T.C

İSTANBUL OKAN UNIVERSITY

FACULTY OF ENGINEERING AND NATURAL SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING & SOFTWARE ENGINEERING



**TRAFFIC SIGN DETECTION AND RECOGNITION**

**THROUGH DEEP LEARNING**

**IN MOTION BLUR AND POOR WEATHER CONDITIONS**

PROGRAM:

Computer Engineering & Software Engineering

PREPARED BY:

Buket KAZMA 180218002

Baran KÜÇÜKDEVECİ 190212012

SUPERVISOR:

Prof. Dr. Semih Bilgen

İSTANBUL

2023

Final version

*“All the information in this project was obtained and presented in accordance with academic rules and ethical principles; I also declare that I have explicitly referenced the sources of all material that does not originate from this study, as required by these rules and principles. "*

*Buket KAZMA*

*Baran KÜÇÜKDEVECİ*

**ABSTRACT**

TRAFFIC SIGN DETECTION AND RECOGNITION

THROUGH DEEP LEARNING

IN MOTION BLUR AND POOR WEATHER CONDITIONS

Road and traffic signs recognition & detection are one of the major tasks of driverless systems. So much so that this task can save lives when accomplished or can cause one’s death. This paper describes a deep learning image recognition and detection implementation proposal on traffic signs in poor weather conditions and under motion blur. The poor weather conditions to be considered are rain, fog, and snow. These conditions are imposed on clean images for data augmentation. The motion blur implies the car's speed. The recognition part aims to build a deep learning model which will be trained by 3 channeled (RGB) images from different classes of traffic signs. The detection model uses YOLO models (v3 and v7) which is trained by 3 channeled (RGB) images from different augmented road pictures.

***Keywords:***Deep learning; Neural Networks; Traffic sign recognition; Traffic sign detection; Image recognition; Driverless systems; Image classification.

**TABLE OF CONTENTS**

[**LIST OF TABLES**](#_heading=h.iyfgdq3xnsdb) **5**

[**LIST OF FIGURES**](#_heading=h.i3aemvsp9tlx) **6**

**ABBREVIATIONS 7**

[**I. INTRODUCTION**](#_heading=h.atkwtak88ajs) **8**

I[.1.](#_heading=h.63us1ejqls1x) Objective of Study 8

I.2. Scope of Study 8

I.3. Significance of Study 9

I.4. Statement of Problem 9

I.5. Literature Survey 9

[**II.**](#_heading=h.44sinio) **SYSTEM STUDY ANALYSIS 14**

II.1. Principles of system analysis 14

II.2 Existing System with Limitations 14

II.3. Proposed System Features 14

[**III. S**](#_heading=h.wy6skm5fnar9)**PECIFICATION OF REQUIREMENTS 15**

III.1. Hardware Requirements 15

III.2. Software Requirements 15

[**IV.**](#_heading=h.wy6skm5fnar9) **DESCRIPTION AND ANALYSIS OF THE PROPOSED SYSTEM 16**

