Traffic Sign Detection and Recognition for Autonomous Vehicles Using Transfer Learning

MSc Research Project
MSc in Data Analytics

Naresh Potla
Student ID: x21126241

School of Computing
National College of Ireland

Supervisor: Aaloka Anant

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Naresh Potla
Student ID:	X21126241
Programme:	MSc in Data Analytics **Year:** 2022 - 2023
Module:	Research Project
Supervisor:	Aaloka Anant
Submission Due Date:	December 15th, 2022
Project Title:	Traffic Sign Detection and Recognition for Autonomous Vehicles Using Transfer Learning
Word Count:	4536 **Page Count** 17

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Naresh Potla
Date:	February 1st , 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Traffic Sign Detection and Recognition for Autonomous Vehicles Using Transfer Learning

Naresh Potla
x21126241

Abstract

Driving a vehicle requires a number of important duties, one of which is identifying and recognising traffic sign boards. They offer details regarding the state of the roads so that you can drive safely and enjoyably. The potential uses for traffic sign recognition in automated driving are numerous. Small traffic signs' ability to be recognized will be impacted by external elements such as lighting, fading colours of indicator markers, climate, and diffraction. One of the main causes of car accidents is carelessness, specifically a failure to comprehend and misrepresenting traffic signs. A driving simulation program in-vehicle has the ability to give drivers new visual feedback for a better drive. This research demonstrates various transfer learning methods like EfficienNetV2L and VGG19 used for classifying the different sign boards. To achieve a better performance model, I applied various data augmentation techniques with the help of ImageDataGenerator, which is available in Keras Library. Achieved an accuracy of 78% using VGG19, and EfficientV2L outperforms the VGG19 model with an improved accuracy of 93%.

1 Introduction

Motivation & Background

Street signs are primarily designed to warn commuters, drawing human focus from the distance while we hurry by with their irregular-shaped forms and bright colours. An automobile that can sense the surroundings and function without operator interaction is an autonomous vehicle. No human operator or commuter is ever required to run the automobile or even to ride in the automobile whatsoever. Wherever a conventional car can travel, an automated car can now go, as well as performing anything a capable pilot can do. The ability to recognize and understand road signs throughout all of their confusing worldwide variants is among the numerous difficulties self-driving cars face.

An Intelligent Transportation System (ITS) that helps in monitoring the automobile as well as the environment may potentially include a roadway and traffic sign recognition system. The primary objective of ITS is the incorporation of information technology into commuting roadways. These technologies can include technology for traffic management and observation, digital information boards, sensors, and in-car location services. Modern innovation is used by intelligent transportation systems to increase highway safety, efficiency, and reduce their toxic emissions.

Subsequently, the wide use and growth of autonomous vehicles (AV) will be possible thanks to the swift evolution of technology and science. For increased monitoring and security, AVs utilize detectors and machine learning software. Technology is helpful in decreasing human involvement and improving the use of modern technology for communication with a clear objective of mechanization. This contributes to increased productivity, cost-effectiveness, safe driving conditions, and, most significantly, environmental friendliness. A revolutionary autopilot system including removable steering and a cheap cost relative to earlier generations of vehicles was introduced by Baidu.

A lot of previous research conducted on this subject is still in use today by important automobile makers. But the challenges that bad lighting, shimmer, spinning, displacement, and diffraction present to the system enhance the complicated variability range which a predictor should navigate. Using the freely accessible German Traffic Sign Recognition Benchmark (GTSRB) dataset, which contains 43 different classes and over 50,000 photographs.

Research Question

How transfer learning methods helps autonomous cars in effectively detecting traffic signs for safer roads?

Objective of Research Question

My research's main objective is to apply previously conducted identification and segmentation studies in the automotive industry with a focus on images in autonomous vehicles. It also emphasizes the applicability of these approaches in other fields of research. This spurred curiosity in merging the problem with an already existing solution because it is difficult to distinguish different traffic sign boards in all climatic conditions, as in other efforts listed in the literature review below. It is crucial to recognize image resizing in order to incorporate it into a transfer learning model, data augmentation, minimize interrupted noises, and uncertainties in the implementation of image processing in order to move the investigation forward.

