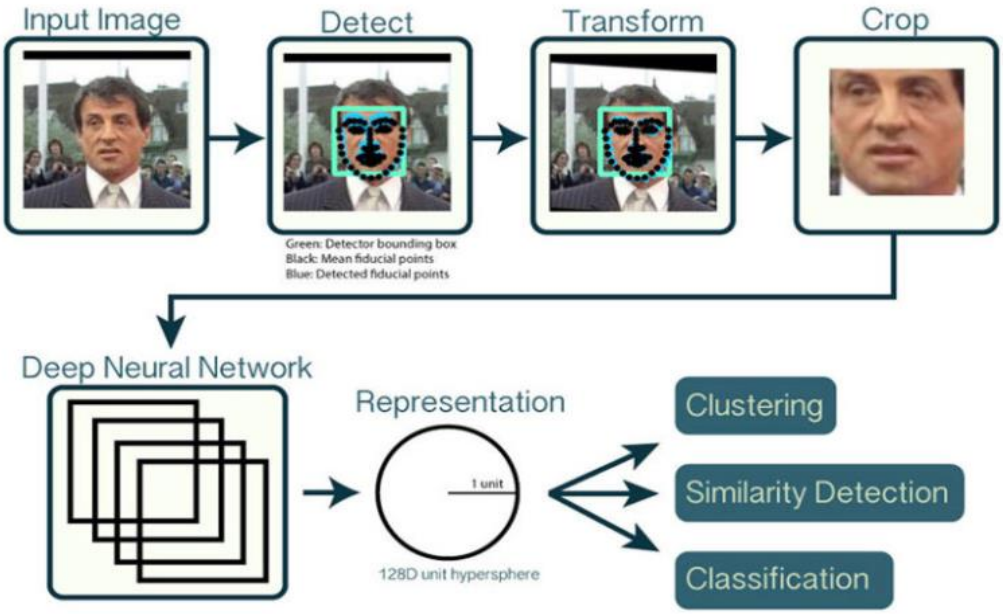


Figure 3-1: Pre-trained Model Pipeline

As shown in Figure 3-1, dlib liberary and deep learning are applied together to:

1. Detect faces
2. Compute 128-d face embedding to quantify a face
3. Train the network
4. Recognize faces in images and video streams

3.1.1 Face Detection

Face detection is a necessary step before applying face recognition as it is important to feed the network with person's face cropped without any objects around him, which makes the model to work efficiently.

Face detector in this model is applied using dlib library, it is divided into two methods:

3.1.1.1 HoG Face Detector in Dlib

This is a widely used model of face detection, based on HoG and SVM features. The model is built out of 5 HOG filters: front looking, left looking, right looking, front looking but rotated left, and a front looking but rotated right. The training dataset consists of 2825 images from the LFW dataset. The **pros** of this method can be explained as follows:

1. Fastest method on CPU
2. Works very well for frontal and slightly non-frontal faces
3. Light-weight model as compared to the other three.
4. Works under small occlusion

This approach essentially works in most cases except for those cases that will be discussed below:

Cons

1. The major drawback is that it does not detect small faces as it is trained for minimum face size of 80×80. Thus, the face size must be more than that in the application. You can however, train your own face detector for smaller sized faces. The bounding box often excludes part of forehead and even part of chin sometimes.
2. Does not work very well under substantial occlusion
3. Does not work for side face and extreme non-frontal faces, like looking down or up.

3.1.1.2 CNN Face Detector in Dlib

This approach uses a CNN based Maximum-Margin Object Detector (MMOD). The training process is very easy for this system, and you do not need a large amount of data to train a custom object detector. It uses a dataset manually labeled, consisting of images from various datasets like ImageNet, PASCAL VOC, VGG, WIDER, Face Scrub. It contains 7220 images.

Pros

1. Works for different face orientations
2. Robust to occlusion
3. Works very fast on GPU
4. Very easy training process

Cons

1. Very slow on CPU
2. The bounding box is even smaller than the HoG detector

However, the model gives the user the ability to choose one of these two methods depending on his requirements or his application.

3.1.2 Embedding extraction

Deep learning works with face recognition by using a technique called deep metric learning. In deep learning, it's known that the network is trained to output a classification or a label for that image.

However, deep metric learning is different, instead of trying to output a single label, the network is outputting a real-valued feature vector. For the dlib face recognition network, the feature vector used to quantify the face is 128-d.

After detecting the face from the given images and initializing the parameters, the network extracts the 128-diminsion encoding for each face in the images as shown in Figure 3-2.

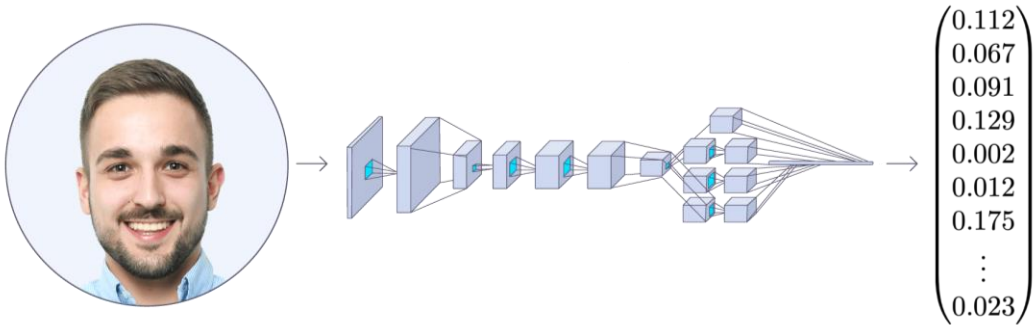


Figure 3-2: Embedding Extraction

3.1.3 Training the model

Training the network is done using triplets, two of these images are of the same person and third image is a random face from the dataset and is a different person as shown in Figure 3-3.

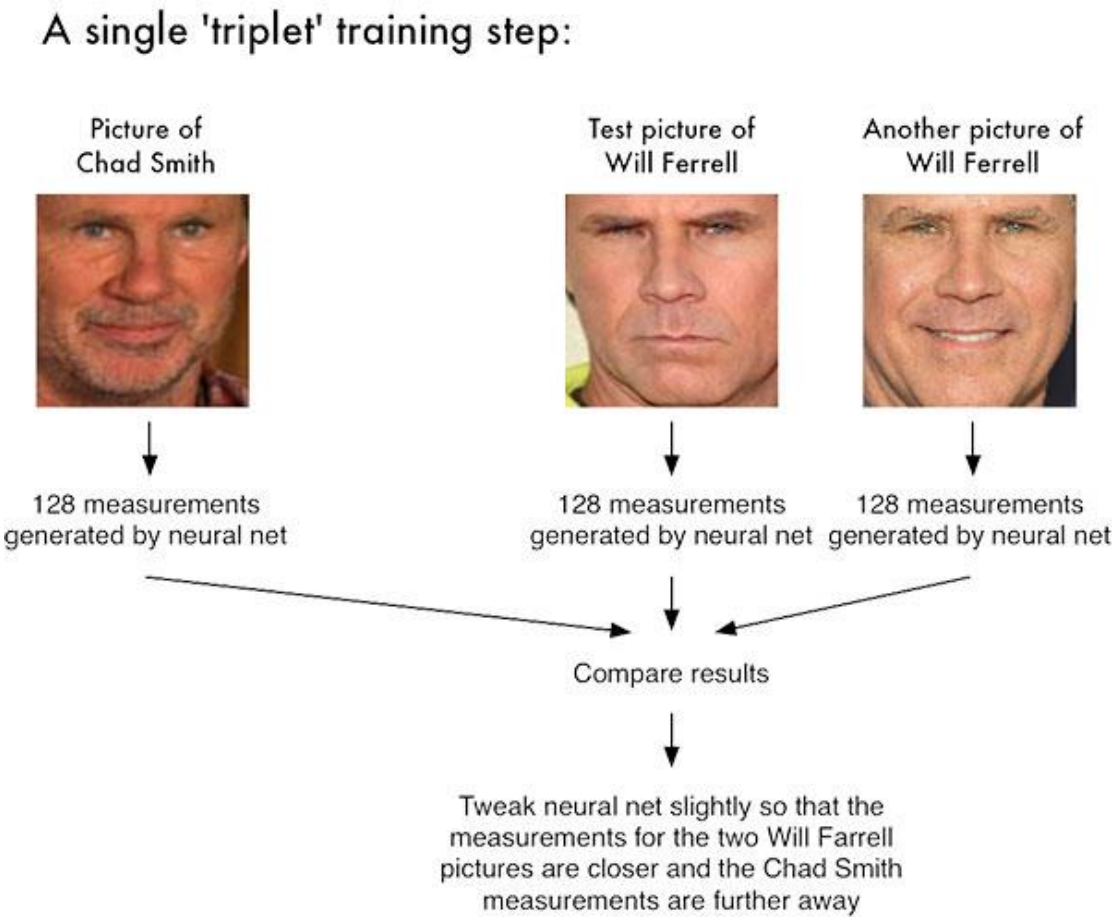


Figure 3-3: Triplet Training

First, the network constructs the 128-dimension embedding for each face. From there, the general idea is to learn the weights of the neural network so that the 128-d embedding of the same person will be closer to each other and farther from the third embedding.

3.1.4 Recognizing faces

Now, the model becomes ready to recognize faces from any image by quantifying the face and extracting 128-d face embedding, then comparing it with all faces embedding in the database to know if this person is one of the known faces or not.

3.2 ResNet

Researchers observed that when it comes to convolutional neural networks "the deeper the better". However, it's been noticed that after some depth, the performance degrades and accuracy gets saturated. To overcome this problem, a deep residual learning framework is introduced which is also called ResNet. The chosen CNN architecture for training is ResNet due to its high accuracy which is needed in face recognition application to identify people correctly.

3.2.1 Overview

Deep Residual Network is almost similar to the networks which have convolution, pooling, activation and fully-connected layers stacked one over the other. The only construction to the simple network to make it a residual network is the ***identity connection*** between the layers which will be discussed later. ResNet won 1st place in the ILSVRC 2015 classification competition with top-5 error rate of 3.57% (An ensemble model) and also won the 1st place in ILSVRC and COCO 2015 competition in ImageNet Detection, ImageNet localization, Coco detection and Coco segmentation. There are multiple versions of ResNet architectures such as ResNet-34, ResNet-50 and ResNet-152 etc. but the choice is on ResNet-34 and ResNet-50 since they are not very deep so they can avoid overfitting. ResNet-34 has top-1 error rate of 21.84% and top-5 error rate of 5.71% while ResNet-50 has top-1 error rate of 20.74 % and top-5 error rate of 5.25% on ImageNet validation set.

3.2.2 Network Architecture

ResNet-34 consists of 34 layers and 21.8 million parameters. It consists of convolutional layers, average pooling and a fully connected layer and has 3.6 billion FLOPs (multiply-adds).

To illustrate the architecture well, two models are described as follows:

Plain Network: This network (Figure 3-4, middle) is based on VGG nets (Figure 3-4, left). The convolutional layers mostly contains 3x3 filters and follow two simple design rules:

- The number of filters is doubled if the feature map size is halved to preserve the time complexity per layer.

- The layers have the same number of filters for the same output feature map size.

Convolutional layers that have a stride of 2 are responsible of downsampling in the network. The network ends with a global average pooling layer and a 1000-way fully-connected layer with softmax. The total number of weighted layers is 34. It is noticed that the plain model has fewer filters and lower complexity than VGG nets since number of FLOPs is only 18% of VGG-19 (19.6 billion FLOPs).

Residual Network: Based on the above plain network, shortcut connections are inserted (Figure 3-4, right) which turn the network into its counterpart residual version. The identity shortcuts can be directly used when the input and output are of the same dimensions (solid line shortcuts in Figure 3-4). When the dimensions increase (dotted line shortcuts in Figure 3-4) [8].

Each layer contains parameters which are composed of weights and biases and are needed in convolution operation, detailed description of layers and parameters are given in Table 3-1 where the image input size is 224×224×3 and batch normalization is applied right after each convolution and before activation.

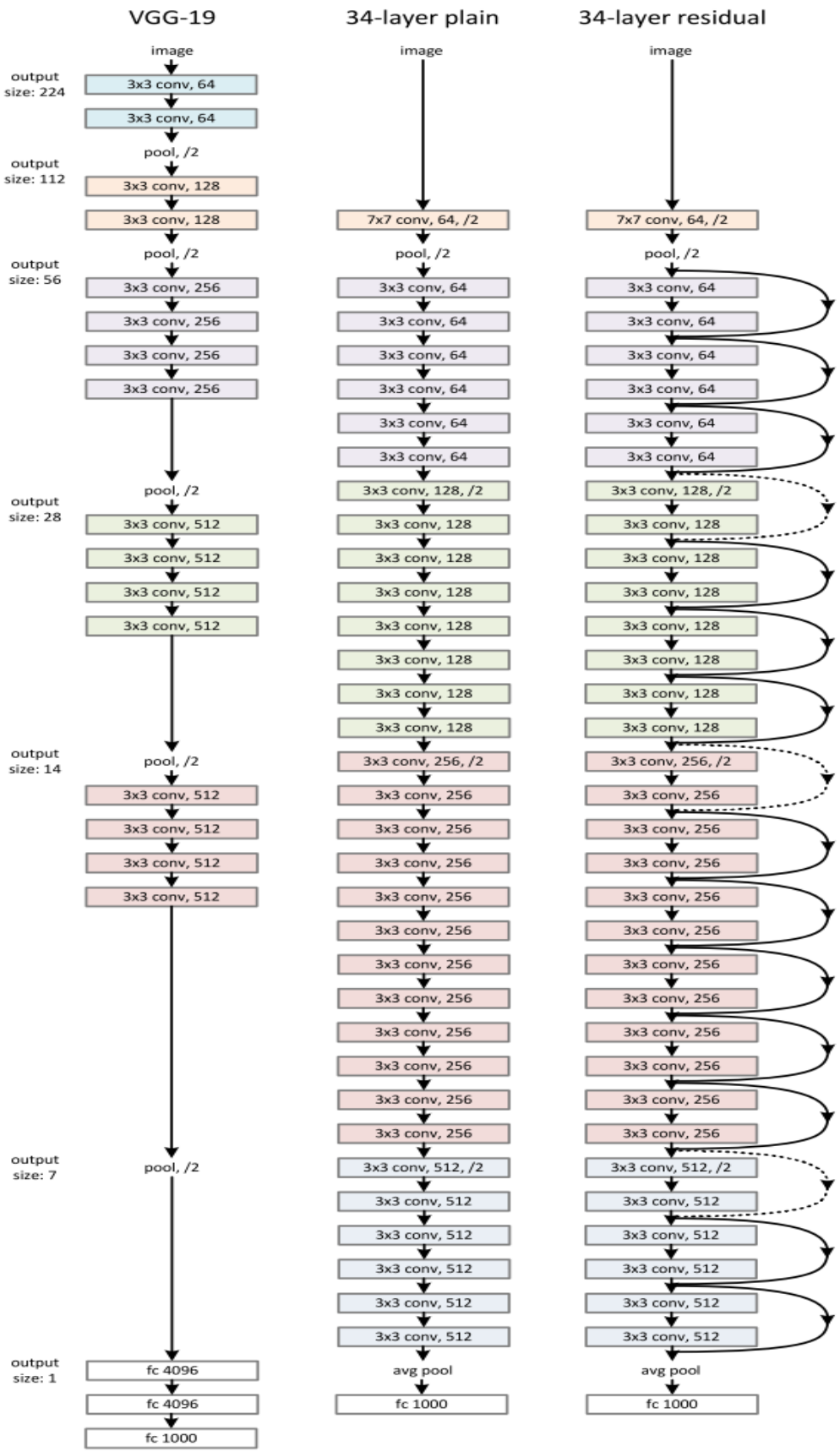


Figure 3-4: Network Architectures. Left: VGG-19 model. Middle: Plain Network. Right: Residual Network