IV.1. Advantages of the Proposed System 16

IV.2. Disadvantages of the Proposed System 16

**V. SPECIFICATION OF THE PROPOSED SYSTEM 17**

V.1. Input Specification 17

V.2. Output Specification 21

**VI. OUR EXPERIMENT RESULTS 24**

VI.1.Convolutional Neural Networks (CNNs) For Traffic Sign Classification 24

V1.2. Convolutional Neural Networks (CNNs) For Traffic Sign Detection 29

VI.3. Comparison 31

**VII. CHOICE AND JUSTIFICATION OF PROGRAMMING LANGUAGE 33**

**VIII. CONCLUSION 34**

[**REFERENCES**](#_heading=h.z7b9v3jmoi31) **35**

[**APPENDIX**](#_heading=h.nt55y2b45ngv) **A: DISTRIBUTION OF TASKS** 38

# List of Tables

**Table 1.** Examples from GTSRB, each number indicating the class that  
belongs to the traffic sign 19

**Table 2.** Models tested on both augmented versions of GTSRB dataset 26

**Table 3**. YOLO v3 results 30

**Table 4.** YOLO v7 results 30

**Table 5.** Comparing augments and models with their accuracy and F1 scores 31

# List of Figures

**Fig. 1.** Structure of traffic sign recognition system. 7

**Fig. 2.** Structure of traffic sign detection system. 7

**Fig. 3.** Countries that have ratified the Vienna Convention on Road

Signs and Signals. 7

**Fig. 4.** Original image from GTSRB 16

**Fig. 5.** Augmented images from GTSRB 17

**Fig. 6.** Original image from GTSDB 18

**Fig. 7.** Augmented images from GTSDB 18

**Fig. 8.** Example of TSR model predictions 21

**Fig. 9.** Correct guess using YOLO v3 darknet 23

**Fig. 10.** Incorrect guess using YOLO v3 darknet 23

**Fig. 11.** Buket Custom Model #1 24

**Fig. 12.** Buket Custom Model #2 24

**Fig. 13.** Buket Custom Model #3 24

**Fig. 14.** Baran Custom Model #1 25

**Fig. 15.** Baran Custom Model #2 25

**Fig. 16.** Baran Custom Model 1 showing loss and validation loss 32

per epoch

**Fig. 17.** Baran Custom Model 1 showing F1 score and validation F1 32

score per epoch

**Fig. 18.** Buket Custom Model 2 showing loss and validation loss per epoch 32

**Fig. 19.** Buket Custom Model 2 showing F1 score and validation F1 score 32

per epoch

**Fig. 20.** Baran Custom Model 2 showing loss and validation loss per epoch 32

**Fig. 21.** Baran Custom Model 2 showing F1 score and validation F1 score

per epoch 32

**ABBREVIATIONS**

**AFF** Attentional Feature Fusion

**AUC** Area Under Curve

**BoVW** Bag-of-Visual-Words

**BPNN** Back Propagation Neural Network

**CC** Challenge Classifier

**CLAHE** Contrast-limited Adaptive Histogram Equalization

**CNN** Convolutional Neural Network

**CTSD** Chinese Traffic Sign Dataset

**CURE-TSD** Challenging Unreal and Real Environments for Traffic Sign Detection

**ETSD** The European Traffic Sign Dataset

**GPU** Graphics processing unit

**GTSDB** German Traffic Sign Detection Benchmark

**GTSRB** German Traffic Sign Recognition Benchmark

**HDD** Hard disk drive

**HOG** Histogram-oriented gradient

**I/O** Input/Output

**LIDAR** Light Detection and Ranging

**LISA-TS** LISA Traffic Sign (Dataset)

**MSER** Maximally Stable Extremal Regions

**R-CNN** Recurrent Convolutional Neural Networks

**RAM** Random Access Memory

**RGB** Red Green Blue Colors

**SSD** Single Shot Detector

**SGW** Simplified Gabor Wavelets

**SVM** Support Vector Machines

**TSC** Traffic Sign Classification

**TSD** Traffic Sign Detection

**TSR** Traffic Sign Recognition

**TSDS** Traffic Sign Data Set

**TT100K** Tsinghua-Tencent 100K Dataset (100K for 100 000 pictures)

**UTBM** University of Technology of Belfort-Montbéliard

**VGG** Visual Geometry Group

**YOLO** You Only Look Once (Model)

# Introduction

* 1. **Objective of Study**

The recognition model is able to classify the signs while attempting to minimize the error from poor weather conditions and car’s speed. It achieves this task using deep learning models. We experimented with different architectures both custom and state-of-art models and tried to find the best model for this task. The overall process was done as shown in Figure 1.

Image

Classification

Output

Fig. 1. Structure of traffic sign recognition system.

The detection model attempts to detect the signs on the road while attempting to minimize the error from poor weather conditions and the car’s speed. It achieves this task using YOLO v3 darknet models. We experimented with different YOLO versions and tried to find the best model for this task. The overall process was done as shown in Figure 2.

Fig. 2. Structure of traffic sign detection system.

Image

Detection

Output

Classification

* 1. **Scope of Study**

The recognition model trained on a newly created dataset. This dataset contains original images and our augmented images. Augmented images include rain, fog, snow, and motion blur. The original images are real-world pictures which use the GTSRB [1] dataset. This dataset adheres to the 1968 Vienna Convention on Road Signs and Signals.

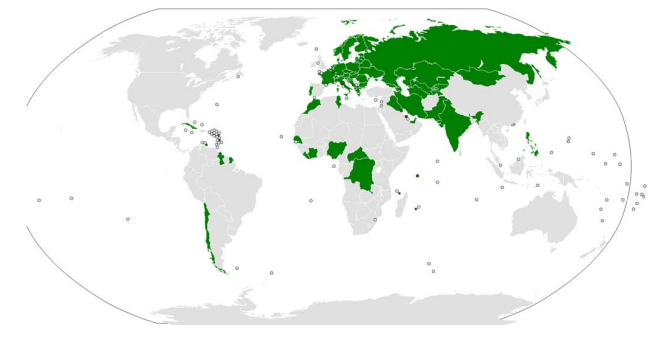


Fig. 3. Countries that have ratified the Vienna Convention on Road Signs and Signals. Data Source: [2]

There are 39209 train images and 12630 test images in GTSRB. The images have a resolution between 16x16 to 256x256. There are 43 classifications in this dataset.

There are 156836 motion blurred augmented images for training, 156836 poor weather augmented images for training, while there are 50520 test images for each augmentation.

The detection model trained on a newly created dataset. This dataset contains original images and both of the augmentation types. Augmented images include rain, fog, snow, and motion blur. The original images are real-world pictures which are from the GTSDB [3] dataset.

There are 600 train images and 300 test images in GTSDB. The images have a resolution of 1360x800. There are 43 classifications in this dataset.

There are 982 motion blurred augmented images for training and 216 motion blurred images for testing; there are 1473 poor weather augmented images for training and 324 poor weather augmented images for testing.

* 1. **Significance of Study**

The purpose of developing this project is to enable TSD and TSC to utilize the algorithm so they can recognize the traffic signs and follow them. This plays a crucial part in making vehicles safer for the roads and TSD and TSC are more widespread as a result. Having the weather conditions in the dataset makes the algorithm better suited for real-world scenarios, where perfect conditions and perfect images of signs are less common. Also, researchers showed that [4] education, monthly income level, and cultural background affect a driver’s comprehension of traffic signs. So, under certain conditions and because of human nature, humans are not 100% accurate when recognizing traffic signs [5]. In the recognition of traffic signs, utilizing TSD and TSC can give more accurate and faster results than humans [6] and save lives.

* 1. **Statement of Problem**

Traffic signs on the road are the most commonly used tools for controlling traffic. These signs display messages in terms of words and/or symbols and are placed to regulate, warn or guide drivers. There aren’t enough solutions to recognize traffic signs without using human sight. The traffic signs we rely on and depend on can be difficult for drivers to see and recognize under poor weather conditions [7]. Some accidents happen because of people failing to see traffic signs [8].

* 1. **Literature Survey**

There are a large number of articles and a vast amount of research efforts on traffic sign recognition under poor conditions. We have searched numerous articles and different approaches. In this section, we have gathered the paper works and the approaches we appreciated the most.

Sabbir Ahmed, et al. [9] suggest an approach that has four main steps: CNN-based challenge classifier, Enhance-Net CNN architecture for image enhancement, CNN architecture for sign detection, and lastly a CNN architecture for sign classification. The challenge classifier part is using transfer learning on VGG16 which is pre-trained on the ImageNet [10] dataset to detect the type of the challenge in the picture. The Enhance-Net part tries to enhance the traffic signs which were not captured clearly due to the challenging weather conditions if the challenge is one of the five CCs. The five main CCs in this project are Rain, Snow, Haze, Dirty lens, and Lens blur which have five different severity levels. There are single Enhance-Net architectures for each different severity level and different agent. This type of enhancement work gives modularity for using pre-trained CNNs for challenge-free conditions to unfavorable ones. The detection and classification part is using SegU-Net. Working with the CURE-TSD dataset, the Experimental result of this project is having an overall precision and recall of 91.1% and 70.71% respectively. This paper is one of the most suitable works for the TSC task under unfavorable conditions.