2 Literature Review

2.1 Image Processing using CNN methods

While driving vehicles to avoid hazards, (Bhatt et al. 2022) proposed a deep learning CNN model for safer roads. In this model, as part of pre-processing, the entire data is split into 3 parts, like transform to Gray scale, histogram equalization and normalization. And also, to overcome the overfitting of the poorly represented classes, he used data augmentation. In this model he used 4 convolutional layers, 2 pooling layers, 1 flattened layer and 4 fully connected layers. By using the final model, 95.15% was accuracy achieved as an end result.

To detect road objects while driving automobiles (Güney & BAYILMIŞ 2022) used Faster R-CNN, a two-stage approach for object detection. The object detection model uses CNN to classify the images in two stages. The region-based models use high state of the art processes to accelerate. In this research, the entire dataset was split into training and validation at 80% and 20% respectively. For testing, they evaluated the model on the test dataset to test 22 different sign classes, like people, cycles, yield and so on. With the evaluated model, they got the accuracy of the model of 89%..

To recognize and classify traffic signs under abnormal environmental conditions, (Reddy et al. 2022) proposed a CNN model for feature extraction as well as for classification. As part of the classification process, initially all the high resolution images are scaled down to small resolution, and also RGB images are converted to Gray scaled images. For traffic sign detection, which is received from the camera, all images are changed from the RGB images into HSV, which is the same as how normal eyes observe the images based on different colours and their shapes.

For recognition and detection of traffic sign boards, (KhabiriKhatiri et al. 2021) used the GTSRB dataset by applying an adoptive colour segmentation depending on the value of neighbourhood mean saturation along with circular Hough transformation. In this process, he used a 12 class CNN model, trained over 17000 images with a split of trains and tested with a proportion of 80 and 20 respectively. The CNN model got 99% accuracy for validation data. 0.54ms is the time taken for classification one image.

For the traffic sign identification and road lines in the Advanced driver assistance system (ADAS) (Nieroda et al. 2022), using heat maps, a quick, easy, and effective way to summarize the data. It is a two-dimensional depiction of data. It was mainly used to analyse the high dimensional data in larger datasets produced by sensors.Heat maps are generated based on the analysis of Traffic Signs and Lane Identification algorithms. They performed the experiment using a python. For generating heat maps for traffic signs took 20 minutes for an 8 hour drive.

In order to detect traffic boards based on image series within inaccurate colour (Swapna et al. 2021) used a colour-based unit partition method along with greatest hue and congestion coefficients. As part of classification, they mainly considered only triangles and circles used for filtration. In this process, they transform all the RGB images into HSV first, which describes the images' hue, purity, and brightness. To recognise the traffic sign from the input image,CNN used it.

To identify traffic boards which are configured in public places in all the challenging conditions, (Ahmed et al. 2021) used the Enhance Net encoder decoder CNN architecture, over CURE-TSD dataset. To evaluate the results, compare the result of Enhance Net with Faster RCNN, Resnet V2, DenseNet and MSA-YOLOV3 models. Using the Enhance Net method, he achieved 91.1% and 70.71% precision and recall, which means an improvement of 7.58% precision and 35.90% recall.

For identification and recognition of traffic boards for safer journeys as part of the advanced driving assistance system (Fan 2021) proposed a system consisting of 8 convolutional neural network layers. Here, to solve the components that influence the precision, he uses group sparse coding technique. In the earlier stage, images were distributed randomly and the model was executed with Adam optimiser. The accuracy of the Multi CNN model achieves approximately 99%.