Table 3-1: Detailed Description of all ResNet-34 Layers and their Parameters

ID	Layer Name	Layer Type	Kernel	Stride	Padding	Output Size	#Params
1	Conv1	Convolution	7×7	2	3	112×112×64	9.728K
	Pool1	Max Pooling	3×3	2	1	56×56×64	-
2	Conv2_a1	Convolution	3×3	1	1	56×56×64	37.184K
3	Conv2_a2	Convolution	3×3	1	1	56×56×64	37.184K
4	Conv2_b1	Convolution	3×3	1	1	56×56×64	37.184K
5	Conv2_b2	Convolution	3×3	1	1	56×56×64	37.184K
6	Conv2_c1	Convolution	3×3	1	1	56×56×64	37.184K
7	Conv2_c2	Convolution	3×3	1	1	56×56×64	37.184K
8	Conv3_a1	Convolution	3×3	2	1	28×28×128	83.2K
9	Conv3_a2	Convolution	3×3	1	1	28×28×128	148.096K
10	Conv3_b1	Convolution	3×3	1	1	28×28×128	148.096K
11	Conv3_b2	Convolution	3×3	1	1	28×28×128	148.096K
12	Conv3_c1	Convolution	3×3	1	1	28×28×128	148.096K
13	Conv3_c2	Convolution	3×3	1	1	28×28×128	148.096K
14	Conv3_d1	Convolution	3×3	1	1	28×28×128	148.096K
15	Conv3_d2	Convolution	3×3	1	1	28×28×128	148.096K
16	Conv4_a1	Convolution	3×3	2	1	14×14×256	330.24K
17	Conv4_a2	Convolution	3×3	1	1	14×14×256	591.104K
18	Conv4_b1	Convolution	3×3	1	1	14×14×256	591.104K
19	Conv4_b2	Convolution	3×3	1	1	14×14×256	591.104K
20	Conv4_c1	Convolution	3×3	1	1	14×14×256	591.104K
21	Conv4_c2	Convolution	3×3	1	1	14×14×256	591.104K
22	Conv4_d1	Convolution	3×3	1	1	14×14×256	591.104K
23	Conv4_d2	Convolution	3×3	1	1	14×14×256	591.104K
24	Conv4_e1	Convolution	3×3	1	1	14×14×256	591.104K
25	Conv4_e2	Convolution	3×3	1	1	14×14×256	591.104K
26	Conv4_f1	Convolution	3×3	1	1	14×14×256	591.104K
27	Conv4_f2	Convolution	3×3	1	1	14×14×256	591.104K
28	Conv5_a1	Convolution	3×3	2	1	7×7×512	1.31584M
29	Conv5_a2	Convolution	3×3	1	1	7×7×512	2.361856M
30	Conv5_b1	Convolution	3×3	1	1	7×7×512	2.361856M
31	Conv5_b2	Convolution	3×3	1	1	7×7×512	2.361856M
32	Conv5_c1	Convolution	3×3	1	1	7×7×512	2.361856M
33	Conv5_c2	Convolution	3×3	1	1	7×7×512	2.361856M
	Avg_pool	Average Pooling	7×7	1	0	1×1×512	-
34	1000-d_fc	Fully Connected	-	-	-	1000	513K

3.2.2.1 Identity Block

The identity block is the standard block used in ResNets, and corresponds to the case where the input activation has the same dimension as the output activation. To illustrate the different steps of what happens in a ResNet's identity block, Figure 3-5 shows the individual steps as the lower path is the main path, and the upper path is the shortcut path which skips 2 layers.

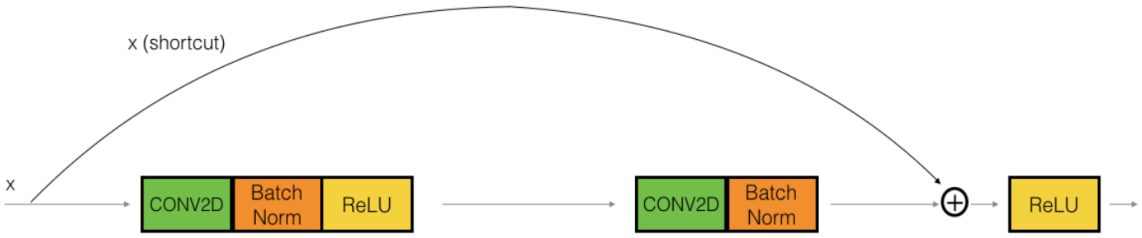


Figure 3-5: Identity Block

3.2.2.2 Convolutional Block

The ResNet convolutional block is the second block type. This type of block is used when the input and output dimensions don't match up. The difference with the identity block is the existence of a convolutional layer in the shortcut path as shown in Figure 3-6. This convolutional layer is used to resize the input in order to make the input and output dimensions match up in the final addition needed to add the shortcut value back to the main path.

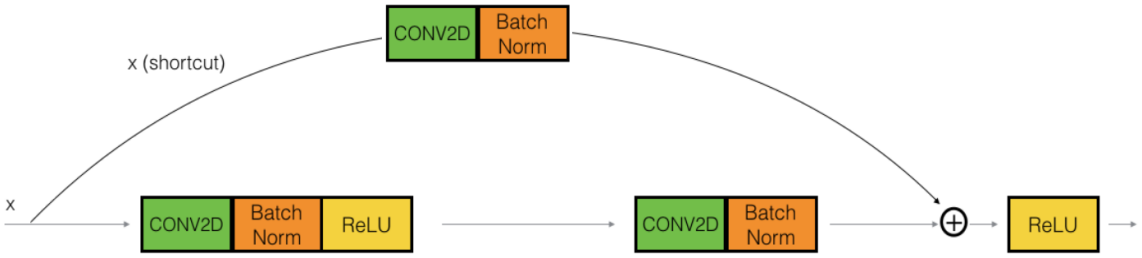


Figure 3-6: Convolutional Block

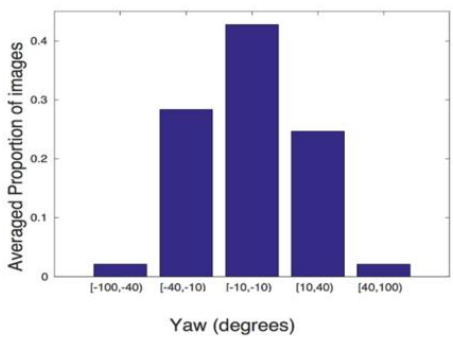
ResNet-50 is similar to ResNet-34, but each 2-layer block in the 34-layer is replaced with a 3-layer block, resulting in a 50-layer ResNet and it has 25.6 million parameters.

3.3 VGGFace2 Dataset

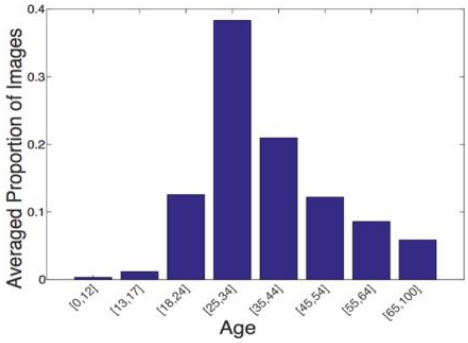
Datasets are an integral part of the field of machine learning. In computer vision, face images have been used extensively to develop facial recognition systems, face detection, and many other applications that use images of faces. Table 3-2 shows the comparison of several face datasets. The chosen dataset for training is VGGFace2 which is a large-scale face recognition dataset. Images are downloaded from Google Image Search and have large variations in pose, age, illumination, ethnicity and profession as shown in Figure 3-7. The dataset contains 3.31 million faces of more than 9000 identities, with an average of 362.6 images for each identity. It contains images from identities spanning a wide range of different ethnicities, accents, professions and ages. All face images are captured "in the wild", with pose and emotion variations and different lighting and occlusion conditions [9]. The dataset is divided into two parts: one for training which contains 8631 classes, and the other is for test which contains 500 classes.

Table 3-2: Statistics for recent public face datasets

Datasets	**# of identities**	**# of images**	**year**
LFW	5, 749	13, 233	2007
YTF	1, 595	3, 425 videos	2011
CelebFaces+	10, 177	202, 599	2014
CASIA-WebFace	10, 575	494, 414	2014
IJB-A	500	5, 712 images, 2, 085 videos	2015
IJB-B	1, 845	11, 754 images, 7, 011 videos	2017
IJB-C	3, 531	31, 334 images, 11, 779 videos	2018
VGGFace	2, 622	2.6 M	2015
MegaFace	690, 572	4.7 M	2016
MS-Celeb-1M	100, 000	10 M	2016
UMDFaces	8, 501	367, 920	2016
UMDFaces-Videos	3, 107	22, 075 videos	2017
VGGFace2	9, 131	3.31 M	2018



(a) pose statistics



(b) age statistics

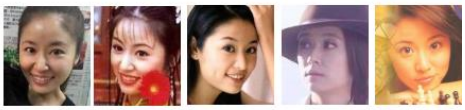
(c) John Wesley Shipp

(d) Leymah Gbowee

(e) Princess Haya Bint Al Hussein

(f) Julio César Chávez Jr.

(g) Roy Jones Jr.



(h) Ruby Lin

(i) Additi Gupta

(j) Lee Joon-gi

Figure 3-7: (a-b) VGGFace2 poses and ages' statistics. (c-j) example images for eight subjects with different ethnicities.

3.4 Training

3.4.1 One Shot Learning

In the previous chapter, we talked about Convolutional Neural Network and how it works. In this section, we will discuss using CNN to apply face recognition.

One of the challenges that faces us that we need to recognize a person given one single image for the person's face, but historically, deep learning algorithms can't work well if we have only one training example. So we have to use one shot learning and similarity function to do this job.

The main idea of one shot learning based on computing similarity between the input image and all images in the database to recognize which person in the database is more likely to be the same person in the input image as shown in Figure 3-8. A good way to do this is to use a Siamese network: through a sequence of convolutional, pooling and fully connected layers, we end up with a feature vector of let's say 128 numbers.

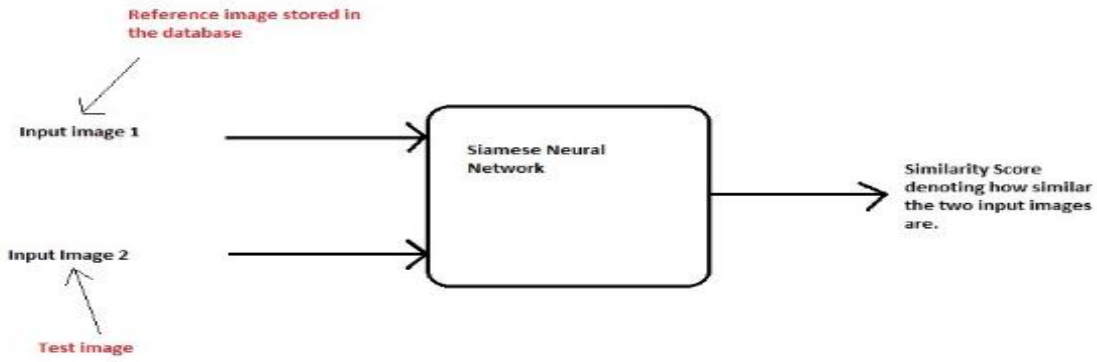


Figure 3-8: One Shot Learning

3.4.1.1 Siamese Neural Network (SNN)

Siamese Networks are a type of Neural Networks which have multiple instance of the same model by sharing same architecture and same weights. Siamese Network solves the problem when we need to add or remove new persons to the data, in Traditional Neural Network we have to update the neural network and retrain it on the new dataset. On the other hand, SNN uses similarity function to compare the encodings of the two images as shown in Figure 3-9. Thus we can learn it to know if they are the same persons or not.

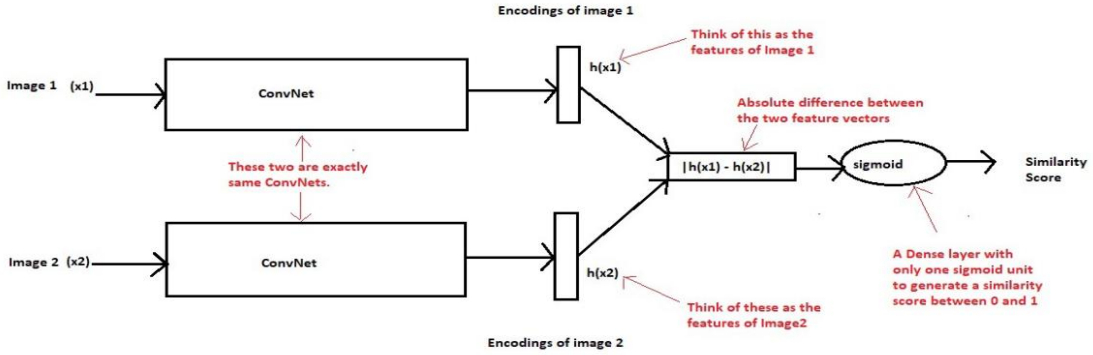


Figure 3-9: Siamese Network

Goal of learning SNN

In deep neural network the word learn means to get the parameters well so that it gives a good encodings of the picture.

We can learn the parameter so that:

$\|f(x^{(1)}) - f(x^{(2)})\|^2$ is small → if the two input images are the same person

$\|f(x^{(1)}) - f(x^{(2)})\|^2$ is large → if the two input images are different person

3.5 Binary Cross Entropy Loss Function

Binary cross entropy is a loss function that is used in binary classification tasks as it answers questions with only two choices as “A or B” as in our case “same person or different persons” so it’s chosen to be used formally. This loss is equal to the average of the categorical cross entropy loss on many two-category tasks. Also called sigmoid Cross-Entropy loss. It is a sigmoid activation plus a Cross-Entropy loss. Unlike softmax loss it is independent for each vector component (class), meaning that the loss computed for every CNN output vector component is not affected by other component values. That’s why it is used for multi-label classification, where the insight of an element belonging to a certain class should not influence the decision for another class.

The binary cross entropy loss function calculates the loss of an example by computing the following average:

$$CE = -\sum_{i=1}^{C'=2} t_i log(f(s_i)) = -t_1 log(f(s_1)) - (1 - t_1) log(1 - f(s_1))$$

The pipeline for each one of the C classes would be as follows. C’ independent binary classification problems is set (C’=2). Sum up the loss over the different binary problems is applied: The gradients of every binary problem are summed up to backpropagate, and the losses to monitor the global loss. s1 and t1 are the score and the ground truth label for the class C1, which is also the class Ci in C. The score and the ground truth label of the class C2 are s2=1 − s1 and t2=1 − t1, which is not a “class” in the original problem with C classes, but a class is created to set up the binary problem with C1=Ci. We can understand it as a background class.

The loss can be expressed as:

$$CE = \begin{cases} -log(f(s_1)) & if \quad t_1 = 1 \\ -log(1 - f(s_1)) & if \quad t_1 = 0 \end{cases}$$

Where f() is the sigmoid function. It can also be written as:

$$\frac{\partial}{\partial s_i}(CE(f(s_i)) = \begin{Bmatrix} f(s_i) - 1 & if \quad t_i = 1 \\ f(s_i) & if \quad t_i = 0 \end{Bmatrix}$$

3.6 Model Design and Hyperparameters

Hyperparameters are all the training variables set manually with a pre-determined value before starting the training, we can also consider the model design components as part of the hyperparameters set which will be illustrated as follows:

3.6.1 Adam Optimizer

Adam is an optimization algorithm that can be used for training deep neural networks to update CNN weights. It's considered different to Stochastic gradient descent which have a fixed learning rate for all weight updates and the learning rate does not change during training. Adam is described as combining the advantages of two other extensions of stochastic gradient descent. Specifically:

- Adaptive Gradient Algorithm (AdaGrad): that maintains a per-parameter learning rate that improves performance on problems with sparse gradients.
- Root Mean Square Propagation (RMSProp): that also maintains per-parameter learning rates that are adapted based on the average of recent magnitudes of the gradients for the weight. This means the algorithm does well on online and non-stationary problems.

