Bangla Number Plate Recognition from Noisy Video Footage Using Deep Learning

by

Mahfuzur Rahman Chowdhury
17101082
MD. Muyeed Shahriar
16101026
Esrat Jahan Meem
17101538
S.M.Faiyaz Hossain
16101034

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

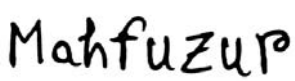
Department of Computer Science and Engineering
Brac University
January 2021

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

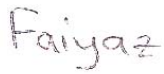
Student's Full Name & Signature:



Mahfuzur Rahman Chowdhury
17101082

MD. Muyeed Shahriar
16101026

Esrat Jahan Meem
17101538



S.M.Faiyaz Hossain
16101034

Approval

The thesis titled "Bangla Number Plate Recognition from Real-Time Noisy Footages using Deep Learning Method" submitted by

1. Mahfuzur Rahman Chowdhury (17101082)
2. MD. Muyeed Shahriar (16101026)
3. Esrat Jahan Meem (17101538)
4. S.M.Faiyaz Hossain (16101034)

Of Fall, 2020 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 23, 2021.

Examining Committee:

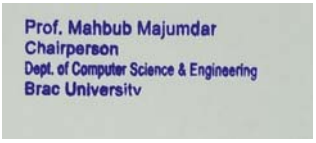
Supervisor:
(Member)

Dr. Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
BRAC University

Thesis Coordinator:
(Member)

Dr. Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)



Mahbub Majumdar, PhD
Professor and Dean, School of Data and Sciences
Department of Computer Science and Engineering
Brac University

Abstract

The automatic number plate recognition of a vehicle from noisy footage has become an important issue in Bangladesh. In Bangladesh, the number of vehicles is increasing at a very high frequency. Moreover, unlike the English number plate, the Bengali number plate consists of two line inputs. As the number is increasing the difficulty of identifying a vehicle is also increasing. There are multiple problems where number plate recognition is very necessary such as, in a crime scene, finding out a lost vehicle, identifying the guilty vehicle in a road accident, etc. The main challenge in this system is to predict the Bangla numbers from very noisy images. Most of the methods of identifying vehicles from noisy data are not as accurate as they should be. In order to reduce such noise, we applied a total of ten filter algorithms but among those four filtration techniques gave us good results those are Conservative Filter, Wavelet denoising filter, Primal-Dual Algorithm, and Autoencoding. In order to detect the characters from the image files extracted from the camera footage, we used two detection algorithms. Those are Haar-Cascade and another one is Contour based Edge Detection algorithm. And to recognize the Bangla number characters from the bangla number plate we used ANN Backpropagation method.

Keywords: Deep learning, Bangla number plate, Noisy data, Conservative Filter, Wavelet denoising filter, Primal-Dual Algorithm, Autoencoding, Haar-Cascade, Contour based Edge Detection, ANN Backpropagation, Modified Non Local Mean Filter.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.
Secondly, to our advisor Md. Golam Rabiul Alam,PhD sir for his kind support and advice in our work. He helped us whenever we needed help.
Thirdly, Md. Golam Rabiul Alam,PhD sir and the whole judging panel of Fall 2020. All the reviews they gave helped us a lot in our later works.
And finally to our parents without their throughout sup-port it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

Δ Delta

ϵ Epsilon

Ω Omega/ Big O

σ Sigma

$ALPR$ Automatic License Plate Recognition

ANN Artificial Neural Network

CNG Compressed Natural Gas

CNN Convolutional Neural Network

$DSLR$ Digital Single-lens Reflex

FFT Fast Fourier Transform

FPN Feature Pyramid Network

GPU Graphics Processing Unit

HOG Histogram of Oriented Gradient

$HOLO$ Holomorphic Filter

ICU Intensive Care Unit

IOU Intersection over Union

$LSTM$ Long Short-term Memory

$MNLMD$ Modified Non Local Mean Denoising

MRI Magnetic Resonance Imaging

$NLMD$ Non Local Mean Denoising

OCR Optical Character Recognition

$R-CNN$ Region Based Convolutional Neural Networks

RNN Recurrent Neural Network

SAT Segmentation with adaptive threshold

SGD Stochastic Gradient Descent

$SIFT$ Scale-invariant Feature Transform

SPP Spatial Pyramid Pooling

SSD Solid-State Drive

$YOLO$ You Only Look Once

α Alpha

δ delta

λ Lamda

∂ Partial Derivative

θ Theta

Chapter 1

Introduction

Automatic number plate detection for vehicle track is one of the challenges we face in Bangladesh. There are numerous vehicles on the road, major or minor accidents and violation of laws is happening so often and it has become so challenging to detect a specific vehicle on the road. For identifying the vehicles who committed crimes or disobey the law is important as well as difficult for Bangladesh because of the fuzzy data. Moreover, another challenging issue is to handle the outdoor illumination condition at different times, and also not all the number plate formats are the same. That's where the technique needs more improvement so that it can give a more accurate result. Bengali number plate detection is not that much enrich, the main reason is lack of the dataset. Also, most of the footage is acquired from the closed-circuit camera and the image quality drops so low in that case the image becomes blurry and full of noise. In order to reduce such noise, we applied a total of ten filter algorithms but among those four filtration techniques gave us good results those are Conservative Filter, Wavelet denoising filter, Primal-Dual Algorithm, and Autoencoding. In the context of Bangladesh, it is mandatory to print one's vehicle number plate in Bangla. In order to detect those Bangla number characters from the image files extracted from the closed-circuit camera footage, we used two detection algorithms. Those are Haar-Cascade and another one is Contour based Edge Detection algorithm.

1.1 Motivation

In our country there are so many incidents on the roads regarding traffic law violation. Unnecessary speeding, risky overtakes, loud horns in such places where it should not be and still there are uncountable situations occur on the roads every day. To handle this sort of situations through taking the law violators into account we thought about something which can help us to eradicate this sort of problems and unwanted situations from our roads. In the context of our country Bangladesh, all the number plates of every vehicle is issued in Bangla language. Because of that even if we want to try and catch those who violated the law the only way for us to locate them down is to identify them somehow with a unique attribute and which is a number plate. Every vehicle on the roads holds a number plate each, those plates consists a portion which identifies from which district the vehicle is registered, a unique 6 digit number along with a Bangla alphabet. It was also possible to detect and recognize the face of the driver through a very high resolution camera

from different points on the roads but that become really complex, rather detecting the vehicle number plate was much more logical and cost effective. So that's what inspired us to choose these plates as the basic identifying media for any vehicle.

1.2 Problem Statement

Bangladesh is a small country with a very large population. The more the population increases the more the demand is increased in every aspect. One of the major aspects is the transportation system or the vehicles people need to travel from one place to another. As the country has an rising economy, many people are being able to afford a personal vehicle of their own. The more number of vehicles require an equal number of number plates. In the context of Bangladesh we have several drawbacks regarding identifying the registration number from the registration number plate of any vehicle.
Firstly, all of the number plates issued for the vehicles are printed in bangla fonts. As we have a very limited amount of data sets regarding bangla font number plates, it is one of the major issues of this project. Afterwards unlike the other countries not all of our vehicle registration numbers are printed in the same media. For example, if a person buys a motorbike today he needs to register his bike first in order to get legal access to the road. But he cannot get his machine engraved number plate in 5 or 6 months, in some cases it sometimes takes even a year. So in this kind of situation what they do is, they print the number on a paper and laminate the paper and attach the paper in the back side of the vehicle. In these sorts of cases the number seems kind of different from a machine engraved number plate because there is no specific font for printing the number in a paper. This kind of printed number requires another huge number of dataset itself.
Moreover we do not have very high resolution cameras equipped in our roads like the first world countries. So this is another big challenge for us to identify a specific number from a moving object (cars, buses, motorbikes etc) through the lens of a very low resolution closed circuit camera. These low resolution noisy footages generate very noisy pictures which are very hard to identify automatically. Existing algorithms are very effective at identifying number plates in clear pictures but in respect to Bangladesh's situation that is not quite possible. Detection of number plate normally consists of 4 parts which are image collection, number plate area detection, breaking down the characters individually and in the end recognizing each of those separate characters[9]. We will try to identify problems with the existing algorithms and try to increase the accuracy of those algorithms.

1.3 Research Objectives

We will be trying to implement deep learning methods along with some image detection algorithms to identify Bangla number plates from noisy dataset. Improving in accuracy will result in better identification of vehicles that are used to commit crimes, guilty vehicles involved in road accidents and even tracking stolen vehicles. Many companies like OpenALPR provide detection of number plates but none of those support Bangla Language. Bangla having a very complex lexical structure makes it harder to detect than the other languages. On top of that noisy picture

makes it even harder.
We will try to find out the best ALPR(Automatic License Plate Recognition) algorithm that works with Bangla Language. Existing Bangla number plate recognition methods work very well with clear and still images[12] but not with noisy data. Our goal is to identify number plates with the same or better accuracy in noisy data and images.

1.4 Research Contribution

In this thesis, a new modified filtering algorithm has been created, as there are not many filtering algorithms to detect Bangla number plate characters at present. Because of that, it carried out a comparative study throughout different filtering algorithms to recognize the characters in the Bangla number plate and came to a decision to create a new modified one that gives more accurate results than any other algorithms before. To make the detection algorithm work, two random datasets have been made and another two datasets have been made from direct Bangla number plates along with the individual Bangla number characters which are more precise than the previous two datasets. A whole new approach has been followed to recognize each Bangla digit and character which is YOLO (You Only Look Once) version 3. Previously no work has been done with the YOLO V3 algorithm for Bangla character detection and recognition. Throughout a detailed comparison with other algorithms, it has been proven that the new modified filtering algorithm works always better than any other algorithm in the perspective of Bangla number plate and character recognition.

1. To recognize Bangla number plates and characters a new modified filtering algorithm has been created.

2. Two datasets for number plate detection and two datasets from Bangla number plates themselves are created.

3. A new approach has been followed to detect and recognize Bangla number plates and characters which is YOLO V3.

4. During the comparison with existing Bangla digit and character recognition algorithms it has been seen that the new modified one always gives a better result, which is why we should use this algorithm.

1.5 Thesis Orientation

1. Chapter 1 is Introduction where the problem statement and research objective is discussed.

2. Chapter 2 is literature review where background and related work are discussed. In the literature review part it reviews the previous work on Image detection.

3. Chapter 3 is Dataset Collection and Preparation where footage collection, footage to image, cleaning image, adding noise, number plate detection, Haar

Cascade, YOLO V3 (Number plate detection), (Algorithm), (Dataset) and Detection Rate are discussed.

4. Chapter 4 is Methodology where applying filter, default 1, default 2, Modifying filter, Labeling Bangla word, Characters and Digits, ANN-Back Propagation and YOLO V3 are discussed.

5. Chapter 5 includes performance analysis and discussion.

6. Chapter 6 is Conclusion where all the future works and scope of improvements are discussed.

Chapter 2

Related Works

Previously multiple approaches have been taken for image recovery, object detection, and number detection. For this thesis, we read some journals that are very much similar to this work and analyze their methodology to solve those particular problems. In the past year, different researchers published their research in this sector. Below we are going to summarise those papers and review them.
The first paper [2] that we read is by Marco Reisert and Hans Burkhardt, in their paper they basically used three different algorithms for image recovery and object detection. For speed reasons they used FFT to implement the first and last convulsions and between those they approximated the differentiation by a finite difference scheme. But they keep the differentiation at a low order which helps them to compute faster than before, but it occurred error which is isotropically. The three filters they used are wiener filter, steerable filter and holomorphic filter. For the test they used a black background with white painting with Gaussian noise ratio of 0.3, 0.4 0.7. In that test wiener filter gives a bad result then the rest of the two. The recovery image is almost the same in terms of both steerable and holomorphic filters. But the sharpness of the holomorphic filter seems more nice than streeable. So to verify their hypothesis they run another test. For this test they use a white background with many black little strips on it. After running the test it is clear that the holomorphic filter is better among them. As a result, for object detection they used holomorphic filters. For object detection they used pollen grains as their object. This time they use third-order filters with degree five. In third-order two variables are holomorphic and one is anti-holomorphic. So we can call it a mixed holomorphic filter. For finding out the local maxima they used 2-D and 3-D voting maps. But here 3-D voting map gives a very bad result. So they took the 2-D voting maps results under consideration. Because of the pixel miss match between the test image and input image they used two different Gaussian functions to match the images. They used the HOLO method to show the result. They also compare their method with two other methods named SIFT and PCASIFT. They compare the error rate of these three methods. For comparing the methods they divided the data set in two groups Porate and Spore. Each group has two sub groups 10px and 20px. Here the error rate of SIFT is higher compared to the other two. But the error rate of HOLO and PCASIFT are almost the same. But they showed three examples where PCASIFT failed to detect the object but HOLO can successfully detect the object. For image recovery this paper's methodology will be very useful for us. But if we talk about object detection then it will not be that much helpful for this project.

The Tesseract OCR engine algorithms utilized [3] in the different phases of the engine square measure explicit to English letters in order, that makes it simpler to help various content that square measure primarily comparable by merely coaching the listing from the new contents. In-depth data of the Bangla and syllabic script scripts is important to integrate the popularity support for these scripts in Tesseract. The engine has its own segmenter to find lines, words and characters. Since Tesseract could be a complete OCR package, once it's trained with the coaching knowledge, it tends to don't ought to be troubled regarding feature extraction and recognition so as to acknowledge the characters within the Bangla script. In most of the cases the image of basic character and modifier incorporates a specific amount of overlap between them. If the segmenter is triple-crown in separating the modifier and basic characters employing a vertical column, then Tesseract is triple-crown in recognition. To perform segmentation, it tends to follow the approach that's specific to the need of the Tesseract engine.

A paper[6] by Yu,Zhao and Wang written for genuine uproarious picture restoration by an effective edge-based bilateral filter. In their paper, they divided all the pixels of noisy images into non-edged based and edge based regions. For noisy images gaussian type filters can be used for smoothing the picture as in noisy images if we take one portion and make a matrix, we can see some values become zero for which the noise was created and in gaussian that noise were suppressed by calculating the center pixel. But in gaussian the main problem is the image becomes blurred as the edge becomes smooth also. For this in this paper, they use bilateral filters. By using bilateral filters they got a much clearer image than gaussian. The algorithm mainly works like firstly, it detects the edge and then applying a bilateral filter, they get the restored image from a noisy image. In bilateral filters, they use two more ways such as non-nearby bilateral filter in the edge district and neighborhood bilateral filter in the non-edge locale. In this paper, they also use the denoising algorithm where the authors use two iterations for enhancing the reestablish execution.

In paper [5] written by Sarfaraz, Shahzad, Elahi, Fraz, Zafar and Edirisinghe was about a productive continuous programmed tag acknowledgment (ALPR) system which is intended to chip away at loud video film got from cameras that are not devoted to using in ALPR. The current ALPR calculations depend on the supposition that the information video will be acquired through a devoted, high goal, rapid camera and is upheld by a controlled catch climate, with fitting camera tallness, center, shade speed and lighting settings. Commonplace video measurable applications may require a vehicle having a specific number plate on uproarious video film acquired by means of non-committed, medium-to-low goal cameras, working under helpless light conditions. A foundation picture is needed to speak to the base condition of the region under assessment for identification purposes. The removed foundation is deducted from each casing to acquire the forefront object. Here the foundation learning is performed by utilizing the dramatic failing to remember strategy. The framework starts its experience calculation by thinking about the absolute first edge as the foundation and updates it with approaching edges. Each refreshed foundation is a weighted amount of the past foundation and the new edge. Thusly, the foundation powerfully adjusts the adjustments in the development of items or luminance conditions in the edge. Commotion present in the frontal area is eliminated by utilizing notable median filtering: a nonlinear technique that applies a sliding window on the edge pixels and eliminates clamor while saving the edge

data. Morphological administrators are applied subsequently to additionally refine the frontal area. After pre-preparing, competitor areas are distinguished by finding the shapes and limits of associated segments in the edge. In the wake of dismissing the bogus locales based on their size, the leftover competitor districts are kept an eye based on their surface similitude to tag like zones. The underlying learning is performed on the initial not many approaching casings by narrowing down the potential sizes of expected plate-like districts satisfying the underlying mathematical imperatives. The last choice of the tag is accomplished by separating pertinent highlights and order. For this reason, they utilized a histogram of situated angles HOG for highlight portrayal. After identification of the tag in the video arrangement, the subsequent stage is to monitor that tag with the development of a vehicle in the continuous edges. The limited plate is pre-prepared for commotion evacuation and upgrade of edge data. After pre-preparing, associated locales are distinguished in the upgraded picture, and bounding boxes are drawn around them utilizing least limit square shape 'MBR'. Thusly, each character is independently encased by a surrounded box.

In paper [4] written by Hasan, they discussed various edge detection algorithms like horizontal and vertical edge detection and canny edge detection algorithm then for Binarization they discussed Sauvola Image Binarization Algorithm and Otsu Image Binarization Algorithm. They discussed symmetry detection for detecting the number plate area of the vehicles.They also discussed ANN(Artificial Neural Network) with its activision function.In their work, they said that their algorithm can detect number plate at a rate of 92.8% which is higher than any other algorithms as they have the accuracy rate of 73.5 %, 85% and 0% respectively. For the segmentation part their algorithm has a success rate of 98.4% which is 246 out of 250. For identifying license plates they used SAT and Duan et al. that have a success rate of 98.4% and 93.75% respectively. Their proposed ALPR framework takes on an average of .16s to recognise the number plate in an input image. They also used the same proposed model for English license plates and that also had an average success rate of 92.7% for detection stage, 100% for segmentation stage and 89% for recognition stage.

In paper [13] written by Xiuxiu Bai, Lele Ye,Jihua Zhu and Taku Komura was about the skeleton filter which can elaborate the shape of objects in low measurements and also exceptionally critical shapes. Skeleton based showcasing seems to be very effective when it comes to recognizing text images. The skeleton filter comprises of a couple of oppositely situated Gabor-like channels, every one of which is made out of a positive and a negative isotropic Gaussian filter. A Gabor filter is a function of a Gaussian kernel modulated by a sinusoidal plane wave. A Gabor-like filter, a generalized Gabor filter structure, consisting of a positive and a negative Gaussian isotropic filter, used to detect paths and edges. Skeleton filters use convolution and non-max suppression operations. The most time-consuming step is the convolution operation, which involves the operation of FFT. The time complexity of FFT is $\Omega(n \log n)$ where n is the data size. In the method the time complexity is $k * \Omega(n \log n)$, where k denotes the number of filter banks. The method can be conveniently parallelized by measuring k channels simultaneously.

Another paper [7] written by Redmon, Divvale, Girshick and Farhadi, they talked about an algorithm called YOLO. Which full form is You Only Look Once. The whole concept of this algorithm comes from how we humans detect objects in real

life. Humans once see an image or watch an senare and by seeing once we can detect all the objects. They also tried to do the same thing with computers. YOLO is extremely fast then other algorithms. In the paper they claim that their network can detect 45 frames per second. Which is almost double of the R-CNN algorithm. YOLO not only detects objects it also tells the name of the object with the probability of it's prediction. YOLO takes an image then divides the image into small rectangular boxes. And each boundary box has five parameters x, y, h, w and confidence. Here, x and y determines the center of the box, h as height and w as wide is calculated relatively by the total height and width of the image. Confidence is calculated by IOU between predicted box and ground truth box. To train they created a network of 28 convolutional layers and 4 fully connected layers with random weight. The also add a layer of weight 0.5 after the first fully connected layer to drop date for avoiding overfitting. But there is a big limitation in this algorithm. YOLO basically calculates the ICU of the predicting box with the truth box. If an object is bigger than the predicting box it combines more than one box to dram a boundary on the object. But if the object is smaller than the predicting box then it fails to detect the object. Which is a major problem. But instead of these problems they authors are able to show us that it is still better than other algorithms by comparing the results of those. They mainly focused on the newly developed R-CNN algorithm with YOLO algorithm. And they are succed to showing us that YOLO is better. One thing in this paper is very uqic that they also showed a faster version of YOLO algorithms. Where they used 29 convolutional layers and fewer filters then YOLO. It gives fast detection but the accuracy is lower than YOLO.

In paper [8] Lehtinen,.Munkberg, Hasselgren, Laine, Karras, Aittala and Aila states about noisy images, from which they will generate a clean image. The writers use simple mathematical logic to signal the reconstruction of Machine Learning. By learning to get clean signals from tainted map signals. According to them, they recover data only by looking at corrupt instances. For this research they train their system by using clean data, here they use a regression model and convolutional neural network. The key noise they concentrate on are pairs of short and long exposure photos of the same scene, partial and complete k-space sampling of magnetic resonance images, fast-but-noise and slow-but-converged ray-tracing of synthetic scenes, etc. The authors train their system with this kind of data. On the basis of noisy results, three key things were done: first, noise reduction, second, denouncement of synthetic Monte Carlo images and restoration of undersampled MRI scans. They use CNN where convolutional layers detect the pattern of the image. The authors use denoising performance for the training era for additive Gaussian noise. They prepared the denoiser for the Monte Carlo way followed pictures made utilizing 64 examples for each pixel. The preparation set comprised of 860 building pictures, and the approval was finished utilizing 34 pictures from an alternate arrangement of scenes.