Faming Shao, et al. [11] suggest an approach that consists mainly of SGW, Maximally Stable Extremal Regions Algorithm, SVM, and CNN. The first part is converting the images to grayscale. That is because; due to the outdoor conditions, color-based approaches often fail. Then, they filtered these grayscale images with SGW to enhance the edges and the shape information. Secondly, they extracted the important part of the images (which keeps the sign information) using the maximally stable extremal regions algorithm. Then, they classified the images with SVM. Finally, they used a CNN to classify the signs into their subclasses. In this project, they used a CTSD dataset having an image size of 1360\*800 and experimented with an accuracy of 99.43%.

Andreas Mogelmose, et al. [12] reckon there is not enough research work and publicly available datasets that focus on the U.S. Traffic Sign Detection and Recognition task. Therefore, they extended the LISA-TS dataset by almost doubling its size. They have utilized the top three solutions from the GTSDB competition. These solutions included the Integral Channel Features detection method. They enhanced this by applying the Aggregate Channel Features detection method too. They used contrast-limited adaptive histogram equalization (CLAHE) for color normalization. Working with an image size of 1280\*960 they achieved AUCs of 98.98, 96.11, and 96.17 for diamond warning signs, stop signs, and no-turn signs respectively.

Aleksey Osipov, et al. [13] had two different approaches to the TSR task for real-life images. First approach is combining the histogram-oriented gradient (HOG) method for making the images clearer, a bag-of-visual-words (BoVW) for image classification, and a back propagation neural network (BPNN). Second approach is using the VGG-16 model. They experimented with a dataset of 1700 images taken from a clean protective glass, and 830 images taken from a protective glass covered with drops of water or dirt on it. First approach achieved an accuracy of 71 to 86% (depending on the type).

Jack Greenhalgh, et al. [14] reckon there is not enough research work which focuses on the textual information on the traffic signs. Their approach has three main stages. First stage is to detect the candidates which is determining the search regions of the image, detecting all possible candidates, and reducing these candidates by contextual constraints. Second stage is to detect the textual candidates out of all remaining candidates. That is done by Maximally Stable Extremal Regions (MSERs) and hue,

saturation, and value color thresholding. These textual candidates are reduced too by using both temporal and contextual information. Last stage is to recognize the textual information. Working with images of size 1920\*1088, they achieved an F measure of 0.93 for detection, 0.89 for recognition, and 0.87 for the entire system.

Zhu et al. [15] have created a dataset called Tsinghua-Tencent 100K, which has over 100 times more images and 32 times the image resolution. The images are also annotated with a pixel mask for traffic signs. This dataset contains illumination changes and harsh weather conditions as well. They have developed two CNNs: one for detection and another for detection & classification simultaneously. Their detection network treats all traffic signs as one category and have achieved 84% accuracy and 94% recall. With panorama pictures that don’t contain any traffic signs, their model identified all of them as not containing a traffic sign.

According to Liu et al. [16], algorithms using machine vision and pattern recognition techniques can be classified under five categories: color-based methods, shape-based methods, color- and shape-based methods, machine learning-based methods and LIDAR-based methods. Each of these categories have subcategories within them as well. Liu, et al. [16] have compared these methods with each other and concluded that machine learning methods are currently the best way to detect traffic signs. They also claim extreme weather has a great impact on the quality of the images captured and extreme weather conditions such as heavy snow, heavy rain and heavy fog were not considered in earlier methods. They suggest new datasets are needed to accommodate these factors and they claim LIDAR-based methods have an advantage over machine learning when it comes to these conditions.

Lin et al. [17] propose a lightweight model for traffic sign detection and recognition. They have used the TT100K dataset for their training. Their goal is to address the high computational requirement needed for real-time performance. From their experiments they have achieved 85% recognition accuracy for various scale targets and categories, scoring higher than Faster R-CNN, ConerNet and CenterNet while having a good real-time performance.

Citlalli Gamez Serna, and Yassine Ruichek [18] have suggested five CNN architectures to classify 164 European traffic signs which follow the Vienna Road Traffic Sign Convention [19]. The ETSD consists of 1280×960 size of 82,476 images from 6 different countries such as Belgium, Croatia, France, Germany, Netherlands and Sweden. The five methods they suggested are: Lenet-5, Idsia Model, URV Model, CNN with Asymmetric Kernels, CNN 8-Layers. The writers highlighted that most of the paper works focus only on Danger or Mandatory signs. Contrarily, in their paperwork they have regarded also the informative signs in order to achieve a complete classification task. For this task, they used an equipped vehicle of the UTBM laboratory to capture various sequences of images around the university campus. They achieved accuracies of 99.37% and 98.99% as the best results obtained for training the GTSRB and our European dataset respectively with the modified CNN 8-layers model.

Yassmina Saadna & Ali Behloul [20] claim deep learning methods for TSD and TSC are superior to color- and shape-based methods. They cite Aghdam et al (2016). [21] for achieving, at the time, the new best accuracy with a 99.55% with single CNN through grayscale images of signs in the GTSRB dataset. Yassmina Saadna & Ali Behloul suggest the need for higher resolution images for TSD, since cars traveling at fast speeds will have a harder time detecting signs in great distances, while also being able to process the number of frames the camera can take in a second. Even though detecting traffic signs with a high recall is important, they emphasize it’s more important to obtain a high precision to decrease false alarms.

Haifeng Wan, et al. [22] proposed a model which is called Traffic Sign Yolo (TS-Yolo) for the TSD under poor visibility and extremely limited vision. In their model, they utilized depthwise convolution (MixConv) and AFF. The MixConv layer can process different patterns with different resolutions. They used the TT100K dataset beside the 1000 images they captured in Shandong. Also, they did a data augmentation task too. As a result of this project, they achieved a precision of 74.53%.

Kaan Kocakanat, and Tacha Serif [23] claimed that the traffic signs are particular to countries even if the countries pursue the same convention. There isn't enough research and public dataset for Turkish Traffic Signs. Because of this difference, they needed a Turkish TSDS. This dataset contains 10842 images from 54 classes. For the detection and recognition task, they utilized the Faster R-CNN Inception-v2-COCO model from Tensorflow. As a result of the project, they achieved an average precision of 67.2% and average recall of 78.3% when trained with 51,217 epochs; an average precision of 76% and average recall of 82.8% when trained with 200,000 epochs.

