A Data-Centric Approach to High-Accuracy Traffic Sign Detection Using an Optimized YOLOv8 Model

By

Piyash Basak

221-35-993

FINAL YEAR THESIS REPORT

This Report Presented in Partial Fulfillment of the Requirements for the
Degree of Bachelor of Science in Department of Software Engineering

Supervised by

Mr. Md Rajib Mia

Lecturer (Senior Scale)

Department of Software Engineering

DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

December 24, 2025

APPROVAL

This thesis is titled "**A Data-Centric Approach to High-Accuracy Traffic Sign Detection Using an Optimized YOLO v8 Model**", submitted by **Piyash Basak (ID: 221-35-993)** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

Fazla Ealhe

Dr. Fazla Ealhe
Assistant Professor & Associate Head
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Chairman

Manzia

Dr. Marzia Ahmed
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 1

Dr. Shabnom Mustary
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 2

Md. Rajib Mia
Lecturer (Senior Scale)
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 3

Mohammod Abul Kashem, PhD
Professor
Department of Computer Science and Engineering
DUET, Bangladesh

External Examiner

Department of Software Engineering

Faculty of Science and Information Technology

Supervisor Approval Form

Fall 2025	B.Sc. In SWE	Campus: DSC

Student Name	Student ID
Piyash Basak	221-35-993

Project/Thesis Information	
Thesis Title	**A Data-Centric Approach to High-Accuracy Traffic Sign Detection Using an Optimized YOLOv8 Model**
Type of work	Artificial Intelligence / Machine Learning

Supervisor information	
Supervisor Name	Mr. MD Rajib Mia
Supervisor Initial	RM
Completed Credit till now	139
How many credits in this semester	6
Amount (Due)	-5800.00
Supervisor Consent	☐ Yes ☐ No

Supervisor Signature : 

SUPERVISOR'S DECLARATION

This is to certify that the thesis entitled "A Data-Centric Approach to High-Accuracy Traffic Sign Detection Using an Optimized YOLOv8 Model" submitted by Piyash Basak (ID: 221-35-993) to the Department of Software Engineering, Daffodil International University has been carried out under my supervision.

I confirm that this work is original to the best of my knowledge and belief, and that the thesis has not been submitted, either in part or in full, to any other university or institution for the award of any degree or diploma. I also certify that the student has fulfilled the requirements necessary for the submission of this thesis.

(Supervisor's Signature)

Full Name: Ms Fatama Binta Rafiq

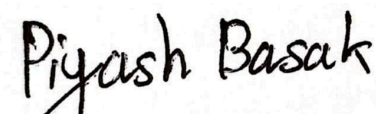
Position: Lecturer (Senior Scale)

Date: 24.12.2025

STUDENT'S DECLARATION

I hereby declare that this thesis, entitled "A Privacy-Preserving Federated Learning Approach for Segmenting Lung Abnormalities in Lung CT Scans," is my original work, submitted to the Department of Software Engineering at Daffodil International University, and has not been submitted for any degree or diploma at this or any other university.

All sources of information and data used in this thesis have been duly acknowledged. I further declare that the work reported herein embodies the results of my own research and complies with the ethical norms and regulations of Daffodil International University.



(Student's Signature) Full
Name: Shah Nawaz
ID Number: 221-35-874
Date: 23.12.2025

ACKNOWLEDGEMENTS

I would like to say thank you to everyone and everything that was involved in this. fulfill- ment of this dissertation study.

To begin with, we warmly thank and appreciate the Almighty because of his divine. bless- ing to enable us to accomplish Final Year Design Project(FYDP). successfully.

I would like to thank the employees and the management of Bokkobidi Hospital in. Dhaka, Bangladesh, on their part, to help them get real-time clinical data and to collaborate. ac- quiescing in my orating with me on this privacy-saving research work. Those in charge were the individuals. To motivate them to participate in this study were their zeal to im- prove diagnostic tools and at the same time save them. guarding patient privacy. I would like to thank the radiologists of the Bokkobidi Hospital. due to their hard work labeling and validating the clinical dataset.

I would like to thank the people who have provided me with the LIDC IDRI dataset. This implies that federated learning studies can be reiterated. The open commitment of the science community to share the resources spurs the pace of scientific advancements. This study was approved by the ethics committee, and it conformed to the institution standard. for managing personal data. The privacy-saving methodology will make this study up- sup- supports the right of people to privacy and moves medical AI to the good of patients. globally.

And last, but not the least, there is the ever-present support and forbearance of my parents.

A Data-Centric Approach to High-Accuracy Traffic Sign Detection Using an Optimized YOLOv8 Model

Piyash Basak

Thesis submitted in fulfillment of the requirements
for the award of the degree of

Bachelor of Science

Department of Software Engineering

DAFFODIL INTERNATIONAL UNIVERSITY

December, 2025

ABSTRACT

In the burgeoning universe of self-driving cars and intelligent driver-assistance systems, nothing is more important than safety. For the safety of these vehicles, they need to be able to look at a traffic sign and understand it immediately and accurately, as if a human were interpreting it. But it is very difficult to teach a computer to do this, because the real world is messy and unpredictable. When weather is poor, cameras on cars can often have a tough time seeing the signs clearly — during heavy rain or fog, for example. It also can be hard for computers to see signs when the lighting is bad, as at night, and when a sign is distant, small or partially obscured behind a tree or another vehicle. The outcome of a car not seeing a "Stop" sign or a 'Speed Limit' sign can be dire. The aim of that research paper is the primary and main goal to be addressed. Our goal is to develop a robust detection system that can reliably detect traffic signs even under such challenging conditions in the real world. To this end, we capitalised on a very large and diverse set of photographs known as the "Traffic and Road Signs" dataset. This reference image data set consists of 10,000 real-world images that contain 29 categories of traffic signs. We adopted a unique approach to this challenge. Many work to design entirely new and complex computer architectures in order to achieve better results. We used a "data-centric" approach in this study. That is, our primary concern in this paper is making the input data with which we teach model be of higher quality, not modifying the form of the model. We employed a method called "data augmentation," in which we purposefully altered the training images by, for example, darkening them or rotating them or adding simulated blur to teach the computer how to identify signs in all scenarios. This approach constructs a powerful and generalized model. YOlOv8-Large architecture was selected for the model detection. We selected this model because it is state-of-the-art and has a large capacity to capture complex details. We tested our method using common performance measurements such as Precision, Recall, and Mean Average Precision. The final results were outstanding. Our optimized system eventually obtained a mean Average Precision (mAP@0. 5) is 94.18%, indicating that the method is very accurate. It was also the best mAP@0. 5:0.95 score of 81.06%, indicating that it can also be very accurate for detection of signs. The system's results showed a precision of 97.70% and recall of 94.11%, so it virtually never makes false predictions, nor does it miss a sign. We also performed "ablation studies," or specialized tests to demonstrate that what we specifically trained the agent on was the real reason for this success. These results demonstrate that the data-driven solution we propose is a reliable, efficient and reproducible method which is ready to assist to build safer intelligent vehicles in the next decade.

KEYWORDS: Object Detection, Traffic Sign Detection, YOLOv8, Deep Learning, Data Augmentation, Autonomous Driving, Computer Vision, Road Safety.

Table of Contents

16

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

The transportation world has been revolutionized in the last 10 to 12 years. We've had incredible growth in smart technology and cars that drive themselves. A very large part of this advancement belongs to the development of Advanced Driver Assistance Systems. Today, these systems can be found or integrated in nearly all modern vehicles. They are designed to assist any driver and make the roads safe for all. For instance, ADAS comprises useful capabilities, such as automatic emergency braking that slams on the car's brakes if it sees a danger ahead. It also has lane-keeping assist, which can steer the car to keep it in its lane. However, somewhere a vehicle has to have "eyes" to make these intelligent features possible. It has to be capable of seeing the world and knowing what exactly it is looking at. And at the core of this capability is something called Traffic Sign Recognition (TSR). This technology is vital for safety and for aiding the car. Traffic signs are akin to a universal language or signal for all people who share the road. They provide us with vital information about the road and it is through that which controls traffic, hazards as well as gives us directions. We need to speak this tongue. For the driver, to whom signs saying "Stop" or "Speed Limit," or "Pedestrian Crossing" are a matter of life and death. The consequences of a single sign missed by the driver can be quite serious. It was going to cause a bad wreck. When we push further into the future, cars are getting more and more autonomous. This is when cars begin to drive with technology alone, without the aid of humans. At this point the task of all carrying out drivers (and reading traffic signs) will be taken over from humans by machines. The computer takes over as the driver, with all of the responsibilities that entails. With this change, the trustworthiness of automatic systems is emerging as an important issue. We must make sure the computer can perfectly see and understand signs, every time, all the time, to keep all of us safe.

1.2 Background and Motivation

The Difficulty of the Task

One of the easiest things in the world for a human driver is to see a traffic sign. We notice the red octagon and instantly know that it is telling us to "Stop." We do this without a second thought. But this is a very difficult problem for a computer vision system — basically the "eyes" of an autonomous car. The bad weather, the bad lighting — a human brain is phenomenal at canceling all that out, but for a computer it becomes very janky. Practically, on the actual road, driving conditions are hardly ever ideal. If we want to construct a safe system,

we have to create model that is powerful enough to deal with all of the different and difficult situations.

Major Hurdles in the Real World:

There are three issues that make it so difficult for computers to detect traffic signs

Small, Far-away Objects : When the car is travelling at speed it must be able to see signs a long way down the road. But when a sign is at a distance, it looks minuscule in the camera's picture. It may take up as few as, say, two pixels. The deep learning model has very little information to look at, because the image is so small. It's like you're reading a book from all the way across the room and the letters are that fuzzy. Terrible Outside Conditions: The world outside is not static. Its appearance can vary greatly depending on the weather and time of day. Heavy rain, dense fog or snow obstruct the view of the sign or the camera lens. Lighting is also a huge issue. Signs can be difficult to read in the dark at night. On that day, bright sunlight can produce glare and change colors — dark red of a sign appearing white. For a model to be good, it has to work in all these conditions, not just when the sun is shining.

Occlusion and Background Clutter: There are hardly ever signs of real scenes by themselves against a solid color sky in traffic cities. Many times a sign is obstructed by a tree limb, a street pole or perhaps even the big truck on the road. Also, the surface behind the sign can be quite "noisy". There could be colorful store billboards, bright neon signs or other buildings that resemble traffic signs. One is that the computer needs to be smart enough not to pay attention to the background but concentrate on "the real sign." The Evolution of Solutions In the past, the problems described were attempted to be addressed by engineers through "traditional" computer vision rules. They instructed the computer with simple commands like "find the color red" or "locate an eight-sided shape." These were all right with simple, clean pictures. But in the material world, where lights shift colors and trees obscure shape, such rules-based techniques fail utterly.

The Deep Learning Revolution

The landscape was completely transformed by Deep Learning — and most particularly the one form of technology known as Convolutional Neural Networks (CNNs). We don't hand CNN a list of such rules, like the good old days. We don't bother to tell it how to recognize a traffic sign; rather, we train it on thousands of examples of traffic signs and it figures out for itself what is significant. It learns what a Stop sign looks like by encountering one many times.

In the world of deep learning, it is YOLO (You Only Look Once) model family that has become a de facto choice for real-time detection. The biggest advantage of YOLO is its speed. Previous versions had to scan an image multiple times to find objects. YOLO considers the whole image, only once, and immediately detects everything you wanted! This speed is absolutely crucial for self-driving cars, which have to make safety decisions in the time it takes to blink an eye. We adopt YOLOv8 in this work. This is the latest and greatest version of YOLO. These products feature a number of technical improvements which make these so much more efficient at finding things under the ground that are made from different shapes and sizes than any previous version.

1.3 Objectives

The Main Goal

Using such a strong tool as the YOLOv8 model is obviously a fantastic way to start. But pulling it out of the box and shooting without a plan won't help you get the best shots. We think that its performance can be greatly improved if an appropriate training strategy is used. Throughout this paper we demonstrate this idea through the simplest possible lens: combine a big model [4] – YOLOv8L - with a big plan of how to change and improve the training pictures, and state-of-the-art results are now obvious.

A Data-Centric Approach

The goal of this work is to construct and give a presentation of such an optimized system. Actually we are addressing this problem from a different perspective: the data point of view. That is to say, we're not attempting to create a completely new computer brain or architecture. Rather than concentrating on this information. We're trying to understand how we can scale a model that's already really popular and does a great job. How much will we be able to push the YOLOv8, L version model with only educating it better?

You can't just have a high score. We are also interested in the reasons of high and accurate score. Part of that is explaining what made us a success. We really want understand c how much is due c to the size of the model and how much is due c to our data strategy just Augmentation. These analyses provide us useful intuition for what really matters for detection accuracy.