Another paper [10] by Xu, Yang, Meng, Lu, Huang, Ying, and L. Huang depicted the tag location and acknowledgment utilizing an enormous arrangement of information. In this paper authors are detecting the number plate of China roadside parking cars number plates by using convolutional neural network (CNN) mostly. The creators work on Chinese city leaving dataset where they gather over 250k extraordinary vehicle pictures by utilizing an Android handheld POS machine and physically clarifying the specific permit number. The main obstacles they faced was

the various weather of China for which the number plates are not always visible clearly so they need to work with blurry data also. In certifiable applications the model that creators produced are for the most part from generally high goal pictures at over 61fps and precision rate is 98.5. The authors proposed this paper with end to end trained roadside parking cars number plates which can be recognized with more accuracy and faster speed. Their algorithm is roughly divided into two major parts which are traditional methods and another one is neural networks models. For processing the image of the number plate they are mainly categorized in two ways where first one is, segmentation free methods and another one is segment first and then recognize. The dataset they utilized comprises of Chinese characters, a letter and five letters or numbers. For CNN algorithms the authors use ten convolutional layers where the input is a RGB image. In their detection module, they take the same width and height image for detecting the license number. The following layer is known as the container indicator layer by the creator where they highlight the guide yield by the last convolutional layer to three kin completely associated layers. For detecting the number plate from the image the last module they use is the recognition module with region of interest where they predict the license number. So basically the authors used three working modules here which is detection, prediction and recognition. The authors firstly divide the dataset into half and they use half of the dataset for pre-training purposes on ImageNet and another set as the default evaluation set. The systems creators enlarge the preparation information by arbitrarily testing multiple times on each picture to expand the preparation set by multiple times. The creators keep the standard convention in item identification Intersection over association and the acknowledgment exactness metric relies upon this. The creators join best in class discovery models with a cutting edge acknowledgment model named Holistic-CNN which gives 98.5% exactness picture every second. Also, the model joined with SSD and Holistic-CNN accomplishes up to 95.2% exactness. Between two module acknowledgment and discovery by sharing element maps they make this quicker which is 61fps.

Another paper [9] by Hossain, Uzzaman and Saif was about recognizing the Bangla number plate with a higher accuracy and low complexity. They divide their project into four main steps. First they made their dataset by taking photos of different Bangla number plates from 1 to 2 meter distance by using different mobile cameras. Their first primary advance was pre preparing. In this progression they convert the RGB picture into grayscale picture. For the gray value they multiply the RGB value with 0.3, 0.59 and 0.11 respectively then sum those three values. After that they convert the grayscale image into binary image by otsu method. Their subsequent advance was number plate extraction from the picture. To limit the tag they utilized Sobel edge procedure on the Grayscale picture. After that the image becomes black background and the letters become white. Then they applied morphological dilation and erosion. These two methods are used for processing images based on their shapes. Because of this reason the rectangular license plate is extracted from the image. Their third step was segmentation of characters. There are two lines in the Bangla number plate. First line speaks to region and type and the subsequent line contains the class of the vehicle and vehicle enrollment number. So they divided the number plate into two parts. Then they used boundary features to draw boundaries to segmentation of characters. Their last step is character recognition. For this they divide the segmentation by drawing as small as possible in the white area of the

image using boundary features. After that they calculate the value of correlation coefficient r from the template image and segmentize the image. After that the output is the best match form the dataset. Their task has a tag discovery precision of 94% and character acknowledgment exactness of 95.74%.
The paper [12] we read by Nazmus Saif, Nazir Ahmmed, Sayem Pasha, Md. Saif Khan Shahrin, Md. Mahmudul Hasan, Salekul Islam and Abu Shafin Mohammad Mahdee Jameel, they used a pre-trained model YOLO (You Only Look Once) that is extremely fast. They just run their neural organization on another picture at test time to anticipate location. Their base organization runs at 45 edges for every second with no group handling on a Titan X GPU and a quick form runs at in excess of 150 fps. YOLO is a lot quicker than R-CNN. Initially, they changed the crude picture into grayscale pictures. At that point they recognized the edge of the tag and from that point the characters lastly from pre prepared information they had the option to distinguish the Bangla characters. At that point they recognized the edge of the tag and from that point the characters lastly from pre prepared information they had the option to distinguish the Bangla characters. Their model has 53 convolution layers. They trained the model for 128 epochs for detecting license plates and 80 epochs for segmentation and recognition. They were able to get 99.5% accuracy which is more than any other model they tested.
While working with alphabet detection [11] using AWD-LSTM if the implementation plan is by using RNN, the attention mechanism where it's not really determined how many attention heads you need to have good performance and it's making it possible to run these algorithms on our own computers and lastly the self-attention boom layer and this is a really neat one.
A one-stage anchor-based detector is generally created of a backbone network, a detection neck that is usually a feature pyramid network (FPN) and a detection head for object classification and localization. This paper [14] 1st revises the detail structure of YOLOv3 and introduces a changed version that replaces the backbone to ResNet50-vd-dcn, that is used because of the basic baseline during this paper. Then introduce a bunch of tricks which may improve the performance of YOLOv3 nearly while not losing potency. DarkNet-53 is 1st applied to extract feature maps at totally different scales. Since ResNet has been widely used and has been studied additionally, They replace the initial backbone DarkNet-53 with ResNet50-vd in PP-YOLO. The authors replace some convolutional layers in ResNet50-vd with deformable convolutional layers within the paper the authors solely replace 3x3 convolution layers within the last stage with DCNs. The authors changed backbone as ResNet50-vd-dcn. Detection Neck then the FPN is used to make a feature pyramid with lateral connections between feature maps. The detection head of YOLOv3 is incredibly straightforward. It consists of 2 convolutional layers. A 3x3 convolutional followed by a 1x1 convolutional layer is adopted to induce the ultimate predictions. The output channel of every final prediction is 3(K + 5), wherever K is a range of categories. The Spatial Pyramid Pooling (SPP) integrates SPM into CNN and uses max-pooling operation rather than bag-of-word operation. YOLOv4 applies SPP module by concatenating max-pooling outputs with kernel size kxk, wherever k = f1; 5; 9; 13g, and stride equals one. Below this style, a comparatively giant kxk max-pooling effectively increases the receptive field of backbone feature. No parameters are introduced by SPP itself, however the quantity of input channel of the subsequent convolutional layer can increase. Thus around two extra parameters

and 1 % extra additional FLOPs are introduced. A higher pre-train Model working a pre-train model with higher classification accuracy on ImageNet could lead to higher detection performance. Here the authors use the distilled ResNet50-vd model because the pre train model. This doesn't have an effect on the potency of the detector. Below a larger batch size setting, the whole detection network is trained with stochastic gradient descent(SGD) for 25K iterations with the initial learning rate being 0.01 and a mini batch of 192 pictures distributed on eight GPUs. The authors used sigmoid activation function.
Finally in a paper [1] by Dauwe, A. and Goossens, B. and Luong, H. Q. and Philips, W. named A fast non-local image denoising algorithm describes that various and different denoising strategies have just been proposed in the previous many years, just to name a hardly any calculations: complete variety, two-sided channel or bit relapse, and wavelet-based methods. All of these techniques gauge the denoised pixel esteem dependent on the data gave in an encompassing neighborhood restricted window. The inspiration to create non-neighborhood strategies is to abuse comparative examples and structures in a picture. This moderately new class of denoising techniques starts from the non-nearby methods. Other denoising techniques dependent on this idea are created in 3d change area sifting and in a preparation based structure. Not at all like fractal-based strategies, had it misused the closeness of little fixes on a similar scale, spatially. On the surface repetitiveness, we can likewise locate this intermittent property in different pieces of the picture, for example, reiteration in various yet comparative articles, along edges and in uniform zones. The non-nearby technique is found completely reasonable to certain applications, a few models are denoising of checked content pictures and huge satellite pictures. The idea of redundant structures has likewise effectively been utilized in other picture handling errands, for example, introduction or super-goal furthermore, distinguishing advanced picture fraud. The non-nearby methods calculation produces best in class denoising results, nonetheless, the technique is computationally unrealistic because of the unreasonable measure of weight figuring between patches. The intricacy of the calculation is $O * (M^2)$ with M the complete number of pixels in the picture. This follows straightforwardly from the way that M loads must be figured for each pixel in the picture. To give some thought regarding calculation time: denoising a 512 by 512 picture on a P4 3.0 GHz with 1024 Mb slam keeps going around 5 hours and 20 minutes. A few papers in writing are committed to the speeding up of the non-nearby methods dependent on pre ordering neighborhoods with normal dim qualities and gradients1 and supplanting the mean squared deviation (MSD) computations with an effective added square picture conspire utilizing quick, Fourier change in a restricted inquiry window. Utilizing a restricted pursuit window based on the handled pixel can quicken the denoising cycle hugely, notwithstanding, in certain applications, we wish to misuse the full inquiry space, for instance, the entire picture or video grouping. Comparable articles can be found arbitrarily in the picture all things considered or pictures can comprise of surface in many parts, for instance, in material pictures. The non-neighborhood property of the calculation is incompletely disappeared due to the restricted pursuit window and is so related back to neighborhood denoising strategies. This likewise centers on the quickening of the non-nearby methods calculation, while remembering that the proposed procedures can likewise be applied for other picture preparing assignments.

Chapter 3

Dataset Collection and Preparation

For this project, we have to work with noisy Bangla License Plate images. But by finding only noisy Bangla License Plate images will not be enough to complete this project because for calculate our accuracy we have to true label the images and it is very hard to true label the images because of the noise in it. To add to this problem there is also a problem of image formatting. We have to collect the images and process it before it goes to the final detection algorithm.

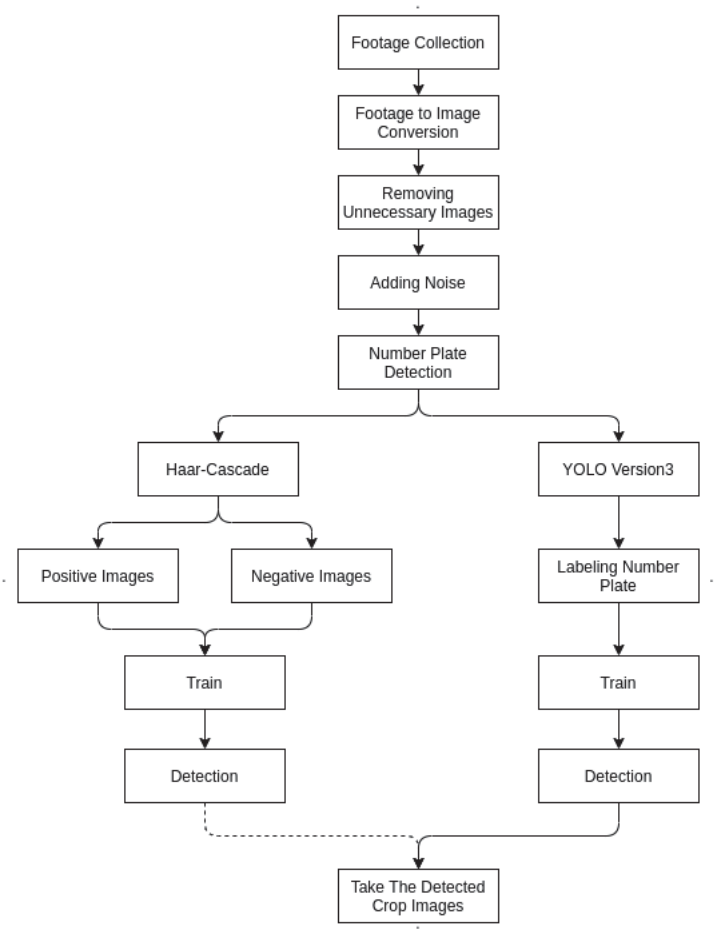


Figure 3.1: Data Collection and Preparation flowchart

3.1 Footage Collection

We take multiple videos in Rampura, Badda and Hatirjheel area of Dhaka. All the videos are captured in three days using multiple devices in daylight The videos were captured in different frame rates such as 12fps, 24 fps, 48 fps, and so on. Each of the clips was about 5 to 15 minutes on average. Summing up all our video clips it was about 4 hours.

3.2 Footage to Images

As we know, video or footage is a collection of simultaneous images. We need images to train our model. So we convert collected images from the videos that we took. To do so we use python code and extract the frames in different frame rates and save them. As we said earlier our videos were 5-15 minutes on average. So we use 1fps, 2fps and 4fps to collect the images. The total number of images was more than 20000.

Figure 3.2: Sample Images

Here are some examples from our image set, we can see that the first three images from the left have vehicles in those images but the last third on the left side does not have vehicles with visible number plates in the image. And we are not working on the green digital number plates which are used in the CNG(Compressed natural gas). That is why, to enhance the size and efficiency of our dataset we need to clean our image sets which is explained in the next part. One thing is most noticeable is our images are in different sizes because we used multiple devices to collect the video footage.

3.3 Removing Unnecessary Images

In video footage the camera captures continuous still images, and because of that we get some images with our desired object and some images that we do not exactly need for our data set. As we are working with object(number plate) localisation techniques we are expecting number plates in our images. But in our primary image set there can be seen, so many unnecessary images which we don't need and it also expands the size of the data set.

Figure 3.3: Unnecessary Images

Here in the example we can see that in some of these images the number plate cannot be seen clearly or some of them do not even have any number plates. So we are deducting these sorts of images from our dataset through these cleaning processes.

3.4 Adding Noise to Images

As we captured our reference video footage with a DSLR(Digital Single-lens Reflex) camera all of our images have high resolution and all the images are clean and clearly visible. But for this thesis we need a noisy image to work with. That is why we extracted clean images from the video footages and applied noise factors in different parameters such as 0.075, 0.1, 0.15, 0.2, 0.25, 0.3 using gaussian in random pixels of the images.

Figure 3.4: Noisy Images

3.5 Object(number plate) Localisation

Object(number plate) localisation is one of the most important part for our research. The main problem is that noisy images do not have proper edges and maximum object localisation algorithms use edges to detect an object. One thing to add here is our main objective is to detect the Bangla characters and digits from the footage. But we are going to detect the number plate first. Because, in a large frame, the number plate is very small. So the threshold value of the number plate is very small compared to the other objects in the frame. But if we can retreat the number plate from the picture and then start looking for the Bangla characters and digits from the number plate then it will give a good result compared to direct the characters and digits directly.

3.5.1 Haar-Cascade

Haar-Cascade is an old object localisation algorithm. Haar-Cascade gives good results in a known environment. The main reason for this is that this algorithm takes two kinds of images. First images are what the algorithm will detect. By convention those images are known as positive images. And the second ones are

what the algorithm should not detect. Also we can call those images negative images.

Haar-Cascade Algorithm

We apply this algorithm using four sub steps. First we select Haar features.

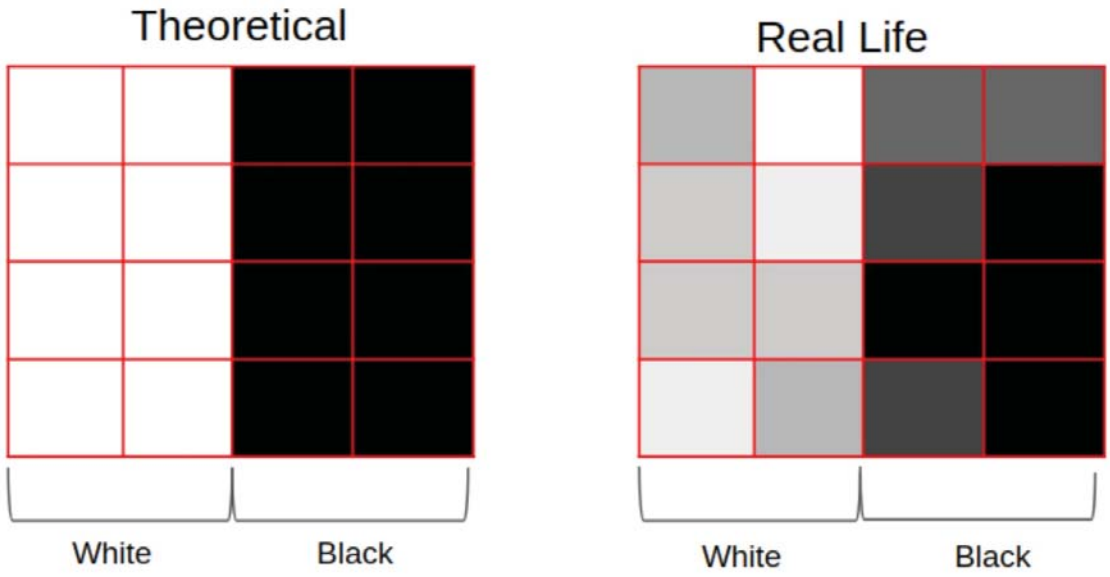


Figure 3.5: Theoretical vs Real Life Haar Features Problem

One In the fig we can see that, in theory Haar features detect the place where the image converted white to black but in real life this is not that scenario. In real life the edges are like the picture of the right side. So we have to calculate Haar features for that.

$$\Delta = Black - White = \frac{1}{n}\sum_{Black}^{n} I(x) - \frac{1}{n}\sum_{White}^{n} I(x) \tag{3.1}$$

Here, in the 3.1 equation we will calculate the mean of black pixels and the mean of white pixels. After that, the difference is Δ .
Now to apply this method in all the pixels of the images it will give us a time complexity of $\Omega(n^2)$. So, we converted the original image to an integral image. In i integral image pixel

$$x_{x,y} = \sum_{i=0}^{x}\sum_{j=0}^{y} x_{i,j} \tag{3.2}$$

Here, in the 3.2 equation for a pixel we are calculating all the pixel values summation from the left to top of that pixel including itself. And then replacing it with the original value. After doing that we use adaboost training to train our features.
Let a set of train example $D_n = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ as input and A strong classifier H comprised of a linear combination of a weak classifier $h_t(x_i)$.

Where, x_i the feature $\epsilon X.y_i$ is the class $\epsilon\{-1, +1\}$ and n is the number of elements. Now to train a haar-cascade we have to give it these two kinds of images. To do so, we cropped use images from our dataset. The cropped image which only contains the number plate or vehicle are the positive images (+1) and the image which has humans, roads, trees etc. are the negative images (-1). Then we used the cascade classification.

Creating Dataset for Haar-Cascade

To create the dataset for Haar-Cascade object localisation we created a total dataset of 1200 images. For the postivate image we used 400 images of the number plate. Here we mainly took the number plate of cars which has a white background and the character and digits are in black, there is also a black boardline in the number plate.

Figure 3.6: Positive Images

Here are some examples of positive images. We can see that there is no fixed image size. And we can also see the different kinds of qualities in the positive images.
We image the number plates of green colors. After creating the positive image data then created negative image data. Here we took 800 images because we took double the number of images in negative data. As haar-cascade gives good results in a known environment we took all the negative images from a single place.

Figure 3.7: Negative Images

Here we can see some photos from negative images. Basically those are the images which our algorithm should not detect. As we are giving the negative images from a particular environment it will create a good feature that we should avoid. In its 800 images we tried to include all the possible necessary objects that can be detected as a number plate.

Creating Classifier

To create a classifier, we first select a fixed image size where each image is 300 pixels wide and 200 pixels high. Then we applied the Haar-cascade algorithm to each image with an epoch of 20. After calculating the cascade Feature we save the result to apply in the further.

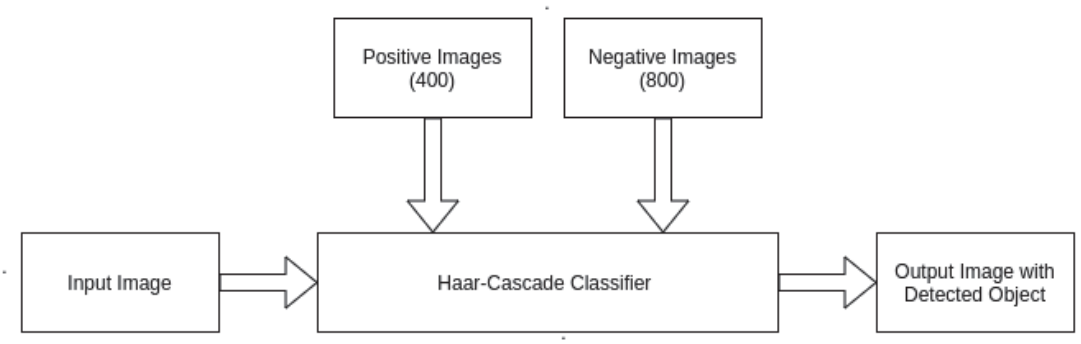


Figure 3.8: Basic work process of Haar-Cascade

Haar-Cascade Result

We applied 100 images from the known environment and 100 images from the unknown environment to our created cascade classifier.

Figure 3.9: Haar-Cascade Result

Here we can see the haar cascade algorithm is not giving good results in any of those images. In the known environment it is detecting the number plate but besides that the algorithm is also detecting false objects as a number plate. On the other hand, in an unknown environment haar cascade gives more bad results. Here our cascade classifier is unable to detect the number plate and also detect false objects as number plate.