Adam uses the squared gradients to scale the learning rate like RMSprop and it takes advantage of momentum by using moving average of the gradient instead of gradient itself like SGD with momentum [10].

Using Adam or non-convex optimization has a lot of attractive benefits as follows:

- Straightforward to implement.
- Invariant to diagonal rescale of the gradients.
- Computationally efficient.
- Little memory requirements.
- Appropriate for non-stationary objectives.
- Well suited for problems that are large in terms of data and/or parameters.
- Appropriate for problems with very noisy/or sparse gradients.

That's why it's chosen to be used in our model.

3.6.2 Learning Rate

The length of a step in the gradient descent mechanism is usually referred to as learning rate. If the search space was to be visualized as a 3D surface, the learning rate is the length of a step taken towards a minimum point of the surface. Learning rate is extremely important because it is the core of the learning process. A high learning rate makes the ML model converge very fast after learning for a short time but later, it does not improve at all or even worse, diverges. If the learning rate is too small, the ML model will converge very slowly and eventually it will reach the optimum. The value of learning rate is determined by tuning then choosing the most suitable value as illustrated in Figure 3-10. A default value for the learning rate is 0.1 or 0.01, and this may represent a good starting point for training a model. So, the chosen value of learning rate in model is 0.01.

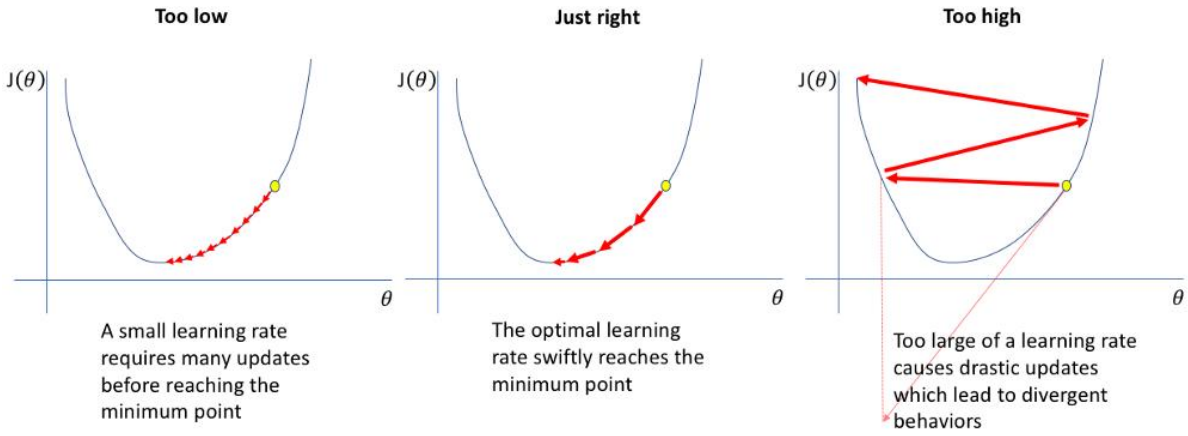


Figure 3-10: Impact of Learning Rate on Gradient Descent

3.6.3 Batch Size

Batch size refers to the number of training examples utilized in one iteration. The batch size can be one of three options:

- Batch mode: where the batch size is equal to the total dataset which in turn makes the iteration and epoch values the same.
- Mini-batch mode: where the batch size is greater than one training example but less than the total dataset size. Batch size usually is chosen as a number that can be divided into the total dataset size.

- Stochastic mode: where the batch size is equal to one training example. Therefore the gradient and the neural network parameters are updated after each sample.

Batch size is a slider on the learning process since small values give a learning process that converges quickly at the cost of noise in the training process while large values give a learning process that converges slowly with accurate estimates of the error gradient. A good default for batch size might be 32 which is used in our model.

3.6.4 Number of Epochs

The number of epochs will decide how many times the weights of the network will be changed. When the number of epochs used to train a neural network model is more than necessary, the training model learns patterns that are specific to sample data to a great extent. This makes the model incapable to perform well on a new dataset. This model gives high accuracy on the training set but fails to achieve good accuracy on the test set. In other words, the model suffers from overfitting. When the number of epochs is less than necessary, the training model doesn't learn enough from the dataset which makes the model gives low accuracy on both training and test set and the model suffers from underfitting. So, the number of epochs should be set as high as possible and terminate the training when validation error start increasing as shown in Figure 3-11. The number of epochs is traditionally large, often hundreds or thousands. In the model, for LFW dataset the number of epochs was 100 with a step size of 68 and for VGGFace2 dataset the number of epochs was 1500 with a step size of 100.

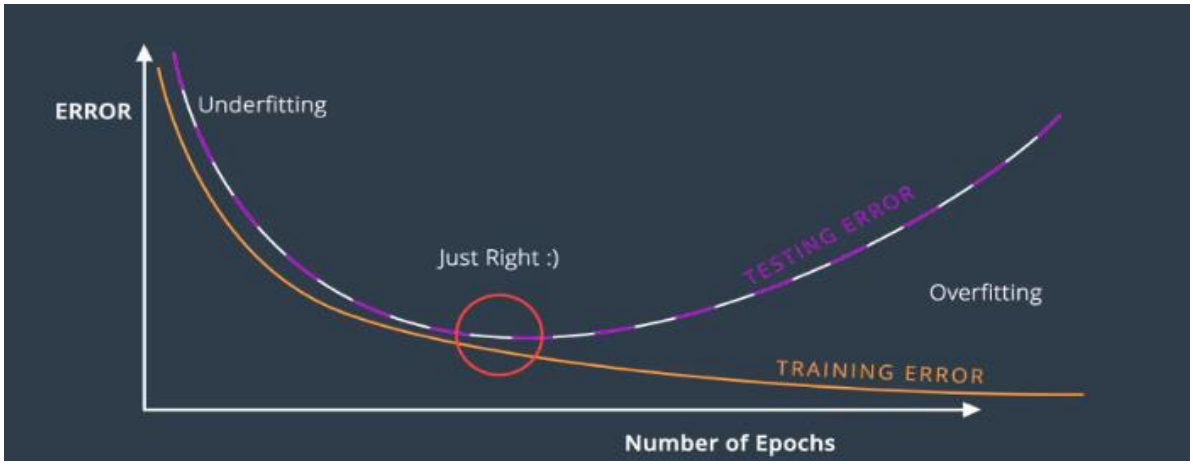


Figure 3-11: Number of Epochs vs. Training and Testing Error

3.7 Liveness Detection

Many companies are now considering biometric face recognition as viable security solution that machine learning engineers can deliver. This innovative technology is showing great promise but also it does have weakness due to the increasing of cybercrime in our increasingly digital world. Paper photographs, screenshots, or 3D facial reconstruction can be easily found and used to spoof facial recognition software, that's why it's important to have anti-spoofing systems in place to reduce theft and mitigate fraud.

In order for face biometrics to fully gain widespread acceptance as a safer form of authentication, it is important to differentiate between a real live face and an attempt to hack the system with an artificial representation of a face, so automated detection of attacks and specifically liveness detection has become necessary for any system based on biometric face recognition. This section discusses the need for liveness, how it works, and approaches.

The most popular face anti-spoofing technique:

Most face spoofing attacks which known as presentation attacks use 2D or 3D (static or dynamic) to hack the system as shown in Table 3-3.

Table 3-3: Types of representation attacks

Type of representation attack	Static	Dynamic
2D	Photographs, flat paper	Screen video or several photographs in a sequence.
3D	3D prints, sculptures, or masks	Sophisticated robots to reproduce expressions, complete with makeup

Nowadays, there is a technological limitation so 2D is more popular than 3D representation attacks and it is important to focus on the techniques that prevent these attacks.

3.7.1 Eye Blink Detection

Blinking detection considers one of liveness detection techniques which is incredibly accurate, so it is very important to implement and use it in this project.

Human blinking can be an easy way to determine if it is a photo attack or not, as it can distinguish between real-life faces and faces on a photo. If the eye at some point is detected open then closed then open, it means that the person has blinked and the program authorize the person as trusted.

Eye Blink Detection can be implemented using two techniques:

3.7.1.1 Deep learning features: Convolutional neural network

In this method blink detection problem is considered as a binary classification problem by training a CNN to recognize which is a closed eye and which is open, after that a function can be built which tries to find a closed-open-closed pattern in the eyes status history to recognize a blink.

This technique is implemented but with unsatisfying accuracy because it needs a large dataset, and the images available for this kind of training is limited, so it was needed to find other method that have good accuracy to use.

3.7.1.2 Eye aspect ratio (EAR)

To build this blink detector, a metric called eye aspect ratio need to be calculated first. In this part eye aspect ratio will be discussed, how it can be used to determine if a person is blinking or not during video frames.

To know what eye aspect ratio means, facial landmarks detection should be applied first to localize the important regions of the face, but in terms of blink detection only two facial structures are important-the eyes [11].

After applying facial landmarks detection and localize the region of the eyes, each eye is found to be represented by 6 (x, y) coordinates.

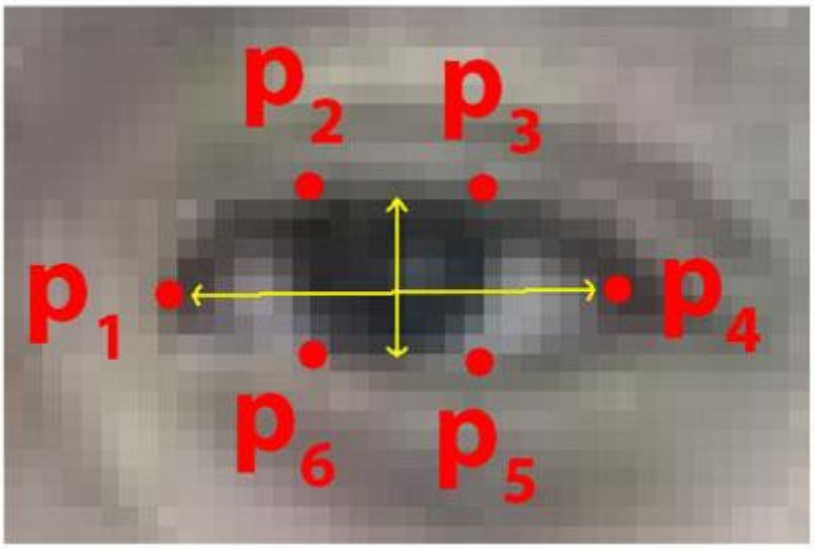


Figure 3-12: Eye Landmarks

Based on Figure 3-12, there is a relation between the width and the height of the eye, this relation is called eye aspect ratio and can be explained as the following equation:

$$EAR = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Where p1, p2…p6 are 2D facial landmarks locations.

The numerator of the equation describes the height of the eye coordinates and computes the length of the vertical axis, while the dominator describes the width and computes the horizontal axis of the eye, weighting the dominator because there is two sets of vertical points but only one set of horizontal points.

Based on this equation, the eye aspect ratio will fall to zero while the eye is closed but it will be approximately constant while the eye is open, so that this equation can show if a blink is taking place, as shown in Figure 3-13.

Using this equation, determining if a person is blinking has become very simple, also it helps us to avoid image processing techniques which have unsatisfying accuracy

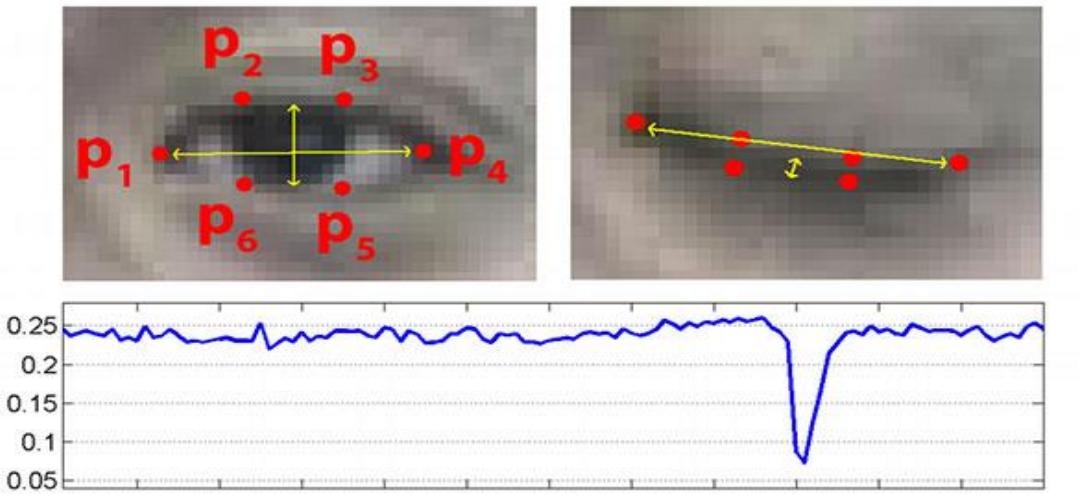


Figure 3-13: Changing eye aspect ratio

Figure 3-13 (left) shows a fully open eye, the eye aspect ratio here would be relatively large and it would be approximately constant over time. However, once the person close his eye, the eye aspect ratio falls to approach zero.

Chapter 4: Simulations and Results

In this chapter, the results of the work are presented and discussed, how the face recognition and blink detection are implemented. First five sections discuss the implementation of face recognition and the results, the last section discusses blink detection. All of these will be discussed as follows:

- Implementation of the pre-trained model on Nvidia Jetson TX2 board.
- How the model works and framework used.
- The preprocessing applied on the datasets.
- Platforms used to run face recognition model.
- Implementation and results of eye blink detection.

4.1 Implementation on Nvidia Jetson TX2 Board

4.1.1 Overview

Jetson TX2 in Figure 4-1, is the fastest, most power-efficient embedded AI computing device. It's built around an NVIDIA Pascal™-family GPU and loaded with 8GB of memory and 59.7GB/s of memory bandwidth. It offers a variety of standard hardware interfaces which make it easy to integrate a wide range of products into it. It is pre-flashed with a Linux development environment. It also supports NVIDIA Jetpack, a complete SDK that includes the BSP, libraries for deep learning, computer vision, GPU computing, multimedia processing, and much more [12].