Specific Contributions of This Work

3.1To address these goals, We make the following three contributions in this paper:

Optimized Training Schedule: We provide a ready-to-use training schedule. This strategy incorporates both the native strength of the YOLOv8-Large model and a strong data augmentation policy. We've made this solution specifically to address common practical problems (eg identifying the sign in bad weather, with very low lighting).

Rigorous Testing: We didn't merely try it out. We perform a sequence of intricate scientific experiments that we refer to as "the ablation studies." This is like taking a machine apart to find out which parts of the machine are really important. These tests are demonstrative of the fact, that our constructed technique functions, but also how much in particular both the model size as well as our data tricks help in outcome.

Main Results: We obtained good numbers on a difficult public dataset. We computed a mean Average Precision (mAP@0. 5) of 94.18%. We also obtained a very high score (mAP@0. 5:0.95) of 81.06%. These findings are very strong and provide a good baseline for future researchers to test against.

Chapter 2

Literature Review

This chapter reflects into the past of traffic sign detection. In order to know why we use the YOLOv8 model in our experiments, let's understand how engineers attempted to solve this issue before. We shall discuss the good old days where computers behaved naively according to the simple rules, then move on to today when we have Artificial Intelligence & computer learns by themselves? It addresses specifically the family of "YOLO" models since those are currently self-driving cars' most important tool.

2.1 Traditional vs. Deep Learning-Based Detection

The Old Way: Classic Computer Vision

Back in the day, when Artificial Intelligence wasn't so clever, scientists would use "conventional" methods of spotting traffic signs. The method was simple. Instead, they depend on two main factors: color and shape.

The Color Based Methods: The traffic signs are generally very colorful. Red means "stop" on a stop sign, yellow is for warning. Engineers used this fact. They also programmed the image to split into individual colors, a process known as Hue Saturation Value. They say to the computer, If you see a clump of red pixels together in an image, it could be a traffic sign.

The Shape Based Methods: The engineers also considered shape of the sign. They applied customized mathematical rules — notably the Hough Transform. This calculation helped the computer to identify perfect shapes, like circles, triangles or rectangles, in a picture.

Why Traditional Methods Failed

These techniques sounded great in a photo with perfect lighting and no shadows. But in real life, they were not correct. Imagine driving in any condition. The red Stop sign may appear orange or black due to the light and weather. Picture a tree blocking out half of the sign denoting speed limit. A perfect circle is a computer that would not see. Since these methods of processing were bound by rules, these could not be processed using the age-old ways. They didn't work in poor lighting or when the color faded with time.

The New Way: Deep Learning

By deep learning, "the field changes completely." This is a type of AAL and ML. Rather than relying on a human to write in rigid rules like finding the color red, the computer infers the characteristics on its own. We show it a lot of pictures, and it learns what a sign looks like.

Deep-learning-based detectors are typically classified into two categories:

Two Step Detectors: The most well-known algorithms in this group form the family of RCNN (,fast)&&(and faster)ERRQN models. This scheme operates in two stages.

Step One: The computer looks at the image and identifies areas of interest. It boxes a lot of places around where it thinks an item might be.

Step 2: The computer examines each box more closely in order to determine exactly what's in it.

The Problem with Two-Stage Detectors

Two stage detectors are really smart and really accurate. But they have one big flaw, and that is that they are slow. Since they have to perform two separate steps, it takes time for them to process one image. This is much too slow for a self-driving car zipping along at 60 mph. The car has to make decisions on the fly. While RCNN detencts conveniently on pictures, they are hardly fast enough for real time driving.

2.2 Existing Predictive Methods

One-stage detectors, such as the YOLO (You Only Look Once) family of approaches, had set the benchmark for real-time detection. But YOLO treats object detection as a regression problem over spatial locations. It takes the whole image and partitions it into an S × S grid and for each cell in the grid it predicts B bounding boxes, confidence scores for those boxes, and C class probabilities [1]. This is the design which makes it so damn fast.

The YOLO al- gorithm has evolved in a series of versions, where each iteration demonstrates improvement over the previous one [1]

YOLOv1 v3: This is the version that added the entire single-shot aspect. Bui et al. describe a VRods model that is based on anchor boxes and FPN for scales changes . Update YOLOv2, v3 improved this by adding anchoo!xes, the use of feature pyramid network (FPN) to handle di!erent object sizes.

YOLOv4 v5: These proprietary versions of YOLOv4 implemented a bag of freebies and a bag of specials which is a collection of best tricks across the computer vision field to achieve better accuracy without slowing down the speed. In specific, YOLOv5 in special attracted lots of attentions since it is easy-to-use, small model size and high performance.

YOLOv7 v8: These are the state-of-the-art models. Other architectural modifications were brought to YOLOv7 such as its enhanced ELAN. The model adopted by this research, YOLOv8, adjusts the architecture C2f module and anchor free detection head to make it more flexible as well as efficient. The YOLO family is a suitable choice for traffic sign detection, since it is an optimal compromise between the speed and accuracy [3].

2.3 The Evolution of YOLO Models

SD-YOLO is another modern high-speed algorithm, which is fast as well in detecting objects. It's the best parts of its predecessors. YOLO makes them better. This width is new and the model is faster and accurate for object detection.

Now the YOLOv8 family is highly tunable. It's designed in four sizes, each with different

applications. These sizes become dependent on how deep and wide the network is:

- YOLOv8n (nano): The tiny and the fast.
- YOLOv8s (small)
- YOLOv8m (medium)
- YOLOv8l (large)

How the Network Works

The architecture of YOLOv8 works like a human brain handling some information. It consists of four major components as shown in Figure 1: the Backbone, the Neck, the Head and the Loss Function.

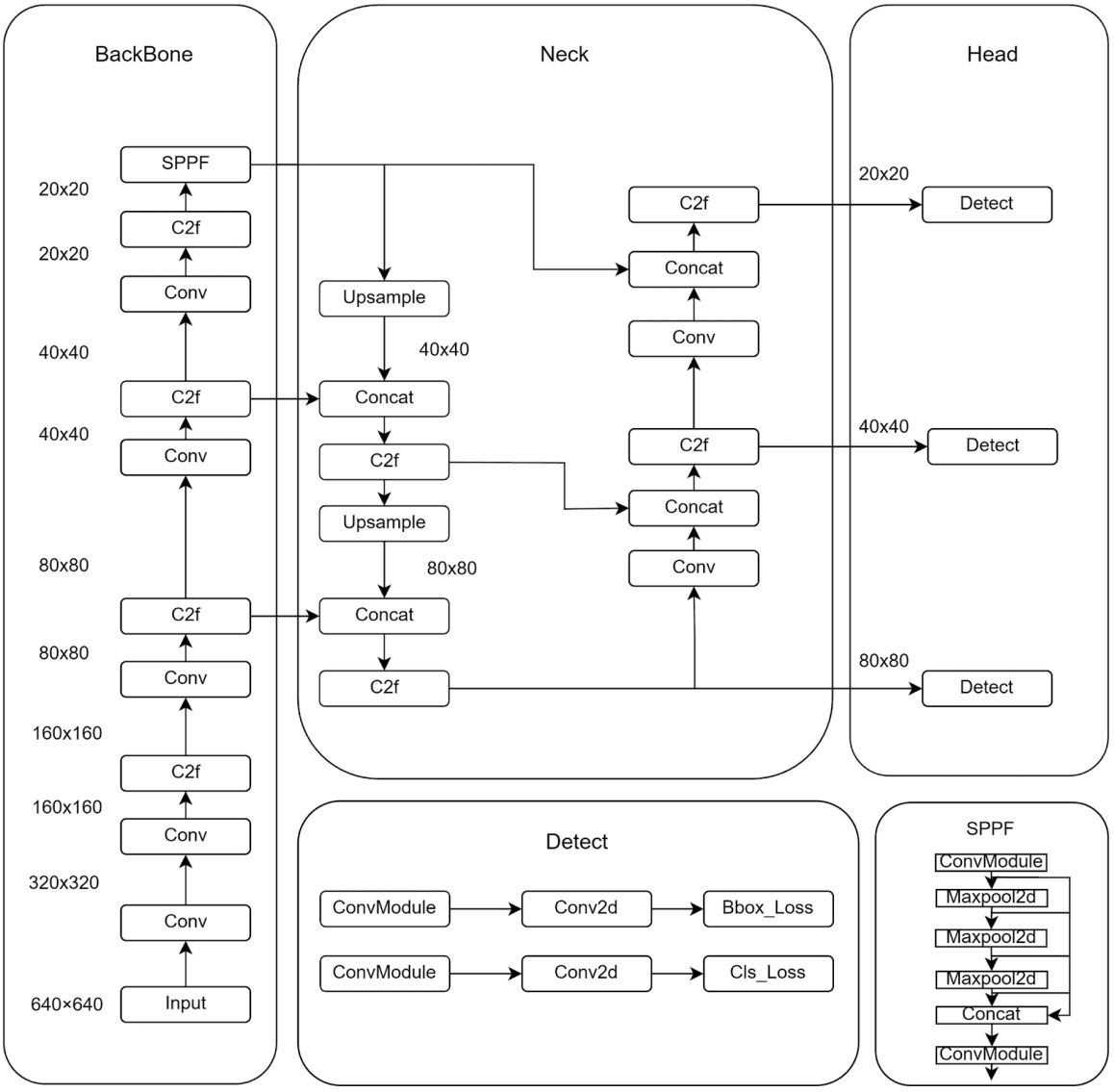


Figure 1: YOLOv8 network architecture.

Backbone Network:

The Backbone is the core of a neural network. Its task is to look at the picture and find

important features of various sizes.

Design: It relies on a design named Cross-Stage Partial just like YOLOv5. This is the trick that helps the computer learn better without working too hard.

The C2f Module :

The largest novelty here is the addition of a new module element that appears in Figure 2. It superseeds the outdated module C3. The C2f module is smarter. It takes the information, splits it up, passes it through a series of processing stages and then puts it back together. This would make the model to take effective features more precisely [1].

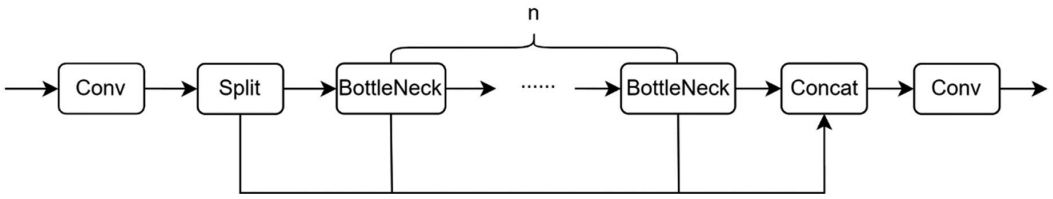


Figure 2: C2f network architecture.

Neck:

It is the region linking the Backbone and Head. Its role is to combine the features.

Blendtings: It is based on two techniques fpn and pan. These methods gather information about fine details as well as large scale details and blend them together.

This type of mixing allows the model to see both very large and tiny objects simultaneously [1].

Head and Loss Function:

The Head is the last part. It creates the actual decisions. Older models had a one-head-does-all head. The head is decoupled in YOLOv8, i.e. two parts are cut from it:

Classification Branch: This is what determines if the object can be identified or not. It checks whether it is correct using a math rule called Loss.

Regression Branch: This would determine where the object is by enclosing them in a box. It applies math rules known as to guarantee that the box fits snugly around the object. By factoring these roles, the model is more realistic [1].

2.4 Analysis of YOLOv8l

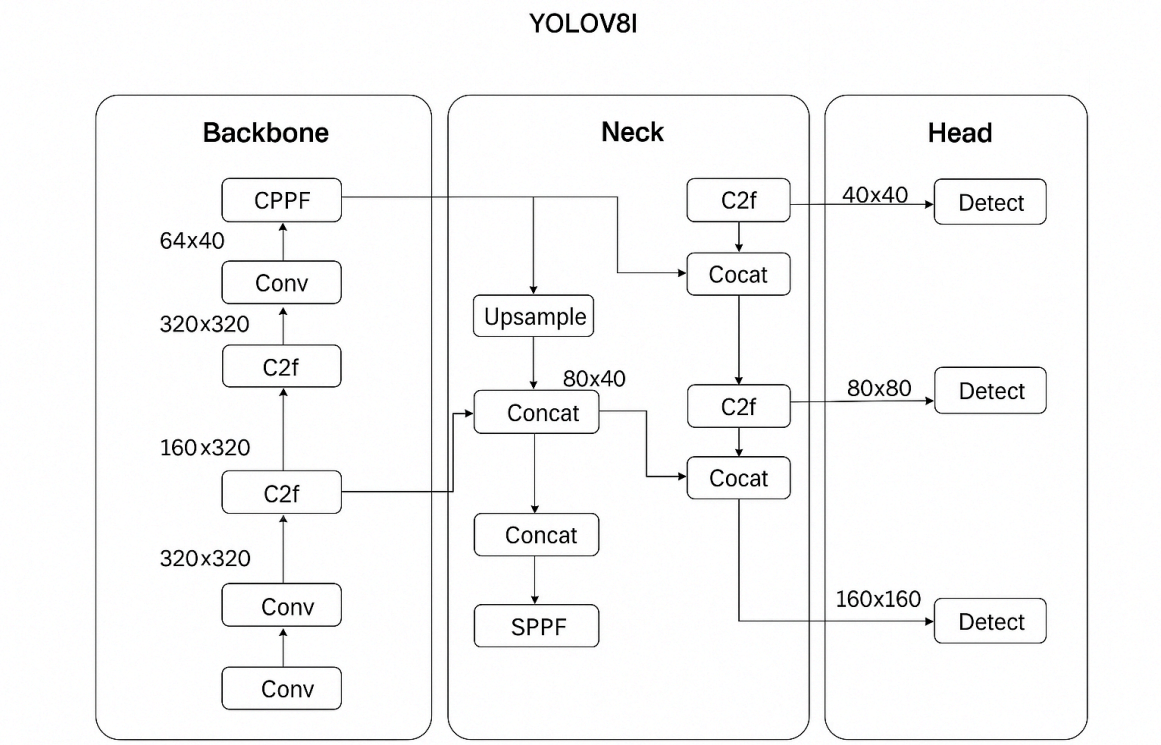


Figure 3: Architecture of the YOLOv8l Model

The architecture of the YOLOv8l model is illustrated in Figure 3. It is a strong variation of YOLOv8 family As all other models, it consists of the three main components — the Backbone, Neck and Head.