	Known Environment		Unknown Environment	
	Positive	Negative	Positive	Negative
True	52	167	37	143
Flase	3	38	1	63

Table 3.1: Confusion Matrix of Haar Cascade Object localisation

From the confusion matrix we can see that from 100 images in a known environment it can detect 52 number plates but it also detect 167 false objects as number plates. And unable to detect 38 number plates. On the other hand in an unknown environment the false negative rate is higher. Where it is unable to detect any number plate in 63 images and also is detecting 143 as number plates which are not.
After analysing the result of haar-cascade we came to the conclusion that it is not a suitable algorithm for this project. So, we are not working with a haar-cascade algorithm for this project.

3.5.2 YOLO V3

Our basic goal is to detect the Bangla number plate from the noisy images. So we need to work with noisy images and detect the number plate successfully. For detecting number plate we used yoloV3. YoloV3 is one of the most strongest object localisation techniques. It uses CNN for object localisation. YOLO applies a single Neural Network to the whole image. This Neural Network divides images into regions and produces probabilities for every region. After that YOLO predicts number of Bounding Boxes that cover some regions on the image and chooses the best ones according to the probabilities. This consists of 53 convolutional layers that are also called Darknet-53. For detection yoloV3 uses 53 more layers that give us 106 layers of architecture. The detections are made at three layers: 82, 94 and 106. Each convolutional layer is followed by a batch normalization layer. There are no pooling layers, but instead, additional convolutional layers with stride 2, are used to down-sample because the use of additional convolutional layers to down-sample feature maps prevents loss of low-level features that pooling layer just exclude. As a result, capturing low-level features helped to improve ability for detection of small objects. YoloV3 makes detection at three different scales at three separate places in the Network. These separate places for detections are layers 82, 94 and 106. Network down samples input image by following factors: 32, 16 and 8 at those separate places of the Network accordingly. These three numbers are called stride of the network and they show how the output at three separate places in the Network is smaller than input to the Network. It takes input as an A X A grid and detects the object where they have two main intermediate layers with grid and colored image with too many bounding boxes. The layer with grid have confidence score. If confidence score is low, bounding boxes are eliminated.

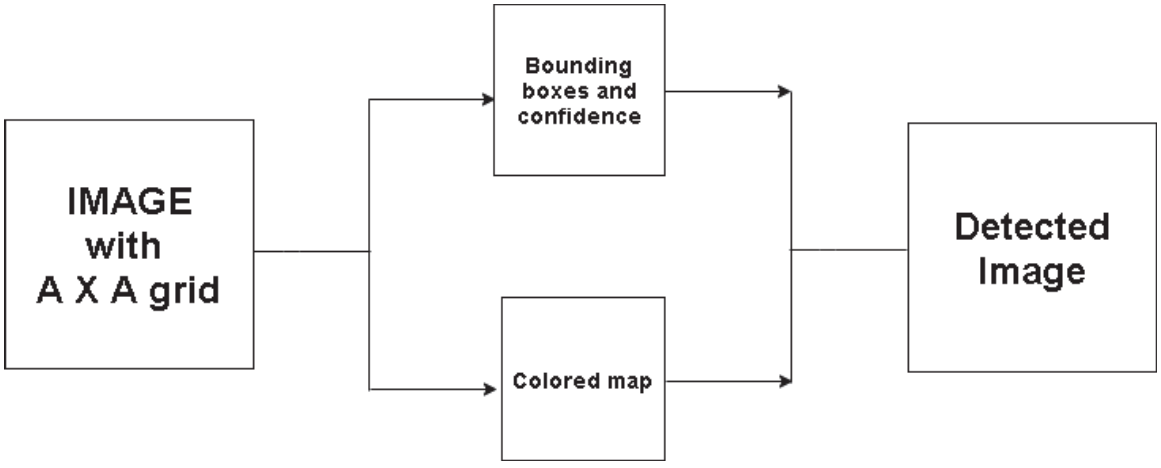


Figure 3.10: Basic YoloV3 architecture

Our image size is 1100x2000 when we detect the image from the noisy image. For YOLOV3, it divides the image into a 13x13 grid of cells. For our model we take our input image size as 416x416. So our cell size was 32x32. Each 32x32 cell is responsible for predicting the number of boxes in the image. In the feature map we have (B x (5 + C)) entities where B is the number of bounding boxes that each cell can predict.

Dataset for Yolo V3

In our dataset we proceed 1200 images for object localisation. As we took the images of speedy vehicles the data was noisy itself. As this work contribution is mainly detection from noisy data so for making it more noisy we add 0.1-0.3 noises. We take 20 percent for testing and 80 percent for training. So use 960 images to train our model and test with 240 images. We labelled the images before training. We used LabelImg for labelling the license plate where we labelled the number plates from the vehicles.

Figure 3.11: Labeled Image

So we labelled these manually using software and saved as a XML file for further processing. Then we trained our model with 960 labelled images.

Detection Rate

Our result was satisfactory for object localisation. We get pretty good accuracy. We had 98.3 percent accuracy on train data and 97.1 on test data for object(number plate) localisation where we used 100 epochs (0 to 99). After the learning phase of the same epoch we calculate the validation loss. Our model validation loss is 0.86 for train data and 0.87 for test data. For step per epoch we use one-fourth of the trained dataset which is 240 images.

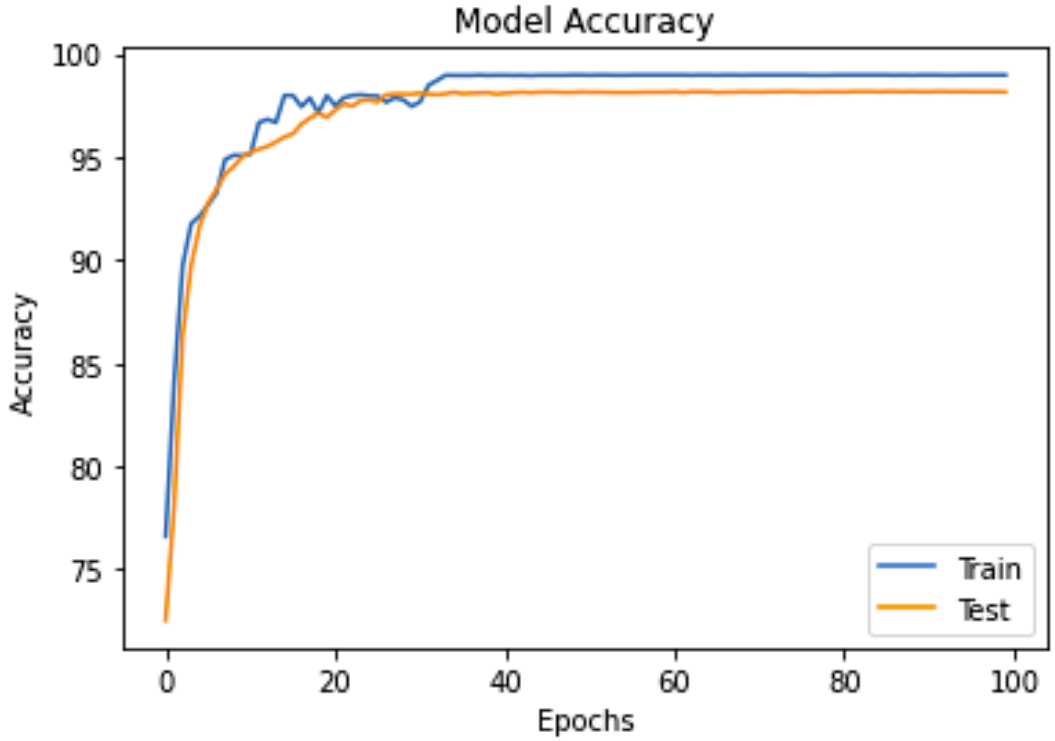


Figure 3.12: Model Accuracy Graph

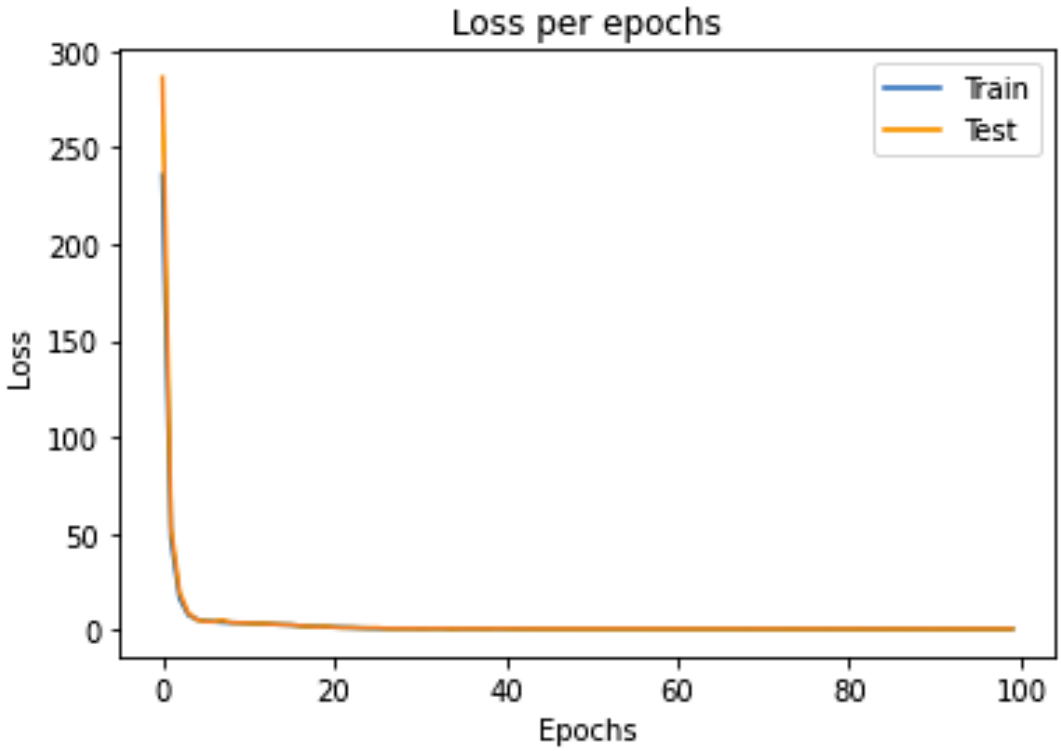


Figure 3.13: Loss Graph

Our model's validation loss graph is a good fitted learning curve as we get minimal

distance between training and testing loss values. The generalization gap is minimal as we can see in Figure 3.13. We can observe our model accuracy in Figure 3.12. Since the enhancements in our model to the train set gazes fairly coordinated upward with upgrades to the approval set, it doesn't appear as though overfitting can be a big issue in our model.

	Known Environment		Unknown Environment	
	Positive	Negative	Positive	Negative
True	100	5	98	9
Flase	0	0	0	2

Table 3.2: Confusion Matrix of Yolo V3 object(number plate) Localisation

In classification we consider true positive and true negative when we get the result as expected that means in true positive the model predicts the positive class correctly and true negative is when the model predicts the negative class correctly. We get 100 true positive and 5 true negative images for the known environment and for the unknown environment it drops a little bit but still we get good results where we get 98 true positive and 9 true negative. False positive and false negative is when our model predicts incorrectly. Our model does not predict any positive and negative class incorrectly in a known environment. For unknown environments our model does not predict any positive class incorrectly but it gets 2 false negatives.

Figure 3.14: Detection in an Unknown Environment.

Figure 3.15: Detection in a Known Environment

In Figure 3.14 we captured the detection for unknown environments as the pictures in this process are from a different location and environment. Figure 3.15 is for known environments which we have already seen in our confusion matrix table.From this we made our customized dataset for bengali number plate to work on our proposed model.

Chapter 4

Methodology

After collecting the number plates we started to pre process our number plate images for a better result. To do so we use 9 types of different denoising algorithms to improve the image quality. After using those filtering algorithms we analyzed our dataset and chose the best ones to work on the next step of our research. In the next step we label the images according to the word, character and digits. After that we applied the created pre processed data on three different detection algorithms to detect the Bangla character, digits and words from the number plate. Here in

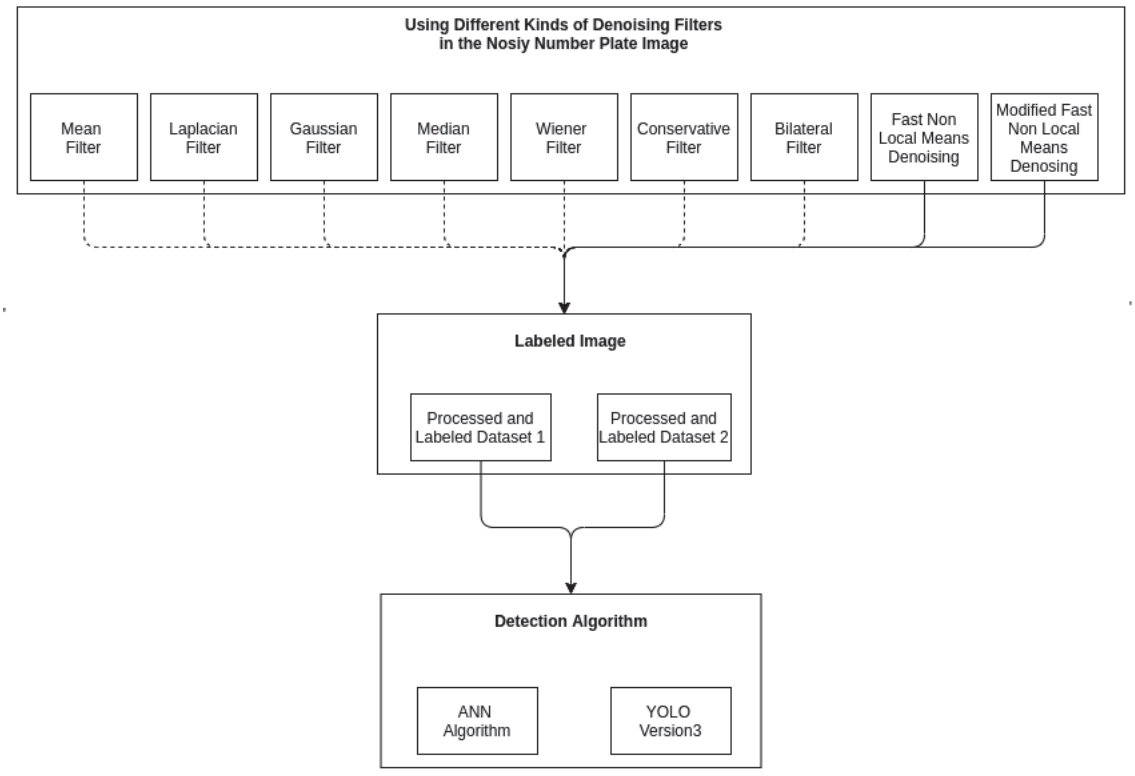


Figure 4.1: System Model for Bangla Number Plate Detection

Figure 4.1 we can see how we are going to work with the collected number plate from Chapter 3. But before going to that the collected number plates are in different shapes and some are so noisy it can not be labeled. So to improve these situations

we first resize all the images. We take 512 in the wide and 256 as the height for each image. Here the image wide is double the the the image height because the ratio of an Bangladeshi number plate is also 2:1. After resizing all the images then we also add some additional noise. Here we add 0.3 factor noise in each image. After that we use those different denoising algorithms.

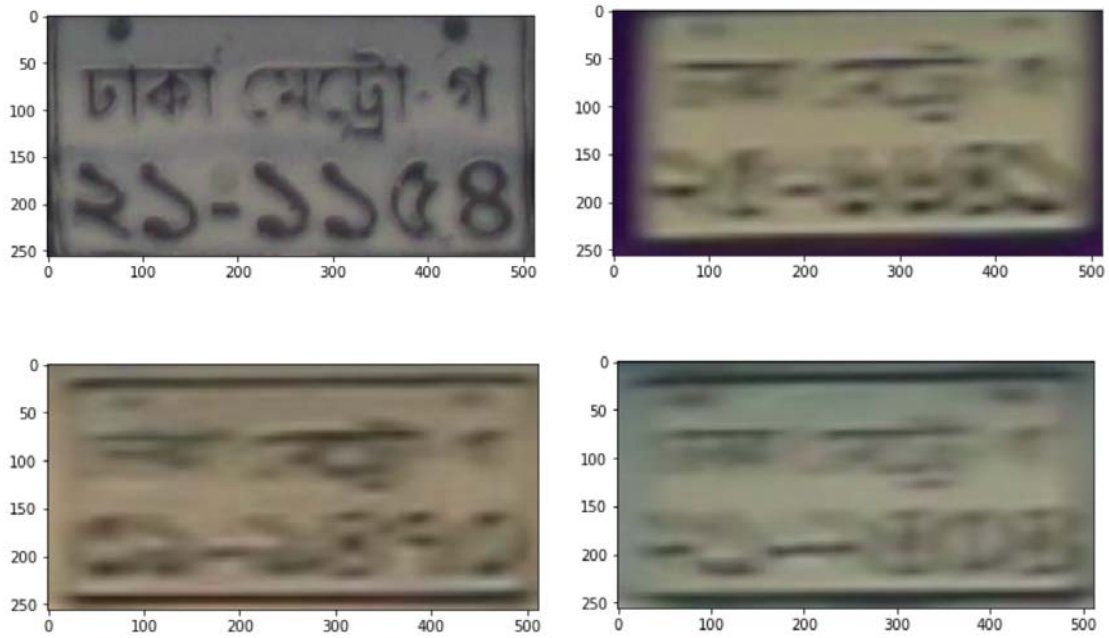


Figure 4.2: 512/256 Images with 0.3 Fator of Noise

4.1 Filter

Now we are going to use different kinds of denoising filtering algorithms which will help the detection algorithm to detect those characters and images. Our main target is to work with color images but we are also considering black and white images if the denoising filter gives an extraordinary result. Also in bangla number plater there is only black and white color. But as we are accurately working on video footage then it will be a backdraw for us.

4.1.1 Mean Filter

The mean filter uses mean value to the pixels to reduce the noise. For the mean value of a pixel, it takes the 8 neighborhood values (for the edge it will be 3 and 6) and it's own value of that pixel then calculates the mean of those values. After that replace the original value with this mean value. In this way, it applies this method to all the pixels of the image.

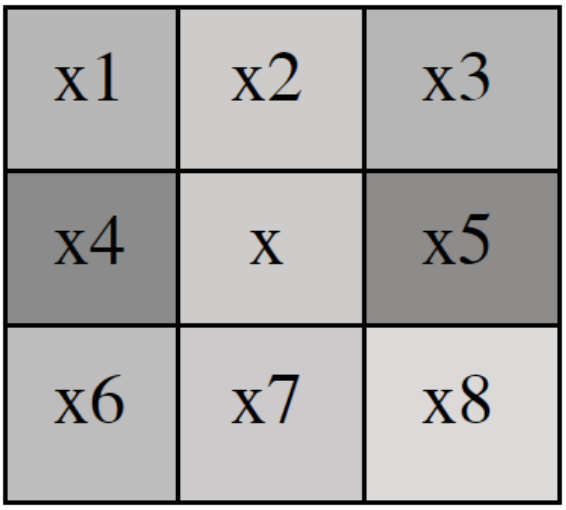


Figure 4.3: Pixels x and it's neighborhood 8 values

So here the new value of x will be,

$$x = (x + \sum_{i=1}^{8} x_i)/9 \tag{4.1}$$

But there are some differ cases in an image.