Yanting Zhang, et al. [24] proposed a new real-word dataset that is called CTSD. This dataset contains information about weather, light conditions, occlusion, distance, color fading and camera angle. Other datasets such as GTSDB, LISA etc. do not explicitly regard these kinds of aspects. Therefore, CTSD has been created from online videos, driving records and phone cameras. Ultimately, they reached 2205 labeled pictures with 3755 signs having 153 different subclasses. Having this dataset, they experimented with six approaches: Haar+Adaboost, HOG+SVM, OverFeat-based, Faster R-CNN, YOLOv2, SSD.

# System Study Analysis

### Principles of System Analysis

1. Understand the issue and define the problem.
2. Research existing topics and papers regarding traffic sign classification and detection.
3. Research existing datasets and models for the TSD and TSR process.
4. Decide and choose public datasets to work with.
5. Understand how to augment traffic sign datasets with fog, rain, snow, and speed.
6. Study to increase precision through experimentation.

### Existing System with Limitations

There are no publicly available TSR and TSD datasets that include all of our weather condition requirements and speed implication. In the GTSRB and GTSDB, there are a limited number of weather conditions that negatively affect traffic sign recognition. Also, the pictures do not have blurs. To overcome this, we need to augment our dataset with these weather conditions and motion blur.

### Proposed System Features

* The model can detect and recognize traffic signs in good weather conditions.
* The model can detect and recognize traffic signs under motion blur.
* The model can detect and recognize traffic signs in poor weather conditions as foggy, rainy and snowy weather.
* The model can perceive an image that includes a traffic sign and label it according to its class with a minimum error.
* Users can upload a picture, which can be classified as one of the traffic signs.

# Specification of Requirements

* 1. **Hardware Requirements**

Following resources are used in our training:

* Google Colaboratory having the listed features:
  + GPU: Nvidia Tesla A100
  + RAM: 16 GB
  + HDD: 50 GB
* Internet connection
* I/O: Keyboard, mouse, monitor
  1. **Software Requirements**

Google Colab

Windows 10

Python 3.8

NumPy

TensorFlow

Keras

Matplotlib

PIL

OpenCV

# Description and Analysis of the Proposed System

* 1. **Advantages of the Proposed System**
* Creating extra images by imposing poor weather agents and motion blur into our dataset improved our prediction precision when faced with real-life scenarios.
* Using cloud computing reduces computing costs.
* Adopting weights from pre-trained models saved time and improved f1-score.
* Utilizing transfer-learning improved f1-score.
* Creating a validation split from the dataset tuned our model’s weights better.
* Using Keras-Tuner helped us choose parameters such as learning rate, dropout rate and number of layers etc. while trying all options and keeping the best.
* Using GTSDB’s road pictures represents a more effective approach to the deep learning process, where the algorithm has to detect the traffic sign instead of being shown only a traffic sign.
  1. **Disadvantages of the Proposed System**
* Augmented images do not look very realistic, since the images in the dataset lack any depth value. Since there is no depth, the augmentations occur as a 2D filter, meaning that snow, rain or fog aren’t intensified when looking at greater distance, which is not realistic. As a compromise, augmentations vary in effect from image to image and are sometimes stronger and sometimes weaker than they should be.
* While the sample size for GTSRB is large, it is mostly similar images with size differences multiplied per class. For example, an image is 29x30 pixels, while there is the almost same image in 30x30 pixels. This results in fewer effective samples in the dataset.
* Imgaug, which is the library used for augmenting poor weather conditions, is no longer up to date with current Python versions and libraries. An older version of Python and its libraries are needed to implement the weather augmentations.
* The GTSDB dataset only has 383 train, 54 test and 108 validation images for a total of 545 images. There weren’t enough images for a successful deep learning model to be developed using this dataset.
* Training detection models take days to get any meaningful model, therefore it is harder to try every model and its implementation.
* With only the limited amount of test images in the dataset, it becomes difficult to get realistic representative results.

# Specification of the Proposed System

* 1. **Input Specification**

**For Recognition Task:**

* Images containing traffic sign images in clear weather conditions and in partially poor weather conditions from GTSRB dataset. It has 39209 train and 12630 test images.
* Images that we augmented by imposing motion blur and poor weather condition agents such as rain, haze and snow into the GTSRB’s original images. There are 156836 motion blurred augmented images for training, 156836 poor weather augmented images for training, while there are 50520 test images for each augmentation.

With these two sources, there are 352881 train and 113670 test images.

However, we can randomly undersample the images if we exceed our computational limits.

Motion blur augmentation task uses both vertical and horizontal direction with different kernel sizes which are 5 and 10 in order to indicate different speeds of vehicles as shown in Figure 5 in comparison to Figure 4.

Weather conditions are augmented with Imgaug, an image augmentation library for Python. Imgaug has classes for weather augmentations, namely Fog, Rain, Snow and Cloud are used for these augmentations, as shown in Figure 5 in comparison to Figure 4.

Fig. 4. Original image from GTSRB [1]

 Fig. 5. Augmented versions of the original image; foggy, foggy alternative, snowy, rainy, vertical and horizontal motion blurred

**For Detection Task:**

* Images containing traffic sign images in clear weather conditions and in partially poor weather conditions from GTSDB dataset. It has 491 train and 108 test images.
* Images that we augmented by imposing motion blur and poor weather conditions such as rain, haze and snow into some portion of the GTSDB’s original images. There are 982 motion blurred augmented images for training and 216 motion blurred images for testing; there are 1473 poor weather augmented images for training and 324 poor weather augmented images for testing.

With these two sources, there are 2946 train and 648 test images.

Motion blur augmentation task uses both vertical and horizontal direction with kernel size 8 in order to indicate different speeds of vehicles as shown in Figure 7 in comparison to Figure 6.