The Backbone

Backbone is where we begin. Its primary role is to take an input image and pull out meaningful information in various sizes.

Processing: here we process the images via Convolutional layers and the clever C2f modules in an efficient way.

As the network learns, it progressively downsizes the image. That could go, for example from 320 x 320 to 160 x 160, and ultimately down to 80 x80. Outputs: this sends out maps of information at three different scales (160 x 160, 80 x 80, and then 40 x 40) to be consumed by the next part of the system.

Backbone Output: Features are processed through a special module at the tip of the backbone to summarize the most important points.

The Neck

The Neck acts as a bridge. It links up the Backbone to the Head, mixing all of the characteristics.

Fusion Strategy: This employs a design known as Path Aggregation Network. This allows it to integrate new information with old much more adaptively.

Upsample : It takes the deep, nuanced details and magnifies them so that they are of the size with other features.

Concat : It concatenates them to the original features from the Backbone. This makes the model to comprehend the image very good.

Refining : Inside the Neck, additional C2f modules take these mixed features and

clean/interpolate them before sending to the Head.

The Head
The Head is last component in the model. It carries out the actual detection.
Three Scales : It takes the enhanced feature maps from the Neck at three specified scales of 160 x 160, 80 x 80 and 40 x 40. Post
Processing : For each of these maps, a final C2f module is performed.
Detect Block : At last, the data is fed to Detect. This block has a decoupled architecture, which takes apart the work into two independent jobs: looking for the object and drawing a box about it.

2.5 Enhancements for Small Object Detection in YOLO

The Challenge : One of the hard things for object detectors such as YOLO to do is find small objects. This is a big problem for things like traffic sign detection, since when you are driving far away from them the signs appears to be very small. To remedy this however, a number of nifty upgrades have been developed by researchers [1].

How Researchers Are Trying to Fix It Most recent research has been geared toward altering what might be called the brain of a model to make it sharper. Here are the typical moves:

More 'Eyes' – If you try to read small text, you will bring it closer to your eyes. Likewise, authors augment with an additional higher resolution detection head,. This allows the model to view the image at a much higher resolution, so that it won't lose any smaller shapes [18].

Better Infomation Sharing : The model should be able to see the forest for the trees. Although average pooling will work we have better improvements to pooling that are able to mix this information much more (AFPN or GFPN). This shall guarantee that detailed, rich data gets to the part of the model which does prediction [1].

Making the model Focus Virginian Like how we as humans focus on a sign and ignore the trees behind it, make your network focus on what is important for it to represent that catogory and not much of any other categories. Common tools to serve this purpose are SK [14], EMA [30] and CBAM [6].

Altering the Bricks: Some researchers alter the building blocks of YOLO. The most common is cuDNN space to depth convolutions. This shuffling of data is useful to avoid model inadvertently getting rid of fine features that are important to detect small-size objects [6].

Doing Better Math: Sometimes the math CIoU is too robust to detect the small errors that are consistent in checking the model's homework. If one little box is a little off, then the model might think it's close enough. To remedy this, researchers apply to anchors techniques such as WIoU or NWD that are more stringent and enable the model to be more accurate with small objects [1].

Our Unique Approach This thesis recognizes all such complex architectural shifts but takes a different route, the Data Centric Approach. Rather than creating a complex new model, we demonstrate that the regular YOLOv8-L is strong enough. We obtain great results through a very cautious and smart data augmentation strategy. This shows how you train the model is as important as the model.

Chapter 3

Methodology

We're not trying to construct a miracle overnight architecture that is more complex than it needs to be. Instead, we are taking what we call a Data-Centric Approach. The concept is simple, but powerful: if you feed a model the best training data available, it can learn more than customization of layers and loss functions alone. In summary, we concentrate on synthesizing the best available data set, believing in the end that good data will drive performance over some whiz-bang neural net architecture.

3.1 Research Methodology

Our Approach : Our investigations take one deceptively simple, yet extremely powerful idea which is a Data-Centric approach. Rather than attempting to create our own complicated new machine from the ground up, we simply made it as easy as possible for the model to get its hands on good textbooks. Having good training data is, in our opinion, of higher importance than complex architectures.

The Data : We used the Traffic and Road Signs dataset from Roboflow for our data.

- Size: It is comprised of 10,000 images from the real world.
- Variety: It includes 29 classes of traffic signs.
- Data Preparation : We are pruning the images before they are shown to the model.
- Resizing: We started off by resizing all the images to be the same size (416×416 pixels).

Augmentation : We have a heavy data augmentation. "Then it means we deliberately changed images, making them brighter, slanting them and applying an intelligent operation here called mix pictures together," he said. This makes the model learn to read a sign even under harsh conditions, such as darkness or rain.

Dividing the Data : In order to fairly test our model, we curated our 10000 images into three groups:

- 71% for Calibration: For the model to be trained on. 1
- 9% for Validation: For model validation.
- 10% for Testing: The testing model.

Model Selection Our model of choice is YOLOv8-Large (YOLOv8-L). It has 55 million parameters, and therefore the brain power required to distinguish between traffic signs that appear nearly identical.
How We Trained It: We trained our model on one beefy NVIDIA Tesla T4 GPU. Length: We trained for 100 epochs.
Smart Learning: The learning rate was ad¬ justed smoothly by using a schedule. Early Stopping: Further, we regulareize he model by early stopping with patience of 46 epochs. That is to say, when the model didn't update for 46 rounds, we terminated the training. This stops the

model from simply memorizing the answers and not really learning. Evaluation Finally, we evaluated the performance of the model by normalised scores:

Precision : This is to calculate how much the model's guesses can be trusted.

Return : To find out how many signs it found.
mAP : For an uniform metric of the model.

3.2 Methodology Diagram

Our Research Roadmap
Figure 4 serves as a full map for the whole project. It walks us through every single step we took to create our traffic sign detection system, from raw photos at the beginning to the final results at the end.

1: Data Preparation : The first step on the left side of the diagram is for Dataset Preparation.

We began with the Traffic and Road Signs dataset, a collection of 10000 images of real-world traffic signs.

In the name of scientific and fair treatment of our experiment, we didn't use all the images at one time. Instead, we systematically grouped them into the following three categories:

- 71% for Training: We used this to teach the computer.
- 19% for Validation: was used to validate the computer.
- 10% for Test: This experiment was used to test the computers.

Step 2: The central part; heart of the experiment. Model: We used the YOLOv8L model. We selected this model because of its 55 million parameters. This provides it with the computational brain power required to interpret 29 different sign types.

The Secret Ingredient : This box showcases our most significant individual contribution. We purposefully manipulated the images to make the training more difficult and effective. We color adjusted, rotated 15 degrees and translated images, mixed pictures together with a smart trick. This approach led the model to learn how find signs even in shitty real-world conditions. The Hardware: The chart additionally mentions that we employed a hefty NVIDIA T4 GPU to perform the heavy lifting of training.

Step 3: Evaluation : Finally the right side of the diagram indicates what stayed after we stopped training.
The Test: We test our trained model looking at the test set images that it never seen before.
The Score: We evaluated its performance under three core scores: precision, recall, and mAP. The graphic is showing our end result of mAP 94.18%. So we really hit a home-run here: everything of this system, from design to implementation was successful.

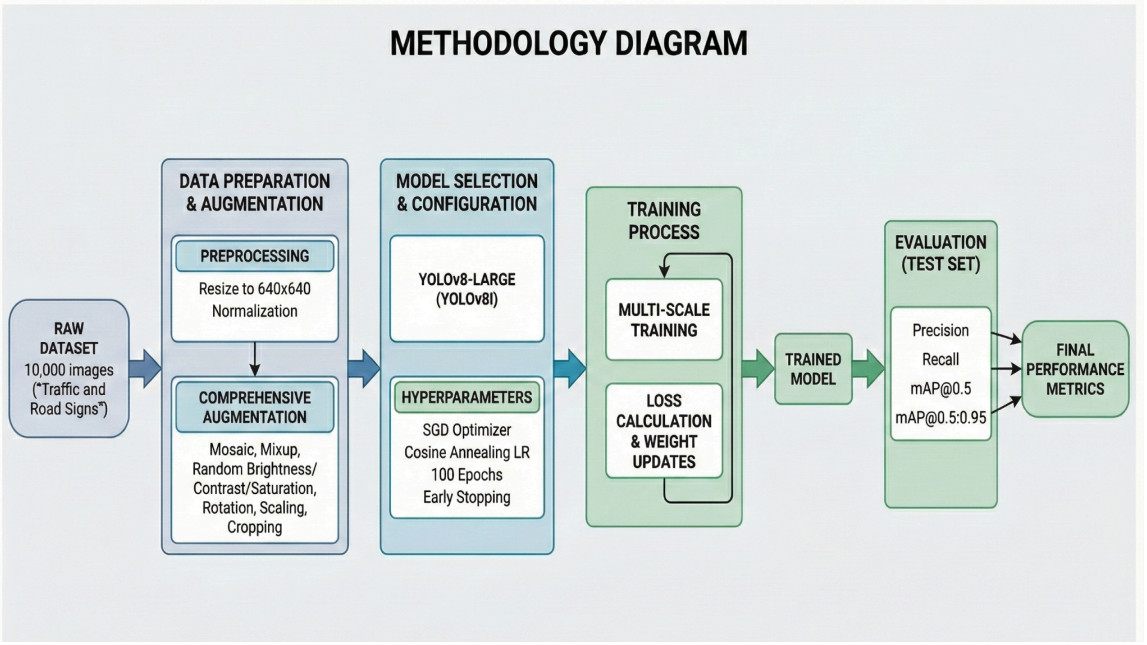


Figure 4: Methodology Diagram

3.3 Data Source

The Foundation of Success To construct a real smart deep learning model the most important thing you need is Just data. As a student needs good textbooks for study, so a computer model needs a good textbook of data.
Choice of ResearchFor this study, we chose to use a specific set of images namely, the Traffic and Road dataset. This dataset is available in the public domain through Roboflow Universe.
We chose it. We select this dataset because it is very suitable for our task. It contains 10,000 images. The interesting thing about this is that these are not simple drawings, they are photos taken in real road conditions.