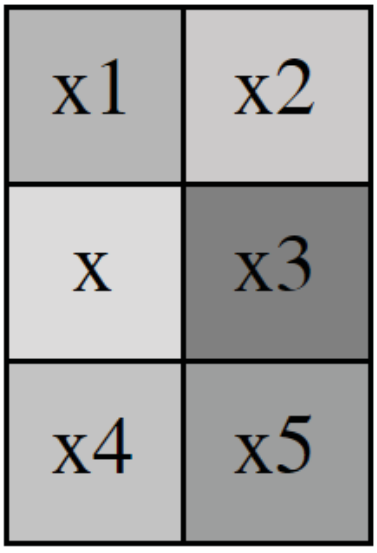


Figure 4.4: Pixels x and it's neighborhood 6 values

Such as, in Figure 4.4 we can see that the pixels are on the one side of the image. So in this case there are only 5 neighbourhood pixels of x.So here,

$$x = (x + \sum_{i=1}^{5} x_i)/6 \tag{4.2}$$

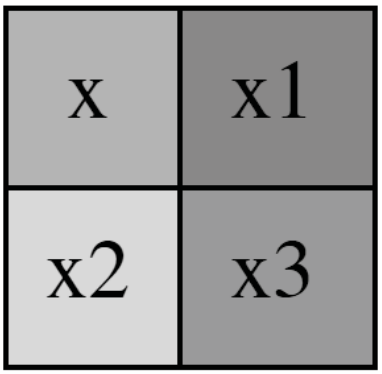


Figure 4.5: Pixels x and it's neighborhood 4 values

Also if the pixels in the connar have 3 neighbourhoods of that pixel. So for that conner pixel the value of x will be,

$$x = (x + \sum_{i=1}^{3} x_i)/6 \tag{4.3}$$

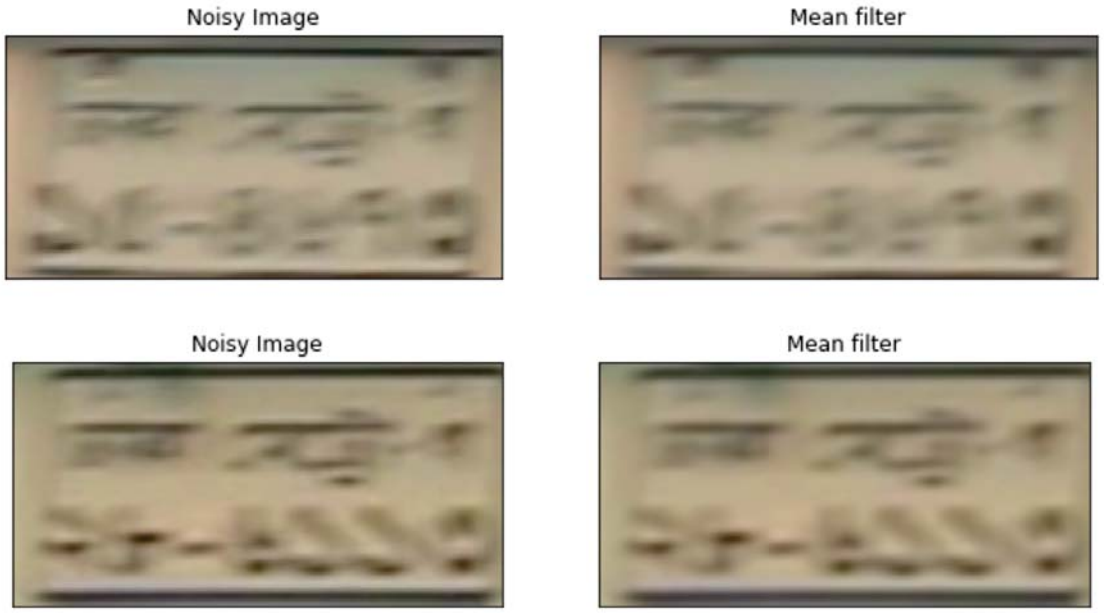


Figure 4.6: Noisy Image as input and Mean filter as output

Here we can see that the mean filter first converts the RGB image into grayscale image. After converting the mean filter applied it's algorithm to every pixel of the image. As a result we can see that the mean filter makes the images more smooth than normal. But it is also distring the edges from the work, character and digits. If the image is more light or the RGB value too high then it will make the whole image white color because maximum pixels are white in color because white has the highest RGB value. So we can not use the mean filter algorithm in our research because it is distorting the characters and digits from the images and our goal is to detect those characters and digits perfectly.

4.1.2 Gaussian Filter

Gaussian filters are also like mean filters. But the gaussian filter used a weighted average of the surrounding pixel. Standard deviation (σ)is used to calculate the changing average of the values. By using this standard deviation and surrounding pixel gaussian filter calculate kernel which represents the district gaussian distribution for the image.Here gaussian filters kernel will be, k=6σ-1 And here standard deviation,

$$\sigma = \sqrt{(n_1^2 + ... + n_m^2 - m)/12} \tag{4.4}$$

in this equation n represents the number of pixels and m is the moving average.

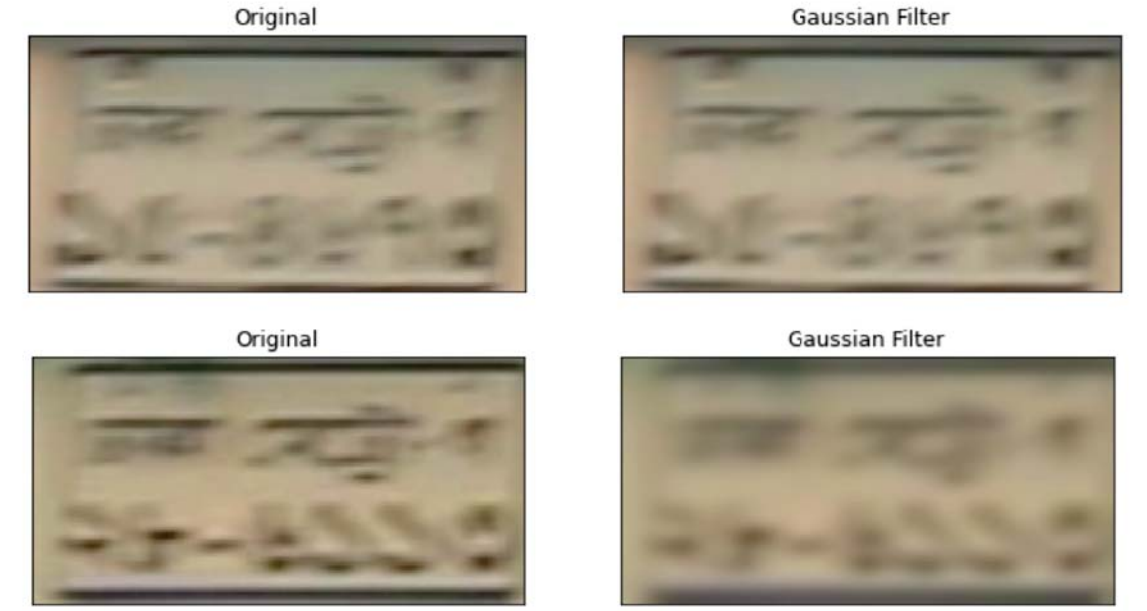


Figure 4.7: Using Gaussian filter in Grayscale noisy image

Here we first used a gaussian filter on black and white image. Here the gaussian filter is able to remove the noise from the image. The images are more smooth then before. but we can see that here in the second images the dark part becomes lighter. So it is removing the edges from the images. Also it will be bad for the dataset because there are lots of noises in the dataset images. So the edge removing will be more than gaussian filter. Also we do not want to work with gray scale images if they do not give any extra ordinary result.

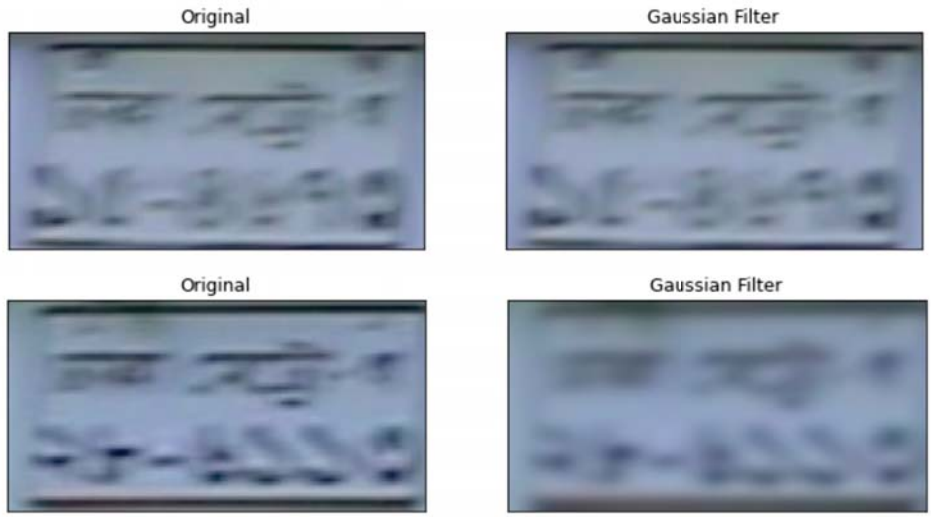


Figure 4.8: Using Gaussian filter in RGB noisy image

Again after applying gaussian filters in color images, we can see that the filter color images are good compared to the gray scale images in terms of maintaining the true color of the object. But if we compare it to the original image then we can see that it is not that much different from the gray scale gaussian filter. Because it is removing the rafness from the image and making it smoother but it is distorting the color of the digit and characters from it. So, it is also not preferable for us to work with. So we can not consider this filtering algorithm which will help us in detecting characters and digits from noisy images.

4.1.3 Median Filter

Median filters are also like mean filters. But the only difference is that it calculates the median value from the surrounding pixels and replaces it with the median value. Now let assume there for a pixel x there are 8 other pixels surrounding it (Figure 4.3). So our median function is going to sort these values and as there are 9 pixels in total it will take the 5th largest value. Let there is a sorted set of 9 pixels,

$$S = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9\} \tag{4.5}$$

Here, S is sorted. So no matter it is ascending and descending order x_5 is the median value of those. So, the filter will replace x with x_5.
And if the pixel is in one side of the image(Figure 4.4) then ,

$$S = \{x_1, x_2, x_3, x_4, x_5, x_6\} \tag{4.6}$$

Here, $x_3$or $x_4$will be median value in the sorted list S. So, $x_3$or x_4 will replace the pixel value. Also if the pixel is in a corner (Figure 4.5) then,

$$S = \{x_1, x_2, x_3, x_4\} \tag{4.7}$$

Where, x_2 or x_3 will be the median value. So, the filter will replace x with $x_2$or x_3.

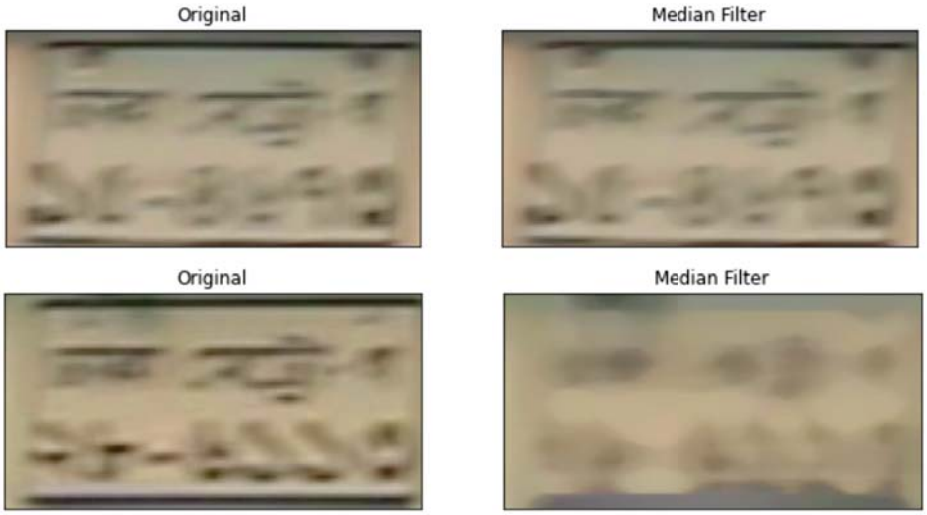


Figure 4.9: Using Median filter in Grayscale noisy image

Here first we convert the RGB image into a gray scale image where each pixel has a value of 0 to 255. Here 0 means the dark value a pixel can have. And by increasing the value of the pixels leads to a whitish color. Here 255 means pure white color. So, here we just start with a pixel and take the neighbourhood pixels value. After that we sort those values in ascending order after that we took the meaning of those. That means in the list of sorted neighbourhoods we took the value which has the same amount of value to it's write and left. One thing is to add here that, we can see that on the second photo it is completely distorted the black color of that image. The main reason is when we took the value of neighbourhood pixels in and took the median one from the sorted list in the edges of a dark pixel it has some white color pixels and when it is converting to replace the value it is not considered the or the neighborhood. As a reason maximum black pixels are replaced by the with pixels. So this filtering method will not be helpful for our research.

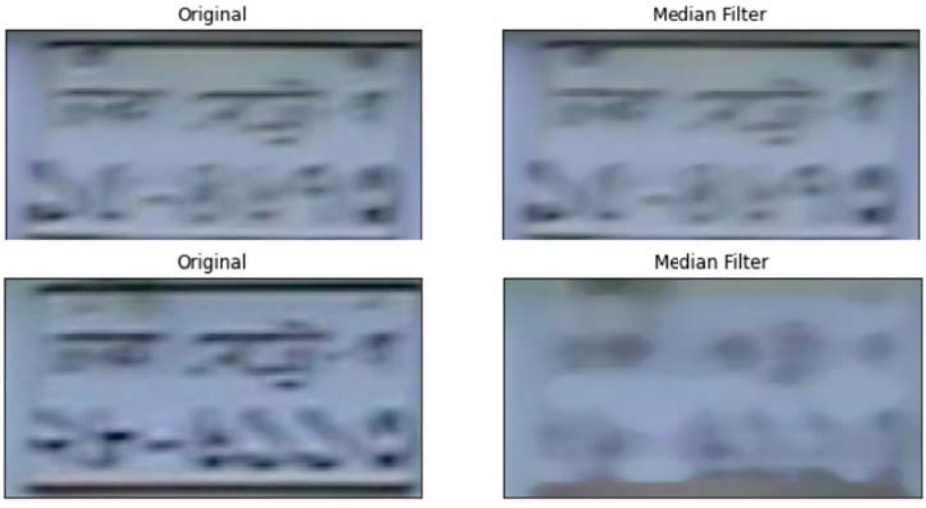


Figure 4.10: Using Median filter in RGB noisy image

Here in Figure 4.10 we used the median filter on color images. Here we can see the same problem as in the gray scale. The Bangla digits, words and character got totally destroyed due to the median color filter. Here the RGB values are replaced in the same process as the gray scale; the only difference is that this replacement is done three times. One for red, one for green and one for blue. But after analyzing the results we are also not considering median color denoising image to apply on the images.

4.1.4 Conservative Filter

Conservation filter is an edge preserving filter. Here the edges are not destroyed. So The characters of the Bangla words, digits and characters will be safe. Let's take "img" as the image name. We also resize all the images 512/256. So here the size will be the same. So the conservative filter algorithm will work as,

Algorithm 1 Conservative Filter Algorithm

$img \leftarrow Input\ Image$ {Take image as input in img.}
$img[a, b] \leftarrow pixels$ {Take an unvisited pixels (a,b) from the img}
while $unvisitedpixels$ **do**
  if $img[a, b]$ in corner **then**
    $array = [x_1, x_2, x_3]$
  else if $img[a, b]$ in one side **then**
    $array = [x_1, x_2, x_3, x_4, x_5]$
  else
    $array = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9]$
  end if
  Sort$(array)$
  $xMax$=$array[lastIndex]$
  $xMin$=$array[firstIndex]$
  x=$img[a, b]$
  if $x > xMax$ **then**
    $x > xMax$
  else $\{x < xMin\}$
    $x < xMin$
  end if
  $img[a, b]$= Next Pixel
end while

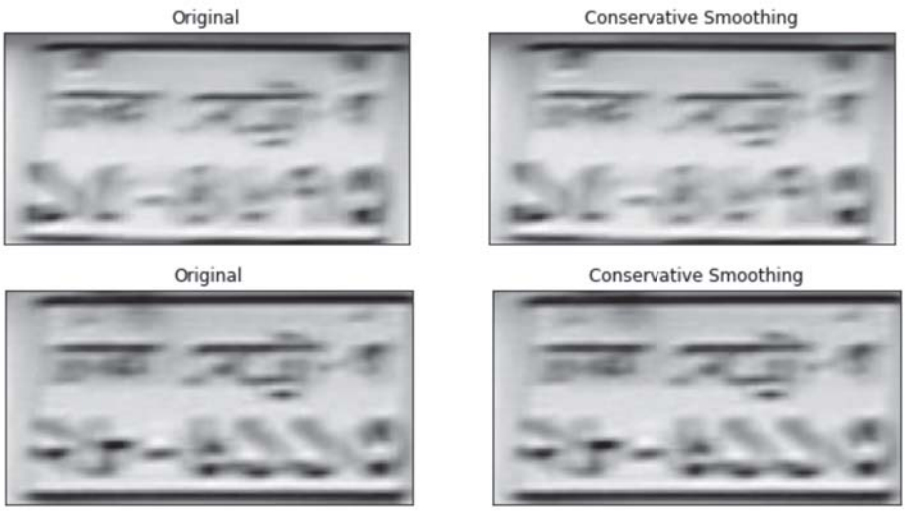


Figure 4.11: Using Conservation filter in Grayscale noisy image

Here, in Figure 4.11 we first convert the RGB image into gray scale image. After that we applied the conservation filter algorithm on that gray scale image. We can see here the conservative filter is not destroying any of the edges of the digits, characters or words from the number plate. Which means that this algorithm will be able to help the detection algorithm to detect the digits and characters. But there is one problem in this filtering system. This algorithm works on gray scale images. As we said earlier we are trying to work on color image only. We will only use a gray scale filter if it gives extraordinary results. So we are not considering this filtering technique.

4.1.5 Laplacian Filter

The Laplacian filter is mainly used to detect the edges from a noise image. Laplacian filter basically highlight the pixels where the value is dramatically changed compared to the neighborhood pixels. Let amuse $I(x, y)$ is the intensity of the values among the pixels. So laplacian will be,

$$L(x, y) = \frac{d^2 I}{dx^2} + \frac{d^2 I}{dy^2} \tag{4.8}$$

Here the laplacian equation is used. We can see here we are taking the 2nd order derivative which means that it is calculating the change ratios of the images. After that when there is a dramatic change in any of the pixel values compared to nearby pixels then it will highlight that particular pixel. Also if there is not much change in the pxials then the algorithm will make those pixels the same value.

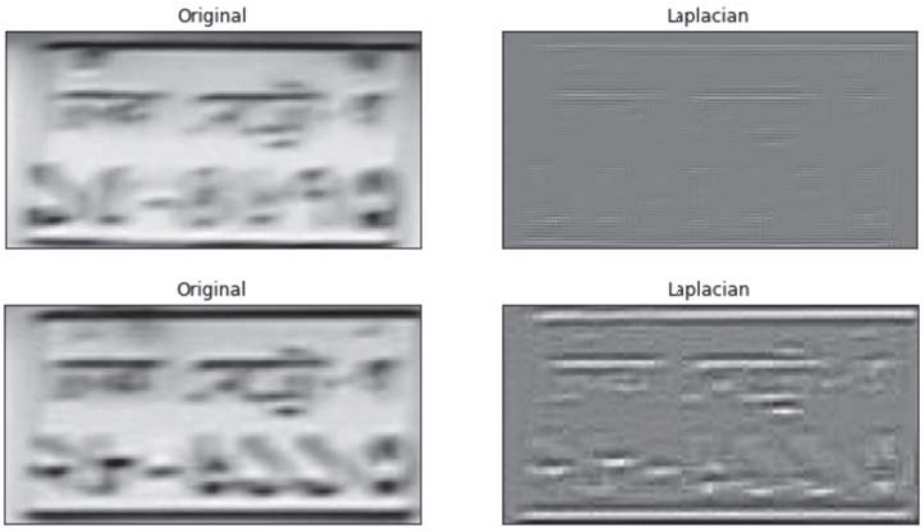


Figure 4.12: Using Laplacian filter in Grayscale noisy image

Here in Figure 4.12 we can see that when we use a laplacian filter in the images it destroys the whole noisy image because in our images the noise margin is very high and the edges are not clear. So the algorithm did not find any dramatical difference in the nearby pixels for any particular pixel. So it did not highlight any big part of the image. Because of this reason we are not considering the laplacian filter for our research.

4.1.6 Wiener Filter

A Wiener filter gives good results when the shutter speed of the camera is slow or the camera is moving or the object in the frame is moving. Also wiener filter assumes that the noise is in a second order stationary. So for pixel (x,y) the wiener function will be, G(x,y)= F(x,y) H(x,y) Here F fourier transformation and H is the blurring function for (x,y).

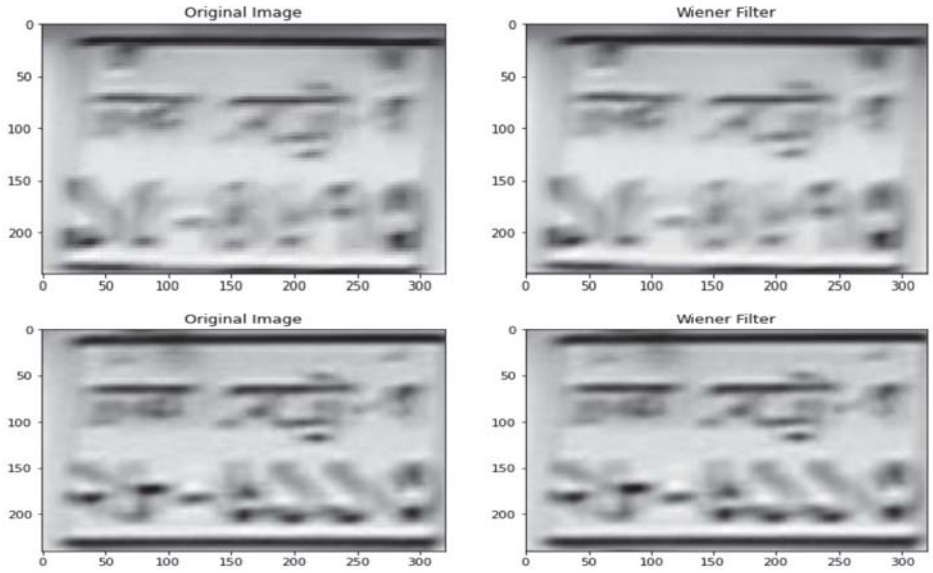


Figure 4.13: Using Wiener filter in Grayscale noisy image

Here in Figure 4.13 we applied wiener filter by converting the image into grayscale. We also reshape the image. Here the image is 320 by 250. We can see here that here the digits are more real than any other algorithms that we used so far. But because it works on a grayscale we are trying to avoid this filter also.