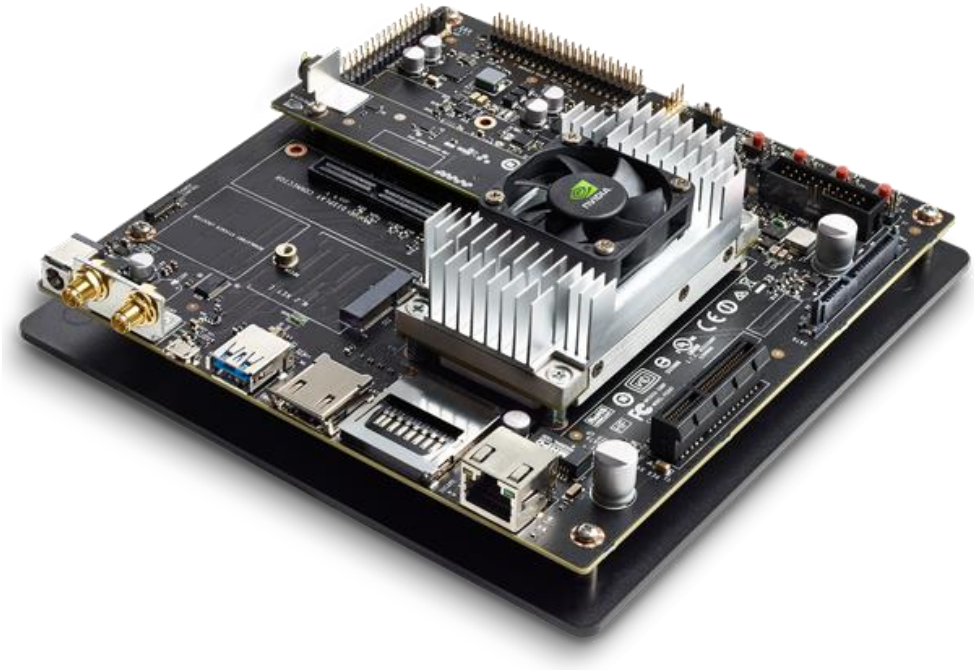


Figure 4-1: Nvidia Jetson TX2 Board

4.1.2 Implementation of the Pre-trained Model

At first, Nvidia Jetson was connected to the required modules as shown in Figure 4-2. Packages of the pre-trained model were installed on Jetson such as dlib and face_recognition libraries. External webcam was connected to Jetson. Then the model was loaded into Jetson and started running. Jetson received frames from webcam to pass them to the model, which in turn processed these frames to recognize them showing results on screen as in Figure 4-3 with accuracy of 96%.

The reason why an external webcam was used although Jetson has an internal camera that while using this camera, the processing time was about 15 seconds which caused a large delay and non-real time operation. On the other hand, the external webcam results a duration time before capturing of 5 seconds while processing time was about 0.5 second per frame.

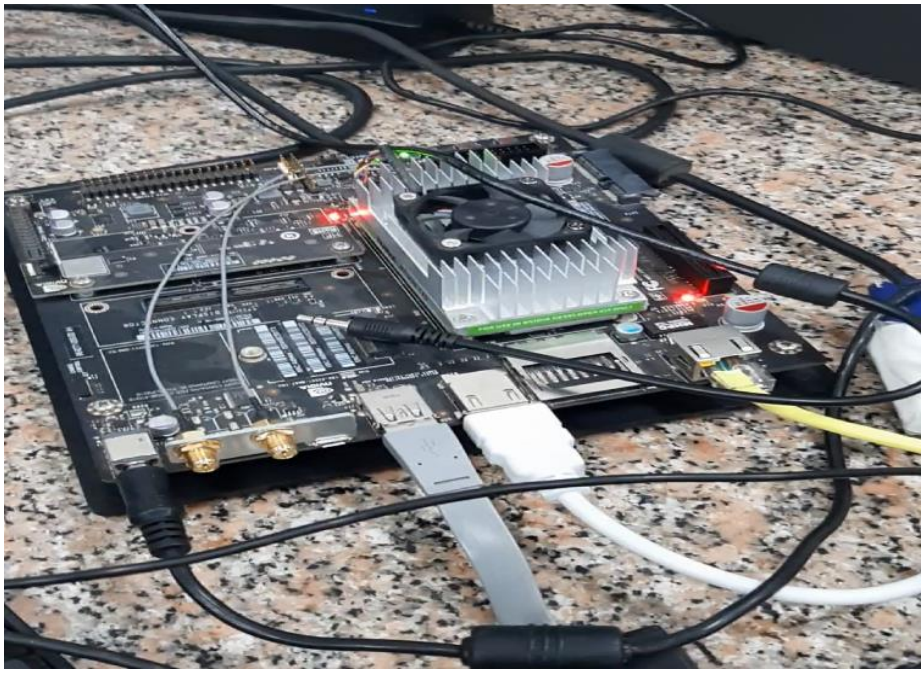


Figure 4-2: Nvidia Jetson Connections

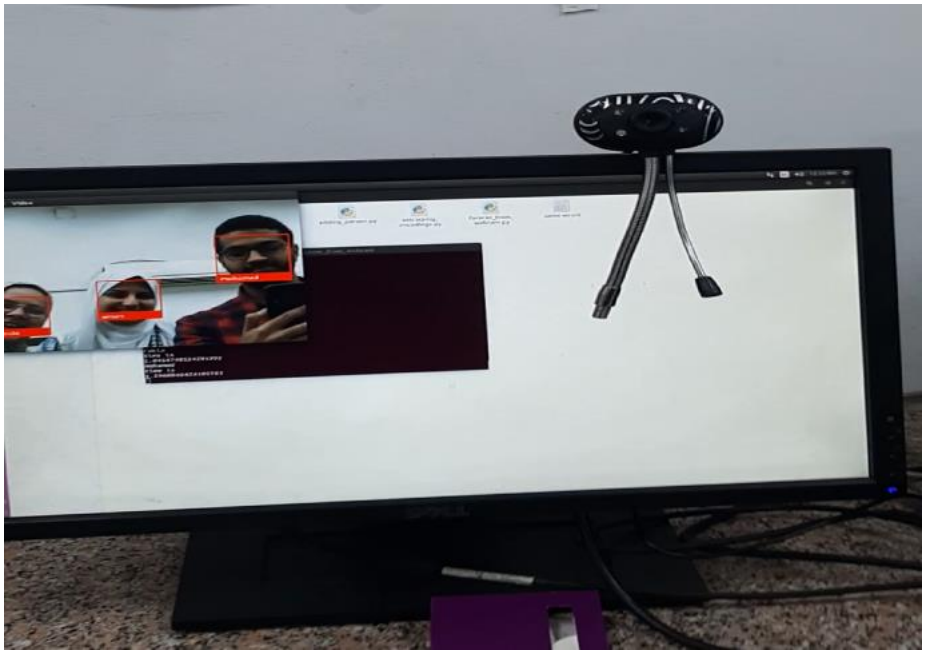


Figure 4-3: Results of the Pre-trained Model

4.2 Model and Architecture

4.2.1 Model

As mentioned before, this model is built using Siamese Network by entering two inputs in the same architecture and then comparing the output of each one, so that:

- Output is 1 if the inputs are for the same person.
- Output is 0 if the inputs are for different persons.

The model was written using:

- Keras framework.
- Model class which helps in using multiple inputs in the same network and then combining the outputs.

How the model works:

1. A base architecture is built, after that two inputs which are referred to as a pair of images are entered to the same base architecture.
2. Each branch outputs the embedding.
3. The two embeddings are combined together and their difference is obtained by absolute difference or by equilidean distance.

Equilidean distance d(p,q) $= \sqrt{(p1 - q1)^2 + (p2 - q2)^2 + \cdots + (pn - qn)^2}$ (1)

Absolute difference d(p,q)= $\sum_{k=1}^{n}(p_k - q_k)$ for embedding of size n. (2)

4. The output of the difference equation is passed to a sigmoid function to output 1 for the same person, 0 for different persons.

4.2.2 Architecture

For the base architecture, more than one architecture have been tested:

4.2.2.1 ResNet-34

We have built ResNet-34 architecture by ourselves, the following points will show how we have built it:

- This architecture has been built using sequential class.
- There are two main blocks that were built for ResNet-34:
    1) Convolutional block:
        - Convolutional block is a stack of convolutional layers for the same number of filters followed by batch normalization layers with activation function ReLU.
        - Shortcut is performed to the input of the first layer to be added to the output of last layer in the block to overcome gradient descent vanishing problem.
        - Output size of this block is same as its input size.
    2) Identity block
        - Identity block is same as convolutional block but in case the output size is different from input size.
        - If the size of output of last layer is not the same size of the input to first layer, a convolution layer is added to the shortcut to manage the input to be added to the output.
- The output of the last block is passed by a fully connected layer to get embedding.

4.2.2.2 ResNet-50

- It is a pre-trained architecture from keras over ImageNet dataset.
- There are two options for using this architecture:
    1) Using the weights directly.
    2) Training the architecture using transfer learning.
- We trained the model on our dataset by transfer learning using the same weights but different fully connected layer to achieve our objective.

4.3 Datasets and Preprocessing

4.3.1 LFW (Labeled Faces in the Wild)

It consists of 2200 pairs of images for training (1100 same identity and 1100 different identities) and a vector of 2200 labels (1 for same identity pair and 0 for different identities) and 1000 pair for testing (500 same identity and 500 different identities) and a vector of 1000 labels (1 for same identity pair and 0 for different identities). The images are cleared and contains only the face and shoulders. The dataset has no images of very low resolution or wrong labels.

For the preprocessing images, resizing has been performed on the dataset to have the intended size of the input of the model. The images were prepared to be fed to the model by collecting them in a list of two NumPy arrays (as two inputs) and a NumPy array of ones and zeros (as the labels). The indices have been shuffled (same shuffle for the two inputs and the labels arrays' indices) in order to train the model correctly

The model was trained on LFW dataset using Keras framework by method "fit" and the accuracy was 99% for the training and 70% for the validation which clearly shows that the model has overfitting issue which encourages us to search for a larger dataset to overcome this problem.

Note that full LFW dataset can be downloaded as directories, each directory contains person's images but will need to form them in pairs and label them.

4.3.2 VGGFace2

It consists of more than 3 million images of more than 8000 identity as a training data and about 250k images of a 500 identity as a testing data.

The data in the form of directories, each directory refers to an identity and contains the images of this person. The images are in different sizes and can have more than one face in the image for the same person or for other persons in the background

For the preprocessing; first, face detection has been performed on the whole dataset to crop the images around the faces only using the mentioned pre-trained model in chapter 3. Then, number of wrong label data was very small relative to the number of images for the whole dataset and the face detection model finished the job of removing the very low-resolution images as it couldn't detect the faces in it.

Then, a function has been created to make pairs from all combinations of the images for the same identity's pairs and it randomly creates different identities' pairs of the same number of the same identity's pairs to make the data balanced. All the data is in the form of list of two NumPy arrays as two inputs and one array has ones and zeros for the labels as 1 refers to same identity and 0 refers to different identities but unfortunately the number of pairs was very huge and it wasn't reliable. So, this function can help in case of small data as it won't waste any images.

Then, a generator function was created to make a batch of pairs randomly while training. By this way, it won't need very large memory as the batch is divided into 2 halves; one for the same identity's pairs and the other the other for different identities' pairs. The pairs were in the form of list of two NumPy arrays that act as two inputs arrays and one array has ones and zeros for the labels as 1 refers to same identity and 0 refers to different identities This function doesn't return and runs a while true loop and it yields a batch of pairs. Both of the two functions shuffle the indices before passing the arrays of the pairs and the labels.

Then, a function has been created to separate the data into parts and train on each part in case of small memory by passing the model, the parameters and the dataset path. The function separates the paths randomly to collections of paths then it starts to train the model on each part separately by calling the method "fit_generator" from Keras.

Finally, the method fit_generator accesses the generator function which in turn will give it the batch of training and validation samples to train and validate the model.

Unfortunately, an issue has been faced that in this case the models can't learn and its training and validation accuracy stay at 50% and many solutions have been tried without any results and the last solution to be tried in the future work is to train the model using a small general subset of the data (about 20k images) to figure out what will happen and if this also doesn't result in different results, another solution will be tried which is augmentation to LFW dataset in order to increase the dataset samples.

4.4 Software and Hardware Platforms

4.4.1 Google Colaboratory

It is one of the most famous free cloud services to help spread machine learning education and research. It is a Jupyter notebook environment that doesn't require any type of setup in order to use and runs completely in the cloud. It also supports many popular machine learning libraries such as Keras, Tensorflow and OpenCV.

As a programmer, the following can be performed using Google Colab.

- Document your code that supports mathematical equations
- Write and execute code in Python
- Import/Save notebooks from/to Google Drive
- Free Cloud service with free GPU
- Create/Upload/Share notebooks
- Import external datasets
- Import/Publish notebooks from GitHub

Thus for these reasons, it was used for training and testing the model but there were some issues that affected training the model such as:

- Runtime restarting
- Limited resources (GPU/RAM/Disk)

4.4.2 Graphics Processing Unit (GPU)

It is a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device. It performs parallel operations. Although it is used for 2D data as well as for zooming and panning the screen, a GPU is essential for smooth decoding and rendering of 3D animations and video. For the issues that were mentioned before in google colab, a fast GPU was needed to speed up the training. So it was acquired from ONE Lab to use Nvidia GeForce RTX 2080 Ti that has high performance represented in:

- Interface: PCI Express x16 3.0
- Cores: 4352 Units
- Core Clocks: Boost: 1755 MHz
- Memory Size: 11 GB

- Memory Speed: 14 Gbps

GPU was connected to CPU to provide interfacing using operating system, and because of that, there was a struggle that the code runs on CPU instead of GPU which obstructed training the model but this issue was solved by some changes in code.

4.5 Eye Blink Detection

As mentioned in the previous chapter, eye aspect ratio method is one of the methods used to detect blinking which is incredibly accurate and fast.

In this part, a Python, OpenCV, and dlib code were written to:

- Perform facial landmark detection.
- Detect blinks in video streams.

This code defines two constants:

- First constant for eye aspect ratio threshold which is the maximum value needed to indicate blink, this value is set to 0.3.
- Second one to indicate number of successive frames with an eye aspect ratio less than EAR threshold that must happen in order for a blink to be registered, this value is set to 3.

The total number of blinks is computed by the code by counting how many times three consecutive frames with closed eyes are occurred.

By comparing the number of actual blinks that can be noticed in multiple videos and the total number of blinks that resulted from the code, we concluded that the accuracy of this code is approximately equal to 100%.

Chapter 5: Background and Related Work

5.1 History

At the beginning of Electronification of automobiles and due to the increase of customer wished features for modern vehicles, automotive manufacturers recognized that the coordination between car ECUs could enhance the functionality of vehicles greatly. This communication was first performed by a dedicated physical connection for each signal that requires transmission, which increased wiring effort and limited data exchange. The introduction of controller area network (CAN) was the solution out of this dilemma. Figure 5-1 shows a visual comparison between the old wiring scheme and the CAN bus network used in modern vehicles.

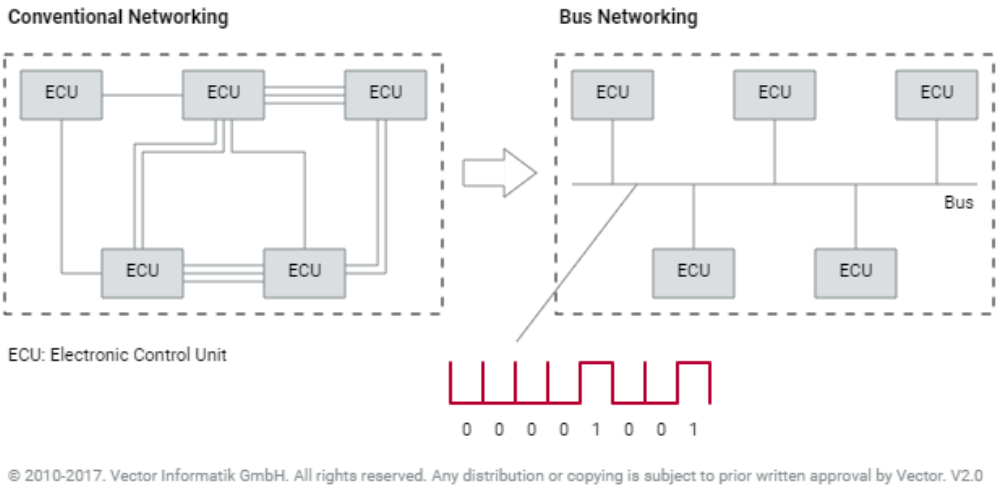


Figure 5-1: CAN network vs. conventional network

5.2 CAN Properties and Details

Controller Area Network (CAN) is a serial multi-master event driven network that allows one sender at a time. CAN is based on broadcasting, while using sender ID to perform arbitration and priority schemes. Data transmission rate is up to 1Mbit/s. It comes in two data rates: low-speed CAN which supports data rate up to 125Kbit/s and high-speed CAN which supports data rate up to 1Mbit/s.