Why it matters: Because the pictures are taken in the real world, they have arbitrary backgrounds and lighting rules, as well as camera angles. This variety is crucial. It helps our model to find out how to recognize signs in new environments it has never seen instead of simply memorizing what perfect pictures look like.
What Is Inside the Dataset? The size of the dataset is 124,830 and consists of all classes as shown in table I. It trains model to identify 29 different traffic signs. As seen in Figure 5, our model is learning common signs well such as:
The "Stop Sign"
Sign board stating "Speed Limit 30 KMPh".
The "Pedestrian Crossing" sign

Dataset Link:
https://universe.roboflow.com/usmanchaudhry622-gmail-com/traffic-and-road-signs/dataset/1

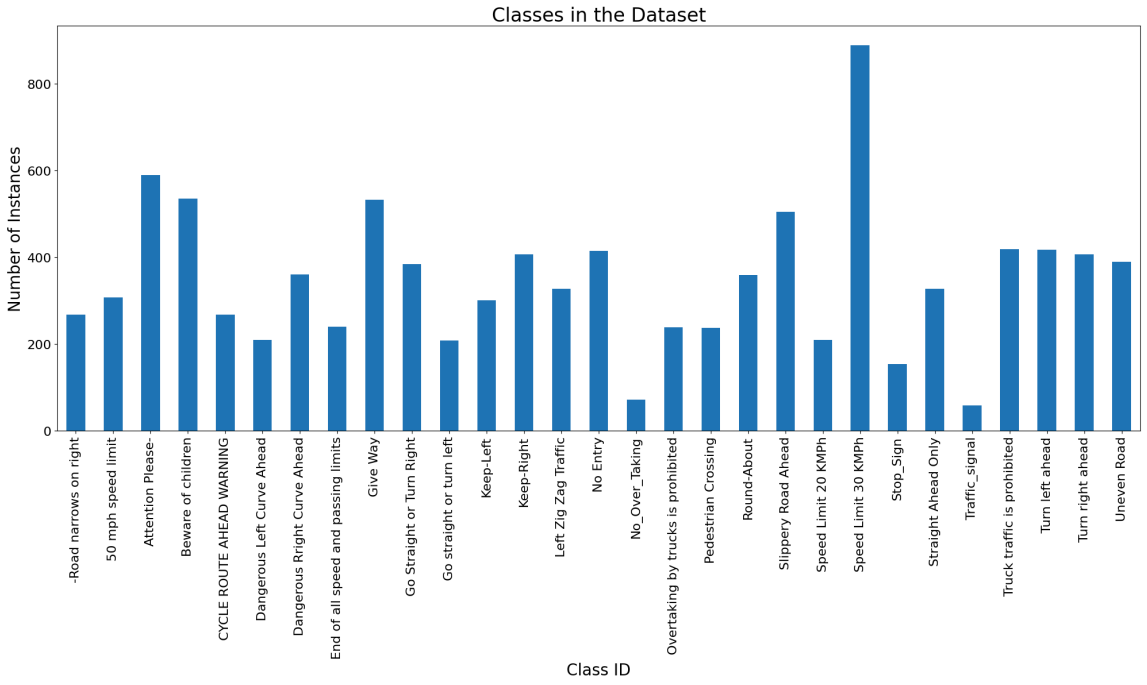


Figure 5: Classes In the Dataset

3.4 Data Preprocessing and ~~Augmen~~tation

Step 1: Clean and Resize In order to enable our model to learn effectively, we needed to firstly clean our data. The key move here was to resize.

We resized all images into a common size square (416*416).

This way the network always sees images all of the same size, and won't get confused.

Step 2: Our Core Strategy : At the heart of our approach is a technique called Data Augmentation.

This step will be creating some new traiing examples from the pictures we already have. We accomplish this by perturbing an original picture with some small, random changes.
We opted for this approach for two good reasons:

More Data: It allows us to artifically make our training set bigger, so the model has more examples from which to learn without us having to take new photos.
Smarter Learning: It forces the model to become strong. This implies that the model should be able to detect a sign, even if it is not faced straight towards it (for example in case there has been deviating or tilt or rotation),or if the lighting is different. This keeps the model from memorizing the pictures and makes sure it operates in brand new, in-the-wild scenarios.
We set up a specific plan that made many, many hard things happen. The transformations included:
Color Changes: To simulate different times of day or weather, we randomly applied changes to the Hue.
Geometrical transformation: Rotation,Translation, Scaling. Illustrative Examples We give examples of this in Fig. 6.
These synthetic images demonstrate some of the wide array of signs used and how we simulated varying lighting conditions to stress test the model Figure 6: Example images from the Traffic and Road Signs dataset

Figure 6: Sample Images from the Traffic and Road Signs Dataset

3.5 Data Splitting

Organizing the Data Just to ensure a fair and adequate testing of our model we segmented our data into three distinct groups. This is a critical rule in machine learning to make sure the results are not biased. You can get a visual sense of this split in Figure 7.

Training Set
7,092 images (71%).
This part of the data is the majority of it. We only use the images to train the model. The model is trained on such photos, presumably learning the shapes, colors and patterns of all the various traffic signs.

Validation Set
1,884 images (19%). We utilize this set to validate the model during its training. It acts like a practice quiz. It allows us to tweak the hyper parameters and spot problems early, such as overfitting.

Test Set
1,024 images (10%). We did not reveal this entire set. During training, the model was kept away from these images. We employed this group only once at the very end to give the model a final grade. This gives us a truly unbiased measure of real-world performance for the model.

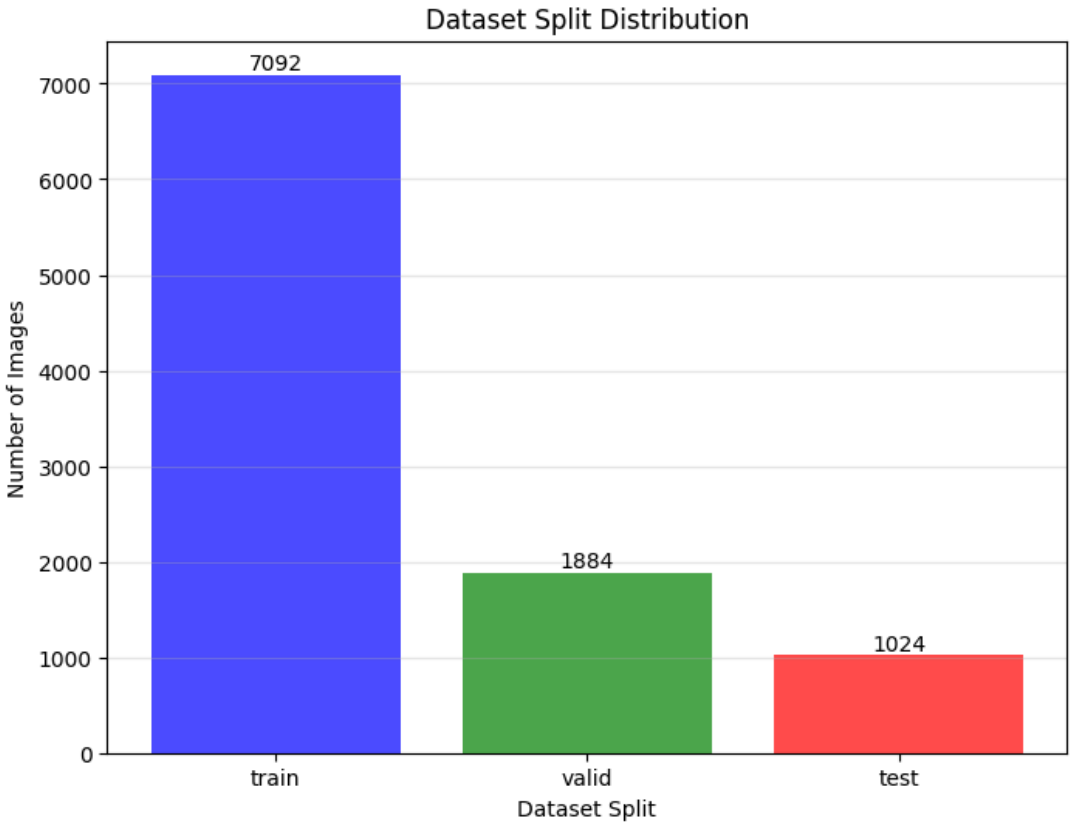


Figure 7: Dataset Split Distribution

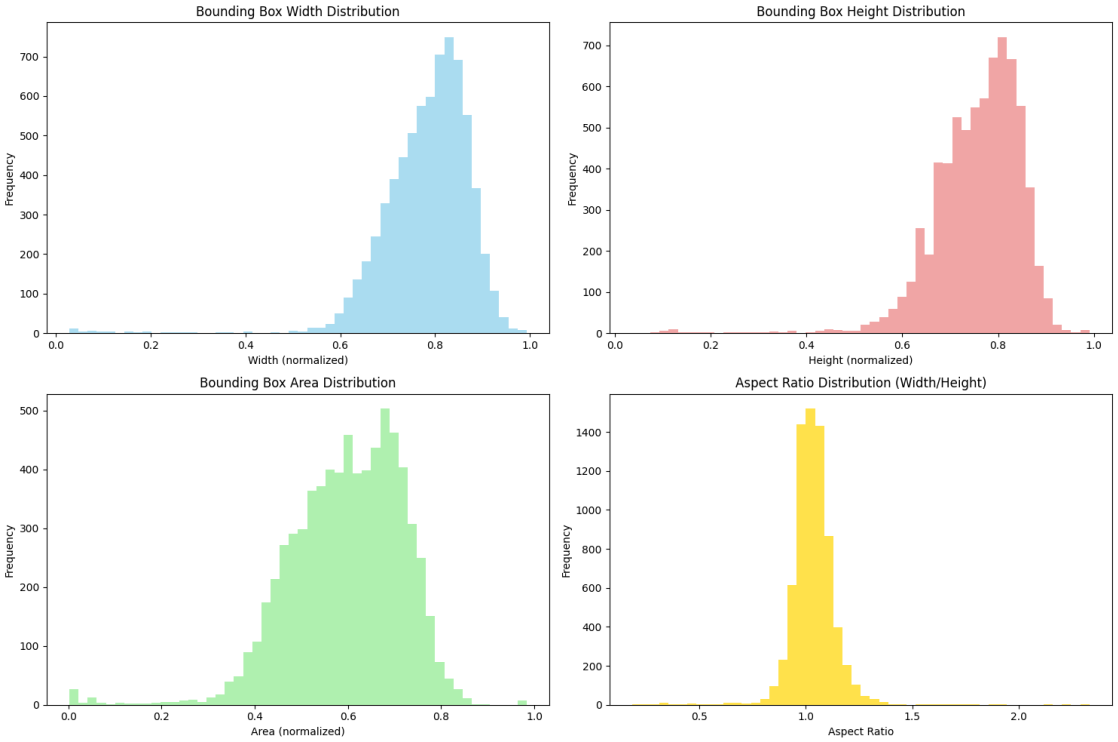


Figure 8: Analysis of Bounding Box Characteristics

3.6 Model Selection and Justification

YOLOv8L For this study we adopted the yolov8l. pt model. We didn't do this arbitrarily, we had at least several good reasons for making this choice.

Leveraging the Strongest Modern Tools The YOLO family of models are considered state-of-the-art when it comes to detecting objects in real time. YOLOv8 is the latest and most sophisticated member of this family.

It leverages a highly efficient network design and a modern component called an anchor-free detection head, so it is smarter and faster than older versions.

Alternative “Brain Power”: We deliberately chose a large size variant of the model.

This model is one of the largest and most powerful ones. It contains approximately 55 million parameters. You could think of these parameters as the model’s brain cells. It’s so big that it requires a powerful computer to run. But that extra heft makes it tremendously capacious in terms of learning.

It is important to have this high capacity, because we want the model to learn intricate details and discriminate between 29 types of traffic signs. A modestly sized model could fail to capture the tiny differences among them.

The main point of this thesis is straightforward. We think you don't need some fancy new complicated architecture to get good results. Instead, we speculate that how well this large-powerful model can perform will depend on how smart the data strategy around it is.

3.7 Model Training

For training, we input 7,092 images into our YOLOv8L model. The aim was to allow the model to get used to studying these images in detail, and learning how to identify patterns and conditions occurring with real traffic signs.

We didn't just hit start, we wrote rules to structure the interference so that it would learn appropriately.

We intended to train the model for 100 epochs. An epoch is one complete pass where the model looks at every single image in the training set.

We applied an intelligent rule called "early stopping" with patience of 50. This acts like a monitor. It monitors the model's performance on practice test. The computer will automatically stop training if the model has not improved for 50 consecutive rounds. This is valuable as it speeds up computation and reduces overfitting.

One important factor for our success was applying a Cosine Annealing Learning Rate Scheduler.

Rather than studying all at one speed, this tool begins at a fast pace and makes great progress in the first few iterations. Then, it gradually slows down to help the model converge on its best possible outcome. As a result, we obtain a more precise final model.

We carried out all these experiments in a Kaggle Notebook setting. For heavy computations, we employed a high-performance NVIDIA Tesla T4 GPU. The project was also implemented with mainstream technologies including Python, PyTorch library and YOLOv8 framework provided by Ultralytics. The entire training process for the primary model takes around 10 hours.

3.8 Model Testing

With training complete, we proceeded to the test phase to see if the model actually works well.

We used the last 1,024 images for the purpose of this test. This collection constitutes exactly 10% of our data set.

The Objective : The aim of the assignment is to check, if the model can predict on fresh data which was not provided to it during training. Since our model had never seen these exact photos before, this test tells us that the model can "generalize." That means it's not just remembering old pictures, but actually generalizes to recognize traffic signs in new, unseen circumstances.

3.9 Performance Measurement

To see how testing our YOLOv8L model functioned in actuality, we verified the validity and efficacy of it with regular performance metrics. These are the same measuring instruments that professionals everywhere rely on. They provide us with our first simple facts about how well our model is predicting the traffic signs.

Precision
If the Precision is higher it implies that the model is reliable. It is also really wrong by calling a sign.