4.1.7 Bilateral Filter

Bilateral filter is an updated version of Gaussian filters. But unlike Gaussian filter bilateral filter is an edge preserving filter. Bilateral filter removes noise from an image besides that this filter does not let affect the edges. In fact it sharps edges more. Bilateral filter allows us to use it on color images. Let's assume x is a pixel in an image. Gaussian filter considers only the neighbourhood pixels. But in bilateral filters we can choose the rang or the neighbourhood boundary.
Let's take a 7 by 7 matrix as a neighbourhood where x is in the middle.

x1	x2	x3	x4	x5	x6	x7
x8	x9	x10	x11	x12	x13	x14
x15	x16	x17	x18	x19	x20	x21
x22	x23	x24	X	x25	x26	x27
x28	x29	x30	x31	x32	x33	x34
x35	x36	x37	x38	x39	x40	x41
x42	x43	x44	x45	x46	x47	x48

Figure 4.14: 7 by 7 Pixels Matrix

Now, like Gaussian filters it will consider the neighbourhood pixels value. Here the closer the pixel is more the weight of that pixel will be. Such as, from the Figure 4.14 we can see that the x_1 pixel is much far away from the pixel in x_{24}. So the weight of the x_{24} will be much higher than the weight of x_1. It implies for that the neighborhood pixels. So, we can say that, the most close eight neighborhood pixels will carry the highest weight. So to calculate the average those pixels will play the most significant rule. But one thing we can notice is that x_{48} has the same value as x but x_{11} has another value. But if we compare it by the distance then we can see

that x_{11} is compared to close then x_{48}. So x_{11} has a higher weight which will play a problematic role for x. To deal with this kind of problem in the Bilateral filter there is another feature called photo-metric distance. Here in photo-metric distance the weight of the pixels are decided by the similarity of RGB values in the pixels. So the pixel x_{15} to x_{48} has the same value or closest RGB value of x then x_1 to x_{14}. So no matter how far the pixel is x_{15} to x_{48} will carry more weights than other pixels in the photo-metric distance domain. After combining the Gaussian distance domain and photo-metric distance domain the result for x value will be replaced with the old value of x. Basically for Gaussian distance it is able to remove the noise from the image and for photo-metric distance it is able to preserve the edges.

Figure 4.15: Using Bilateral Filter in Color Images

Here in Figure 4.15 we used bilateral filter on the images. Here we took a 15 by 15 matrix as the neighbourhood domain. After that bilateral calculate the Gaussian distance for every pixel and give them a weight according to the distance and after that it calculates the photo-metric distance. The one thing is to notice that the pixel value is RGB. so it also calculates this distance in three terms individually. Because of that reason the algorithm is a little bit slow. After calculating the photo-metric distances it also assigns a weight to all the neighbourhood pixels. After that it combines the two weights and changes the value of the pixels according to the weight of the nearby pixels. So we can see the done a good job in edge preserving but also it is not able to make the image snooker then original.

4.1.8 Non Local Mean Denoising Filter

Non local mean denoising filter is a popular denoising filter which is used in medical sectors such as MRI. We are also using this filter because our images are color images but their are basically white and black color in it same as maximum medical images. Let's take an image as input. And the image has 512*256 pixels total. So the non local mean denoising algorithm will be,

Algorithm 2 Non Local Mean Denoising Algorithm

$img \leftarrow input\ image$ {Take image as input in img.}
$neighbour \leftarrow input\ integer$ {Take integer as input in neighbour.}
$matrix \leftarrow size(neighbour * neighbour)$
$pixel \leftarrow first\ pixel\ of\ image$
while *image has unvisited pixel* **do**
  $matrix[neighbour/2, neighbour/2] = pixel[red, green, blue]$
  $matrix \leftarrow other\ neighbourhood\ of\ pixel\ as\ in\ image$
  $divide\ image \leftarrow neighbor * neighborSizeSegments$
  while *unvisited segments* **do**
    Calculate photometric distance for all pixels in segment
    Compare to matrix and segment photometric distance of pixel coordinate wise
    if every pixels photometric distance is close **then**
      assign weight for those pixels
    end if
    $newMatrix{=}size(neighbor * neighbor)$
    while *unvisiter weighted segment* **do**
      while *unvisited cell in newMmatrix* **do**
        $newMatrix[cell]{=}newMatrix[cell] + segment[cell]$
      end while
    end while
    $newMatrix{=}newMatrix/(totalweightedsegment)$
    while *unvisited cell in newMatrix* **do**
      $newPixel{=}newPixel + newMatrix[cell]$
    end while
  end while
  $pixel{=}newPiexl/(neighbor * neighbor)$
end while

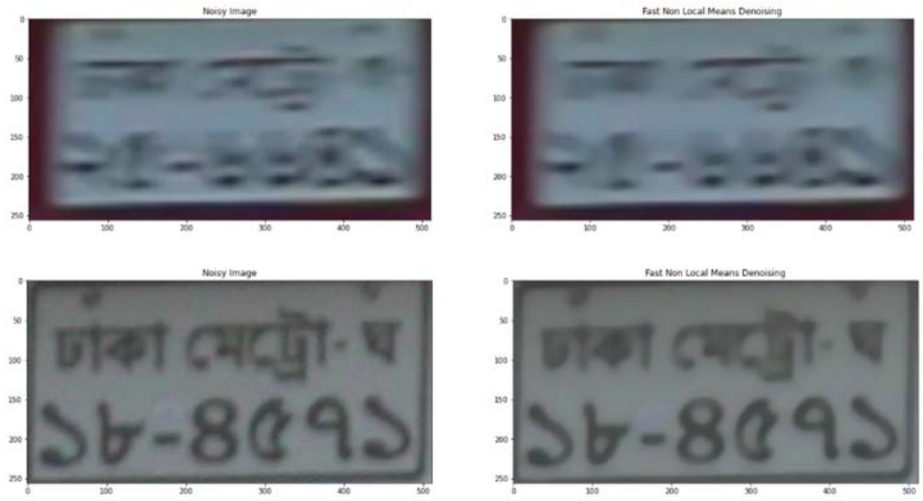


Figure 4.16: Using Non Local Mean Denoising Filter in Color Images

Here in Figure 4.16 we use the NLMD filter in those colour images. Here we take 15*15 as the matrix size. Then the algorithm takes a pixel then considers that pixel as the middle value of the neighborhood matrix. After that it takes other values from the neighborhood pixels of that pixel. One thing is to method here is that matrix maintains the sequences of the neighbourhoods. After that the algorithm divided the image into 15 by 15 segments. Then the algorithm checks every segment and checks the values similarity of the segment with the matrix. So, we can say that if the pixels are in the white position of the image then there is a high chance that the matrix will have the white pixels values. So when the algorithm is taking the segment it will take for the segments that have white pixels value. Again if there is a white pixel as a nosing in the middle if the character then the matrix will have mostly the black pixel values. So the algorithm will take mostly the segments with black values as consideration and assign weight to those pixels of that segment. After that the algorithm will collect all the segments that got weight and calculate the mean value of each coordinates segment then again calculate the mean of those mean value and finally replace it with the pixel. So, we can clearly see that this algorithm is removing nosing along with preserving the edges and the digits are more clear the noisy image. Also a plus point is that this filter can be used in RGB scale. So we are considering this filter for the next step of our research.

4.1.9 Modified Non Local Mean Denoising Filter

We also modified non local mean denoising filters. Our main goal is to apply this modified filter on our dataset to get a better result than any other filter can give us. To create this filter we first analyze our dataset images. Then we analyze every digit and character close and start to find some similarities and the patterns in the front of Bangladeshi number plate. Based on our assumption we created this modified filter. Let's take an image as input. And the image has a height of 256 and wide of 512. So the our modified non local mean denoising algorithm will be,

Algorithm 3 Modified Non Local Mean Denoising Algorithm

```
img ← input image {Take image as input in img.}
neighbour ← input integer {Take integer as input in neighbour.}
segment ← image[301 <= height <= 400 and 101 <= wide <= 200]
while segment has unvisited pixel do
    x = pixel[red, green, blue]
    matrix ← other neighbourhood of pixel as in image
    array[newIndex] = ((x[0] + x[1] + x[2])/3)
end while
Sort(array)
upperLimit=array[150]
lowerLimit=array[0]
while image has unvisited pixel do
    x = pixel[Red, Green, Blue]
    avg = (x[0] + x[1] + x[2])/3
    if avg <= upperlimet and avg >= lowerlimet then
        pixel RGB value = pixel RGB value/2
    end if
end while
matrix = neighbour * neighbour
pixel = first pixel of image
while image has unvisited pixel do
    matrix[neighbor/2, neighbor/2] = pixel[Red, Green, Blue]
    matrix = other neighbourhood of pixel as in image
    Divided image into neighbor*neighbor size segments
    while unvisited segments do
        Calculate photometric distance for all pixels in segment
        Compare to matrix and segment photometric distance of pixel coordinate
        wise
        if every pixels photometric distance is close then
            assign weight for those pixels
        end if
    end while
    newMatrix = size(neighbour * neighbour)
    while unvisiter weighted segment do
        while unvisited cell in newMatrix do
            newMatrix[cell] = newMatrix[cell] + segment[cell]
        end while
    end while
    newMatrix = newMatrix/(total weighted segment)
    while unvisited cell in newMatrix do
        newPixel = newPixel + newMatrix[cell]
    end while
    pixel = newPiexl/(neighbour * neighbour)
end while
```

Figure 4.17: Using Modified Non Local Mean Denoising Filter in color image

Here in Figure 4.17 we used our own modified algorithm. Which is edge preserving and removes the noise from the image. It has a quince fracture. In the Bangla number plate the digit and characters are written in black. But there is some place in every character and digits that has more black color on it then the other side of that character. We can see the highlighted dark place in the fig 4.14. So we used this to make the places darker. To do this we just take a place in the image where we are sure to find a digit. For our data set the place is wide 101 to 200 and height 301 to 400. Then we took the average of the RBG value for each pixel. After that we sorted the values in ascending order. As we know that RGB value =[0, 0, 0] means most dark. So then we take all the pixels of the image and for each pixel we check whether the pixel's RGB value average is less than or equal to the 150th and greater than or equal to the 1st value of the sorted list. If it is then we just make the RGB value half. In this way we are making the dark places darker. Besides that the non local mean filtering is also applied on the images but here as some dark pixels are boosted so those are those value got separated will calculating the photomatix similarity of those value. As this filter is creating some unique feature for every digits and character so amour assumption is that those feature will help the deep learning algorithm to detect Bangla characters and digits from number plate.

4.2 Data Pre Processing

In chapter 4.1.1 to 4.1.8 we discussed and analyzed 8 different kinds of denoising algorithms. Where we find Non Local Mean Denoising filter the best one. Beside that after analysing our dataset we also modified Non Local Mean Denoising filter.

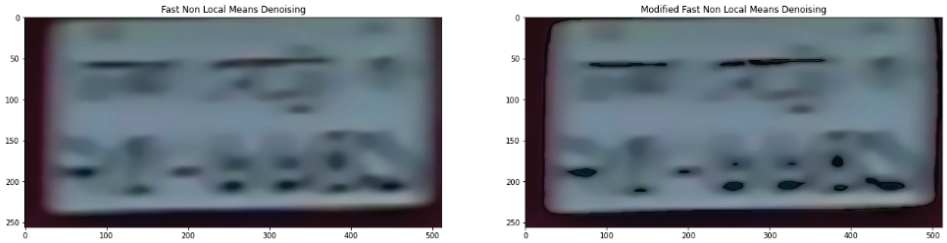


Figure 4.18: Fast Non Local Mean Denoising and Modified Fast Non Local Mean Denoising

Here we decided to go with both of the algorithms. We will compare the results of those modified filters with the existing one and analyze the result. So we applied those two filter individually in every image of our dataset.

4.2.1 Filter Result Comparison

Following is an in depth comparison between the two denoising filtration methods. The one is Fast non local means denoising method and another one is Modified Fast non local means denoising method. Talking about these two filters the Fast non local means denoising filter is an image denoiser which works to make hazy and blurry image more clear to read and understand its inside content. But there is another denoising method which is Modified fast non local means denoising method which gets the job done with higher accuracy and gives sharper and an image with more prominent edge.

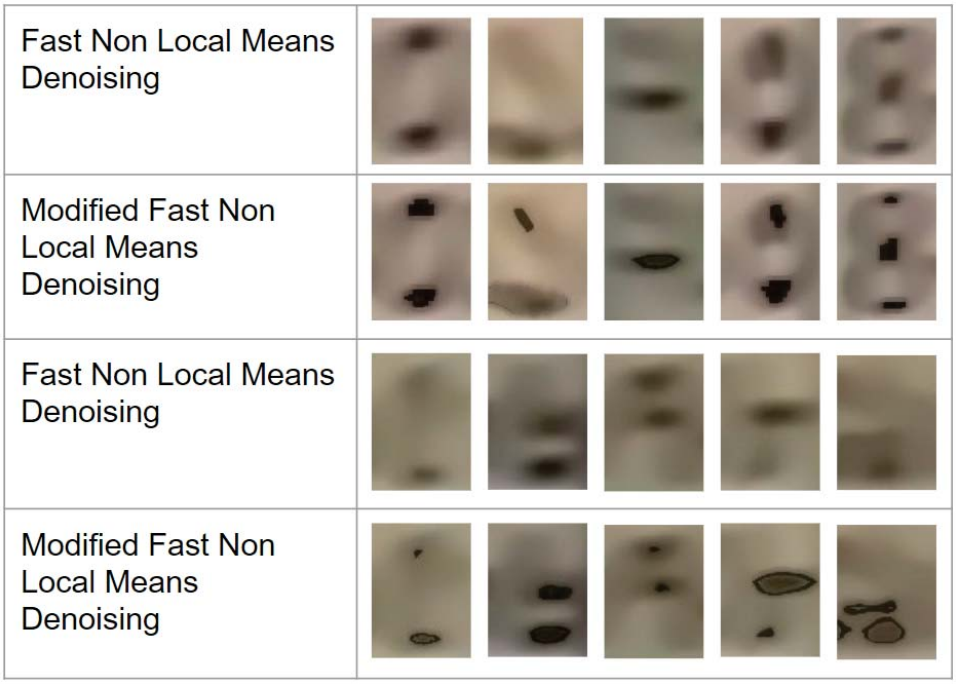


Figure 4.19: Comparison table between two methods

If we are to talk about individual number character detection process here we can see that, in terms of the character 0 it can be seen that in the Fast non local means

denoising method there is no significant upper and lower edge visible whereas in terms of the Modified fast non local means denoising method the upper and the lower edge of 0 is more prominent and sharper. The character 0 can easily be separated from other characters because the other characters do not have the same kind of prominent dark portions in the exact same area like the character 0. Now talking about the character 1, in case of Fast non local means denoising method the character do not have any distinguishable characteristics by which it can be recognized with just a glance because the image here do not have any sharp edge and also it is sometimes also confusing with the Bangla character 9. Because of that this character 1 should hold a proper individual attribute by which it can be easily identifiable. In terms of the other denoising method which is the Modified fast non local means denoising method the character 1 has a prominent dark line just about the diagonal leaning on the left portion of the picture moreover the bottom portion of this character is cleared so that it can create an effortless difference between the character 1 and 9. So it is clear that in terms of this character 1 the Modified fast non local means denoising method is clearly ahead of the Fast non local means denoising method. Moving onto the next character which is 2, the Fast non local means denoising method has more blurry and edgeless confusing picture than the Modified fast non local means denoising method ones. Because in the Modified fast non local means denoising method in the middle of the character 2 there is a dark horizontal dash can be seen which is a distinguishable attribute of the character 2 and these important details has made the detection
and recognition of these characters much easier and simpler. Now moving on to the next character 3 which has two major attributes in the Modified fast non local means denoising method which is in the bottom and in the top. The bottom portion has a comparatively larger dark area than the top portion, because the formation of this character is done in such a way that the bottom area is bigger than the top area. Which is why it is easily distinguishable when the character shows this sort of dark portion in the detection process, it can be easily said that the character is 3. In terms of the character 4 it is sort of easier to detect and recognize this character in Bangla because it surrounds almost the highest area among the other characters and also it has two circular shapes within the character. So in this case when we use the Modified fast non local means denoising method it delivers more significant attributes in three portions of this characters which are the top, middle and in the bottom. In this three area three dark semi horizontal dashes can be seen because of forming two circular shapes side by side. And by this significance it is clearly more recognizable than the version from the fast non local means denoising method. So in case of character 4 the modified fast non local means denoising method unconditionally works much better than fast non local means denoising method. The character 5 in the Bangla number system is pretty confusing with the character 0 because both of them look almost identical from a defocused point of view. To clear out this sort of confusion during the detection process a unique portion in the character 5 has been selected to highlight by the Modified fast non local means denoising method. As we can see that the bottom portion of the character 5 has a dark horizontal dash and in the upper portion there is a very small dark portion which is easily distinguishable from the character 0. In terms of 0 both the upper and the bottom portion has almost a similar kind of dark area in size and intensity. But in terms of the number 5 the bottom portion is comparatively bigger than the upper portion

which gives us a clear idea that the character is 5 with these attributes. In terms of the character 6 there are two dark portions which are in the bottom part and in the middle part. Both the dark portion is horizontal in orientation and the upper portion is slightly smaller than the lower portion in the Modified fast non local means denoising method. But there is no such distinguishable difference in the picture produced from fast non local means denoising methods. Moving onto the number 7, in the fast non local means denoising method the picture is still blurry and non distinguishable because it do not hold any prominent attribute to be recognized but in the Modified fast non local means denoising method the middle and the upper portion of the character 7 has two small separated dark portions which gives us a clear thought, that this character is 7 in Bangla. Then comes the number 8, it has a clear distinguishable feature in the Modified fast non local means denoising method which is a long horizontal dash in the middle and a small dark portion in the bottom right corner. These two features help us to understand that this character is Bangla number 8. In terms of the numbers this is the last one which is 9 and one of the most confusing one of them all. The Bangla 9 is so similar to the Bangla character 1 but our method which is Modified fast non local means denoising method handles the situation very well through making the edges more sharp and creating some prominent dark areas in such areas that is clearly understandable that the character is 9 and not 1.

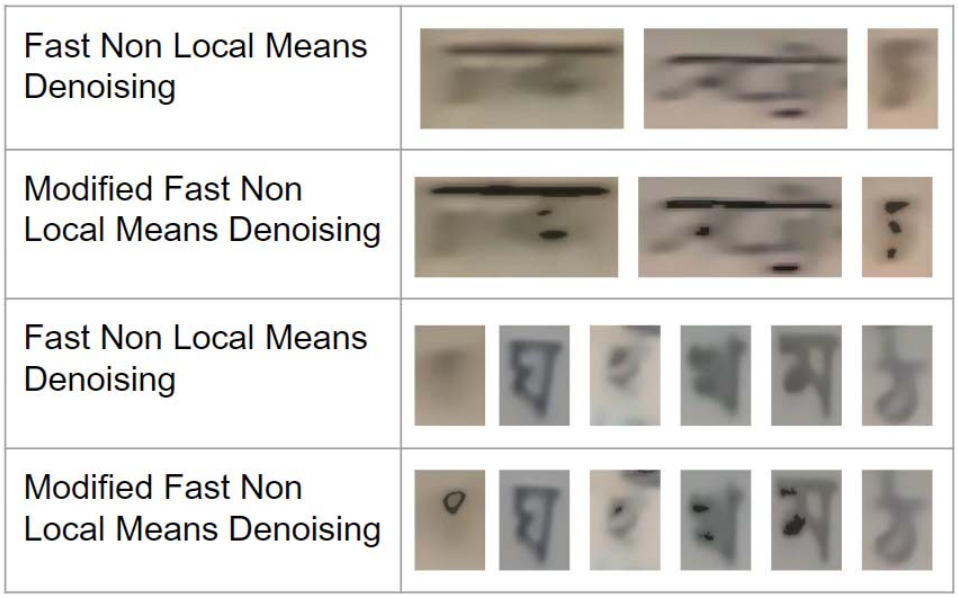


Figure 4.20: Comparison table between two methods

Dhaka is one of two major words in our dataset and the another one is Metro but if we look closely in the Modified fast non local means denoising method The e is a proper horizontal dark line in the bottom right portion in the Metro image but in case of the Dhaka image the horizontal line in that portion is missing. From this basic distinguishing feature Dhaka and Metro these two words can be easily recognized. Then comes ga, ca, gha, ho, kha, mo, tho these are Bangla alphabets used in the number plate system in Bangladesh. From the view of Modified fast non local means denoising method shows almost similar results in terms of edge refinement and character recognition and clarity to fast non local means denoising

method. But the key improvement can easily and clearly be seen in the number detection portion and one more thing to mention is that the background portion which is the white portion is more white and clear in the Modified fast non local means denoising method.

4.2.2 Labeling Filtered Image

Data labelling is necessary for data preprocessing for supervised learning systems. We make two datasets for labelling. Data labelling is a manual process with the help of software. We use labelImg for our help in data labelling. We made two similar datasets to compare the results of fast non local means denoising and our modified fast non local means denoising filter. Our two datasets are exactly the same labelled image, we make the duplicate to test our own modified filter. Our main focus was edge detection of the characters so that it's easier to identify the data from the noisy image. As our modified filter successfully added some more features to our noisy images, it was easier for us to understand the bengali letters specially the bengali numbers. After adding our modified filter we can label the image from the noisy dataset easily.