CAN frames exist in three types: data frame, remote frame and error frame. CAN Data frame is used to transmit user data. It can hold a maximum payload of 8 bytes in the data field. The data field is then framed by other fields important for the operation of the CAN protocol. Those fields include message identifier (ID), data length code (DLC), cyclic redundancy check sequence (CRC sequence), end of frame (EOF) and RX acknowledgment field. The standard CAN frame is depicted in Figure 5-2.

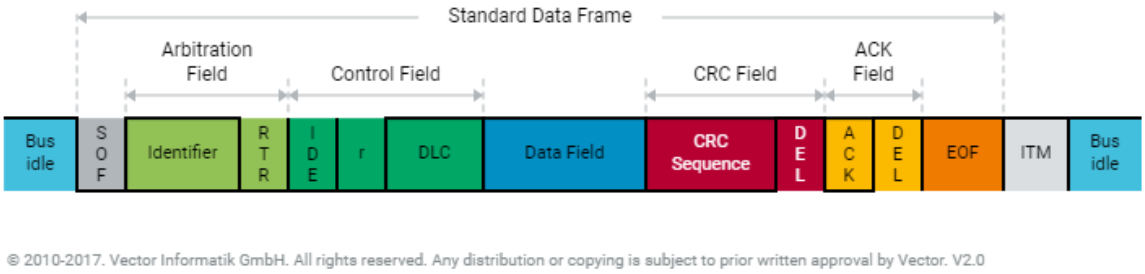


Figure 5-2: Standard CAN data frame

CAN remote frame (shown in Figure 5-3) have the same structure as the data frame except it misses the data field. The remote frame is sent to request user data from another ECU. The third type of CAN frames is CAN error frame (depicted in Figure 5-4). It is sent to indicate an error during communication. It consists of error flag and error delimiter.

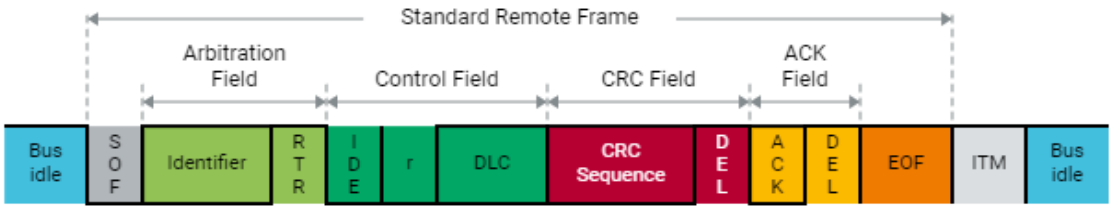


Figure 5-3: Standard CAN remote frame

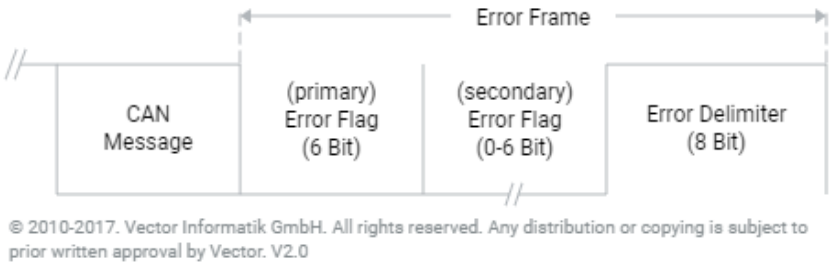


Figure 5-4: CAN error frame

Communication in CAN network is based on content-related addressing. Instead of giving IDs to the communicating nodes, the data and remote frames are given the IDs. The broadcasting nature enables a message to be visible to all connected nodes, but each receiver is independently responsible for selecting the messages to receive by the internal acceptance filter. This filtering process is based on the frame ID.

The ID of a CAN frame can exist in two formats based on two modes of operation: standard ID mode in which the ID field consists of 11 bits, and extended ID mode in which the ID field consists of the standard 11 bits ID field as the base ID, and an additional field of 18 bits called the extended ID. The difference between standard ID field and extended ID field is illustrated in Figure 5-5.The extended ID mode is mainly for the purpose of future expanding. The number of messages is expected to increase, which corresponds to more message IDs needed.

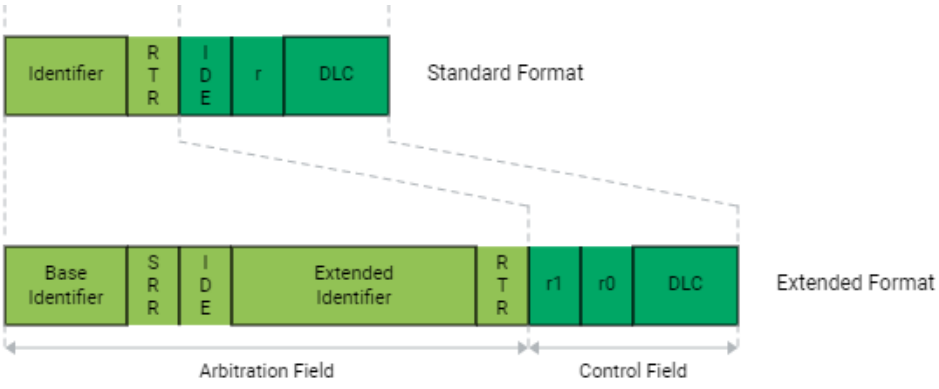


Figure 5-5: Standard ID vs. Extended ID

The ID field is also responsible for arbitration and prioritization. When more than one entity tries to access the bus simultaneously logical “0” is the dominant over logical “1”; which means an entity writing 0 at a time will overwrite the other if it was writing 1. Based on the previous rule, bit wise arbitration occurs and the frame with lower ID will dominate the bus and the other sender has to wait. As a consequence, the priority scheme for the CAN protocol gives higher priority to messages with lower ID value [13].

CAN bus provides high reliability and speed for real time applications. In addition it is a lightweight and robust protocol. The decentralization of operation is also one of CAN bus advantages, which guarantees network availability and prevents single point of failure. Those aforementioned properties are of great importance while dealing with safety critical applications. On the other hand CAN bus suffers from a wide attack surface. In the next section the CAN bus vulnerabilities will be discussed.

5.3 CAN Bus Vulnerabilities

As mentioned before CAN bus was not mainly designed to stand against attacks. It was designed when CAN attack threats were not of a great possibility. Nowadays, the new technological trends in automotive domain dictate that CAN bus endurance against attacks is now a must. Security analysis of the CAN network have been performed to define the probable attacks and vulnerabilities as a first step in the way to secure CAN [14].

From security point of view a secure communication should meet those five criteria:

- **Data Integrity:** this mean that the data is received exactly as it was sent without being altered in the communication channel.

- **Authentication:** all entities participating in the communication should be detected authentic or trusted.

- **Confidentiality:** the data transmitted on the communication channel should be secure against intruders who try to overhear it

- **Availability:** to assure that the system is available for communication throughout different circumstances.

- **Nonrepudiation:** the security solution should prove that the parties in the communication cannot deny the authenticity of the message that was organized.

Applying those criteria to CAN-bus the main drawbacks are Lack of authentication and data confidentiality. Sending bare data on the broadcasting communication channel opens the door in front of sniffing attacks in which an attacker can listen to the messages, record them and define the different content and relate it to different vehicle functionalities. Now the attacker knows exactly what message to send if a certain action is wished. This side by side with the fact that the CAN protocol don't naturally concern with sender identification, instead it uses message identification. An intruder can broadcast a spoofed message on the bus which the other nodes can hear and act upon. Moreover, replay attack can be launched in which the attacker sends a recorded message repeatedly to force a certain action in the car. One of the other critical attacks in CAN network is Denial of service attack (DOS) this attack addresses the CAN arbitration and priority scheme. The attacker can send a malicious message with high priority ID repeatedly in order to occupy the bus. This can result in a system failure. Table 5-1 contains the main attacks in the CAN threat model and the system vulnerability leading to it.

Table 5-1: CAN bus vulnerabilities and resulting attacks

Attack	**System vulnerability**
Sniffing attack	Lack of confidentiality
Spoofed messaged injection	Lack of authentication
Replay attack	No message freshness guarantee
Denial of service attack	The arbitration and priority scheme

5.4 Related work

Securing in vehicle CAN communication against attacks has gained strong interest over the last few years. Many researches were conducted to find reasonable ways to secure communications over the CAN network without losing the main advantages of this communication protocol. This matter was found to have many tradeoffs between maintaining the reliability and real time constrains of the unsecured CAN network which we cannot compromise in automotive domain, and achieving high levels of security at the same time. This chapter will discuss some of the related work introduced in this topic.

5.4.1 VeCURE

VeCURE was one of the proposed methods to apply CAN bus security concepts [15]. This method assumes that the main adversary goal is to inject spoofed messages and let the receiver believe it is from a legitimate ECU; to fool the car to perform a certain action based on the spoofed message, or to start a replay attack –DOS attack is not covered in this method-. All the work is under the assumption that the sources of attack are compromised ECU or through the OBD-II port.

The suggested method VeCURE categorizes the connected ECUs to Low trust group and high trust group based on what they are connected to. If an ECU is a suspicious node based on the assumed sources of attacks, it is in the low trust group and the rest are in the high trust group. The high trust group nodes share a secret key K_h which is not known to the low trust group. This secret key is used to generate message authentication code for each message between the high trust group members, which allow them to communicate securely to each other. The communication pattern allowed by the trust group based authentication is shown in Figure 5-6. To prevent replay attacks the message authentication code generated is a function of a counter called message counter. Message counter is a counter defined for each message and is updated at sender and receiver each successful transmission of this message; to guarantee message freshness.

To overcome the issue of message authentication code calculation delay pre-calculation of the heavy computational part is performed, so it is ready even before it is needed. The idea of using shared key introduced is also useful when it comes to the cost of key distribution, while it is not as secure as using private keys. This method uses an extra frame with each message frame for authentication, which is considered a large added overhead that lowers the overall performance. Another disadvantage is that this method doesn't address overhearing and data confidentiality.

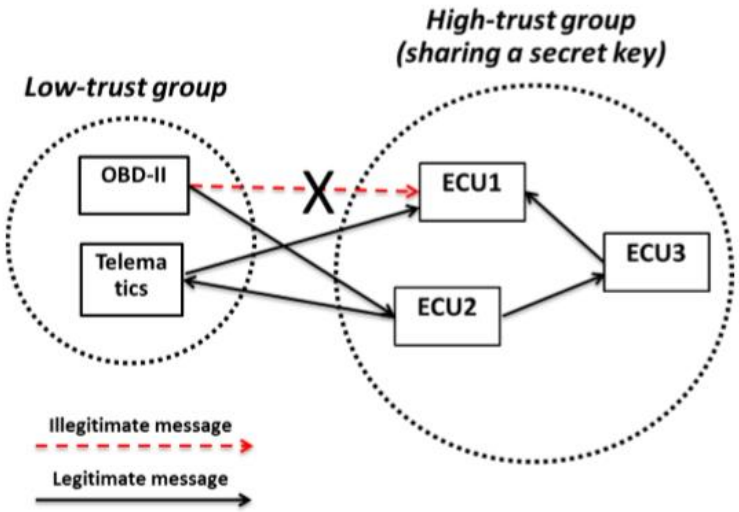


Figure 5-6: Communication between low trust group and high trust group in VeCURE

5.4.2 CaCAN

Centralized authentication system for CAN – CaCAN – is one of the methods proposed for the purpose of CAN security [16]. This method uses an extra node which they name HMAC-CAN controller. The HMAC-CAN controller monitors the bus to guarantee transmission of authentic messages only and destroy unauthentic messages. This method is similar to VeCURE in the idea of using message authentication code (MAC). It also uses message counters to guarantee messages freshness. The MAC – part of HMAC and part of the message counter- are transmitted with the message payload. The frame fields for this method are depicted in Figure 5-7. Although the overhead here is better than in case of sending a separate authentication frame, but the fact that part of the payload must be freed means that the usual CAN messages with 8 bytes payload should be modified or separated into two frames. This method did not consider message confidentiality, and adding additional node is not advisable when considering the backward compatibility with systems that already exist.

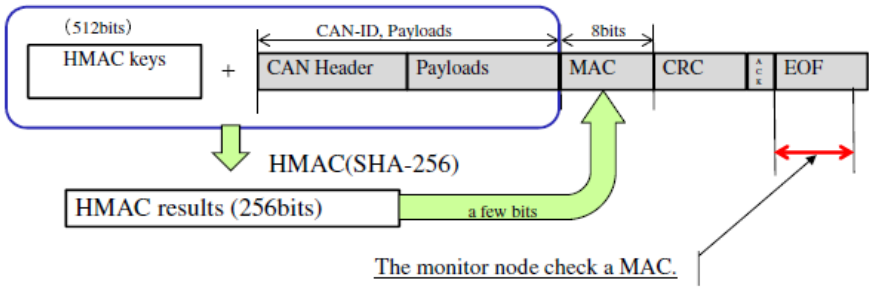


Figure 5-7: CaCAN frame

5.4.3 VuLCAN

The VuLCAN method uses secure computing concepts [17]. It uses Sancus which is a light-weight and open source for trusted computing for IOT. Sancus forces that private data section can only be accessed by its corresponding code section, and it employs a three level key hierarchy using an embedded cryptographic core. It also forces that Critical application software is included in the protection domain so that it is anti-tampering.

This method guarantees authenticity by a separate frame sent before the data frame, this frame includes 64-bits message authentication code. The message authentication code is constructed using ID, payload and anti-replay counters. Data confidentiality is considered and data is protected using encryption with 128-bits symmetric key. While the secure computation concepts lead to more secure system, but this adds complexity to the system. The complexity will in turn lead to more computation and delays. This method is also obviously far from backward compatibility.

5.4.4 CANAuth

CANAuth method suggests using CAN+ standards [[18]. CAN+ concept (illustrated in Figure 5-8) is to use the duration between sampling instances for each transmitted bit to transmit excess bits –usually authentication bits- with higher transmission rate. Message authentication code is transmitted over CAN+ achieving authentication with zero additional overhead compared to original unsecure CAN. CANAuth also consider message confidentiality by using encryption. Using CAN+ solves some of the strict tradeoffs in the issue of CAN security, but this protocol is still not used and will require modification in the CAN controller.

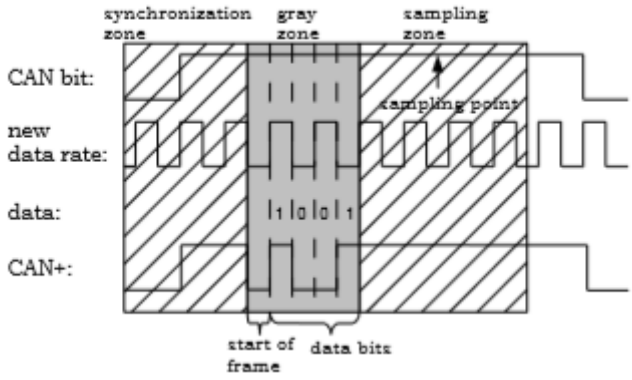


Figure 5-8: Transmission of data bit for normal CAN and CAN+ protocols

5.4.5 WooAuth

WooAuth [19] guaranteed authenticity by using message authentication code (MAC), and data confidentiality by using encryption. To guarantee message freshness message counters are used. The MAC Is sent in the extended ID field and the CRC field, which is an acceptable overhead. The MAC generated in a way to guarantee the integrity of data and can replace the CRC in data validity check.