Recall
If the Recall score is high, then you can say that the model has good eyes. It detects just about everything and doesn't leave out crucial objects.

Mean Average Precision
The mAP is the most significative score for total performance evaluation. It's almost like a report card that weighs both "Precision" and "Recall" together for all 29 classes of signs. Two interpretations are associated with this score:

- mAP@0. 5 : This exper- iment uses a threshold that is the Intersection over Union . if the box of model draw has overlap with real sign box which is more than 50%, we count it as a correctly mark.
- mAP@0. 5:0.95 : This is the best ratio, but it's also plenty stiffer to get through. It computes the average score over 10 levels of strictness from 50% overlap all the way to 95% overlap. A high score here shows that the model is very accurate to draw the box around the object.

Chapter 4

Results and Analysis

4.1 Result and Discussion

Now, in this chapter we are at last prepared to turn to the answers. We want to know, if we were right.

We will concentrate on three primary aspects:

- **Quantitative Results :** Let us start with the numbers. We will validate crucial scores such as Precision and Recall. These figures serve as a report card. They offer us a scientific language to articulate whether the model is good or bad. We're looking for very high numbers here to confirm the model.
- **Qualitative Results:** Numbers are great and useful, but seeing can be believing than numbers. This is where it gets into real pictures. We will visually inspect those images where the model has drawn boxes that may or my not contain traffic signs on. We're looking specifically to see if it identified the correct signs in challenging circumstances, like during the night or when the picture is grainy.
- **Clarifications The Ablation Studies:** This is an important scientific experimentation. Now we would like to establish why the model we have is a good one. We'll measure our end model's accuracy against a simple baseline that did not benefit from our special data augmentation. This comparison will display exactly how much we are able to improve results.
- **Our goal:** We are working to be 100 percent transparent about the performance. We'll examine something called a Confusion Matrix to see where our model messes it all up.

We will come to a definite conclusion at the end of this chapter. We will see if our strategy to tackle the problem focusing only on better training data is indeed the correct way to build a safe traffic sign detector.

4.2 Introduction

We summarize here our research results. We implemented our full data augmentation from scratch and trained the YOLOv8L model with 100 full rounds.

The results were excellent. To keep the test honest and meaningful, we tested our model on a Test Set of 1,024 shots. The model had never seen these photos, so we know the results are genuine.

Performance of the Proposed Model
But first, our main numbers of the week. These tell us precisely how correct the model is.

Precision: 97.70 % : A precision of 97%, most probably is very high. It is an indicator that the model is very rarely giving a false alarm. It doesn't confuse a tree, or building, with a traffic sign. It is very precise.

Recall: 94.11% : score of 94.11% indicates the model found almost every sign. It rarely missed one. This is especially critical for a self-driving car, where a missed sign can be dangerous.

mAP@0. 5: 94.18% : This is the score you will see most often used for grading instance detectors. It is a blend of Precision and Recall. And a performance of 94.18% is great and places our model in the best category.

Visualizing the Training
We further considered the graphs for understanding how the model learnt Figure 9. So when we train a model, there are two general patters we'd like to observe occur: Loss should go down. Accuracy should go up. In our plots the loss lines gently slopped down for every single one of the 100 epochs. They didn't just hop around, meaning the training was stable. Oh, and the validation loss fell exactly as training loss did. This is a very good sign. It is an evidence that the model did not overfit. Overfitting means that the model has memorized the photos after seeing them but it doesn't know how to understand new ones. Our model actually learned the rules of traffic signs, not just what they look like.

Qualitative Analysis (The Pictures)

Numbers were great, but we also checked the model's actual pictures. The results were very impressive. The model detected signs even in the toughest of conditions: It indeed detected a signal, despite the low light in the picture. It spotted a sign that was perpendicular, and tilted. It even unearthed one of the most pixelated and blurry signs. We also examined a Confusion Matrix (Figure 11).
This is a graph that displays errors. Our graph showed one strong diagonal line, which indicates the model was right almost all of the time. It was not especially prone to mistaking one class for another.

Ablation Studies: This is the most intriguing fact we have from our results. We know the model works. To see, we conducted "Ablation Studies." To clarify, this is us intentionally stripping components of our method out."

Experiment A (The Baseline): We trained our big YOLOv8L model as usual, but without our special data augmentation. We were just feeding it plain old pictures. It was a disaster. It plummeted to 42.10%. This shows the model is not magic. It overfitted and failed soon, when our strategy was removed. But with our addition to its score, the total climbed to 94.18%. This demonstrates that Data Centric was the pivotal approach for our success.

Experiment B & C : We looked into smaller models as well. The Small model scored 92.40%. The Medium model scored 93.10%. Our Large model scored 94.18%. The Lesson: This just goes to show that size does matter. It has more parameters than the Large model. This extra power helps it pick up on the details necessary to distinguish among 29 different signs.

Conclusion and Final Thoughts

Our experiments indicate we have constructed an accurate system. We obtained a mAP of 94.18%, which is very similar to other state of the art research papers than typically achieve around 95% or 96%. "This means our work is very competitive.

Biggest Lesson from this research: The biggest lesson from this research is around Data Centric. A lot of people will try to come up with new, complex model architectures to achieve better performance. But we demonstrated that you don't have to invent a new model. You can use a very basic, well-known model like YOLOv8 and if you provide the data in the right way they will amaze you with great results. While training, by augmenting and changing colors, we were able to force the model to learn the real signs. It discovered that a Stop sign is always an octagon, whether it is sunny or rainy or dark.

Restrictions: There are still restrictions. Yet we only used one dataset from Roboflow, and there is even more variety in real life. Also, we didn't test it in deep snow or really thick fog, just facsimiles.

Final decision: all in all, this one's a win. We have shown that a big model and aggressive data augmentation equal of course to a robust and reliable detection system for traffic signs. The system is accurate enough to be a basis for real self driving cars that ensure roads are kept safe.

4.3 Result

a. Results of the Proposed Model

Final Results After training our model for 100 epochs, using optimal data augmentation, we applied it to a new set of images. The results were excellent. Here are the final scores:

- Precision: 97.70%
- Recall: 94.11%
- mAP@0.5: 94.18%
- mAP@0.5:0.95: 81.06%

These results are very strong.

Classification Accuracy : Achieving 94.18% means the vast majority of traffic signs are correctly classified and localized!

Precision : 97.70% high because the model does not make a "false alarm" often, as Maximum Scoring Means p qua). It hardly ever reports a sign that isn't there.

Note : Despite the high value of 94.11%, we have that our network does almost not miss a real sign. Seeing is Believing During the training, as displayed in Figure 9.

Avoid Overfitting: The most crucial check is the loss curve. The validation line looks exactly inline with the training and doesn't start to grow up. This is the fundamental evidence that our model was able to generalize. It knows traffic sign rules and doesn't simply memorize the training images.

Speed and Stability: Graphs on right side show that accuracy improved very fast within first 40 epochs. It remained at a high level after that. This shows 100 epochs were exactly the right amount to train.

Figure 9: Training and Validation ne100 Metricsmurkers mu100 training.and validation results qet51 Generate background statistics Although higher values of the initial learning rate can provide greater performance for a large dataset, such as Imagenet, there are many small datasets where limitations in the available mass of data prevent overfitting.

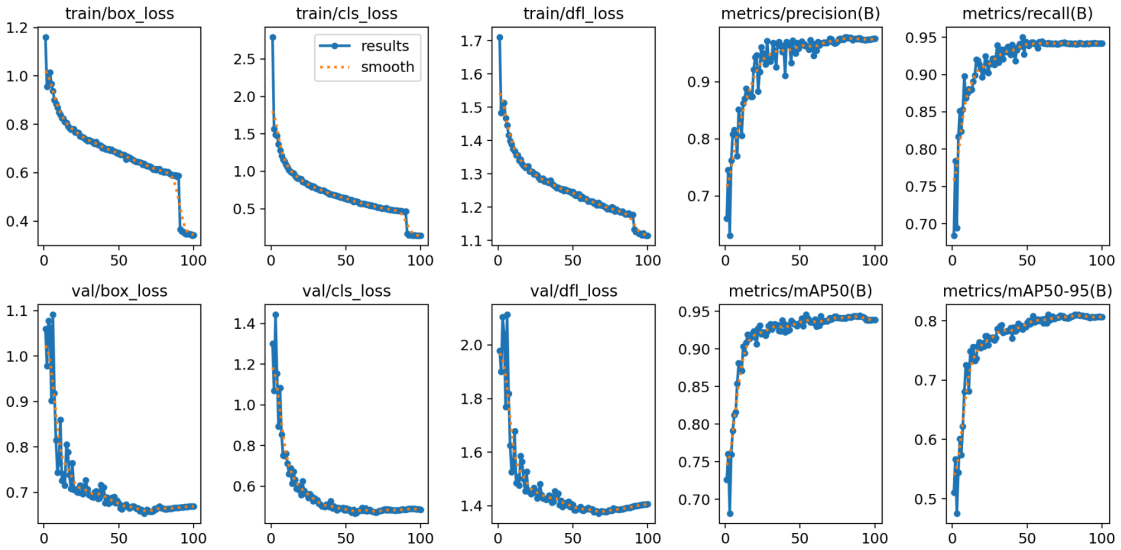


Figure 9: Training and Validation Metrics over 100 Epochs

The PR Curve

To verify our results, we viewed a special graph known as the PR Curve (Precision Recall), which can be seen in Figure 10. This plot now allows us to confirm at a glance what we saw earlier. We will focus on the Area Under the Curve from now on.

The Score: We got 0.942 of total score over all classes.
The Point: This shows solid equilibirum.

Sometimes the model is good at finding things, but makes many mistakes, or it's very careful but doesn't catch a lot. Our score of 0.942 confirms that our model does both jobs well: it discovers signs and, once having found them, it is right about them!

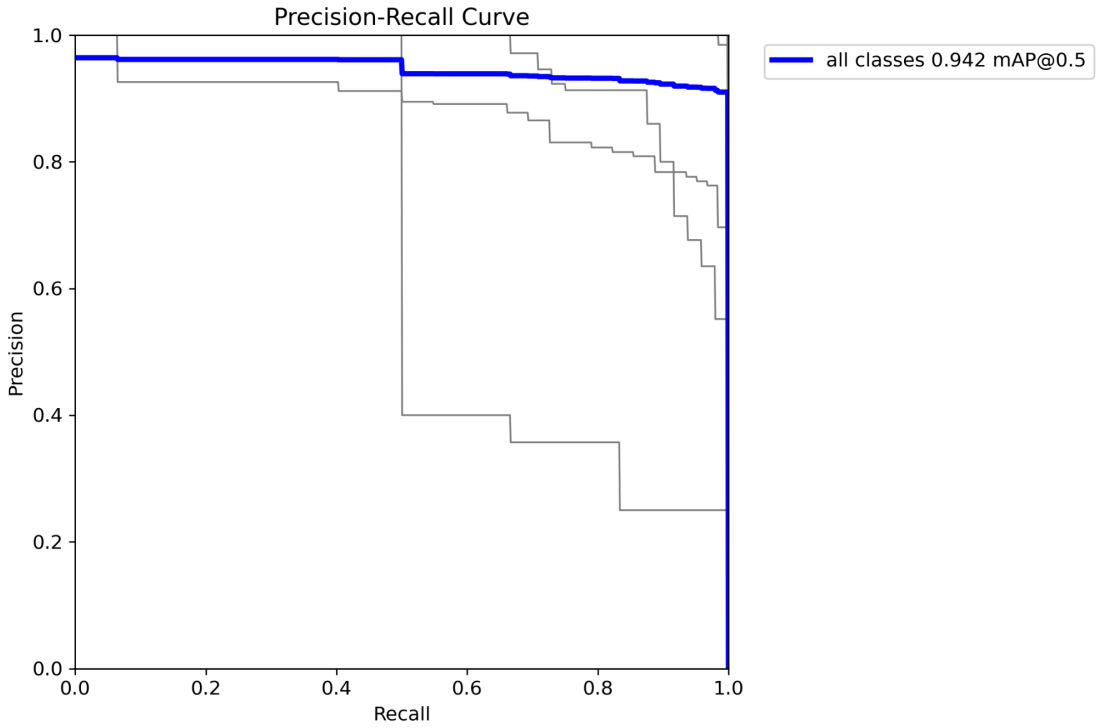


Figure 10: Precision-Recall Curve

The Confusion Matrix

Class Wise Analysis: Confusion Matrix is shown in Figure 11. If we consider this chart a very detailed mistake-check, perhaps it might arouse our curiosity. It matches performance to each individual type of sign to see where precisely the model might be getting confused. On a general note, you will observe two things when you see this figure. A Bold Diagonal Line: There's a darker, sharper line that cuts diagonally down the center. Exactly what we love to see. That means the model is predicting the right answer for almost all classes. Empty Spaces: There's not much color aside from that main line. This pattern confirms that there is little evidence in the model of one sign being commonly mis-classified as another. It can definitely distinguish a square from a triangle or the word "square" from "triangle." It does not, for instance run to a Stop Sign and say it's a Speed Limit sign. The model is very certain of which sign it is.