Weather conditions are augmented with Imgaug, an image augmentation library for Python. Imgaug has classes for weather augmentations, namely Fog, Rain, Snow and Cloud are used for these augmentations as shown in Figure 7 in comparison to Figure 6.



Fig. 6. Original image from GTSDB [3]



Fig. 7. Augmented versions of the original image; foggy, snowy, rainy, and motion blurred images.

* 1. **Output Specification**
* Our TSR model outputs one of the class labels based on its prediction.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 | 25 | 26 | 27 |
| 28 | 29 | 30 | 31 | 32 | 33 | 34 |
| 35 | 36 | 37 | 38 | 39 | 40 | 41 |
| 42 |  | | | | | |

Table 1. Examples from GTSRB, each number indicating the class that belongs to the traffic sign

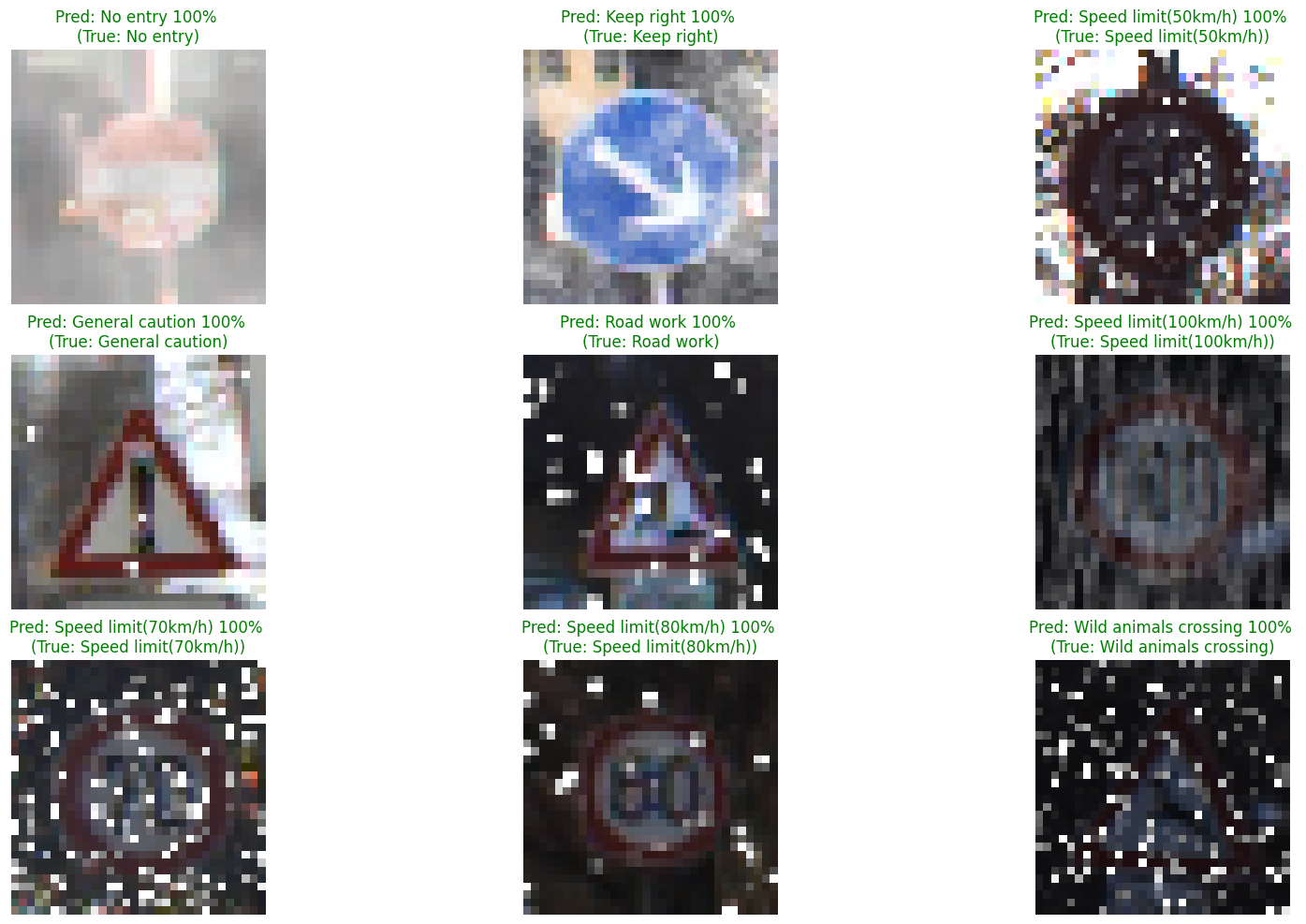


Fig. 8. Example of TSR model predictions

* Our TSD model boxes the traffic sign part from the whole road image. It outputs the name of the sign and precision for this prediction.



Fig. 9. Correct guess using YOLO v3 darknet

Fig. 10. Incorrect guess using YOLO v3 darknet

# Our Experıment Results

* 1. **Convolutional Neural Networks (CNNs) For Traffic Sign Classification**

**A. ResNet-50**

We tried ResNet-50 from He. et.al [25] with ImageNet [10] pretrained weights and without pretrained weights on each augmented version of the dataset, however the results are below 20% accuracy and F1-scores, therefore we ended experimenting on this model.

**B. MobileNet**

We tried MobileNet from Andrew G. Howard. et.al [26] with ImageNet [10] pretrained weights and without pretrained weights on each augmented version of the dataset, however the results are below 20% accuracy and F1-scores, therefore we ended experimenting on this model.

**C. Tiny VGG**

We tried TinyVGG from Xing Fang [27] without pretrained weights on each augmented version of the dataset. We removed the last 2 Convolutional2D and the last MaxPooling2D layer, since our images were smaller than the images used by Xing Fang.

On the motion blurred dataset (original + augmented), we trained the model for 10 epochs using Adam optimizer. The overall test accuracy and F1-scores are 91.36% and 87.89% respectively.

For the poor weather condition augmented dataset (original + augmented), this model achieved 84% accuracy and 81% F1-score in the test dataset.

**D. VGG16**

We tried VGG16 from Andrew Zisserman. et.al. [28] with ImageNet [10] on each augmented version of the dataset.