This method uses long term symmetric key distributed between ECUs to calculate system variables e.g. message authentication codes and encryption keys. The distribution of those long term keys happens once at the start of each driving session. This process happens when ECUs take turns to communicate to a gateway node and share a random number with it, calculate MAC and key based on the shared numbr, and then verify that the MAC and key were identical after calculations at both the gateway and the ECU it perform key distribution with. This scheme guarantees that no errors happen in the initial key distribution phase which will mean failure in the communication between ECUs while the driving session.

Towards more security they perform periodic refreshment for the encryption keys during the driving session, but the process of key refreshment actually stops the communication on the bus for the period of refreshment protocol, which may not be the best practice when it comes to the strict real time requirements of automotive CAN bus. Although WooAuth is considered one of the most successful CAN security introduced methods with high security level, it lacks backward compatibility because of the added gateway node.

Eventually a comparison between the most popular proposed CAN security solutions was held. Table 5-2 shows the system evaluation for introduced encryption methods [20]. Table 5-3 shows the system evaluation for introduced authentication schemes and the resulting security level [21].

WooAuth has shown the best results the only drawback was that it is incompatible with existing CAN network due to the added gateway node. Adding more hardware needs changes in the in vehicle network.

Table 5-2: Evaluation of cryptography solutions according to identified requirements

Solution	Authentication	Integrity	Confidentiality	Backward compatible	Replay attack resistance	Real-time requirements
LiBrA-CAN[13]	✓	✓	X	X	X	X
WooAuth [14]	✓	✓	✓	X	✓	✓
Vecure[15]	✓	✓	X	✓	✓	-
CaCAN [16]	✓	✓	X	X	✓	-
VatiCAN [18]	✓	✓	X	✓	✓	X
VulCAN [19]	✓	✓	X	✓	✓	-

Table 5-3: Evaluation of authentication solutions according to identified requirements

Message Authentication Solution	IR 1 Cost Effectiveness	IR 2 Backward Compatibility	IR 3 Repair and Maintenance	IR 4 Implementation Details	IR 5 Acceptable Overhead	Approx. Security Level[1]
CANAuth [7]	✗	✗	✗	✗	✗	Strong
SchweppeAuth [8]	✗	✗	✓	✓	✗	Strong
LiBrA-CAN [9]	✓	✗	✓	✓	✗	Strong
LinAuth [10]	?	?	?	✗	?	Medium
MaCAN [11]	✓	✓	✓	✗	✗	Medium
CaCAN [12]	✓	✗	✓	✓	?	Weak
VeCure [13]	?	✓	?	✓	?	Medium
WooAuth [14]	✓	✗	✓	✓	✓	Medium
VatiCAN [15]	✓	✓	✓	✓	?	Medium
WeisglassAuth [16]	✓	✗	✗	✓	?	Medium

Chapter 6: System Design

Our proposed system aims to fill the gaps in CAN bus security; in order to have more secure and safe in-vehicle communication system that is ready to be connected to the outer world through WIFI, GSM, Bluetooth, etc…

The proposed system collects the most successful aspects in the previously mentioned and evaluated methods. WooAuth is the start point of the introduced solution. Mainly, some of WooAuth concepts are adopted with more simplification, and with the avoidance of the extra added node.

Starting from the pre-discussed threat model for CAN bus, the proposed system is based on some assumptions:

Assumptions:

- Initially stored secrets are stored in anti-tampering memory and cannot be read or change by intruder.
- Tampering the code running on the ECUs is not easily possible
- Threats come from an external node attached to the bus or through OBD connected device-which may have access to WIFI, GPS, GSM or Bluetooth

The main countermeasures concluded from the previous work are authentication, encryption and anti-replay counters. Those three measures stand against most of the vital attacks that threaten CAN bus security. In our proposed system SHA-1 hashing algorithm is used to generate authentication codes, encryption keys. AES algorithm is used for encryption and decryption of data. In the next section the discussion of SHA-1 algorithm and AES algorithm is held. The reasons behind choosing them and some introduced modifications are also discussed.

6.1 SHA-1

6.1.1 Overview

Secure Hash Algorithm -1 (SHA-1) [22] was proposed by U.S. National Institute for Standards and technology (NIST). Hashing functions are mainly used to map variable size input data to a fixed size code. In SHA-1 this operation is based on performing many bit-wise operations using the input data to alter a number of initially defined variables (five variables, 32-bits each) called chaining variables. The values of the chaining variables after running the hash are the hashing result. The input to SHA-1 is usually blocks of data with large size and the output is 160 bits dictated by the defined standard. The operations from input to output are standardized by four stages.

To measure the quality of any hashing function some properties were defined:

- **Preimage resistance:**
  Means that for an output y, it is computationally infeasible to find any input x_o that hashes to y (i.e. x_o such that h(x_o)=y is not feasible to calculate).

- **2nd-preimage resistance:**
  Means that it is computationally infeasible to find any second input x_1 that have the same hash value y as x_o (i.e. x_1 to achieve that h(x_1) = h(x_o) is not feasible to calculate)

- **Collision resistance:**
  Means that it is computationally infeasible to find any two distinct inputs x, x_o which hash to the same output, (i.e. such that h(x) = h(x_o)).

The SHA-1 algorithm is considered good when it comes to the pre-mentioned requirements for a hash function. It is not considered the strongest when we talk about the data hashing applications for example. There are more sophisticated algorithms for data hashing -and other hashing applications- which is based on more complicated operations and in turn needs more computational resources.

In our system SHA-1 is not used for message -data- hashing. It may be said that the hashing function is used here as random code generator. This decision was based on two properties of SHA-1:

- SHA-1 is a one way hashing function i.e. the adversary cannot know the input given the output –which is important for the inputs of the hash function to remain secret -.
- SHA-1 output changes significantly with a bit or a few bits change in the input.

Starting from this different utilization for SHA-1 function, the properties which gain the most concern in our system are preimage and 2^{nd}-preimage resistance; because the output of the hashing function -which is used to define the secrets that the system members share - is needed to be impossible to guess by any intruder. This is also defended by the fact that SHA-1 is a one way function. The collision possibility is not a great concern for our system.

6.1.2 Advantages (why chosen)

Choosing between the different hash functions the main concern was the speed; because as mentioned before for our system some hashing functions properties can be compromised, but we can never compromise the speed of the algorithm under the limited resources of embedded ECUs in automotive real time applications.

Comparing SHA-1 to other algorithms of the same family, it appears in Table 6-1 that SHA-1 is better –most importantly - in speed. Also it is better in collision and preimage resistance as shown in Table 6-2.

Table 6-1: Summary of selected hash functions based on MD4

Hash function	Relative speed
MD4	1.00
MD5	0.68
SHA-1	0.28

Table 6-2: upper bounds on strength of selected hash functions

Hash function	Preimage	Collision
MDC-2	$2.\ 2^{82}$	$2.\ 2^{54}$
MDC-4	2^{109}	$4.\ 2^{54}$
Merkle	2^{112}	2^{56}
MD4	2^{128}	2^{64}
MD5	2^{128}	2^{64}
SHA-1	2^{160}	2^{80}

6.1.3 Reduction

SHA-1 is mainly designed to deal with large input blocks of data. In our case we use the hashing function to serve as random number generator to generate unpredictable codes, so there is no data to perform hashing on. Instead, the available variables in the system will be given to the hashing algorithm as an input. As a result a size reduction in the input to SHA-1 algorithm was necessary.

The size reduction for hashing function in most of the traditional applications of hashing will lead to severe impact on the definition of the hashing function and will weaken its strength. While here the function is used just to generate random codes, so the size reduction will not be very harmful and it will also lead to faster computation which is desired whenever real time applications are mentioned.

6.2 AES

6.2.1 Overview

Advanced Encryption Standard (AES) was announced by the National Institute of Standards and Technology (NIST) in November 2001 [23]. First there was the Data Encryption Standard (DES) which was published in the 1970s. The DES algorithm has a symmetric-key encryption/decryption with a key of 56 bits length, which was the main disadvantage of the DES as it makes it insecure and vulnerable. Electronic

Frontier Foundation in collaboration with distributed.net managed to publicly break a DES key in 22 hours and 15 minutes due to its short key length in addition to weaknesses in cypher. DES has been withdrawn as a standard by NIST. To determine which algorithm would follow DES, NIST called for different algorithm proposals as a competition and the best of all was the new AES or Rijndael (named after its inventor) that won because of its security, ease of implementation and low memory requirements. AES also has a symmetric-key encryption/decryption which means that the same key is used in either encryption or decryption of the digital data. AES has three versions with different key lengths: 128, 192 and 256 bits, all have a fixed block of data with 128 bits length as an input and an output. We will consider the 128 bits key length in our work.

6.2.2 How the algorithm works [24]:

1- **AES algorithm** (in Figure 6-1**)**

It consists of 10 rounds of encryption (12 rounds for 192-bits key and 14 rounds for 256-bits key). The 128 bit key is expanded to 11 versions called round keys with 128 bits length each. Each round includes transformation using the corresponding round key to ensure the security.

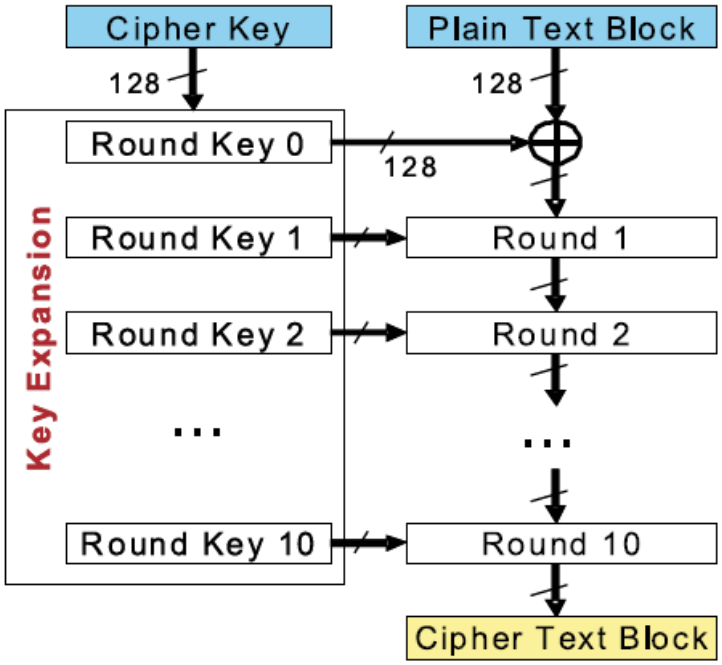


Figure 6-1: Basic concept of AES algorithm

Starting with an initial round during which the first round key is XORed with the plain text, nine equal rounds follow. Each round will be consisting of the following operations:

- Substitute bytes
- Shift rows
- Mix columns
- Add round key

The tenth round is similar to rounds one to nine, but the Mix columns step is omitted. These four operations are explained in details in the following sections but first the structured key and input data phase should be illustrated during which the data (state) and the key are structured in 4x4 matrix of bytes as follows in Figure 6-2:

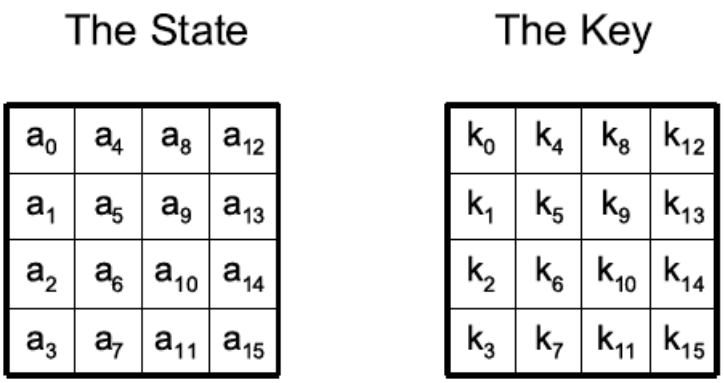


Figure 6-2: Structure of data and key

2- Substitute bytes:

One of the major reasons of the security of the AES is that byte substitution is not linear (illustrated in Figure 6-3). This operation is considered with substituting the data bytes with the corresponding values from a table named the substitution box (SBox).

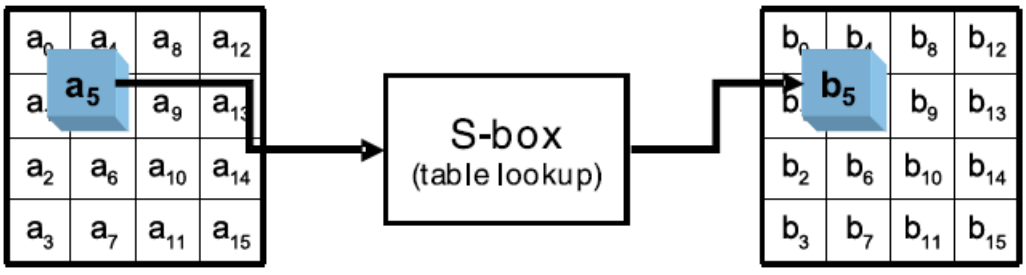


Figure 6-3: Substitute bytes operation

3- Shift rows:

Processes for shifting the different rows of the 4x4 matrix constructed (Figure 6-4). The first row is left unchanged, the second is shifted one byte to the left, the third is shifted two bytes to the left and the fourth is shifted 3 bytes to the left.

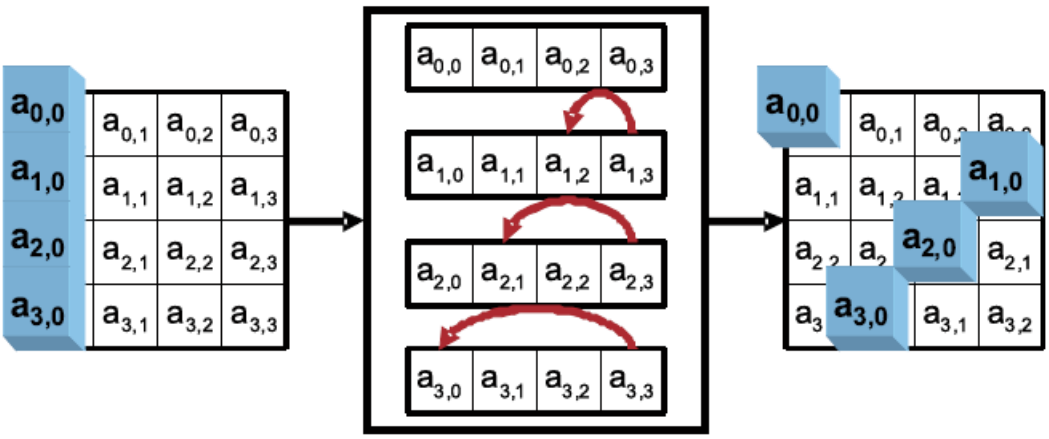


Figure 6-4: Shift rows operation

4- Mix columns:

Opposed to the shift rows operation, now working on the columns of the matrix is considered (Figure 6-5). This maybe the most complex operation for software implementation. As a principle only matrix multiplication is needed. But in order to make this operation reversible, the normal addition and multiplication is not used. Instead Galois field operations are used. Without going into mathematical details, the most important part to know about Galois field is that the addition corresponds to XOR operation and the multiplication corresponds to more complex equivalent.