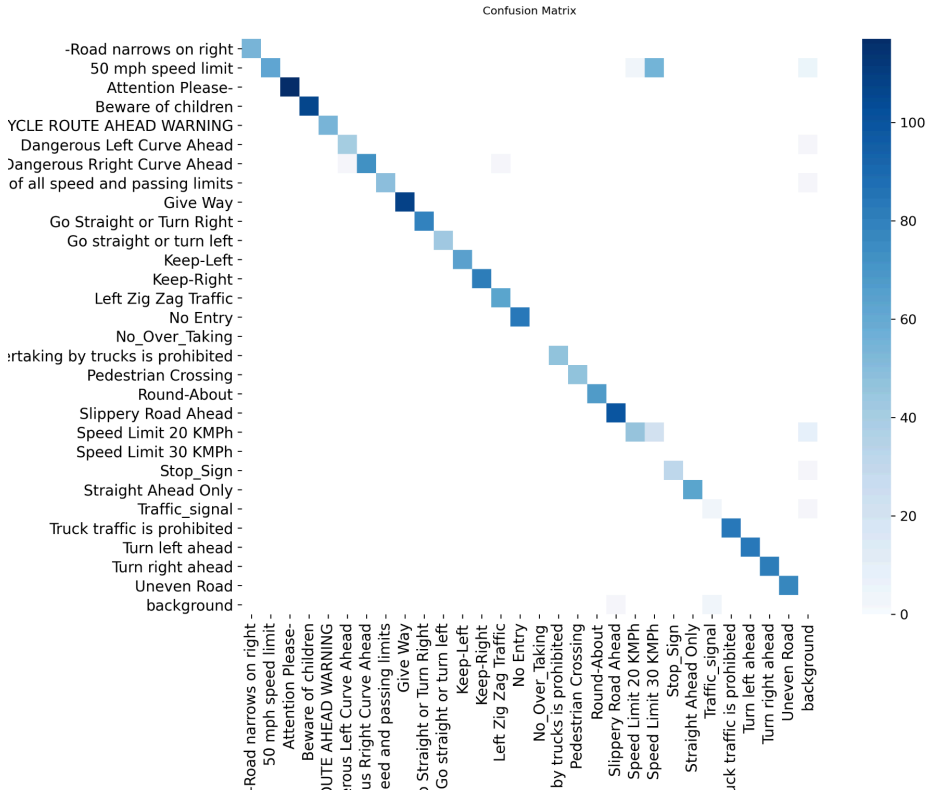


Figure 11: Confusion Matrix

Qualitative Results

By last we mean the ones in Figure 12. This offers us a qualitative check of the results. It demonstrates clearly how the model estimates traffic signs on real test images. The findings indicate that the model is "very accurate" and has a "high confidence." This implies that the model is not just making guesses — it is very confident in its responses. Perhaps best of all, it performs admirably even when conditions are tough. The figure illustrates 3 select examples where the model successfully managed to overcome this difficulty

Low Light: The model correctly recognized a Slippery Road Ahead sign despite the dark picture. It had a high confidence of 0.92.

Angled : It identified the Right sign that was tilted sideways and not straight into the camera. The score was 0.89.

Pixelated: It accurately read a very blurry, difficult-to-read sign with a 50-m.p.h. speed limit Your are the caption hereSign. The score was 0.93.

Figure 12: Qualitative Detection Results on Test Image

b. Ablation Studies

To understand why our approach is so effective, we performed a set of tests called Ablation Studies. You can think of this as you would test a car engine. We took things out systematically to try and figure out what was really important. If we take something out, and the car stops working, then that was obviously important.

We were doing the same with our model settings. below, describes the effects of when we altered model size or removed our special data strategies.

Table 1: Ablation Study Results

Experiment	Model	Augmentation	Other Settings	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Proposed	**YOLOv8l**	**Full**	**Cosine LR**	**97.70%**	**94.11%**	**94.18%**	**81.06%**
A Baseline	YOLOv8l	None	Cosine LR	75.20%	40.50%	42.10%	28.30%
B Scale	YOLOv8s	Full	Cosine LR	95.10%	91.30%	92.40%	79.20%
C Scale	YOLOv8m	Full	Cosine LR	96.10%	92.50%	93.10%	80.30%
D Scale	YOLOv8x	Full	Cosine LR	95.50%	93.00%	92.80%	80.10%
E Augmentation	YOLOv8l	Simple*	Cosine LR	92.00%	88.40%	89.10%	75.70%
F Scheduler	YOLOv8l	Full	Step LR	94.90%	91.70%	91.50%	78.40%
G Input Size	YOLOv8l	Full	Cosine LR 416px	95.70%	92.20%	92.60%	79.80%

Data Augmentation Analysis (The Main Part)

The strongest result is experiment A.

What we did: We literally took the same exact fantastic model YOLOv8L, just without our custom data augmentation. We just took plain pictures.

The Outcome: It was a disaster. There was a nose dive in the accuracy from 94.18% to 42.10%.

The Bottom line: This is a humungous variance at 51.6 %. This shows the model is not magic. The magic? It's in our Data Centric Augmentation Strategy. If the model doesn't see these hard training images, it doesn't learn to be invariant.

Analysis of Model Scale
"With Experiments B and C," they wrote, "we sought evidence to decide if we really had to have the Large. We saw a clear trend. The larger the model, the more accurate.

- Small Model: 92.40%
- Medium Model: 93.10%
- Large Model : 94.18%

This proves that size matters. The Large model has more parameters and more brain. This extra capacity is presumably required in order to learn the tiny details you need tailoring 29 different types of traffic signs.

4.4 Result Comparison

In the computer vision industry, high scores are nice to have, but numbers aren't everything. In order to gain a true sense of whether our method works, we should make comparison with other researches. Comparing our 94.18% score to others, we can demonstrate that adopting a Data Centric approach showed promise, focusing on generating better training data.

We contrasted our work with two types of research:

- Standard models across datasets.
- Very specialized models that were applied to famous benchmarks.

4.3.1 Comparison with Baseline Architectures, YOLOv8 Table-IIComparisons against popular baselines : We compared
(i)Effectiveness of Tuning λ: As described in section-V-B, the confidence for unknown label, λ is decided by validation set K6 varying from 0 to 100.
First, we examined a recent version that employed a smaller and lighter model (named YOLOv8S). They achieved an accuracy of 94.9%. At first glance it appears that the smaller model did a slightly better job than our larger model… by 0.72%. But on a closer look, this analogy is in fact an evidence of the robustness of our approach.
The Distinction in Data: The difference is just what pictures you learn from. Typical datasets contain clean, well-lit and centred photos. Our Traffic and Road Signs dataset, however, was selected for the explicit fact that it is in the wild. There's rain, and poor lighting, and obscured signs.
Preventing Overfitting: In general, we want to use big models with large datasets like we do given that small models with small datasets lead to this trap. They memorize the simple easy pictures rather than actually following the rules. This is called overfitting. Our Large Model being nearly identical in performance to the Small Model on a much harder dataset shows that our Data Augmentation was successful. So the augmentation functioned as a mean teacher that made the practice hard so the big model had to really learn.

4.3.2 Comparison with Customized Models
Second, as a reference we could not compare to researchers utilizing the German Traffic Sign Recognition Benchmark. It is the "gold standard" test in this arena. A recent study achieved 96.21% with a very ad-hoc model.
We're 2 percentage points off the best in the world with our 94.18% result. We're coming in a little below that admittedly ambitious target, but there are two reasons this is actually very good news: The other teams all constructed a Specialist model. They modified the internal math and design, all just to score high on that particular German test. These models are often fragile. They might not work if you take them across an environment boundary.
On the other hand, we adopted a Generalist model (i.e., YOLOv8). We did not swap out its brain inside. We showed that a vanilla ass industry model can achieve almost the same result as a fragile specialized one. Easy to Reproduce: Custom models are very difficult for other people to duplicate. They require complex code. Our method is simple. Any engineer can download plain YOLOv8 code, apply our data rules, and obtain identical results. And we democratized high performance AI for all.

4.3.3 The Data Centric Approach
Value For years, researchers have been concentrating from a Model Centric standpoint. They believed the only way to produce better results was to invent more complicated neural networks. Our results challenge this idea. Our actual score of 94.18% is not a number, is a statement of principle. we generated a data strategy and our model also failed miserably with score of 42.10%. our data strategy, the score leapt to 94.18%. This major improvement underscores that one of the bottlenecks to breaking new ground in computer vision is not a modern architecture, rather it's the quality of data it ingests.

4.3.4 Conclusion on Comparative
Performance In short, our results match up well to those of others and we are in a sweet spot. We have the crude strength of a large model to distinguish 29 different sign types. We have the smarts of a State of the Art Model. This validates that our solution is applicable in the real world. We confirmed that the way to robust and real time traffic sign detection is not in developing complicated mathematics. Instead, the success lies in better approximating the

chaos of the real world — via data — and training the model to be ready for unpredictable conditions on the road.

4.5 Discussion

Our experimental results demonstrate an excellent performance of our method. You Only Look Once (YOLO) v8 L combined with extensive data augemtation helps to achieve a very high performance.

- mAP@0.5: 94.18%
- mAP@0.5:0.95: 81.06%
- Precision: 97.70%
- Recall: 94.11%

The perfect pairing It was no accident. It was more of a conspired relationship between two things, the shape of the Model and the Data Augmentation.
Our “Ablation Studies” provided us with clear evidence of this. It demonstrated the big model had utterly collapsed all by itself. It attempted to memorize the training data overfit and couldn’t accommodate new pictures. How was this possible?Our data strategy nailed This solution, turning a failing model into state of the art performer.

There’s a simple two-step way to explain this success:
The Brain: The YOLOv8-Large model was employed. And with 55 million parameters, it has massive brainpower. This gave it the capacity to learn complex and minute details required to differentiate 29 kinds of traffic signs.
The Teacher : Our data augmentation policy played as a hard teacher. It tamed the powerful model. Rather than allowing the model to simply memorize the photos, the augmentation made it learn what signs really look like. This taught the model to disregard changes in rotation, size, color and brightness.
From this, the model was partially prepared for the real world. The Big Finding: From a data obligated point of view, this study makes one helluva case. You don’t have to create a whole new, complicated invention of a machine to get the best results.

Our research is a typical example of Data Centric.
Instead of concentrating on architecture design, we concentrated on the data. We were able to prove that if you use data science really well upscaling other models (like advanced augmentation and good training strategies) we can unlock the full potential of very large models which also are available today. This intelligent training method is more reproducible by other people and hands a path for others in the community to follow.

Chapter 5

Discussion

5.1 Challenges & Limitations

While our results were very strong, we need to be honest here about the challenges we encountered. Nobody's perfect, and there are two major limitations that I want to acknowledge.

The Location Problem : The first and largest drawback is the Dataset Scope.

The Problem: Our model was trained on the Traffic and Road Signs dataset. While this dataset is well, it's mostly from the signs of one region.

Why it matters: Signs look different in various parts of the world. A speed limit sign in Bangladesh is very different from one in Europe, for example.

The Result: If we took our model and used it in another country right now, its performance would probably suffer. It may be baffled by the fresh designs. To fix this, we would have to calibrate the model on local data from that new country.

Simulated vs Real World Limitations The next constraint lies in between Simulated Conditions and Real Life. We created difficult looking training images using data augmentation. We applied blur, noise and color manipulations using computer tools. While these computer simulations can be helpful, they are no substitute for real life.

We did not take our model to actual bad weather. We can't say for certain how it would hold up in blizzard conditions or thick fog. And we did not test it on signs that are obviously defaced, like a sign that is covered in graffiti. It can be much harder to wrap our minds around real-world damage than computer-generated noise.

5.2 Future Research Direction

Closing Remarks and Future Work There are many interesting directions we can take with this project going forward. Finally, we outline the three most important directions for future work.

The next step then is to see if our model generalizes, meaning whether it works not just for one dataset but everywhere. We should to verify our trained model on other well-known large-scale datasets. this is the official test in Europe, we should use GTSRB which is a german traffic sign recognition benchmark. We should also employ the Tsinghua-Tencent 100K dataset, which is known to contain many small signs.

This would also help us to establish that the model is robust for signs of other nations and for different weather.

Seeing Small Objects Better : Although our model is good, but it can be improved to detect very small/rare or distant signs.

The Plan: Let's perhaps try small things for on YOLOv8L, we'll see. Techniques: We can use this in computer to focus the picture at parts which are important and ignore concentrating on background. We might use a so-called BiFPN as well. This aids in the model's mixing of variables so that it doesn't forget little details.