On the motion blurred dataset (original + augmented), we trained the model for 7 epochs using Adam optimizer. The overall test accuracy and F1-scores are 93.83% and 90.80% respectively.

For the poor weather condition augmented dataset (original + augmented), this model achieved 88% accuracy and 83% F1-score in the test dataset.

**E. VGG19**

We tried VGG19 from Andrew Zisserman. et.al. [28] with ImageNet [10] on each augmented version of the dataset.

On the motion blurred dataset (original + augmented), we trained the model for 2 epochs using Adam optimizer. The overall test accuracy and F1-scores are 93.00% and 90.15% respectively.

For the poor weather condition augmented dataset (original + augmented), this model achieved 87% accuracy and 83% F1-score in the test dataset.

**F. Our Custom Models**

We tinkered with the TinyVGG architecture, since it has much fewer parameters than the alternatives. The custom models we built baselining the TinyVGG architecture can be seen below, in Figure 11, 12, 13,14 and 15.

For the poor weather condition dataset (original + augmented), Baran’s custom model #1 has 91.33% accuracy and 89.80% F1-score in testing, giving the best overall result for this augmentation.

On the motion blurred dataset (original + augmented), after training Baran’s custom model #1 for 7 epochs using Adam optimizer it achieved 95.17% accuracy and 93.21% F1-score in testing, giving the best overall result for this augmentation. On the other hand, Buket’s custom model #2 achieved 94.47% accuracy and 92.51% F1-score, which is the best overall result from her trials.

Input

3x3 Conv s=2

BatchNorm

BatchNorm

3x3 MaxPool s=2

FC

Softmax

3x3 Conv s=2

3x3 Conv s=2

BatchNorm

32x32x3

15x15x32

15x15x32

7x7x64

7x7x64

3x3x64

1x1x128

1x1x128

1x1x128

Input

3x3 Conv

3x3 Conv

2x2 MaxPool

BatchNorm

3x3 Conv

3x3 Conv

2x2 MaxPool

32x32x3

30x30x16

28x28x32

14x14x32

14x14x32

12x12x64

10x10x128

BatchNorm

5x5x128

FC

BatchNorm

Softmax

5x5x128

1x1x512

1x1x512

Input

5x5 Conv

2x2 MaxPool

3x3 Conv

2x2 MaxPool

3x3 Conv

2x2 MaxPool

FC

Softmax

32x32x3

28x28x32

14x14x32

12x12x64

6x6x64

4x4x64

2x2x64

1x1x256

Fig. 11. Buket Custom Model 1

Fig. 12. Buket Custom Model 2

Fig. 13. Buket Custom Model 3

Input

3x3 Conv

Batch Normalization

3x3 Conv

Batch Normalization

2x2 MaxPool

Dropout

3x3 Conv

Batch Normalization

2x2 MaxPool

Dropout

Flatten

32x32x3

30x30x64

30x30x64

28x28x128

28x28x128

14x14x128

14x14x128

12x12x256

12x12x256

6x6x256

6x6x256

Dropout

Dense

Batch Normalization

Dropout

Batch Normalization

Dropout

Dense

Batch Normalization

512

256

256

256

256

128

128

Dropout

Dense

128

128

Dense

Batch Normalization

9216

512

512

Softmax

43

Input

3x3 Conv

3x3 Conv

2x2 MaxPool

Dropout

3x3 Conv

3x3 Conv

2x2 MaxPool

Dropout

Flatten

Dense

Dropout

Dense

Softmax

32x32x3

30x30x32

28x28x64

14x14x64

14x14x64

12x12x64

10x10x128

5x5x128

5x5x128

3200

512

512

43

Fig.14. Baran Custom Model #1

Fig. 15. Baran Custom Model #2

**TSR Results**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Input size | Parameters | GTSRB+Motion Blur | | GTSRB+Poor Weather | |
| Accuracy | F1 | Accuracy | F1 |
| TinyVGG | 32x32 | 1.52M | 91.36% | 87.89% | 84.84% | 81.12% |
| VGG16 | 32x32 | 15.00M | 93.83% | 90.80% | 88.04% | 83.45% |
| VGG19 | 32x32 | 20.31M | 93.00% | 90.15% | 87.60% | 83.32% |
| Baran’s Custom #1 | 32x32 | 5.27M | 95.17% | 93.21% | 91.33% | 89.80% |
| Baran’s Custom #2 | 32x32 | 3.69M | 93.82% | 91.36% | 91.01% | 88.94% |
| Buket’s Custom #1 | 32x32 | 0.13M | 85.79% | 82.23% | 76.7%% | 71.24% |
| Buket’s  Custom #2 | 32x32 | 1.76M | 93.82% | 91.36% | 90.60% | 88.85% |
| Buket’s  Custom #3 | 32x32 | 0.20M | 90.04% | 85.96% | 84.27% | 80.14% |

Table 2. Models tested on both augmented versions of GTSRB dataset

* 1. **Convolutional Neural Networks (CNNs) For Traffic Sign Detection**

1. **YOLO-v3**

We trained our new dataset which consists of the original GTSDB [3] dataset and our augments. We used the darknet framework for this approach. Since the parameters learned differ each run, we had 2 separate runs to see the variance. After a certain point, the best weights obtained are very similar and the model stagnates. It stops improving and can even become less accurate. The best precision and F1-score achieved are 61% and 48% respectively in various runs.

**B. YOLO- v7**

In YOLO v7, the accuracy and recall were performing worse than YOLO v3 for a given time. Each epoch took 10 minutes to process, taking much longer than YOLO v3 epochs which takes ~10 seconds. In our preliminary run without any augments, the results were underwhelming for the time spent, so it was abandoned.