2.2 Traffic Sign Identificatin using YOLO

For accurate identification of sign boards from distinct places using 5G technology for autonomous vehicles, (Ma et al. 2021) proposed YOLOv3, which is based on Deconvolution feature fusion framework. In this model, it was derived by combining a deep features map that was sampled and then merged with a shallow features map. For detection of the target, the deconvolutional fusion module restores the current prediction layer. To form this, a convolutional layer with dimensionality reduction was used. The experimental results show that DFF-YOLOv3 models achieve a mean accuracy of 75%.

(Fernando & Sotheeswaran 2021) used You Only Look Once Version4(YOLOV4) model in his research to detect traffic signs under challenging conditions like different lighting conditions, twisted signs, and radiance. In this model, feature extraction was elaborated by using a backbone architecture, which is CSPDarknet53. Here the Dense block features are bifurcated by using cross stage partial connections. To enhance the accuracy and performance of the model, a total of 161 layers contributed. As a final result, out of 43 classes, for 18 classes, they got 100% and the overall accuracy of the model was 84.7%.

To detect real time traffic signs and pedestrians, (Güney et al. 2022) proposed a YOLOv5 model which is a single stage recognizer, and also a high speed and identification accuracy for its performance on mobile platforms. To evaluate the model's speed and accuracy, it was implemented with training and evaluated on GPU embedded platforms. The model was trained on by a Tesla P100 GPU for over 8 hours over 850 epochs and also tested on Jetson Nano and Jetson Xavier.

For accurate detection and identification of sign boards with more speed and less processing time, (Yuan et al. 2022), proposed the YOLOv5SA2 method, which includes YOLOv5 as a base architecture. In this, initially, the initial augmentation strategy was applied for imbalanced classes and then, for horizontal links, used a path aggregation module for the Feature Pyramid Network. Finally, to fix the aliasing effect, an attention detection head module was used on the TT100K dataset. The model got 87.3% mAP with a constant maintenance of 87.7 frames per second.

For detecting traffic objects precisely and most accurately with high input frame rate, (Kumagai & Goto 2022) used the YOLOv4 method. In this process, the input image stream is initially passed through the YOLOv4, the cropped images are then passed through the CNN. Here the YOLO method puts the image into grids, anticipating the boundary, predicting and choosing the boundary. The performance of the evaluated by comparing the mAP for YOLOv3 and YOLOv4 as 96.4% and 98.8% respectively.

2.3 Transfer learning

Simply, the recognition and detection of traffic signs with different and uneven samples is the biggest challenge. To overcome this issue, (Li et al. 2022) implemented the VGG16 model, which is a light weight two-stage sign recognition framework. As part of the initial stage, he presented getting all the traffic sign categories and then, implemented another stage to recognise digital characters getting on the sign boards. He evaluated all the experiments on the TT100K dataset, which includes 61 classes segmented into training and test data between 3896 and 1961 respectively.

3 Research Methodology

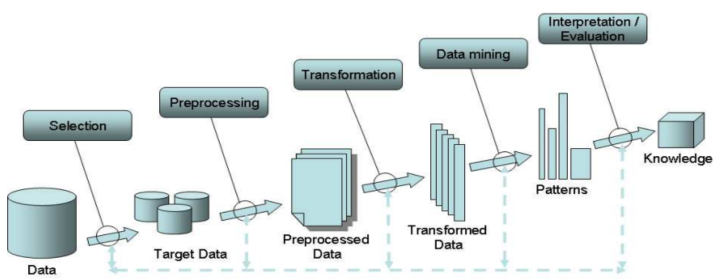

Figure 1: KDD Methodolgy

The systematic method of identifying characteristics or information from a large number of databases or datasets is known as knowledge discovery from data (KDD).As per (De Martino et al. 2002), KDD is an information management technology that combines a number of data management technologies, including database administration, data warehousing, statistical machine learning, predictive analysis, and others like graphics and parallel computations.