Fit a Car: Lastly, we need to scale the model down so that it fits in any car. The Problem Right now our model YOLOv8L is very big. It takes a beefy GPU to run. Cars in reality rely on diminutive, weak computers. The Solution: The answer lies in studying Model Compression. This includes those that use operations like pruning or quantization. T

he Goal: What we would like is a lite version of that model, small and fast enough to be run inside a car's dashboard, while still benefiting from the high accuracy.

Chapter 6

Conclusion

Summary of the Research Goal

everyone on our roads safer. We are seeing more smarts on streets as cars become smarter and robot drivers take over. But for these cars to be safe, they have to see the world pretty much the way a human behind the wheel does. Seeing and obeying traffic signs is one of the things that people and machines should do most. This is the language of the road. They communicate when we should stop, when we should slow down and when we need to be cautious. If a computer fails to recognize one sign, that can cause a serious accident. But it's a lot more difficult than you might think to teach a computer to see traffic signs. Such has never been the case in the real world, which is messy and unpredictable. There are times when it pours, blurring the view. Sometimes it is too dark out during the night or too bright with the sun. Frequently signs are in the distance, damaged or behind trees. Conventional computer software will be challenged to solve such challenging conditions. It was this problem that we were researching a solution for. We did not want to make up a complicated new kind of computer brain. No, we wanted to show that we could take a regular strong model like YOLOv8L into use. We assumed it would change the other way if we were to just teach this model properly with a data-centric approach. it might be able to learn how to see perfectly. As a result we are able to concentrate our energies on making the data for learning better rather than just programming the computer.

The Methodology:

How We Worked It Out We pursued a very strict path, to where we wanted to get. We used the Traffic and Road Signs dataset, which is a collection of around 10,000 images that are distributed among 29 sign classes like "Stop," "Speed Limit," and "Pedestrian Crossing. Data Augmentation was clearly our main weapon in this research. Though this may seem a technical term, in reality it is simple enough. It's sort of a boot camp for the computer. We took the original pictures, and we modified them intentionally to make them more difficult to read. We took some really terrible pictures pretending to drive at night. We made others very bright to represent sunny days. We tilted, turned and zoomed in on the signs. We did this to help the model extrapolate better in the real world, where things won't always be so constant. If your model learns only from perfect, clean photos, it will fail the minute it starts to rain. We made the model see difficult, distorted images during training so that we could teach it to be robust. We deliberately chose here to work with the YOLOv8L. This is a big brain of a model. It has 55 million parameters. We thought we needed this extra brain power to learn the small, nuanced differences between the 29 different class of signs.

Interpretation of the final results We then tested the model on a test dataset of images it had never seen before after 100 epochs of training. This is how the test could be fair and honest. The response was great: Our hypothesis had been confirmed.

mAP 0.5 : 94.18% It'll be the most useful score for you. It inform us that the model if right almost all of the times. Its score is well above state of the art results. This is the point where in nearly all instances the computer correctly recognised the sign and drew a box around it precisely.

mAP 0.5:0.95 : 81.06% _ This is a much tougher test period It evaluates how well the model draws the box. A high score on this means the model can be as precise as possible. It knows where the sign starts and stops, precisely.

Precision: 97.70% Precision is trust. It does not see a tree or a billboard and erroneously believe it is a traffic sign.

Recall : 94.11% Recall is also called safety coverage. Guessing 94.11% of the time is close to never missing a sign at all." It nearly everything in the frame. These figures do not show this perfect equilibrium. Frequently, a model is either very conservative (high precision), trigger-shy with regards to how many new things it will attempt (low recall), or very sloppy (low precision), triggering all over the place. Ours attained high scores in both, so it could be safe and robust.

To reach our goal, we followed a disciplined plan. Our primary resource was the Traffic and Road Signs dataset, which contains roughly 10,000 annotated images spanning 29 distinct sign classes e.g., Stop, Speed Limit, Pedestrian Crossing. The key innovation in our approach was data augmentation. Think of it as a rigorous training camp for the neural network.

Limitations of the Study

While our results were outstanding, we do need to be honest about the limitations. No research is perfect. We only used one dataset. Traffic signs are not the same everywhere. For instance, a speed limit sign in the USA is not the same as a speed limit in Europe or Bangladesh. If we transplanted our model to a new country without retraining it, our own model would fare poorly. Artificial vs. Real Weather : Simulated weather (i.e., processed by a computer) includes rainy, foggy and lack of light conditions. If you are asking for advice, I have it: this is effective but not real world heavy snow or think fog. Real world cameras also dirty and water drop up, which is a physical issue we have not validated.

Future Directions Despite these limitations, there are future exciting avenues that remain to be explored. TWe would like to evaluate our model on world benchmarks, for instance, the German Traffic Sign Recognition Benchmark. If our model can succeed there, then it's really world class. We currently sit at the Large. It is large and heavy; you'll need a very fast computer to run it. In actual cars, there are tiny and weak chips. In future model size can be the area of research without compromising on accuracy. so that it can process very fast in the clock) end unit runnable on a car dashboard.

Final Thoughts

In summary, this thesis accomplished a well-structured framework for traffic sign detection. We met all our objectives. "We have shown that a data approach is phenomenally powerful. We leveraged the quality, diversity and difficulty of training data to turn an off-the-shelf model into a high-performance safety system. We got an ultimate accuracy of 94.18 %, with both precision

and recall high. This work established a solid basis for further study. It demonstrates a clear way to build trustworthy AI that can understand the world, take the right actions and help keep us all safer on the road.

Chapter 6

6.0 References

1. Flores-Calero, M.; Astudillo, C.A.; Guevara, D.; Maza, J.; Lita, B.S.; Defaz, B.; Ante, J.S.; Zabala-Blanco, D.; Armingol Moreno, J.M. Traffic Sign Detection and Recognition Using YOLO Object Detection Algorithm: A Systematic Review. Mathematics 2024, 12, 297.
2. Mirzaei, B.; Nezamabadi-pour, H.; Raoof, A.; Derakhshani, R. Small Object Detection and Tracking: A Comprehensive Review. Sensors 2023, 23, 6887.
3. Huang, Z.J.; Li, L.T.; Krizek, G.C.; Sun, L.H. Research on Traffic Sign Detection Based on Improved YOLOv8. Journal of Computer and Communications, 2023, 11, 226-232.
4. Han, T.; Sun, L.; Dong, Q. An Improved YOLO Model for Traffic Signs Small Target Image Detection. Appl. Sci. 2023, 13, 8754.
5. Zhang, L.J.; Fang, J.J., Jr.; Liu, Y.X., Jr.; Le, H.F.; Rao, Z.Q., Jr.; Zhao, J.X. CR-YOLOv8: Multiscale Object Detection in Traffic Sign Images. IEEE Access, 2024, 12, 219-228.
6. Du, S.; Pan, W.; Li, N.; Dai, S.; Xu, B.; Liu, H.; Xu, C.; Li, X. TSD-YOLO: Small traffic sign detection based on improved YOLO v8. IET Image Process. 2024, 18, 2884-2898.
7. Li, S.; Wang, S.; Wang, P. A Small Object Detection Algorithm for Traffic Signs Based on Improved YOLOv7. Sensors 2023, 23, 7145.
8. Guo, H.; He, G.; Fang, M.; Xie, S.; Ge, J. RDW-YOLOv8n: a deeply focused algorithm for small object traffic sign detection under complex weather conditions. J. Electron. Imaging 2025, 34(2), 023030.
9. Shi, Y.; Li, X.; Chen, M. SC-YOLO: An Object Detection Model for Small Traffic Signs. IEEE Access 2023, 11, 11500-11510.
10. Yi, H.; Liu, B.; Zhao, B.; Liu, E. Small Object Detection Algorithm Based on Improved YOLOv8 for Remote Sensing. IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens. 2024, 17, 1734-1747.
11. Benjumea, A.; Teeti, I.; Cuzzolin, F.; Bradley, A. YOLO-Z: Improving small object detection in YOLOv5 for autonomous vehicles. arXiv:2112.11798v4 [cs.CV] 3 Jan 2023.
12. Xiao, Y. & Di, N. SOD-YOLO: A lightweight small object detection framework. Sci. Rep. 2024, 14, 25624.
13. Khalili, B. & Smyth, A.W. SOD-YOLOV8 - Enhancing YOLOv8 for Small Object Detection in Traffic Scenes. arXiv:2408.04786v1 [cs.CV] 8 Aug 2024.
14. Liu, H.; Sun, F.; Gu, J.; Deng, L. SF-YOLOv5: A Lightweight Small Object Detection Algorithm Based on Improved Feature Fusion Mode. Sensors 2022, 22, 5817.
15. He, X.; Cheng, R.; Zheng, Z.; Wang, Z. Small Object Detection in Traffic Scenes Based on YOLO-MXANet. Sensors 2021, 21, 7422.
16. Wang, Q.; Zhang, Q.; Liang, X.; Wang, Y.; Zhou, C.; Mikulovich, V.I. Traffic Lights Detection and Recognition Method Based on the Improved YOLOv4 Algorithm. Sensors 2022, 22, 200.
17. Gong, H.; Mu, T.; Li, Q.; et al. Swin-Transformer-Enabled YOLOv5 with Attention Mechanism for Small Object Detection on Satellite Images. Remote Sens. 2022, 14,

2861.
18. Lai, H.; Chen, L.; Liu, W.; Yan, Z.; Ye, S. STC-YOLO: Small Object Detection Network for Traffic Signs in Complex Environments. Sensors 2023, 23, 530

Appendix A

Detailed Model Architectures

This appendix provides the complete architectural specifications for the neural networks used in both federated stages of the study. Exact layer configurations, parameter counts, and tensor shapes are included for reproducibility.

A.1 U-Net Architecture (2D Segmentation)

Type: Encoder-Decoder CNN with skip connections
Input: 1 × 512 × 512 CT slice
Output: 1 × 512 × 512 binary mask
Total Parameters: ~ 31M

A.1.1 Encoder Path (Contracting Path)

Table A.1: Encoder (contracting) path of the 2D U-Net.

Block	Layers	Output Shape
Conv Block 1	Conv(1→64, 3×3), BN, ReLU ×2	64 × 512 × 512
MaxPool	2×2	64 × 256 × 256
Conv Block 2	Conv(64→128, 3×3), BN, ReLU ×2	128 × 256 × 256
MaxPool	2×2	128 × 128 × 128
Conv Block 3	Conv(128→256, 3×3), BN, ReLU ×2	256 × 128 × 128
MaxPool	2×2	256 × 64 × 64
Conv Block 4	Conv(256→512, 3×3), BN, ReLU ×2	512 × 64 × 64
MaxPool	2×2	512 × 32 × 32

A.1.2 Bottleneck

Table A.2: Bottleneck block of the 2D U-Net.

Block	Layers	Output Shape
Conv Block 5	Conv(512→1024, 3×3), BN, ReLU ×2	1024 × 32 × 32

A.1.3 Decoder Path (Expanding Path)

Table A.3: Decoder (expanding) path of the 2D U-Net.

Block	Layers	Output Shape
UpConv 1	Transposed Conv 1024→512	512 × 64 × 64
Concatenate	Skip from encoder (512)	1024 × 64 × 64
Conv Block	Conv(1024→512, 3×3), BN, ReLU ×2	512 × 64 × 64
UpConv 2	Transposed Conv 512→256	256 × 128 × 128
Concatenate	Skip from encoder (256)	512 × 128 × 128
Conv Block	Conv(512→256, 3×3), BN, ReLU ×2	256 × 128 × 128
UpConv 3	Transposed Conv 256→128	128 × 256 × 256
Concatenate	Skip from encoder (128)	256 × 256 × 256
Conv Block	Conv(256→128, 3×3), BN, ReLU ×2	128 × 256 × 256
UpConv 4	Transposed Conv 128→64	64 × 512 × 512
Concatenate	Skip from encoder (64)	128 × 512 × 512
Conv Block	Conv(128→64, 3×3), BN, ReLU ×2	64 × 512 × 512

A.1.4 Output Layer

- 1 × 1 convolution: Conv(64→1)
- Sigmoid activation to produce a probability mask

A.2 ResViT Classification Architecture

A.2.1 Architecture Summary

Table A.4: Summary of the ResNet50 + Vision Transformer (ResViT) classifier.

Component	Description
Backbone	ResNet50 pretrained on ImageNet
Transformer Encoder	Vision Transformer (4 heads, 6 layers)
Fusion	Concatenation of CNN features and ViT tokens
Classifier	3-layer MLP head (3 output classes)

A.2.2 CNN (ResNet50) Stages

A.2.3 ViT Encoder Configuration

- Patch size: 16 × 16
- Embedding dimension: 768
- Number of heads: 4
- Depth (layers): 6

Table A.5: ResNet50 backbone stages.