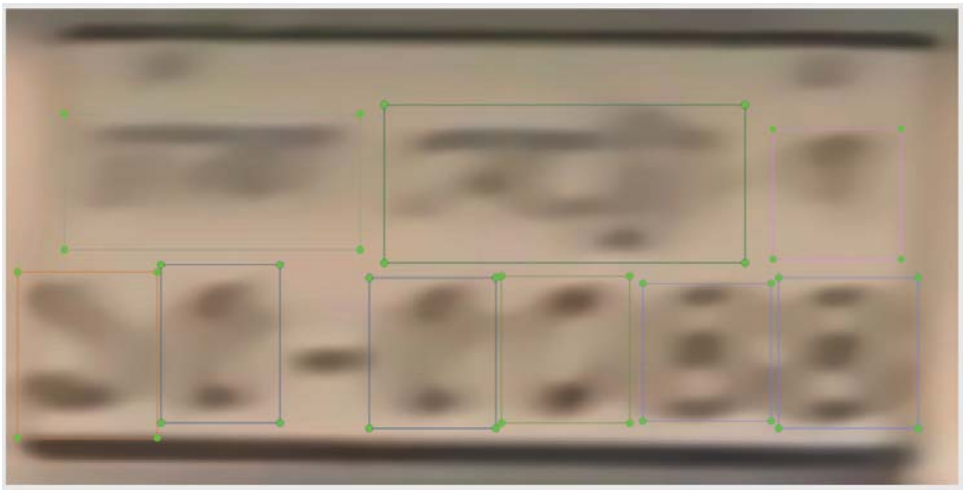


Figure 4.21: Labeling for Fast Non Local Mean Filter Applied Images

Above we can see that this Figure 4.21 is slightly blurry and seems quite hard to distinguish each and every character separately. In order to create a precise dataset the person who does the data entry, needs to understand each and every character easily and clearly. If the characters are confusing and can't be understood easily then it will become a huge gap in getting a good and more accurate result. Because we create a dataset so that we can generate a future proof result or predict and avoid any unwanted situation. But if the dataset is not carefully and appropriately labelled it will not be able to generate an expected and promising result. To avoid this situation we applied a new modified filtering method of our own and we have found that it gives more accurate results than the previous filtering method.

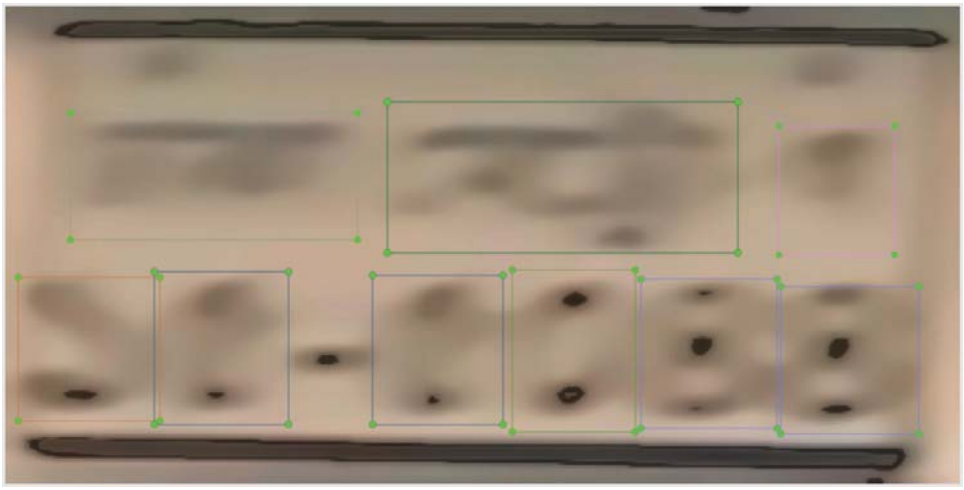


Figure 4.22: Labeling for Modified Non Local Mean Filter Applied Images Filter

Here in this Figure 4.22, it is clearly visible that all the characters are easily noticeable and distinguishable. It will obviously be easier for the person who will label the image files and the labeling will be more fast, accurate and precise than before. The key portions to mention here is that, in the previous image the contrast between the background and the characters were pretty low and it was really hard for someone to visualize them easily. Moreover if we look at the edges of each character individually we can easily notice that the edges are more sharp and visible compared to the conventional filtering method. On the other hand, in the second picture it is noticeable that the numbers and the words bear more high contrast value which helps to recognize and detect each character and numbers more easily than before.

4.3 Model Specification

To use our two pre processed data we choose two deep learning models. We will apply both kinds of filtered data in each of the models.

4.3.1 Artificial Neural Networks (ANN)

Here we used ANN algorithm and after that we performed backpropagation on it to minimize the error. By this way we detected the pattern. ANN backpropagation is for neural net training purposes. For training our neural net we need to think about loss function, activation function, hypothesis function and optimization function with learning rate 0.1 and weights initialized as 1 and will change the weight through backpropagation for getting a good result. We used the "Kalpurush" Bangla font to write and check our results.The error function in classic backpropagation is the mean squared error,

$$E(X, \theta) = \frac{1}{2N} \sum_{n=1}^{N} (\hat{y}_i - y_i)^2 \tag{4.9}$$

where y_i is the target value for input-output pair (x_i, y_i) and y_i is the computed output of the network on input x_i. We need to perform derivation for our backpropagation. The original formula for ANN without bias is,

$$a_i^k = \sum_{j=1}^{rk^-1} w_{ji}^k o_j^{k-1} \tag{4.10}$$

Backpropagation tries to minimize the error function by using formula[4.9]. For backpropagation we need to derive formula[4.10] where we get,

$$\frac{\partial E(X,\theta)}{\partial w_{ij}^k} = \frac{1}{N}\sum_{d=1}^{N} \frac{\partial}{\partial w_{ij}^k}(\frac{1}{2}(\hat{y_d} - y_d)^2) = \frac{1}{N}\sum_{d=1}^{N} \frac{\partial E_d}{\partial w_{ij}^k} \tag{4.11}$$

The derivation of error function is,

$$\frac{\partial E_d}{\partial w_{ij}^k} = \frac{\partial E_d}{\partial a_j^k}\frac{\partial a_j^k}{\partial w_{ij}^k} \tag{4.12}$$

For backpropagation algorithm we do partial derivatives,

$$\frac{\partial E_d}{\partial w_{ij}^k} = \partial_j^k O_i^{k-1} \tag{4.13}$$

for final layer's error term we calculate

$$\delta_l^m = g_o^l(a_l^m)(\hat{y_d} - y_d) \tag{4.14}$$

For hidden layer error calculation we compute

$$\delta_l^m = g^l(a_j^k)\sum_{i=1}^{r^{k-1}} w_{jl}^{k+1}\delta_l^{k+1} \tag{4.15}$$

We need to update the weights for getting the result where we compute

$$\Delta w_{ij}^k = -\alpha\frac{\partial E(X,\theta)}{\partial w_{ij}^k} \tag{4.16}$$

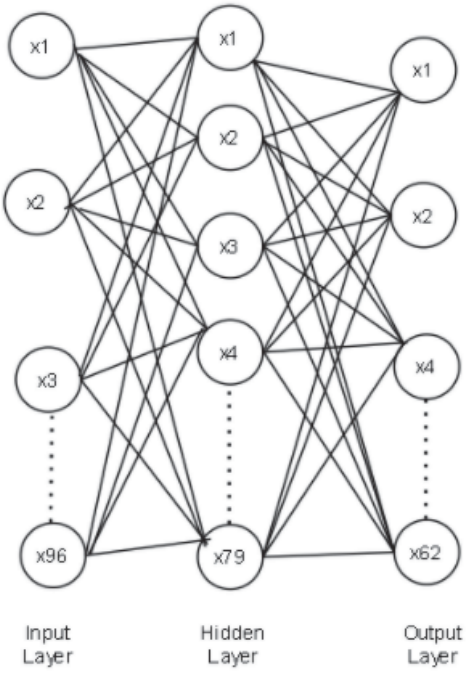


Figure 4.23: ANN algorithm

In ANN each neuron has a particular weight, weights are assigned to the neurons and one additional weight for bias. The network input layers are used for the training dataset. We initialized the weights of each neuron between 0 to 1. ANN logistic function. After forward propagation, it calculates the error function by backpropagation. The algorithm for ANN is as follows-
Step 1: Initialized the network with weights between 0 to 1
Step 2: Forward propagate, it includes three steps

i. Neuron activation
ii. Neuron transfer
iii. Forward propagation

Step 3: Back propagate error, it mainly calculated the transfer derivation and then error back propagation.
Step 4: Train the network by updating the weights.
Step 5: Predict the result.
Step 6: Apply back propagation for getting better results.
Step 7:check the accuracy of the model.

Here we used openCV2 and pytesseract in the python3 environment to detect the Bangla characters. In pytesseract it uses an ANN algorithm which has 3 layers. One input layer, one output layer and one hidden layer. Input layer there are a total 96 neurons, the hidden layer has 79 neurons and the output layer 62 neurons and the number of epochs is 280. By using this library we were able to detect Bangla numbers and letters. One of the important things here is if the edges or the number plate is not detected well then the algorithm gives an error. So we have to be careful to deal with this.

4.3.2 YOLO V3

We use fast non local image denoising filters for number plate denoising as we are working with noisy data. We can write the YOLOV3 algorithm in four simple steps.
Step 1: Input image (608,608,3).
Step 2: The input image goes through the algorithm and gives us output (19,19,5,85).
Step 3: After flattening the last two dimensions, the output volume (19,19,425)-
i. Grid is 19 x 19 and the input image gives 425 numbers.
ii. 425 = 5 x 85 , prediction for 5 boxes corresponding to 5 anchor boxes.
Step 4: We select boxes based on score thresholding and non-max suppression. The loss function for YOLOV3,

$$Lossfunction = \lambda_(coord) \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(x_i - \hat{x}_i)^2] + \lambda_(coord) \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] + \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_(coord) \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} (C_i - \hat{C}_i)^2 + \sum_{i=0}^{S^2} 1_{ij}^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \tag{4.17}$$

This loss function uses IoU which is intersection over union. YOLOV3 uses its predefined bounding The bounding box responsible for detecting the number plate whose anchor has the highest IoU with the ground truth box.

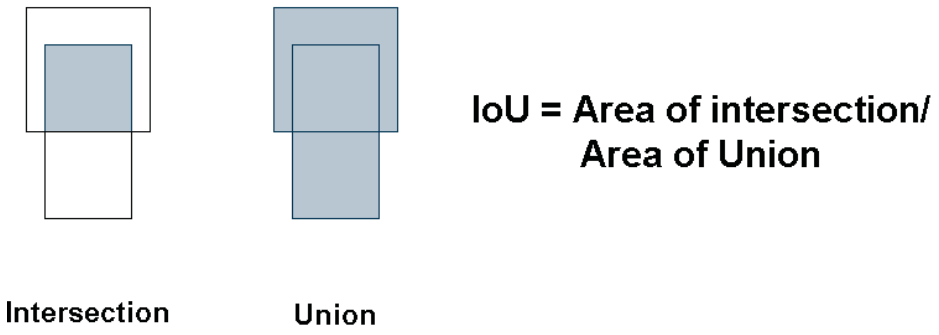


Figure 4.24: IoU

Bounding box predictions are done with the help of Output equations.

$$b_z = \delta(t_x) + c_z \tag{4.18}$$

$$b_y = \delta(t_y) + c_y \tag{4.19}$$

$$b_w = p_w exp(t_w) \tag{4.20}$$

$$b_h = p_w exp(t_h) \tag{4.21}$$

$$p_\gamma(Object) = \delta(t_o) \tag{4.22}$$

YOLOV3 does not predict the absolute coordinates, YOLOV3 predicts the offsets by normalizing the dimensions of the cell from the feature map. YOLOV3 works from the top left corner like a sliding window where it considers this as a square matrix and slides it according to the stride.

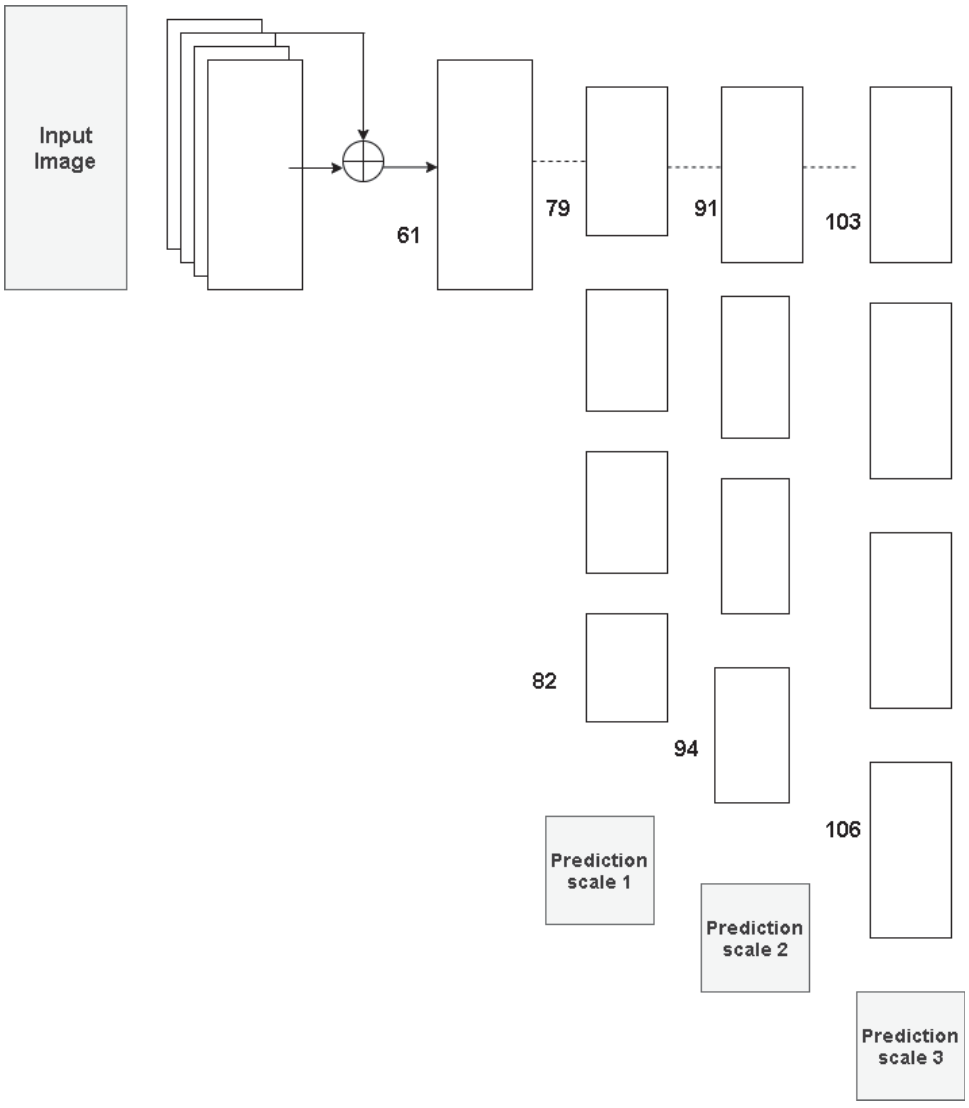


Figure 4.25: YOLOV3 architecture

We trained 760 images and tested with 330 images where we take one-fourth of the train images as epoch steps.

Chapter 5

Performance Analysis and Discussion

In this chapter we are going to discuss the result and analyze the reason behind the results. We created two datasets with the same labeling by applying two filters. Then we applied those two into a pre-train assisting model for which there is already a python library and then we also train the data into another algorithm and analyze the result.

5.1 ANN Algorithm Result for Both NLMD and MNLMD FIltered Data

Here we used opencv2 and pytesseract in the python3 environment to detect the bangla characters. In pytesseract it uses an ANN algorithm which has 3 layers. One input layer, one output layer and one hidden layer. Input layer there are a total 96 neurons, the hidden layer has 79 neurons and the output layer 62 neurons and the number of epochs is 280. By using this library we were able to detect Bangla numbers and letters. One of the important things here is if the edges or the number plate is not detected well then the algorithm gives an error. As this work with noisy images it is not possible to detect those. Most of the images gives an error for this existing pre train model.

```
License Plate : টাকা মেট্রো-ভ
১১-১৯৮১

<matplotlib.image.AxesImage at 0x7fccdf4faf10>
```

Figure 5.1: ANN ALgorithm in Pytesseract Result

Here in Figure 5.1 we can see that ANN algorithm is giving a good result in this clear image. But when we test our image in this algorithm only 4 out of 100 noisy images it can read. But not correctly. So we can conclude that this existing model does not work on the noisy image. Though we use two denoising filters but no denoising filter can remove all the noise from the image. So both the Non Local Mean Denoising and Modified version of Non Local Design filter failed to work on this process. We also tried the raw noisy image in this algorithm. But the results are the same. Only the clear image like Figure 5.1 can give a good result on this algorithm. But our goal is not that.

5.2 YoloV3 Result for NLMD Filtered Data

Our result for YOLOV3 using non local mean denoising is satisfactory. We apply NLMD for filtering technique and then we apply YOLOV3 for object detection. Here there were 19 kinds of labels in our dataset, where 10 labels were Bangla digits, 7 labels were Bangla characters and 2 was Bangla words.

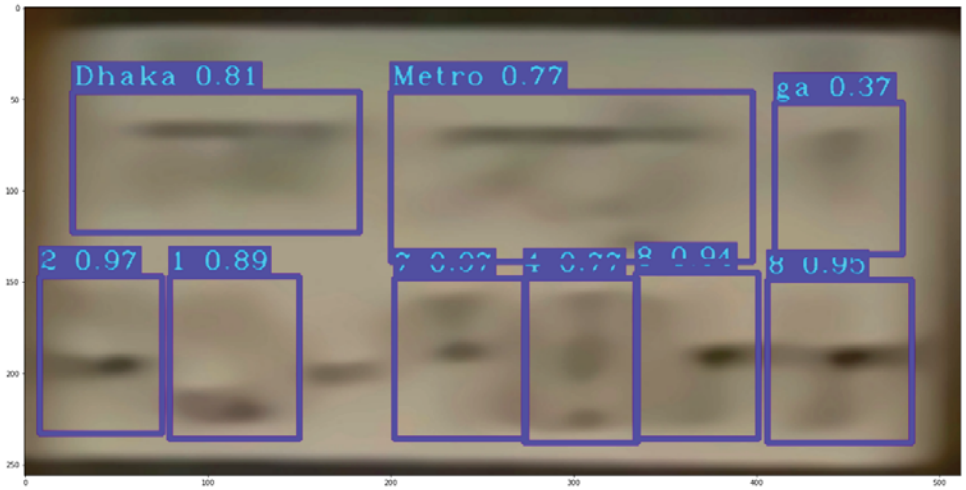


Figure 5.2: Detection in NLMD Filtered Image by YOLOV3

After training those pre-processed data in YOLOV3 object detection model. We get results like (Fig 5.2). Here we can see the through the image is very blurry but the algorithm is able to detect the characters, words and digit perfectly.

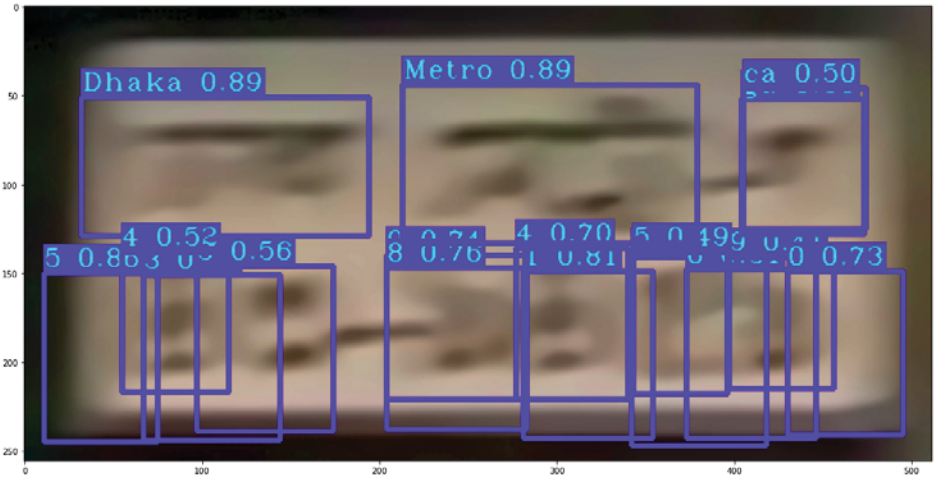


Figure 5.3: Error in Detection in NLMD Filtered Image by YOLOV3

But there are some images where the algorithm detects multiple objects in a single digits or character place. Such as, Fig 5.3. There are also some images where the algorithm is detecting some objects and missing some objects. This type of problem can also be counted as errors. To analyze this seranio we randomly took 100 images. And build 2 separated confusion matrices. The first confusion matrix is for the first line of the image, which contents the bangla words and characters. The first two are words followed by one character. Then the second one is for the second line of the image which contains the Bangla digits. There are a total six digits in one Bangla number plate.