**TSD Results**

**YOLO v3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Runs | Epochs | Precision | Recall | F1-score |
| Run 1 | Epoch 3000 | 0.44 | 0.34 | 0.38 |
| Epoch 3900 | 0.45 | 0.33 | 0.38 |
| Epoch 4800 | 0.45 | 0.32 | 0.37 |
| Epoch 5600 | 0.48 | 0.28 | 0.35 |
| Epoch 7000 | 0.49 | 0.33 | 0.39 |
| Epoch 7600 | 0.58 | 0.32 | 0.41 |
| Epoch 8600 | 0.55 | 0.31 | 0.40 |
| Epoch 9400 | 0.55 | 0.35 | 0.42 |
| Epoch 10000 | 0.55 | 0.37 | 0.44 |
| Epoch 11000 | 0.61 | 0.39 | 0.48 |
| Epoch 12500 | 0.49 | 0.32 | 0.39 |
| Epoch 13000 | 0.48 | 0.35 | 0.40 |
| Run 2 | Epoch 2500 | 0.35 | 0.34 | 0.34 |
| Epoch 3200 | 0.52 | 0.43 | 0.47 |
| Epoch 3400 | 0.48 | 0.45 | 0.46 |
| Epoch 3900 | 0.47 | 0.42 | 0.44 |
| Epoch 4700 | 0.47 | 0.43 | 0.45 |
| Epoch 5200 | 0.48 | 0.38 | 0.43 |

Table 3. YOLO v3 results, green highlights as best results per category

**YOLO v7**

|  |  |  |
| --- | --- | --- |
| Epochs | Precision | Recall |
| YOLO v7 800 epochs | 0.40 | 0.40 |
| YOLO v7 1200 epochs | 0.38 | 0.36 |

Table 4. YOLO v7 results

**VI.3. Comparison**

In this comparison, we have combined our motion blur and weather augmentation datasets with the original GTSRB dataset for the recognition task as shown in Table 5. Adding both of the augments to training makes little difference to the scores of GTSRB & Poor Weather. GTSRB & Motion Blur scores are lowered when combined with the poor weather conditions, while motion blur scores have gone lower than the score increase in poor weather. The newly added accuracy and F1 scores are lower than GTSRB + Motion and higher than GTSRB + Poor Weather as expected. Figures 16, 17, 18, 19, 20 and 21 show the changes in loss and F1-score during training processes for different models.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model | Input Size | Parameters | GTSRB + Motion Blur | | GTSRB+Poor Weather | | GTSRB+Motion+Weather | |
| Accuracy | F1 | Accuracy | F1 | Accuracy | F1 |
| Baran’s Custom #1 | 32x32 | 5.27M | 95.17% | 93.21% | 91.33% | 89.80% | 92.95% | 91.40% |
| Baran’s Custom #2 | 32x32 | 3.69M | 93.82% | 91.36% | 91.01% | 88.94% | 90.89% | 88.69% |
| Buket’s  Custom #2 | 32x32 | 1.76M | 94.47% | 92.51% | 90.60% | 88.85% | 92.80% | 91.04% |

Table 5. Comparing augments and models with their accuracy and F1 scores.

Green values are the newly added comparisons.

Since there were only 491 train images in the GTSDB dataset, we combined all augments together with the original dataset for the detection task. Therefore, we do not have a comparison between versions of the dataset for detection.

|  |  |
| --- | --- |
| Fig. 16. Baran Custom Model 1 showing  loss and validation loss per epoch on x-axis | Fig. 17. Baran Custom Model 1 showing F1 score and validation F1 score per epoch on x-axis |
| Buket Custom 2Fig. 18. Buket Custom Model 2 showing loss and validation loss per epoch on x-axis | Fig. 19. Buket Custom Model 2 showing F1 score and validation F1 score per epoch on x-axisBuket Custom 2 |
| Fig. 20. Baran Custom Model 2 showing loss and validation loss per epoch on x-axisBaran Custom Model 2 | Fig. 21. Baran Custom Model 2 showing F1 score and validation F1 score per epoch on x-axisBaran Custom Model 2 |

# Choice and Justification of Programming Language

**Python** is the most popular language used in scientific computing, data science and machine learning. It achieves great productivity with its performance by using low-level libraries and APIs. Over the last decade, Python has also grown in general purpose computing as well, with the support of the scientific computing community. A majority of machine learning and deep learning libraries are now exclusive to Python. It is a high-level programming language which is easy to read and learn, while also being incredibly powerful and efficient. The Python community and of its libraries are also very active, which makes it a desirable language to use for scientific computing [29].

According to a poll in 2019 which surveyed more than 1800 people, people voted Python for the top analytics, data science, and machine learning software [30].

# Conclusion

After doing a literature survey of deep learning, traffic sign recognition & detection and datasets available, a conclusion was reached on selecting the GTSRB and GTSDB datasets for the project’s implementation. Augmentations have been created using the methods described in input specifications. Advantages and disadvantages have been stated for the methods used for the project. For reasons mentioned in section VI, Python has been chosen as the programming language that will be worked with. Hardware and software requirements have been specified in section III. Input and output specifications have been listed in section V according to our data augmentation. After testing a multitude of models on classification, the best models have been laid out in the report. YOLO v3 is the model used for the detection task. To improve upon our model and progress, we recommend using a larger dataset with unique images.

# References

[1] Houben, Sebastian, et al.*, “*Detection of Traffic Signs in Real-World Images: The German Traffic Sign Recognition Benchmark”, *GTSRB*, 2011. https://benchmark.ini.rub.de/.

[2] United Nations, “Vienna convention on road signs and signals,” 1978.

[3] Houben, Sebastian, et al., “Detection of Traffic Signs in Real-World Images: The German Traffic Sign Detection Benchmark”, GTSDB, 2012. https://benchmark.ini.rub.de/.

[4] Al-Madani, H., & Al-Janahi, A.-R.,” Assessment of drivers’ comprehension of traffic signs based on their traffic, personal and social characteristics”, *Transportation Research Part F: Traffic Psychology and Behaviour,* 5(1), 63–76, 2002.

[5] Al-Rousan, T.M.; Umar, A.A. Assessment of Traffic Sign Comprehension Levels among Drivers in the Emirate of Abu Dhabi, UAE. Infrastructures 2021, 6, 122.

[6] Stallkamp J., Schlipsing M., Salmen J., and Igel C., Man vs. computer: “Benchmarking machine learning algorithms for traffic sign recognition”, *Neural Networks*, (0):–, 2012.