KDD applies specific principles; the initial stage has always been the selection of the dataset (or part of the dataset) from which the information should be extracted. In the second phase, the homogeneous data is gathered from different sources, if any, it is by using data migration or data synchronous tools. In the third step, for exploratory data analysis, we should get rid of the noise or unrelated data. The fourth step, the Transformation phase, translates the input data into formats or patterns. To create the patterns, the data should be unique and oriented in a single type. In order to do data mining and take out the needed information or format hidden, this transformation phase assists us with data sorting and arrangement, which can be helpful for further business tasks. Data mining uses machine learning technologies to identify trends and identify inferences, and graphical tools are used to plot, diagram, and also visualize the conclusions drawn from the data. The following phase is evaluation or interpretation, which enables information to be gained from the data using BI tools and can be utilized to create a demonstration that highlights patterns. For this research project, I got the German Traffic Sign Benchmark dataset (GTSRB), which is available from Kaggle, a public website. The dataset consists of 43 classes of different traffic signs. The GTSRB is divided into train and test categories consisting of 39209 and 12630 images respectively. The dataset was gathered from the real world and deals with different kinds of images: low light, less brightness, dark background and so on.

4 Design Specification

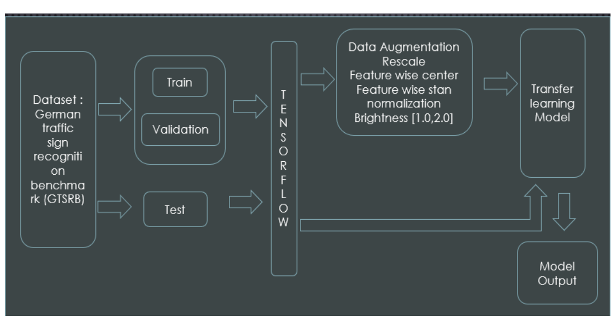

Figure 2: Implementation Architecture

In this research project, I describe the specification of the project and implementation steps are shown in Figure 2. As per the KDD methodology, the approach and specification of each phase of the project are explained below.

- As per KDD methodology, the experiment starts by collecting GTSRB dataset from the public website, Kaggle. It has 43 classes, which consist of different kinds of traffic signs like speed limits, diversions, Yield and so on.
- The GTSRB dataset collected from real world scenarios split into train and test folders consists of 39209 and 12630 images respectively. During the experiment, I am going to split 80% data for training and 20% data for validation.
- Next, I am using ”ImageDataGenerator” to perform data augmentation for creating the images during the runtime with featurewise center, featurewise std normalization and brightness within the range of 1 to 2.
- Next, I feed these augmented images to the transfer learning models, EfficientNetV2L and VGG19 models used for testing accuracy and execution time over the train dataset for the traffic signs.
- Finally, the model which is built on train images, is going to be tested against the test data images for the end output, called test accuracy, which can be used for evaluation with the help of confusion matrix

5 Implementation

The most crucial element of the research project is implementation, where we will try to use the technique and architectural design developed from the studies conducted in the past research papers in the associated work portion to carry out the experiment.

5.1 Environment setup

Due to the lack of computing resources, the implementation and execution of all the code is done in a Kaggle notebook. At Kaggle, I got a 4.1 GB Disk, 13 GB of RAM and also a GPU T4*2 Graphical processing unit and selected python language. Also installed, version of TensorFlow 2.9.2 library to implement deep learning models. The Imported Keras module from Tensorflow library, is extensively recommended for image processing.

5.2 TensorFlow

It is a free and open-source software library for artificial intelligence and machine learning. Deep neural network training and inference are its core research areas, while it can be used for many other tasks as well. TensorFlow was developed by the Google team for use in internal Google development and research.

As mentioned in the design specification, I am dealing with the problems of deep learning, which requires Keras, which is a Python-based deep learning API. It was created with the goal of facilitating quick experiments on the top of the Machine Learning framework TensorFlow. By using TensorFlow, the user can produce structures that explain how data flows through a series of nodes. All the nodes in the structure mean mathematical functionality, and there is a multidimensional data array between the nodes. It allows the execution of calculations across a range of hardware, including CPU and GPU.