	Positive	Negative
True	272	17
False	2	10

Table 5.1: Confusion matrix for first line of the number plate using NLMD

With NLMD we get the result correctly for 272 characters and words. Only 17 characters and words of the model can detect but cannot detect them correctly. We get 2 characters and words for false positives where it is detecting the character or word but not in properly. Also there were 10 characters and words where the model does not detect anything.

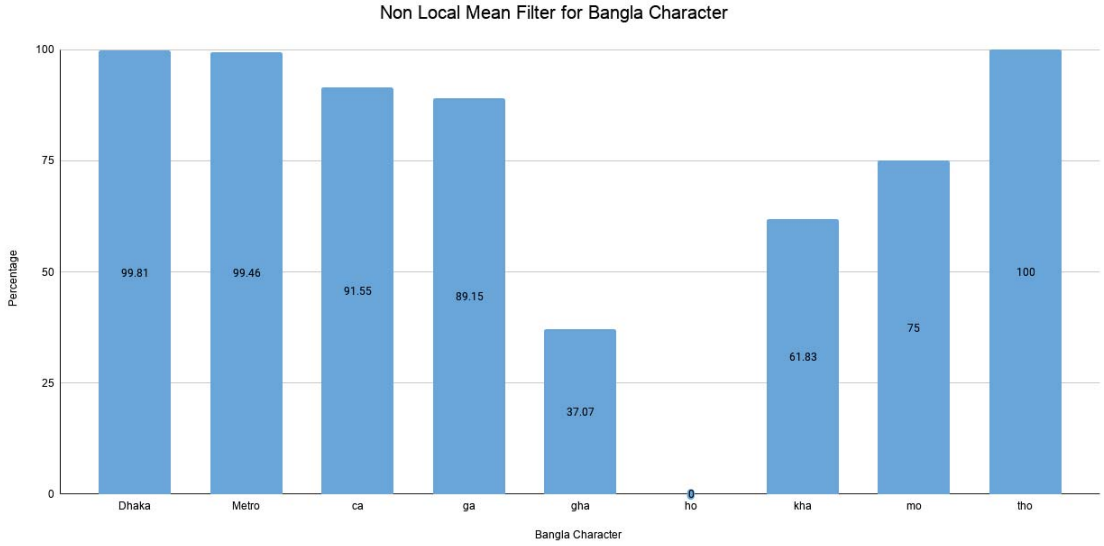


Figure 5.4: Accuracy Bar Diagram For Non Local Mean Filtered Dataset on YOLOV3 Model on Characters

In Figure 5.4 we can see the bar diagram of the total nine labels for the first line of the Bangla Number Plate. This data is based on the 330 images in the test images of our dataset. Here the two words "Dhaka" and "Metro" both have accuracy near to 100 %. But in terms of character out of 7 only 3 characters have the accuracy greater than 85% and also we can see the "ho" has accuracy 0%.

	Positive	Negative
True	543	19
False	25	2

Table 5.2: Confusion matrix for the second line of the number plate using NLMD

Now for the second line we make the confusion matrix like Table 5.2 where our model works perfectly for 543 digits and 19 digits are not detecting correctly. Our model is detecting multiple things where it has nothing to detect or only one digit to detect for 25 images and it gives false negatives for 29 images.

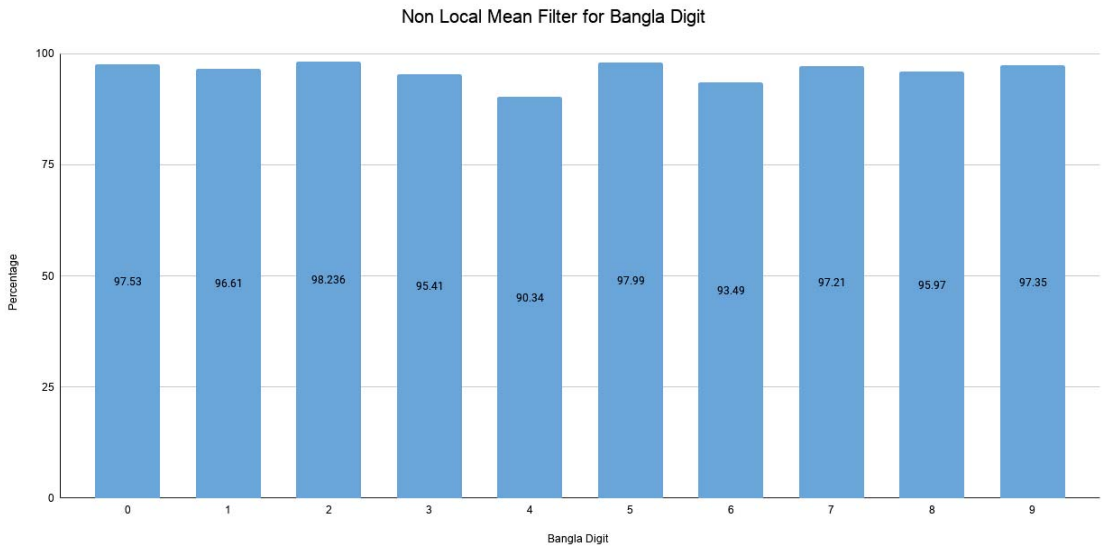


Figure 5.5: Accuracy Bar Diagram For NLMD Filtered Dataset on YOLOV3 Model Digits

Here in Figure 5.5 we can see that all of the Bangla digits have an accuracy greater than 90%. Which is a very good result

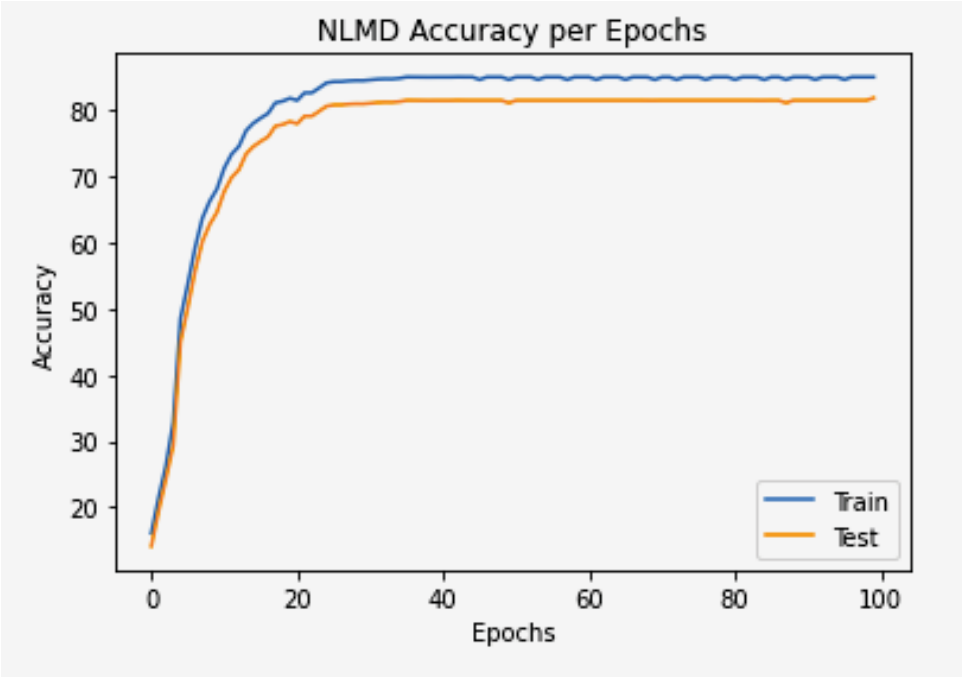


Figure 5.6: Accuracy on Train and Test set of NLMD Filtered Data

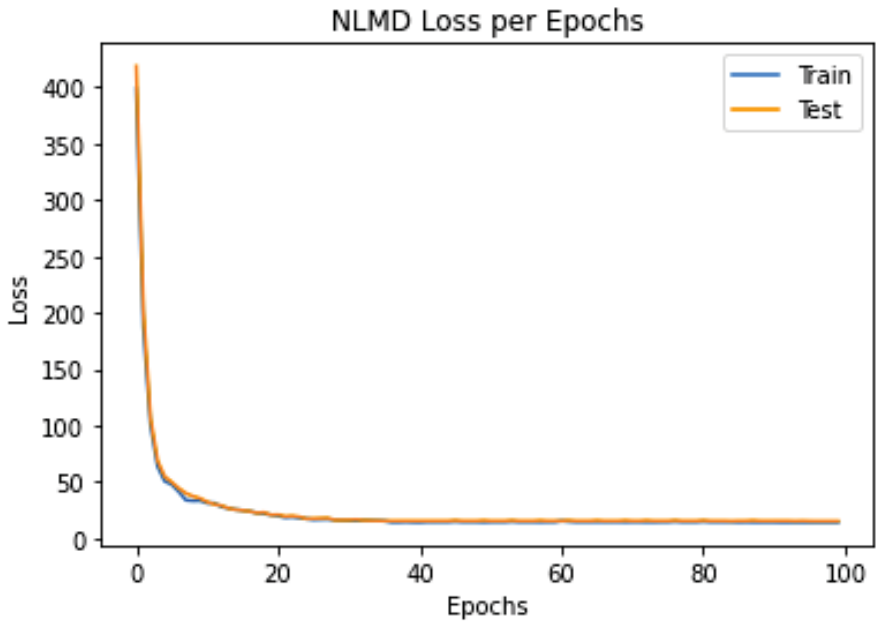


Figure 5.7: Loss on Train and Test set of NLMD Filtered Data

Our training loss per epoch for NLMD is 18.96 and test loss per epoch is 20.23 (Figure 5.7). We get pretty good accuracy with this technique. Our train accuracy for NLMD is 84.43 and test accuracy for NLMD is 82.76(Figure 5.6).

5.3 YoloV3 Result for MNLMD Filtered Data

We modified the NLMD filter technique and applied that to our dataset and we got better results with our customized filter as our filter detected the digits and character from the images by bolding some of the unique places of the digit and characters.

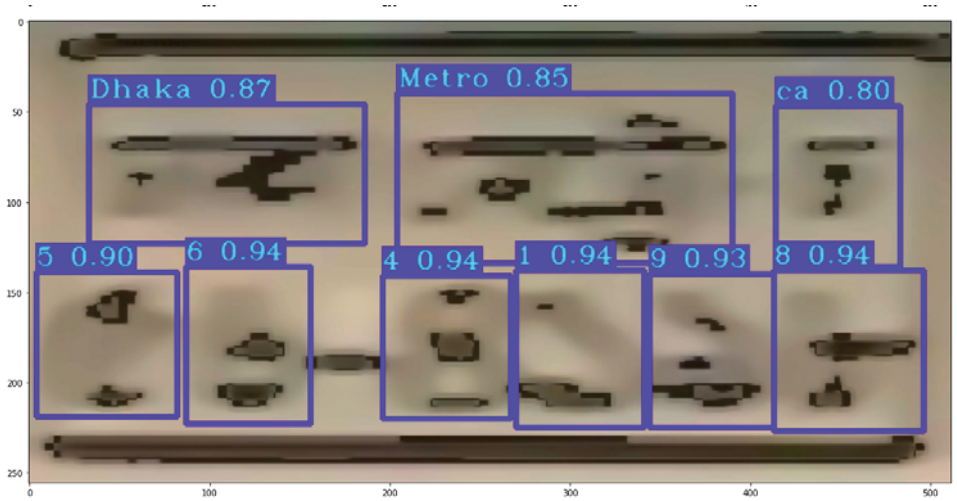


Figure 5.8: Detection in MNLMD Filtered Image by YOLOV3

After training the modified non local mean denoising filters images in YOLOV3

object detection model. We use some images to see the result in action. Such as in Figure 5.8.

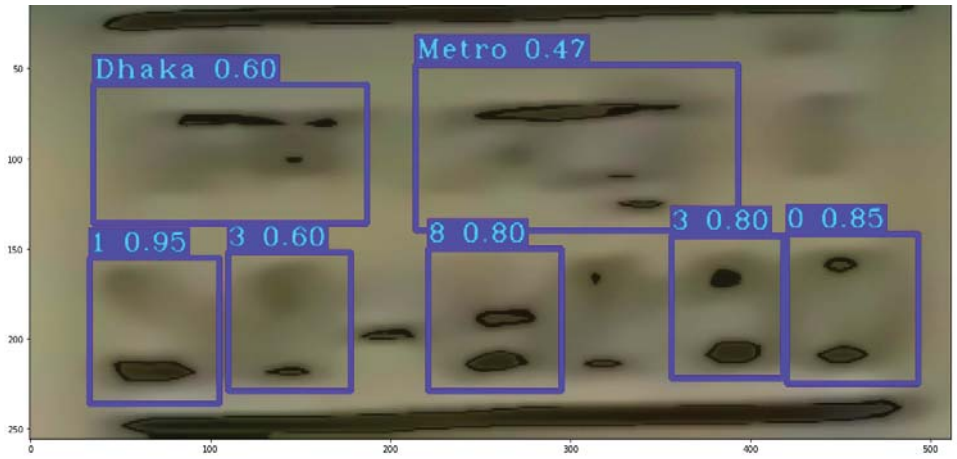


Figure 5.9: Error in Detection in MNLMD Filtered Image by YOLOV3

But there are some images where the algorithm misses some of the digits or characters to detect. Such as, Figure 5.9. There are also some images where the algorithm is detecting multiple objects in a single digits or character place. This type of problem can also be counted as errors. To analyze this scenario like Chapter 5.2 we randomly took 100 images to build two separated confusion matrices. One for the word and character. Another one for digits.

	Positive	Negative
True	276	14
False	9	3

Table 5.3: Confusion matrix for the first line of the number plate using MNLMD

With MNLMD we get the result correctly for 276 words and characters. Only in the 14 images can the character but cannot detect the characters or words correctly from the first line. We get 9 images for false positives and 3 images where we have nothing to detect our model is detecting something.

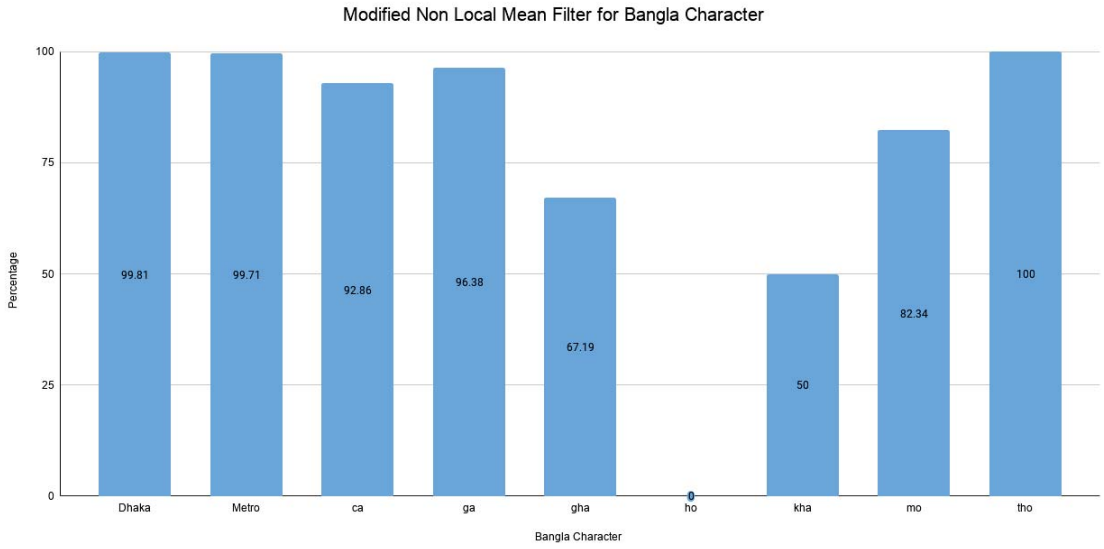


Figure 5.10: Accuracy Bar Diagram For MNLMD Filtered Dataset on YOLOV3 Model on Characters

In Figure 5.10 is the summary of individual accuracy for MNLMD filtered dataset. Here the accuracy is based on our test images, we can see that out of 9 labels 6 has an accuracy over 90% and one has accuracy over 80% and the rest of the 3 labels are less than 70%.

	Positive	Negative
True	551	16
False	22	26

Table 5.4: Confusion matrix for the second line of the number plate

Now for the second line we make the confusion matrix like Table 5.4 where our model works perfectly for 551 digits and 16 digits are not detecting correctly. Our model is detecting two things where it has nothing to detect for 22 digits and it gives false negatives for 26 digits.

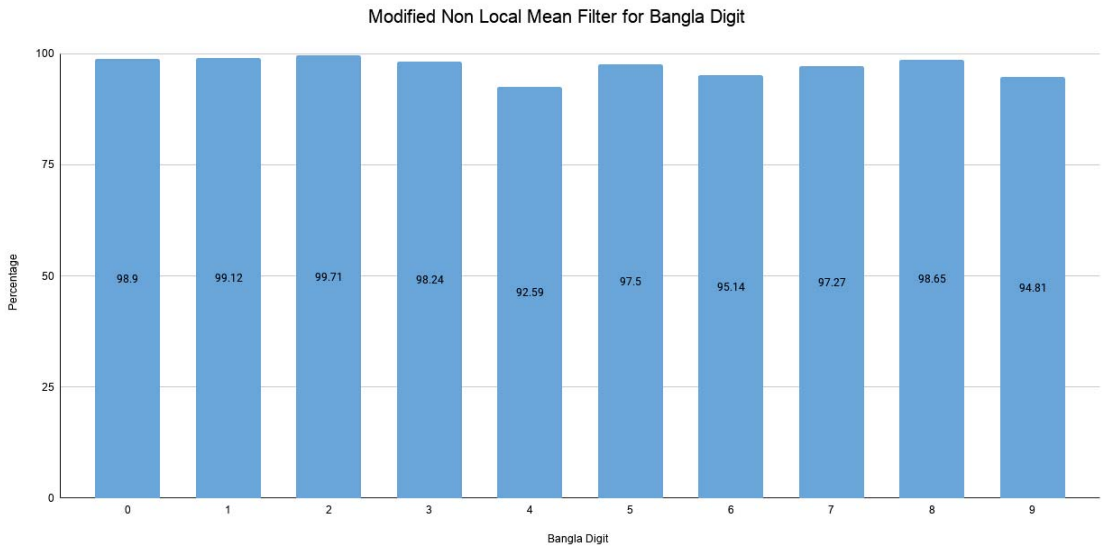


Figure 5.11: Accuracy Bar Diagram For MNLMD Filtered Dataset on YOLOV3 Model on Digits

In Figure 5.11 is the summary of individual accuracy for MNLMD filtered dataset. Here the accuracy is based on our second line of the test images. We can see that out of 10 Bangla digits we got an accuracy over 95% for 9 digits. Only the Bangla digit "4" is giving us an accuracy of 92%. which is also pretty good.

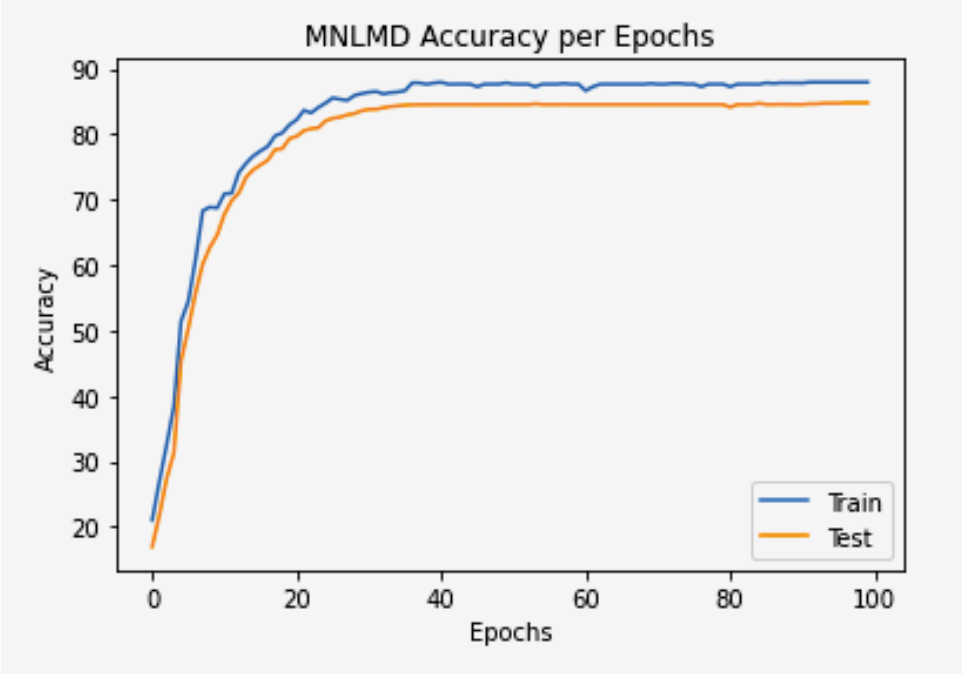


Figure 5.12: Accuracy on Train and Test set of MNLMD Filtered Data

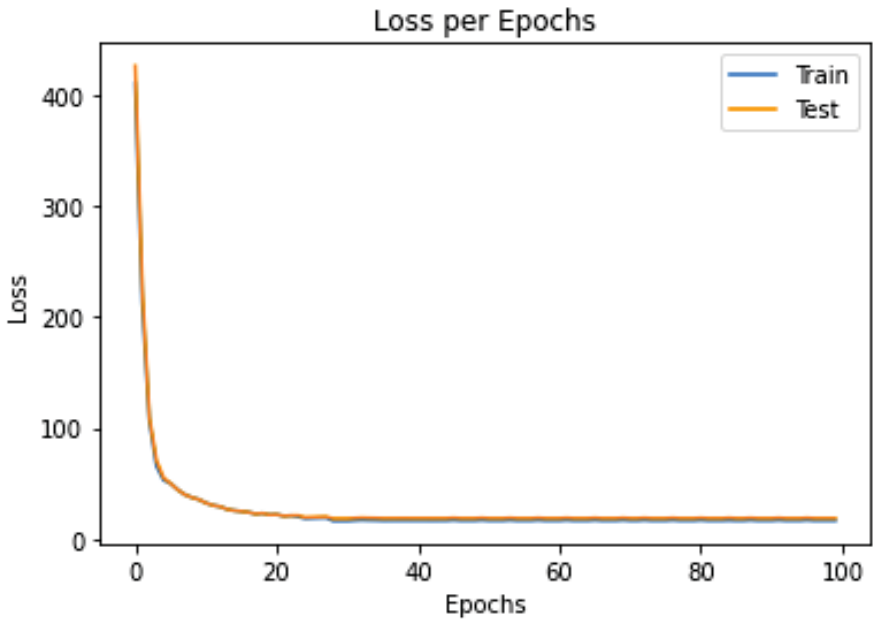


Figure 5.13: Loss on Train and Test set of NLMD Filtered Data

We plot the MNLMD train and test accuracies are 86.36 and 85.14 respectively (Figure 5.12). Also we get 14.33 and 16.67 loss on train and test respectively.