[7] R. Belaroussi and D. Gruyer, "Impact of reduced visibility from fog on traffic sign detection," 2014 IEEE Intelligent Vehicles Symposium Proceedings, 2014, pp. 1302-1306.

[8] Sato, Ryuhei & Domany, Keisuke & Deguchi, Daisuke & Mekada, Yoshito & Ide, Ichiro & Murase, H. & Tamatsu, Yukimasa. (2012). Visibility estimation of traffic signals under rainy weather conditions for smart driving support. Conference Record - IEEE Conference on Intelligent Transportation Systems. 1321-1326.

[9] Ahmed, S., Kamal, U., & Hasan, M. K., *“* DFR-TSD: A Deep Learning Based Framework for Robust Traffic Sign Detection Under Challenging Weather Conditions*.”, IEEE Transactions on Intelligent Transportation Systems,* 2021.

[10] Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, & Li Fei-Fei. (2009). “ImageNet: A large-scale hierarchical image database”. *IEEE Conference on Computer Vision and Pattern Recognition,* 2009

[11] Shao, F., Wang, X., Meng, F., Rui, T., Wang, D., & Tang, J. ,” Real-Time Traffic Sign Detection and Recognition Method Based on Simplified Gabor Wavelets and CNNs. Sensors”*,*Army Engineering University of PLA,Nanjing, 2018.

[12] Mogelmose, A., Liu, D., & Trivedi, M. M. ,“Detection of U.S. Traffic Signs.”, *IEEE Transactions on Intelligent Transportation Systems, 16(6).3116–3125.* 2015.

[13] Osipov, Aleksey, Ekaterina Pleshakova, Sergey Gataullin, Sergey Korchagin, Mikhail Ivanov, Anton Finogeev, and Vibhash Yadav, 2022, "Deep Learning Method for Recognition and Classification of Images from Video Recorders in Difficult Weather Conditions", *Sustainability* 14, no. 4: 2420.

[14] Greenhalgh, J., & Mirmehdi, M. ,“Recognizing Text-Based Traffic Signs.” *IEEE Transactions on Intelligent Transportation Systems,* 16(3), 1360–1369. 2014.

[15] Zhu Z., Liang D., Zhang S., Huang X., Li B. and Hu S., "Traffic-Sign Detection and Classification in the Wild," 2*016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2110-2118.

[16] Liu C., Li S., Chang F. and Wang Y., "Machine Vision Based Traffic Sign Detection Methods: Review, Analyses and Perspectives," in IEEE Access, vol. 7, pp. 86578-86596, 2019.

[17] Lin, S.; Zhang, Z.; Tao, J.; Zhang, F.; Fan, X.; Lu, Q. Traffic Sign Detection Based on Lightweight Multiscale Feature Fusion Network. Sustainability 2022, 14, 14019.

[18] Serna, C. G., & Ruichek, Y., “Classification of Traffic Signs: The European Dataset.”, M.S. Thesis, University Bourgogne Franche-Comté, University of Technology of Belfort-Montbéliard, F-90010 Belfort, France, 2018.

[19] Economic Commission for Europe-Inland Transport Committee, ‘‘Convention on road signs and signals,’’ United Nations Treaty Ser., vol. 1091, p. 3, Nov. 1968.

[20] Saadna, Y., Behloul, A. , “An overview of traffic sign detection and classification methods. “, *Int J Multimed Info Retr 6*, 193–210, 2017.

[21] H.H. Aghdam, E.J. Heravi, D. Puig,” A practical approach for detection and classification of traffic signs using Convolutional Neural Networks”, *Robotics and Autonomous Systems,* 2016.

[22] Haifeng Wan, Lei Gao, Manman Su, Qinglong You, Hui Qu, Qirun Sun, "A Novel Neural Network Model for Traffic Sign Detection and Recognition under Extreme Conditions", *Journal of Sensors*, vol. 2021, Article ID 9984787, 16 pages, 2021.

[23] Kocakanat, K. & Serif, T., ”Turkish Traffic Sign Recognition: Comparison of Training Step Numbers and Lighting Conditions”, *European Journal of Science and Technology*, (28), 1469-1475, 2021.

[24] Zhang, Y., Wang, Z., Qi, Y., Liu, J., & Yang, J., “ CTSD: A Dataset for Traffic Sign Recognition in Complex Real-World Images.*”, IEEE Visual Communications and Image Processing (VCIP),* 2018.

[25] Kaiming He, Xiangyu Z., Shaoqing R., Jian S.,” Deep Residual Learning for Image Recognition”, *Microsoft Research*, 2015.

[26] Andrew G. H., Menglong Z., Bo C., Dmitry K., Weijun W., Tobias W., Marco A., Hartwing A., “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications”, *Google Inc*., 2017.

[27] Fang, X.“Understanding Deep Learning via Backtracking and Deconvolution”. *Journal of Big Data*, 2017.

[28] Karen Simonyan & Andrew Zisserman, “Very Deep Convolutional Networks For Large-Scale Image Recognition”, *ICLR*, 2014.

[29] Raschka, S.; Patterson, J.; Nolet, C. Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence. Information 2020, 11, 193.

[30] Piatetsky, G. Python Leads the 11 Top Data Science, Machine Learning Platforms: Trends and Analysis. 2019. Available online: https://www.kdnuggets.com/2019/05/poll-top-data-science-machine-learning-platforms.html (accessed on 26 December 2022).

# APPENDIX: Task Distribution

**Buket’s tasks: Clear weather**

* Data augmentation for motion blur on clear weather dataset
* Preparing the final dataset with these conditions
* Searching deep learning models for this variation of the dataset
* Implementing deep learning model & building the model
* Fine tuning the models
* Comparing between top performing models and making a visualization

**Baran’s tasks: Poor weather**

* Data augmentation for poor weather: fog, rain and snow on clear weather dataset
* Preparing the final dataset for these conditions
* Searching deep learning models for this variation of the dataset
* Implementing deep learning model & building the model
* Fine tuning the models
* Comparing between top performing models and making a visualization

**Common tasks:**

* Literature survey
* Traffic Sign Detection
* Comparing between the results of clear weather with motion blur added and poor weather results
* Preparing the report