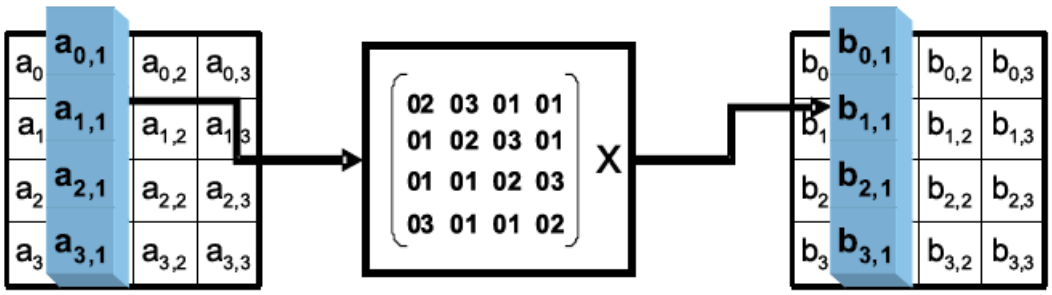


Figure 6-5: Mix columns operation

5- **Add round key:**

A simple process (in Figure 6-6) considering XORing each byte of the data with the corresponding from the expanded round key.

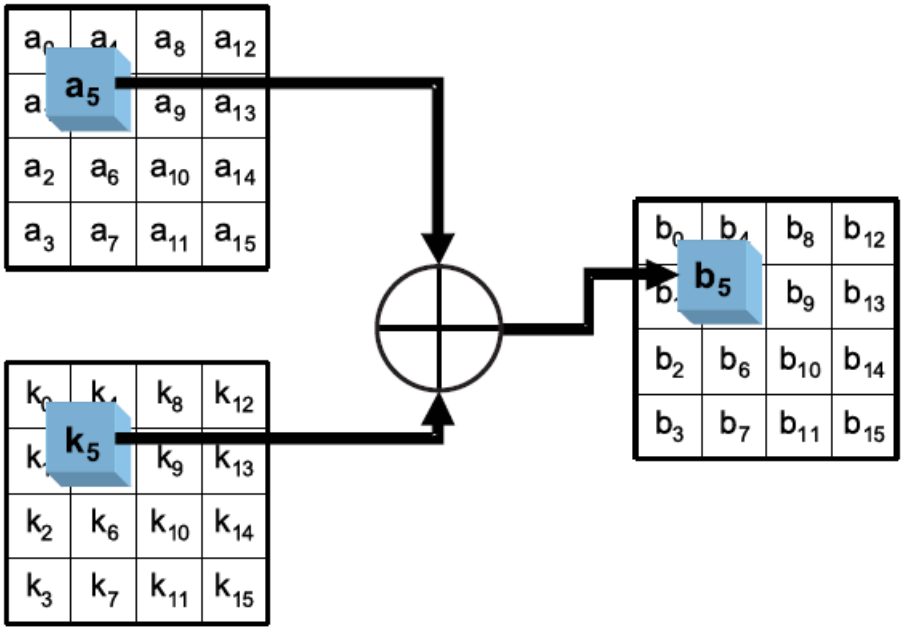


Figure 6-6: Add round key operation

6.2.3 Key expansion process:

As mentioned before the 128-bits original key is expanded into eleven 128-bits round keys. To compute key (n+1) from key (n) two steps are performed:

1- Compute the new first column of the next round key (Figure 6-7): first all the bytes from the old fourth column are substituted by the bytes substitution process. Then they are shifted vertically by one byte and XORed to the old first column and that results in generating the first new column.

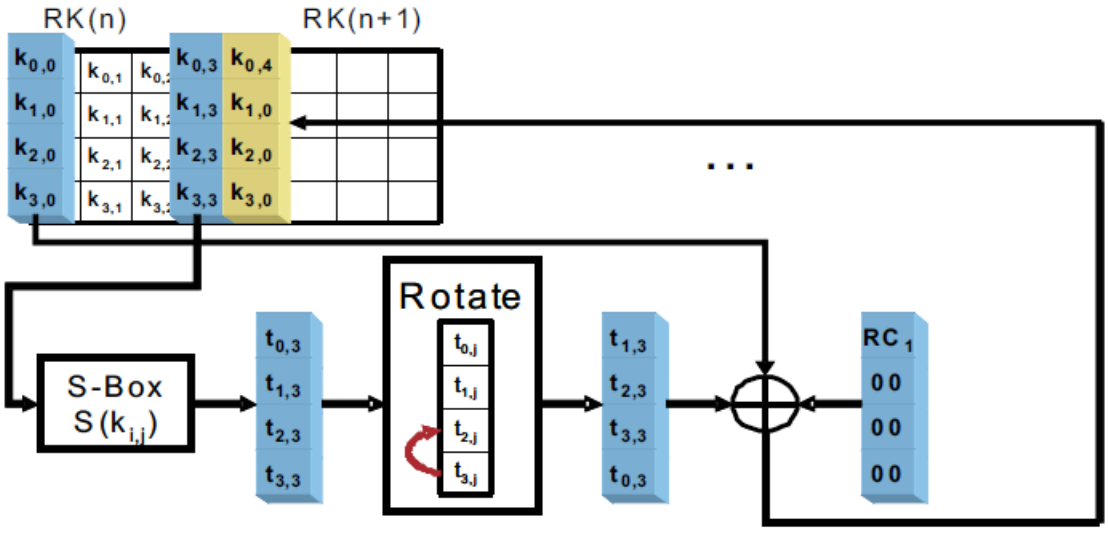


Figure 6-7: First column generation

2- New column 2 to 4 are calculated as:

- [new second column] = [new first column] XOR [old second column]
- [new third column] = [new second column] XOR [old third column]
- [new fourth column] = [new third column] XOR [old fourth column]

This process is shown in Figure 6-8.

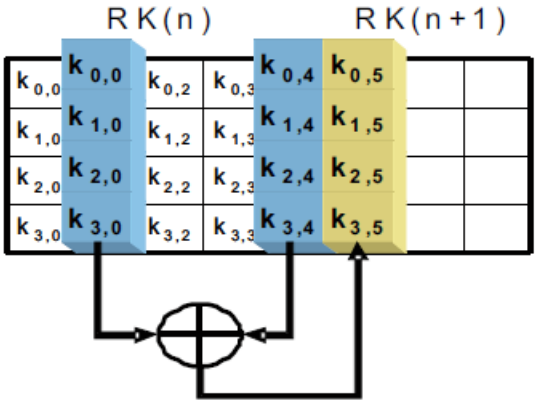


Figure 6-8: 2 to 4 columns generation

For the decryption process, all the before introduced steps are reversed using the RBox (reverse box) instead of the SBox.

6.2.4 Advantages (why chosen)

The AES is now the worldwide most used standard due to its high level of security and efficient data encryption. Its large key length makes it immune against the hacking attacks, in addition to its complex operation discussed before, which makes it so hard to track back and break. Also its acceptable usage of the memory and the processing unit as shown in Table 6-3 makes it so efficient for a lot of applications.

Table 6-3: AES memory usage and cycles

	Encryption	Decryption
Clock cycles needed	~6600	~8400
Flash usage (bytes)	1839	2423
RAM usage (bytes)	80	80

6.2.5 Reduction

As mentioned before the AES works on fixed 128-bits block of data as an input and an output. For the CAN bus with variable data size and maximum data length of 64 bits we had to modify the AES software implementation to accept variable input data length, and to generate variable data length output. Working with the same implementation without modifying the data size leads to wrong output so although that modification may affect the memory usage and the cycles needed (not much) but it's necessary and still acceptable for our system.

6.3 Parts of the system

As mentioned before the system proposed here consists of three main elements to face most of the defined attacks. Those three elements are message counter –anti replay counter-, ECUs authentication and data encryption.

6.3.1 Anti-replay counters

Replay attack as defined previously is when the attacker uses a message that was legally sent on the bus and repeat sending it to force a certain action. In our system this legally sent message will be properly encrypted and authenticated as it came from an authorized sender at the first place. As a result, there is no reason for the receiver to suspect the message or even define the happening attack.

A powerful solution is to use message counters for each message. This message counter will be synchronized at sender and receiver and incremented every successful transmission. This counter should be sent within every message -in the extended ID field-. When the receiver gets a message with a certain ID, it checks the counter value in it and compares it to the value stored at the receiver. If the value is less than or equal to the stored counter value associated with this message, then it is obviously an old message used by an attacker. In this case the repeated message will be dropped and the attacker will not be able to force the action meant by this message.

In our proposed system 8-bits message counter is associated with each message. The counter is resetted to zero every driving session and the related counter is sent with every message in the extended ID field of the CAN frame.

6.3.2 Authentication

Authentication is necessary for communicating parties to guarantee the source of each message. CAN bus nature is based on message ID, and there is nothing to define the sender's identity. This opens the door in front of an intruder to send legitimate messages as If they come from their original source.

To guarantee authenticity it is necessary for the receiver to be able to verify the sender's identity. In order to achieve this, the proposed system defines an authentication code for each communicating pair. This authentication code is an 8-bit code shared between sender and receiver and is considered as a signature to guarantee that the message originates from a trusted sender. This 8-bit authentication code is sent within every message frame in the extended ID field side by side with the message counter. The receiver checks the authentication code received and compares it with the stored authentication code associated with the trusted message sender. If the message doesn't originate from its trusted sender, the receiver knows it is an attack and drops this frame.

Most of the published papers in CAN bus security depend on message wise authentication code, in which a code is generated for each transmitted message and is based on the transmitted data. This approach needs to perform calculations on each message frame, which may lead to a significant delay. As mentioned before those delays are not really welcomed in real time automotive applications.

We instead chose the computationally light option of generating authentication code for each sender and receiver pair at the beginning of each driving session, and use this fixed 8-bit code for the rest of the driving session. The idea that the authentication code is fixed is not as powerful as having a message based varying code, but having the message counter by its side the total 16-bits of MAC and message counter will be changed each transmitted message. Moreover, those 16 bits will be sent in an encrypted form. This will guarantee that during the lifetime of a driving session it will not be easy for an adversary to reveal the encryption key, authentication code and message counter -all three of them- to be able then to launch an attack. This is acceptable because the data rate limitation of the CAN bus also limits the adversary capabilities when trying to attack the system.

The proposed solution is based on operating the CAN protocol in extended mode. The modifications made on the extended ID field are illustrated in Figure 6-9.

Figure 6-9: Modified extended ID field

6.3.3 Encryption

Data confidentiality means not to be able to overhear the data transmitted on the bus, which is of an increasing importance because of the demanding automotive applications that may contain personal user information.

Depriving the adversary from the ability to easily overhear what is being transmitted on the bus not only guarantees data confidentiality, but it also makes it difficult for an adversary to launch other types of attacks. All CAN bus mentioned attacks start at the ability of attacker to monitor the communication and define the frames associated with each task in the automobile. After monitoring the bus for a sufficient time the adversary will usually be able to define the structure of the frame and the variables in each type of messages to manipulate the task associated with this message. Blocking this way

against the attacker will make it hard for an adversary to reveal the secrets in the CAN messages and how exactly they are functioning, which will greatly lower the possibility of attacks. The adversary will need first to reveal the encryption key which is a time and resources extensive task, especially when using AES algorithm for encryption.

At the beginning of a new driving session each communicating pair generates encryption key to use during this session. This key is used to encrypt data at the sender and decrypt it at the receiver.

6.4 System operation

Each part of the system was previously discussed separately. The flow of operation will put those pieces together to form a complete security approach for in vehicle CAN bus. First of all, initial key storing phase needs to be held once in the automobile fabrication process. Then, at the beginning of every driving session each sender and receiver communicates to define an authentication code and encryption key to communicate with during the session. At the end of the second phase the vehicle is ready to operate in a secure manner under the security measures taken. A refreshment process for the long term stored information is also held at each driving session. In the following, each phase will be separately discussed.

Phase 1: Initial key storing phase

At the car manufacturing and ECUs programming process each ECU in the vehicle will hold two secrets in a sealed long-term memory which is assumed not to be manipulatable by intruders. The two secret keys are:

- **Initial session key:** a global key of 128 bit is stored in all car ECUs and is changed from a vehicle to another.
- **ECU number:** a unique 64 bits key assigned as identification to each ECU.

Those stored keys will be used to generate authentication code and encryption keys in phase 2.

Phase 2: Authentication code and encryption key distribution

Starting from the stored secrets each communicating pair (transmitter of a certain message and the intended receiver for this message) generates a common identification code that the transmitter will attach to the message to let the receiver know it is authentic, as well as encryption key to communicate with.

The flow chart of authentication and key distribution phase is illustrated in Figure 6-10. The phase starts when the sender retrieves its own unique ECU number, encrypts it with the initial key stored at all ECUs and transmits it to the receiver. The receiver gets the encrypted transmitter ECU number, decrypts it and both the sender and receiver will use the initial stored global key and the ECU number of the transmitter as inputs to our reduced SHA-1 function. SHA-1 function will perform one way –not reversible-bit wise operations using these inputs. The output of SHA-1 comes as 160 bits which will then be divided into 8 bits as the authentication code (MAC) and 128 bit encryption key. The last 32 bits of the 160 bit output will be used as a random number; this random number is needed in the process of stored keys refreshment.

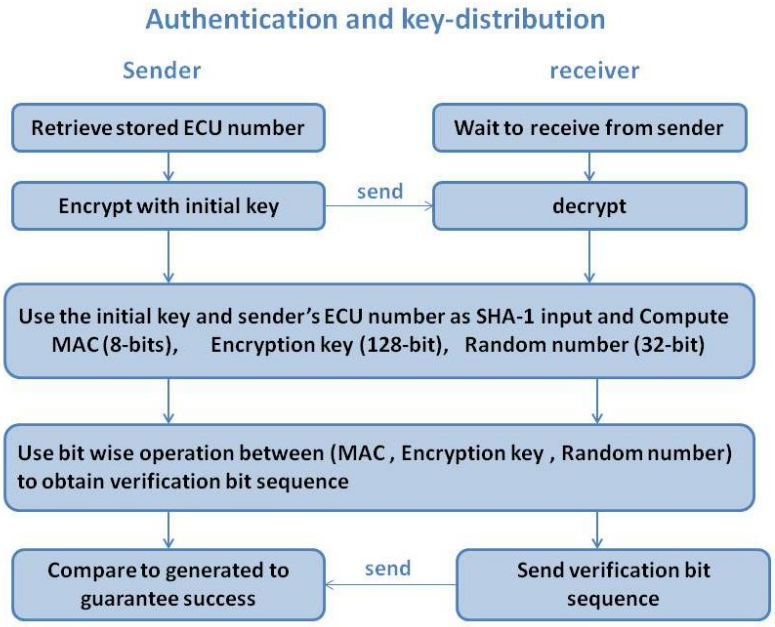


Figure 6-10: flow chart of the initial key distribution phase

Now both sender and receiver should have generated the same authentication code and encryption key. There exists a possibility that an error may occur in the ECU number

transmission, the encryption or the SHA-1 algorithm that may result in different authentication code and encryption key at sender and receiver. In this case those two entities will not be able to communicate through the driving session, which is not acceptable. To overcome this issue a check happens to guarantee that the generated secrets are identical.

For this check, after the two ECUs get done with the SHA-1 they use the outputs generated (MAC, Encryption key and random number) in an identical bit wise operation at sender and receiver. The result of this bit wise operation (verification bit sequence) is then sent back from receiver to sender. The sender checks the received value with the generated value, if equal the communication succeeded, if not the process needs to be repeated.

It worth mentioning that this method is followed so as not to transmit any of the secrets (session encryption key and message authentication code) on the communication bus. The only transmitted secret is the ECU number which is transmitted in an encrypted manner, and will shortly be updated before being used in the next driving session.

All communicating pairs take turns to generate, share and verify a MAC and a key to use through the starting driving session. This approach may be time consuming but it is also acceptable to happen once at the beginning of each driving session.

Phase 3: Car operation

After finishing the second phase of key distribution and authentication, the vehicle is ready to normally operate. The communication between ECUs now happens in a secure manner following this pattern of operation shown in Figure 6-11. The sender adds the message authentication code and the message counter in the extended ID field. The message is then encrypted using the session key and sent. The receiver receives the message, decrypts it and then extracts the MAC and counter from the extended ID field.