5.4 Comparison of NMLD and MNMLD Dataset Results

This is one of the most crucial parts of our research. Here we will discuss the impact of existing non local mean denoising filters and our modified version of non local mean denoising.

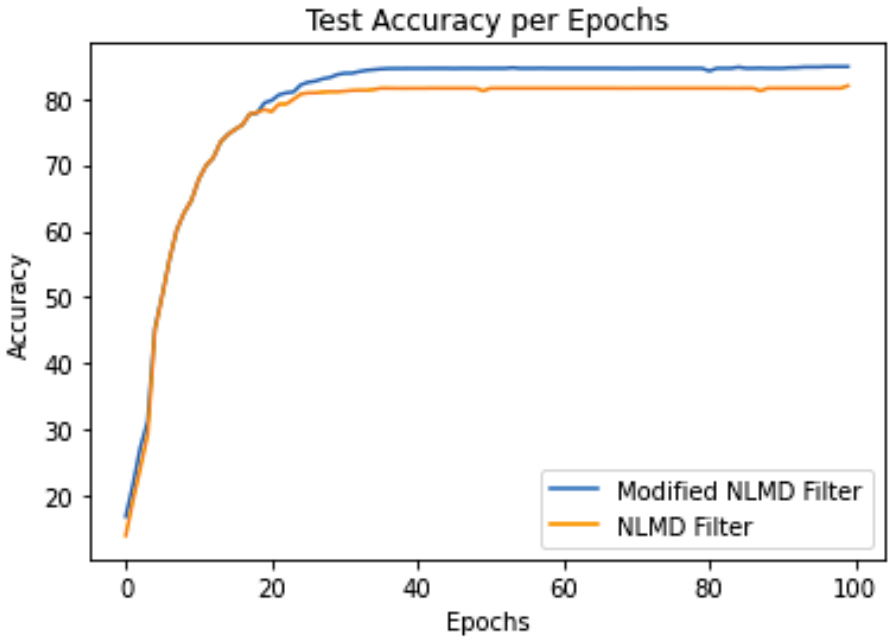


Figure 5.14: Graph of Test Accuracy for Modified NLMD and MLMD

Here in Figure 5.14 we can see that the test accuracy is MNLMD was higher tough all of the epochs. And at the end we have 85.14

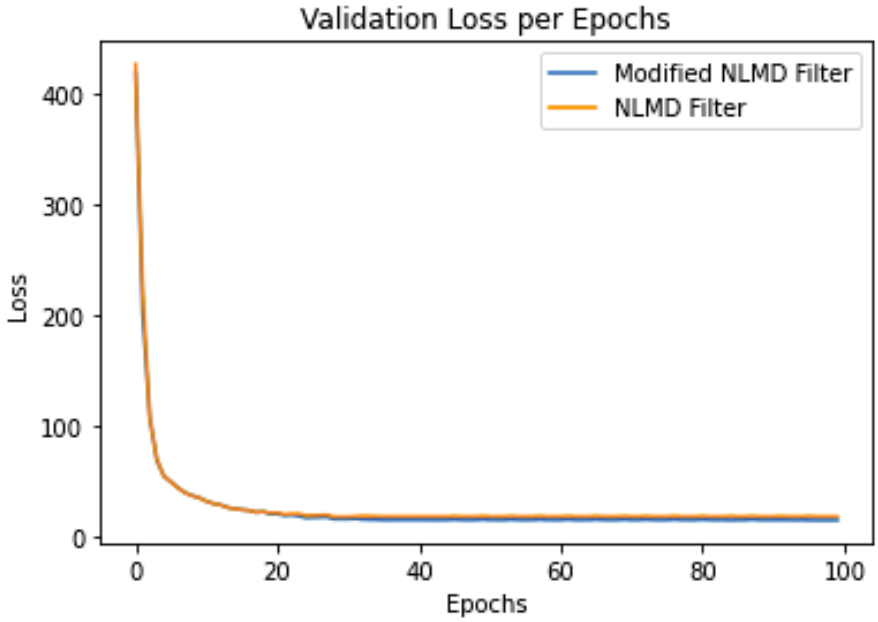


Figure 5.15: Graph of Test Loss for Modified NLMD and MLMD

In Figure 5.15 we can see that we got almost 4 less loss for MNLMD then NMLD. So we can say that in overall test accuracy and validation loss we got a better result in MNLMD filtered data than NLMD.

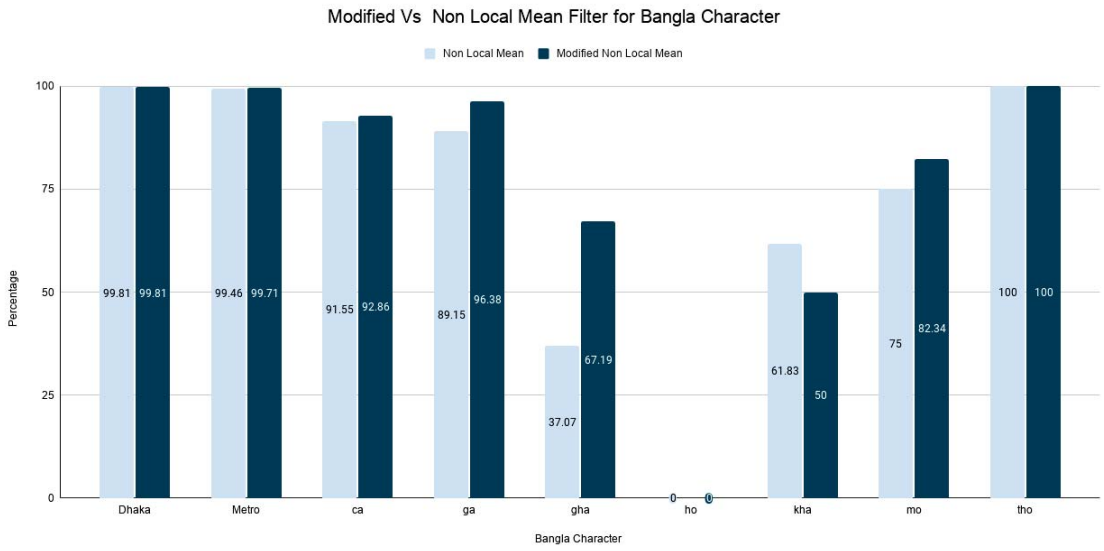


Figure 5.16: MNLMD Filtered vs NLMD Filtered Image for Bangla Characters

To analysis the result more deeply we take the 330 images from both dataset and applied them into the train model build by same dataset. Here one thing is important to mention that all images of both datasets are same. But the only difference is that in one we used NLMD filter and in another we use MNLMD. Now for the first line of the number plates we can see that out of 9, 7 labels MNLMD filtered data is giving more accurate results then NLMD. Rest of the 2 labels is giving same amount of accuracy.

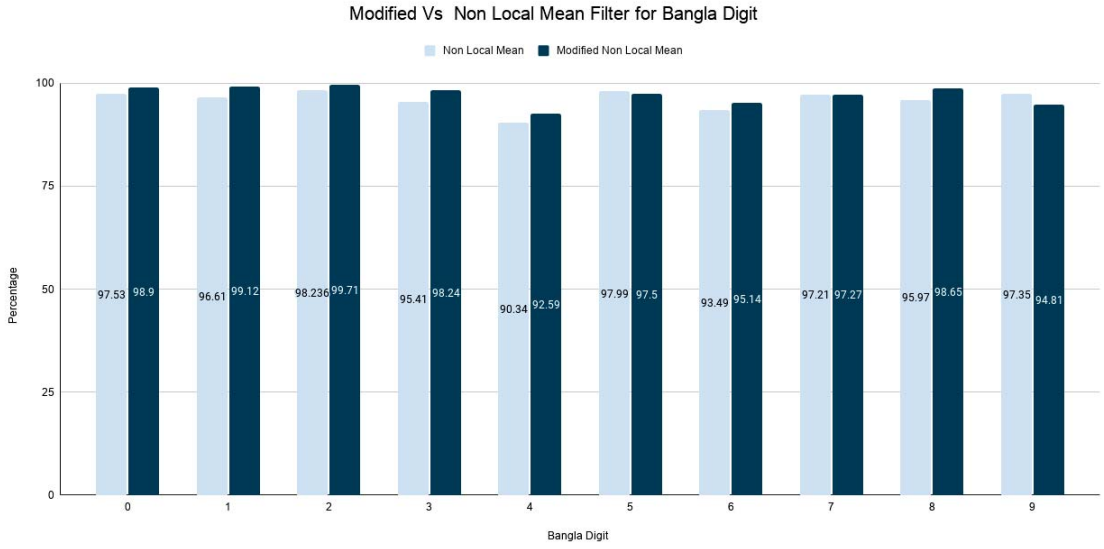


Figure 5.17: MNLMD Filtered vs NLMD Filtered Image for Bangla Digit

In Figure 5.17 we show that the accuracy rate for both MNLMD and NLMD filtered data on the second line of the number plate. We can see the in total 10 digits MNLMD is giving better accuracy in 8 digits and other 2 are almost the same. So by analysis this result we can say that modified non local mean denoising filter

algorithm is more appropriate for detecting Bangla number plate digits and character from noisy image.

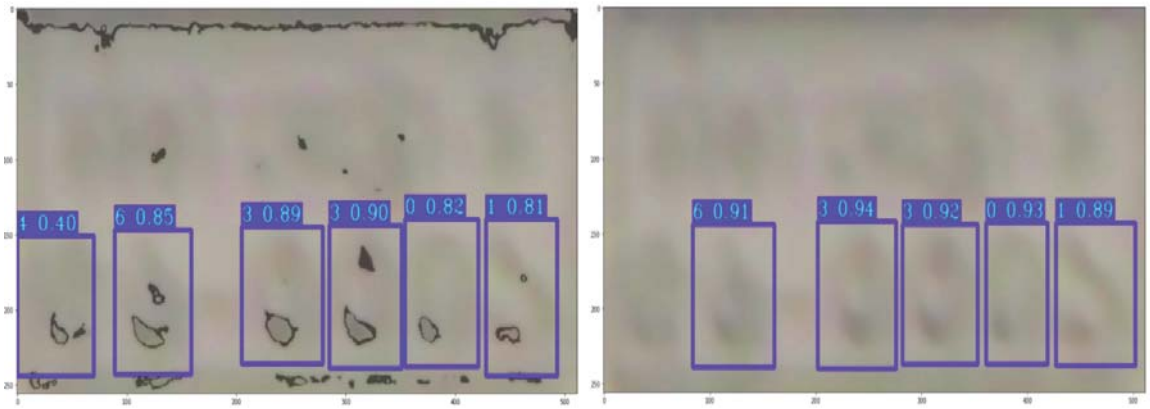


Figure 5.18: Result visualization for MNLMD and NLMD

Here in Figure 5.18 we used a very noisy image where the character and digits are very hard to detect. Both MNLMD and NLMD we can see that it is unable to detect the first line. But for the second line MNLMD can detect all six of the digits where MNLMD can not detect the first one. The main reason is, if we look at the first digit of the second line of both detected images then we can see that for MNLMD it creates an extra feature and the feature helps the algorithm to recognize the digit. By this way of creating features in all over the dataset images MNLMD performance better in the algorithm where NLMD fails.

5.5 Error

We calculate the error in three parts. For number plate detection we try with two algorithms.We apply Haar Cascade and YOLOV3 for detection. Haar Cascade doesn't work much well in our model so we choose the best one which is YOLOV3 for our model. For further process we did not work with Haar Cascade for poor accuracy. We calculate the first error in this process for YOLOV3.

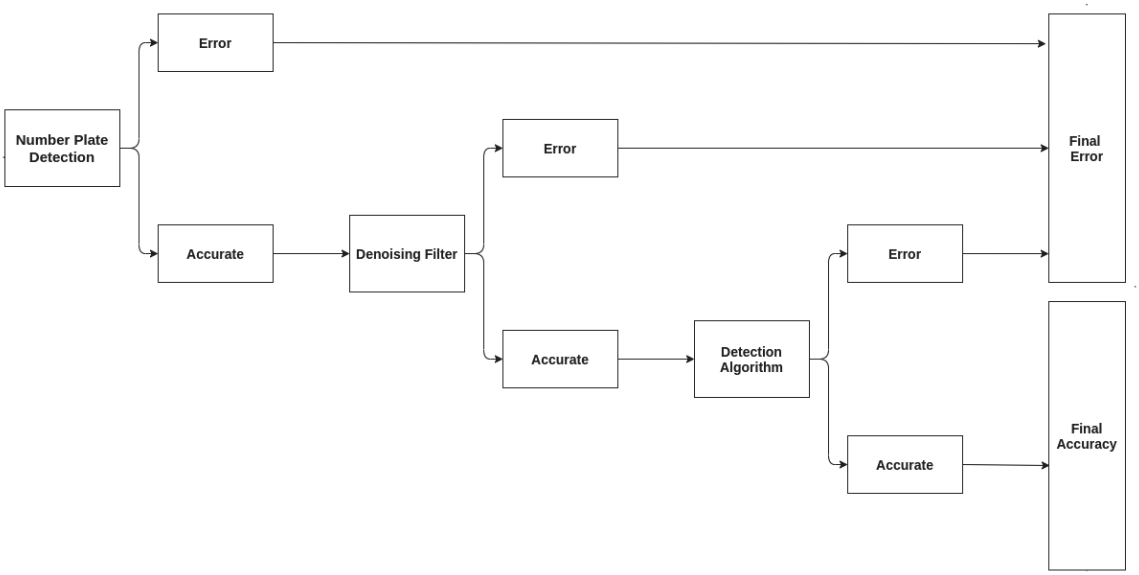


Figure 5.19: Flow of Final Error and Accuracy

After the number plate detection we work with a denoising filter which was one of the most challenging parts for us. We apply eight different filtering techniques for choosing the best one for our model. After trying with eight filters we choose the best one for denoising and modify that algorithm according to our dataset. After getting our filtering technique we get our second error and we move to the detection of the digits and characters which is the main motive of this work. We apply YOLOV3 here and get pretty good results for digit accuracy especially. We get more than ninety percent accuracy for digit detection after applying our modified filter and YOLOV3 for detection. But for Bengali characters we get less than fifty percent accuracy because of the limitations of datasets. Still we get more than eighty percent accuracy overall which is pretty satisfactory. We calculate the accuracy in this stage and get another and last error for our model. For final error calculation we get three errors in total as we divided this work in three main categories. Error increases exponentially so we need to take the best algorithm for minimizing the error and to get the best result with our dataset.

Chapter 6

Conclusion

In our paper we tried to study different types of algorithms and compare them to find out an optimal way to detect Bengali number plates from noisy footage as there is no significant work with Bengali number plate detection. We are in a global pandemic right now so collecting the data was really difficult for us. We make our own dataset from the Dhaka road for our model. Our model was for noisy dataset so another challenge we faced was training the dataset with suitable filters for which we build our own modified filter to detect the edge of the letters and alphabets. We trained our data with our two sets of dataset and then we added noise for making our dataset more noisy. We made our own test dataset to show the desired output. This model will help to identify stolen vehicles, identify vehicles involved in road accidents and also to track down criminals even from the noisy images. Finally, our goal is to detect Bangla number plates from a noisy dataset with a fast and less complex model. For our future work we can work with real time data from CCTV footage as our model is able to detect from noisy images with good accuracy. For CCTV video footage we can expand this work in future. We are detecting the number plates with 14 FPS, but for human eyes we need at least 25 FPS speed. Our plan is to work with this in future.

Bibliography

[1] A. Dauwe, B. Goossens, H. Q. Luong, and W. Philips, "A fast non-local image denoising algorithm," in *Image Processing: Algorithms and Systems VI*, J. T. Astola, K. O. Egiazarian, and E. R. Dougherty, Eds., SPIE, Feb. 2008. DOI: 10.1117/12.765505. [Online]. Available: http://dx.doi.org/10.1117/12.765505.

[2] M. Reisert and H. Burkhardt, "Equivariant holomorphic filters for contour denoising and rapid object detection," *IEEE Transactions on Image Processing*, vol. 17, no. 2, pp. 190–203, Feb. 2008. DOI: 10.1109/tip.2007.914218. [Online]. Available: http://dx.doi.org/10.1109/tip.2007.914218.

[3] M. A. Hasnat, M. R. Chowdhury, and M. u. Khan, *Integrating bangla script recognition support in tesseract ocr*, 2009. [Online]. Available: http://hdl.handle.net/10361/635.

[4] M. Mahmudul Hasan, *Real time detection and recognition of vehicle license plate in bangla*, http://lib.buet.ac.bd:8080/xmlui/handle/123456789/3314, 2011.

[5] M. S. Sarfraz, A. Shahzad, M. A. Elahi, M. Fraz, I. Zafar, and E. A. Edirisinghe, "Real-time automatic license plate recognition for cctv forensic applications," *Journal of Real-Time Image Processing*, vol. 8, no. 3, 2011. DOI: 10.1007/s11554-011-0232-7. [Online]. Available: http://dx.doi.org/10.1007/s11554-011-0232-7.

[6] H. Yu, L. Zhao, and H. Wang, "An efficient edge-based bilateral filter for restoring real noisy image," *IEEE Transactions on Consumer Electronics*, vol. 57, no. 2, 2011. DOI: 10.1109/tce.2011.5955208. [Online]. Available: http://dx.doi.org/10.1109/tce.2011.5955208.

[7] Y. Wen, R. H. Chan, and T. Zeng, "Primal-dual algorithms for total variation based image restoration under poisson noise," *Science China Mathematics*, vol. 59, no. 1, 2015. DOI: 10.1007/s11425-015-5079-0. [Online]. Available: http://dx.doi.org/10.1007/s11425-015-5079-0.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Jun. 2016. DOI: 10.1109/cvpr.2016.91. [Online]. Available: http://dx.doi.org/10.1109/cvpr.2016.91.

[9] M. Jaber, M. Hasan, and A. F., "Bangla digital number plate recognition using template matching for higher accuracy and less time complexity," *International Journal of Computer Applications*, vol. 181, no. 29, 2018. DOI: 10.5120/ijca2018918140. [Online]. Available: http://dx.doi.org/10.5120/ijca2018918140.

[10] Z. Xu, W. Yang, A. Meng, N. Lu, H. Huang, C. Ying, and L. Huang, "Towards end-to-end license plate detection and recognition: A large dataset and baseline," in *Computer Vision – ECCV 2018*. Springer International Publishing, 2018. DOI: 10.1007/978-3-030-01261-8_16. [Online]. Available: http://dx.doi.org/10.1007/978-3-030-01261-8_16.

[11] S. Merity, "Single headed attention rnn: Stop thinking with your head," in. Nov. 2019.

[12] N. Saif, N. Ahmmed, S. Pasha, M. S. K. Shahrin, M. M. Hasan, S. Islam, and A. S. M. M. Jameel, "Automatic license plate recognition system for bangla license plates using convolutional neural network," in *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, IEEE, Oct. 2019. DOI: 10.1109/tencon.2019.8929280. [Online]. Available: http://dx.doi.org/10.1109/tencon.2019.8929280.

[13] X. Bai, L. Ye, J. Zhu, L. Zhu, and T. Komura, "Skeleton filter: A self-symmetric filter for skeletonization in noisy text images," *IEEE Transactions on Image Processing*, vol. 29, 2020. DOI: 10.1109/tip.2019.2944560. [Online]. Available: http://dx.doi.org/10.1109/tip.2019.2944560.

[14] X. Long, K. Deng, G. Wang, Y. Zhang, Q. Dang, Y. Gao, H. Shen, J. Ren, S. Han, E. Ding, and S. Wen, "Pp-yolo: An effective and efficient implementation of object detector," in. Jul. 2020.