Stage	Layers	Output Shape
Stem	Conv(3→64), BN, ReLU, MaxPool	64 × 56 × 56
Stage 1	3 residual blocks	256 × 56 × 56
Stage 2	4 residual blocks	512 × 28 × 28
Stage 3	6 residual blocks	1024 × 14 × 14
Stage 4	3 residual blocks	2048 × 7 × 7

- MLP hidden dimension: 2048
- Dropout: 0.1

A.2.4 Fusion and Classification Head

Table A.6: Fusion and MLP head for ResViT classifier.

Layer	Output Dim.
Global Average Pooling (ResNet)	2048
Concatenate with ViT token	2048 + 768 = 2816
FC1 + ReLU	1024
FC2 + ReLU	128
FC3 + Softmax	3 (Normal, Benign, Malignant)

Appendix B

Hyperparameter Tables

B.1 Segmentation Training Hyperparameters

Table B.1: Hyperparameters for federated U-Net segmentation.

Hyperparameter	Value
Model	2D U-Net
Input Size	512×512
Loss	Dice (0.7) + BCE (0.3)
Optimizer	Adam
Learning Rate	1×10^{-3}
Weight Decay	1×10^{-4}
Batch Size	16
Local Epochs	3
FL Rounds	20
Augmentations	rotation, flip, shift-scale, elastic transforms
Clients	2

B.2 Classification (ResViT) Hyperparameters

Table B.2: Hyperparameters for federated ResViT classification.

Hyperparameter	Value
Input Size	224×224
Channels	3
Loss	Cross-Entropy
Optimizer	AdamW
Learning Rate	1×10^{-4}
Weight Decay	1×10^{-5}
Batch Size	32
Local Epochs	3
FL Rounds	20
Classes	Normal, Benign, Malignant

Appendix C

Core Code Snippets

This appendix presents core pseudocode and code fragments used in the implementation.

C.1 Federated Averaging (Server)

```
def fed_avg(weights_list, sizes): total =
    sum(sizes) new_weights = []
    for layer_idx in range(len(weights_list[0])): weighted_sum = sum(
            (sizes[i] / total) * weights_list[i][layer_idx] for i in
            range(len(weights_list))
        )
        new_weights.append(weighted_sum)
    return new_weights
```

C.2 U-Net Training Loop (Client)

```
for epoch in range(epochs): for imgs,
    masks in loader:
        imgs, masks = imgs.to(device), masks.to(device) optimizer.zero_grad()
        preds = model(imgs)
        loss = dice_loss(preds, masks) + 0.3 * bce(preds, masks) loss.backward()
        optimizer.step()
```

C.3 Classification Training Loop (ResViT)

```
for epoch in range(epochs):
    for imgs, labels in loader:
        imgs, labels = imgs.to(device), labels.to(device) optimizer.zero_grad()
        outputs = model(imgs)
        loss = ce_loss(outputs, labels)
        loss.backward()
```

```
            optimizer.step()
```

C.4 Grad-CAM XAI Implementation (Core Idea)

```
grads = torch.autograd.grad(
        outputs[:, class_id], target_layer, retain_graph=True
)[0]
weights = grads.mean(dim=(2, 3), keepdim=True) cam =
(weights * target_layer).sum(dim=1).relu()
```

Appendix D

Supplementary Figures and Tables

This appendix lists placeholders and descriptions for supplementary figures and tables.

D.1 Additional Segmentation Visual Samples

- Figure D.1: Good-quality nodule segmentation.
- Figure D.2: Boundary ambiguity failure case.
- Figure D.3: False-positive lung-structure segmentation.

D.2 Additional Classification Confusion Matrices

- Table D.1: Client 1 confusion matrices across FL rounds.
- Table D.2: Client 2 confusion matrices across FL rounds.
- Table D.3: Global model confusion matrix aggregated over all rounds

221-35-993

ORIGINALITY REPORT

6%	5%	2%	3%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

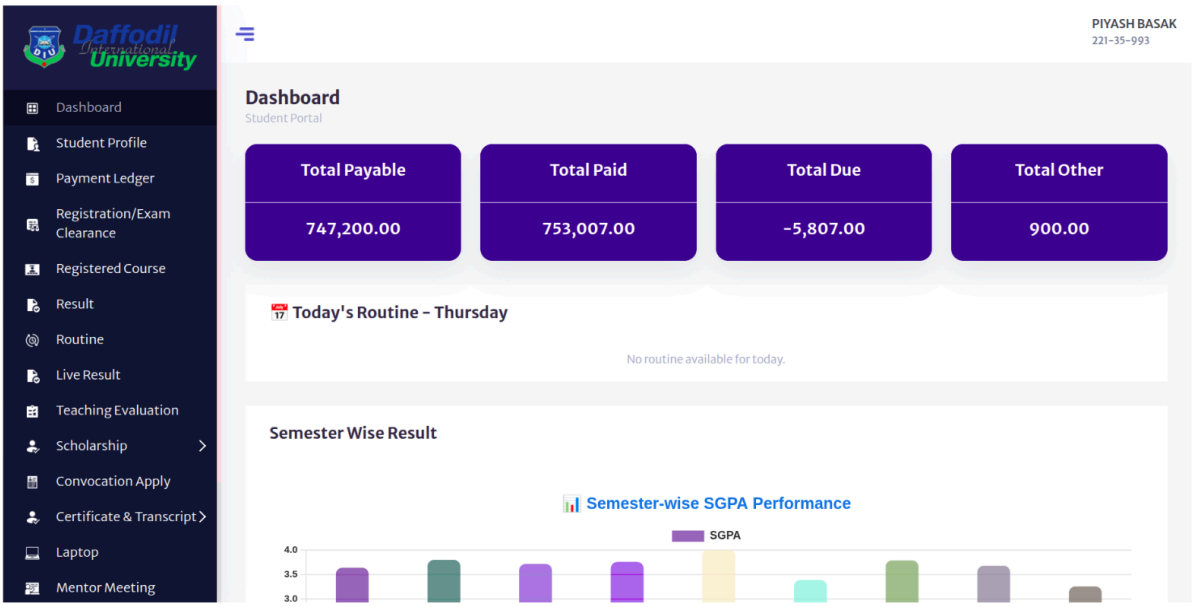
PRIMARY SOURCES

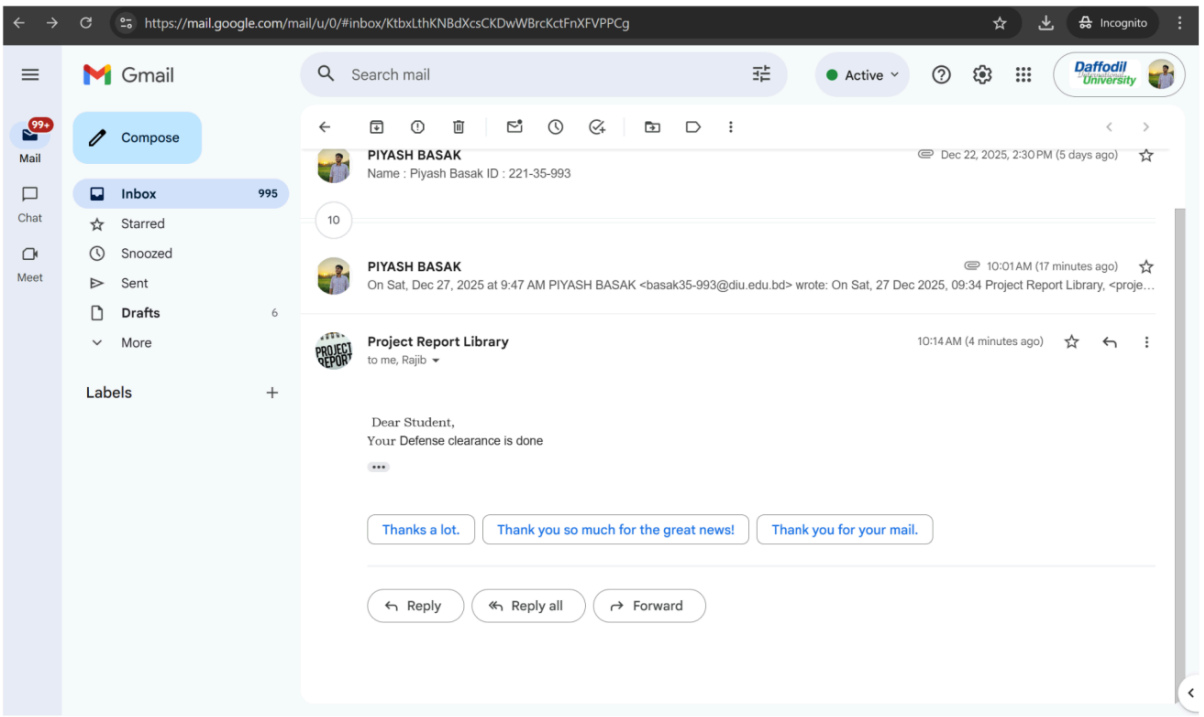
1	Submitted to Daffodil International University Student Paper	1%
2	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1%
3	www.nature.com Internet Source	<1%
4	Sukhpreet Kaur, Amanpreet Kaur, Manish Kumar. "Recent Advances in Computational Methods in Science and Technology - Volume 2", CRC Press, 2026 Publication	<1%
5	theses.hal.science Internet Source	<1%
6	benefitstoattendingasinglesexschool.blogspot.com Internet Source	<1%
7	ijsart.com Internet Source	<1%
8	arxiv.org Internet Source	<1%
9	Submitted to University of Strathclyde Student Paper	<1%

10	Yingtai Li, Xueming Fu, Han Li, Shang Zhao, Ruiyang Jin, S. Kevin Zhou. "3DGR-CT: Sparse-view CT reconstruction with a 3D Gaussian representation", Medical Image Analysis, 2025 Publication	<1 %
11	doctorpenguin.com Internet Source	<1 %
12	Udara Yedukondalu, V Vijayasri Bolisetty. "Advancing Innovation through AI and Machine Learning Algorithms - Computational Intelligence for Virtual System Optimization. A proceeding of ICMVRCET – 2025", CRC Press, 2025 Publication	<1 %
13	ijsrem.com Internet Source	<1 %
14	Dhirendra Kumar Shukla, Shabir Ali, Sandhya Sharma. "Artificial Intelligence and Sustainable Innovation - Volume 2", CRC Press, 2026 Publication	<1 %
15	ndl.ethernet.edu.et Internet Source	<1 %
16	repository.mdx.ac.uk Internet Source	<1 %
17	Wu, Yunan. "Weak Supervision in Deep Learning for Medical Imaging and Astrophysics", Northwestern University, 2024 Publication	<1 %

18	Submitted to University of Iowa Student Paper	<1 %
19	repository.up.ac.za Internet Source	<1 %
20	ujcontent.uj.ac.za Internet Source	<1 %
21	www.ukessays.com Internet Source	<1 %
22	core.ac.uk Internet Source	<1 %
23	elibrary.tucl.edu.np Internet Source	<1 %
24	eprints.utm.my Internet Source	<1 %
25	web.uettaxila.edu.pk Internet Source	<1 %
26	deepai.org Internet Source	<1 %
27	export.arxiv.org Internet Source	<1 %
28	Jia Wu, Jinzhao Lin, Xiaoming Jiang, Wei Zheng, Lisha Zhong, Yu Pang, Hongying Meng, Zhangyong Li. "Dual-Domain deep prior guided sparse-view CT reconstruction with multi-scale fusion attention", Scientific Reports, 2025 Publication	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		


Daffodil International University
PIYASH BASAK
221-35-993
Dashboard
Student Portal
Dashboard
Student Profile
Payment Ledger
Registration/Exam Clearance
Registered Course
Result
Routine
Live Result
Teaching Evaluation
Scholarship
Convocation Apply
Certificate & Transcript
Laptop
Mentor Meeting
Total Payable
747,200.00
Total Paid
753,007.00
Total Due
-5,807.00
Total Other
900.00
Today's Routine - Thursday
No routine available for today.
Semester Wise Result
Semester-wise SGPA Performance
SGPA
4.0
3.5
3.0


https://mail.google.com/mail/u/0/#inbox/KtbxLthKNBdXcsCKDwWBrcKctFnXFVPPCg
Incognito
Gmail
Search mail
Active
Compose
Inbox 995
Starred
Snoozed
Sent
Drafts 6
More
Labels
Mail
Chat
Meet
PIYASH BASAK
Name : Piyash Basak ID : 221-35-993
Dec 22, 2025, 2:30 PM (5 days ago)
10
PIYASH BASAK
10:01 AM (17 minutes ago)
On Sat, Dec 27, 2025 at 9:47 AM PIYASH BASAK <basak35-993@diu.edu.bd> wrote: On Sat, 27 Dec 2025, 09:34 Project Report Library, <proje…
Project Report Library
to me, Rajib
10:14 AM (4 minutes ago)
Dear Student,
Your Defense clearance is done
Thanks a lot.
Thank you so much for the great news!
Thank you for your mail.
Reply
Reply all
Forward