If the MAC is identical to the pre-shared MAC, the message is authentic. If the counter value is one count incremented from the last stored value, the message is fresh and not a replay message. The authentic fresh messages are accepted, while the rest is discarded. The operation will keep going this way until the end of the driving cycle.

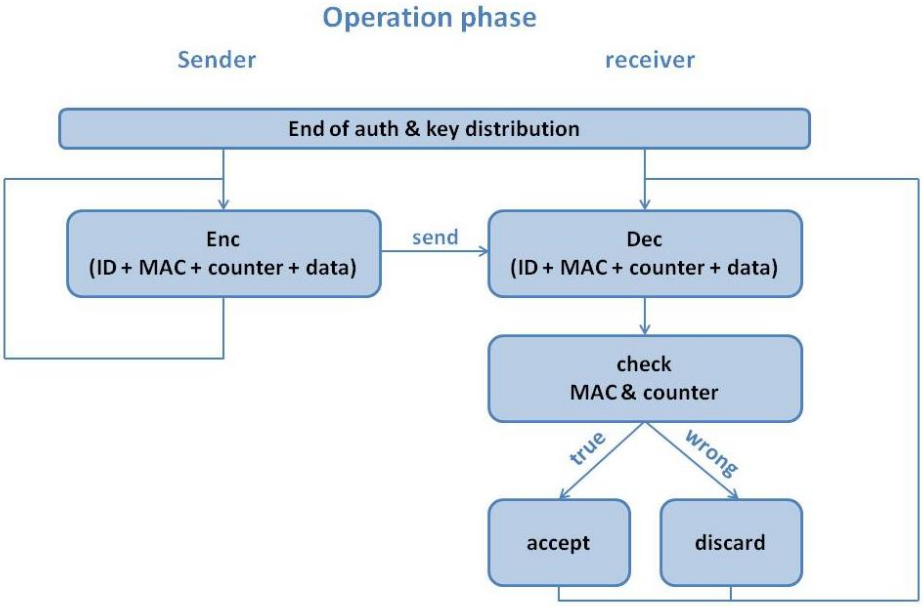


Figure 6-11: Car operation phase

Initial key & ECU number update

To increase security, the lifetime of the long-term memory stored variables (initial key and ECU number) is defined to be one driving session. Those two variables are updated as soon as they are used. The update phase is performed using a bit wise operation (computationally light weight) between the old values and the random value generated in the SHA-1 output for each driving session.

Chapter 7: Simulations and Results

After defining the countermeasures to be taken to secure our CAN network, and defining how exactly each part of the system will function. The next step was the software implementation and evaluation for the system. Software C implementation code was written performing all the phases of our system operation. The chosen target was **Tiva™ TM4C123GH6PM Microcontroller ARM Cortex-M4F Based.**

To evaluate the system performance many essential factors should be evaluated. Now we will go through these factors with concerning our system:

1- **Backward compatibility:**

Means that the proposed system is compatible with the existing systems and could be easily integrated with the existing automotive networks. Our system didn't require any modified hardware, and depends only on software implementation so it could be easily integrated with the existing automotive environment with only software modification-achieving backward compatibility.

2- **Confidentiality and integrity of the data:**

As mentioned before one of the most important requirements of the system is to guarantee the integrity of the data -that it's not manipulated-, in addition to the confidentiality of the data -that it's considered private between its sender and the interested receivers so that no other one could sniff and extract information. By the encryption/decryption process the confidentiality is guaranteed since there is not anyone that can understand the data except who has the key to decrypt it. Integrity is also guaranteed through the authentication and encryption/decryption processes, and the data is vulnerable only for the communication drops and not for anyone to manipulate it, as in this case it would turn into unacceptable data by the receiver.

3- Authentication:

As mentioned before the most important factor to resist message injection attacks is to have an authentication protocol existing in the network. With that protocol, every node can recognize its sender, and accept data only from those who are trusted. With our authentication protocol explained previously, every node has a MAC that is known by its receivers, so no any other data from nodes that are not authenticated would be acceptable.

4- Replay attack resistance:

Using the message counter in our system made the freshness of messages in the system guaranteed. Every node has the last value of the message counter and shall not accept any similar message with equal or lower counter value, so our system is immune against replay attack as any un-updated message shall not be accepted.

5- Repair and maintenance:

As the whole system is only software based architecture, it can easily be tested, repaired and guaranteed for maintenance. Also any future improvements in the hardware and specially CAN controllers could be used for our benefit as the system doesn't rely on any specific hardware so it's compatible with any chance for enhancement.

6- Overhead:

The overhead added by our system is considered to be at the start of the driving session only -in the phase of authentication and key distribution-. After that phase no overhead packets is generated in the data transfer phase as we used the extended ID for the MAC and no need for any specified packets for the security reason. The only added overhead will be the time for the encryption and decryption of the messages, but it is still in the acceptable range so we can say the system is overhead free.

7- Real-time performance:

Here is the most important system evaluation factor as there is no real-time performance vulnerability allowed in the automotive industry, because a very small delay could lead to a dramatic end. So that was our first concern in our implementation to reach the very optimized software.

Code composer studio was used to debug and measure the performance of the introduced software. The computation time of AES and SHA-1 was measured in clock cycles. Then the whole pre-driving protocol of authentication duration was measured also in clock cycles, and here are the results:

- AES algorithm : 12.5k cycles for encryption, 18k cycles for decryption
- SHA-1 algorithm: 15k cycles
- The total pre-driving session phase: 156k cycles in a single connection between one sender and one receiver.

To have better intuition of those numbers, the clock cycles was used to calculate the computation time in milliseconds for different crystal frequencies –different ECUs computational power-. For variable frequency ranges the needed time tabulated in Table 7-1.

Table 7-1: Time measurments

Frequency	8 MHz	24 MHz	48 MHz	250 MHz
AES encryption	1.56 ms	0.52 ms	0.26 ms	0.05 ms
AES decryption	2.25 ms	0.75 ms	0.375 ms	0.072 ms
SHA-1	1.875 ms	0.625 ms	0.3125 ms	0.06 ms
Pre-driving session	19.5 ms	6.5 ms	3.25 ms	0.624 ms

As shown in Table 7-1, the higher ECU crystal frequency will lead to better timing performance. For the pre-driving session protocol in authentication and key distribution phase the timing of the process between each sender and receiver is graphed in terms of crystal frequency in Figure 7-1.

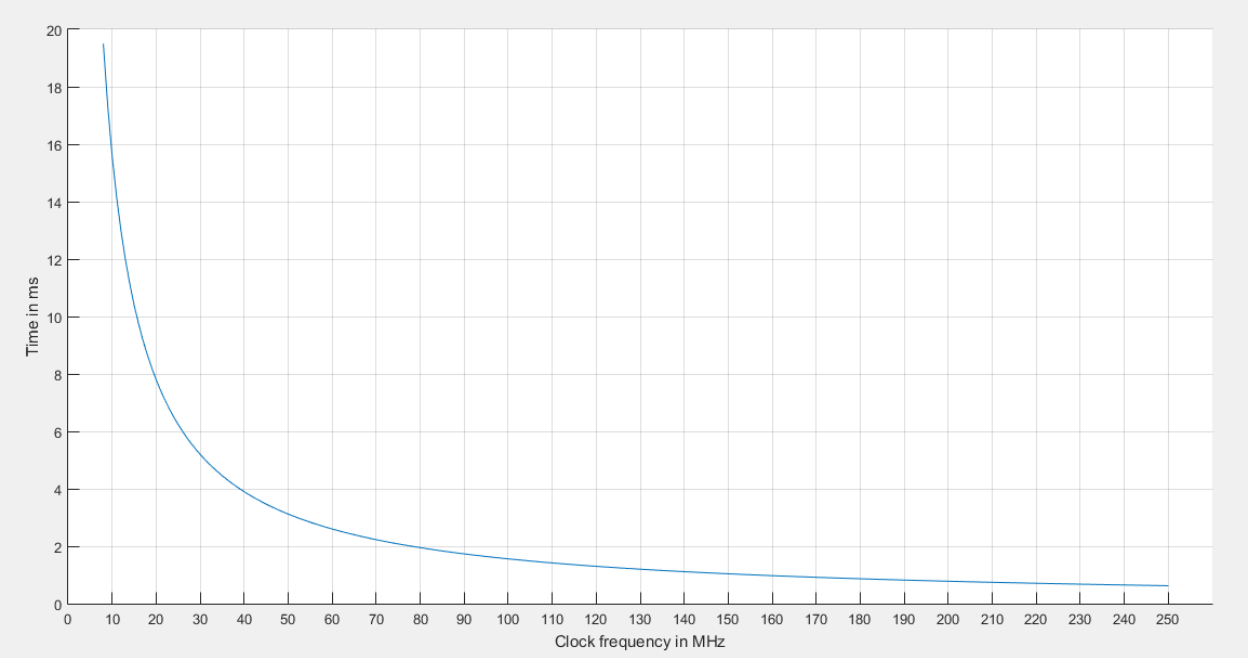


Figure 7-1: Pre-driving session performance with frequency

The pre-driving session added time is needed only once through the whole driving session at its beginning. The previous time measurements are for single TX-RX configuration and should be generalized on a real automotive network environment. The real-time results even if on a single TX-RX scale, are very promising as only some time needed at the start of the driving session, would lead to a secure communication for the whole driving session. The AES added time is applied to all messages during the driving session. Having reasonable crystal and good performing controller, the real-time performance will be very promising and will have efficient results in the real world.

Chapter 8: Conclusion and Future Work

8.1 Conclusion

In this thesis, we introduced a security system to stand against car theft attempts, which is based mainly on three modules:

- Face recognition module which defines the person in front of the wheel, and makes sure it is the authorized owner.
- Liveness detection to make sure that there is a live person in front of the camera.
- CAN bus security to limit the ability of those attacks.

Face recognition part was discussed in details, showing the pipeline of our work and the effort to achieve best performance and best accuracy. Our model also was discussed in details, the architecture used, the dataset which we train the model on and the framework used in training. So, as a result we have a complete face recognition system starting from capturing and preprocessing frames, applying face recognition on it and comparing with identities in the database, then take the action of having the permission to move or lock the car. The system has also the ability to add or remove trusted persons. For liveness detection, it was shown that it is implemented using eye aspect ratio algorithm which is real-time and incredibly accurate.

Controller area network security will keep being a field to improve and enhance. It needs extensive testing to fill in all the gaps that may happen while applying the security concepts. It is also needed to find the optimal point at which sufficient security is achieved within the accepted alteration to the system performance.

8.2 Future Work

For a car security system, face recognition model should have high accuracy as possible to prevent thieves from accessing the engine also it should be general as possible to recognize any face since people can be different in age, skin color, etc. Also it should recognize faces from multi poses with high accuracy to be fast as possible.

These reasons encourage to train on smaller dataset then to increase it or to use augmentation on small dataset in order to increase the number of samples.

On the other hand, the liveness detection is considered a security system for the face recognition itself as it prevents hacking this system using an image of the owner, but unfortunately it depends on an algorithm that uses eye blinking which might be passed through a video of the owner. So, from here it needs to be enhanced to be unable to be passed using images or videos.

So for the face recognition part we can conclude the future work as:

- Training on smaller part of vggface2 dataset then increasing it, train the model for a large amount of time or training on augmented LFW pairs dataset.
- Liveness detection enhancement.

In CAN bus security the tradeoffs between security level and fast performance will always exist. As previously stated the real time requirements of an automobile cannot be compromised. A car with no strict real time reactions can be a death machine. Starting from this point the proposed CAN bus security solution was built adopting the simplest ways to achieve security requirements. Till now, no proposed CAN bus security method can claim to achieve both very high security level and sufficient real time performance.

Our method chose to cure the most common vulnerabilities of CAN network with a simple way first, and then to build up from this point. What needs to be done next is to test the proposed system extensively in a real vehicle environment, to define points of strengths and weaknesses and to give an intuition on how much computation can be added to achieve better security without rendering the real time performance. Some additional work can also be done to cover more attack surface.

It is eventually obvious that with the existing hardware CAN bus security will always suffer from limitations. Some future hardware modifications in CAN controller can be a really good step to have more flexibility and to achieve really sufficient security levels that enable car owners to drive securely with their connected cars. It cannot be denied that hardware modification will badly affect the backward compatibility and may lead to the need of huge system modifications, but on the other hand it will leave more room for security improvements which is really demanding in the field of connected cars.

We can conclude the future work as:

- Testing the performance in a real vehicle environment.
- Denial of service attack solution.
- Considering future CAN controller modifications.

References

[1] "ImageNet Large Scale Visual Recognition Competition (ILSVRC)," Image-net.org, 2017. [Online]. Available: http://www.image-net.org/challenges/LSVRC/.

[2] "statista," 2007. [Online]. Available: statista.com.

[3] C. Miller and C. Valasek, "Adventures in Automotive Networks and Control Units," 2014.

[4] "Trustpilot," 2018. [Online]. Available: https://tech.trustpilot.com/forward-and-backward-propagation-5dc3c49c9a05.

[5] "towards data science," 2017. [Online]. Available: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

[6] "CS231n," [Online]. Available: https://cs231n.github.io/convolutional-networks/.

[7] "Analytics Vidhya," [Online]. Available: https://medium.com/analytics-vidhya/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5.

[8] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2015.

[9] Q. Cao, L. Shen, W. Xie, O. M. Parkhi and A. Zisserman, "VGGFace2: A dataset for recognising faces across pose and age," in *IEEE International Conference on Automatic Face & Gesture Recognition*, 2018.

[10] "Machine Learning Mastry," 2017. [Online]. Available: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/.

[11] "pyimagesearch," 2017. [Online]. Available: https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/.

[12] "NVIDIA," 2017. [Online]. Available: https://developer.nvidia.com/embedded/jetson-tx2-developer-kit.

[13] "Vector E-learning," 2010. [Online]. Available: https://elearning.vector.com/.

[14] O. Avatefipour and H. Malik, "State-of-the-Art Survey on In-Vehicle Network Communication “CAN-Bus” Security and Vulnerabilities," 2018.

[15] Q. Wang and S. Sawhney, "VeCure: A Practical Security Framework to Protect the CAN Bus of Vehicles," 2014.

[16] R. Kurachi and Y. Matsubara, "CaCAN - Centralized Authentication System in CAN," 2016.

[17] J. V. Bulck, J. T. Mühlberg and F. Piess, "VulCAN: Efficient Component Authentication and Software Isolation forAutomotive Control Networks," 2017.

[18] A. V. Herrewege, D. Singelee and I. Verba, "CANAuth - A Simple, Backward Compatible Broadcast Authentication Protocol for CAN bus," 2011.

[19] S. Woo, H. J. Jo and D. H. Lee, "A Practical Wireless Attack on the Connected Car and Security Protocol for In-Vehicle CAN," 2015.

[20] M. Gmiden, M. H. Gmiden and H. Trabelsi, "Cryptographic and Intrusion Detection System for automotive CAN bus: Survey and contributions," in *International Multi-Conference on Systems, Signals & Devices (SSD'19)*, 2019.

[21] N. Nowdehi, A. Lautenbach and T. Olovsson, "In-vehicle CAN message authentication: An evaluation based on industrial criteria," 2017.

[22] A. J. Menezes, P. C. v. Oorschot and S. Vanstone, "Hash Functions and Data Integrity," in *Handbook of Applied Cryptography*, 1996, pp. 321-376.

[23] National Institute of Standards and Technology, "Announcing the Advanced Encryption Standard (FIPS PUB 197)," 2001.

[24] U. Kretzschmar, "AES128 – A C Implementation for Encryption and Decryption," 2009.