For the model implementation, I used to install TensorFlow 2.9. 2 version in my Kaggle notebook, because it supports high level APIs. With the help of these, we can construct Machine Learning models using Neural Networks. To adopt best practices for augmentation, model recording, for monitoring the performance and retraining the model, TensorFlow can be used.

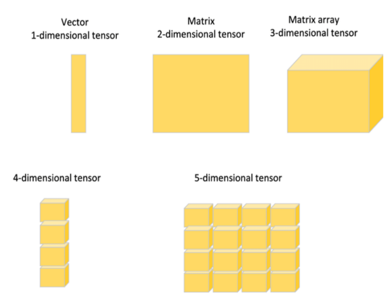

Figure 3: Multidimensional array of Tensors

5.3 Image Processing

The GTSRB dataset contains 51840 total images. It was split into train and test folders with images consisting of 39209 images for training and 12630 images for testing. Again, the training dataset bifurcated into 80% for training and 20% for validation.

Machine learning usually makes use of the data preparation process known as normalization. Normalization is the technique of scaling all the fields in a dataset to a uniform value. Whenever the ranges of features differ, it is necessary. To normalize the pixel data, which was within the range of 0 to 255 in my dataset, I used the ImageDataGenerator class for making pixel values range from 0 to 1, which is suitable for neural network models. To standardize every image, I used the featurewise_center parameter, which subtracts the entire dataset mean value from every element in the multidimensional array and the remainder values are divided by a standard deviation value by defining the featurewise_std_normalization as true.

To adjust the brightness for all the input images, I used the brightness_range parameter, which takes a list of values, representing the lower (0) and upper (1) ranges. Where 0 means no brightness and 1 means the actual image without any change, anything above just makes the image brighter. Hence, for all my images I choose the brightness range [1.0,2.0]. Now, I divide the training dataset into trains and validation with resized images consists of 64*64 pixels with a batch size of 64.

Figure 4: Input images after adjusting the brightness range [1.0,2.0]

Figure 5: Input images after adjusting the brightness range [0.0,1.0]

The above figures show the difference in brightness ranges. The images are displayed brighter when setting the brightness range [1.0,2.0] and for the brightness range [0.0,1.0] it just displaying actual images. The images getting shaded for brightness range beyond 2.

5.4 Models Implemented

Before implementing any model, we should understand the discrepancies between transfer learning and machine learning. In all the machine-learning model approaches, we need to train the model from scratch. To do this, everyone should have high configurations for computations, which is very expensive, and also to get the better performance, we have to train the model with a dataset which is a bigger size. Whereas, in the case of transfer learning, we need to implement the model from scratch. The model gains knowledge from one model, and it is transferred to another model. For instance, you could use a model that is identical to the one you trained to predict additional things like a wallet if you have trained a straightforward classifier to determine whether images contain a bag. By this, we can use the same huge amount of computing time and resource costs.

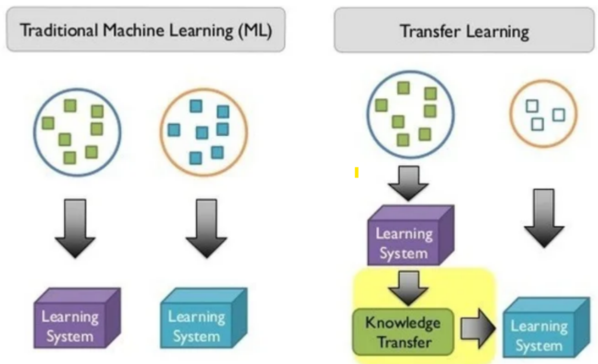

Figure 6: Machine learning Vs Transfer learning

Based on the architecture of EfficientNetv2, it has higher training speed and improved attribute effectiveness than other pre-trained models (refer below graph).

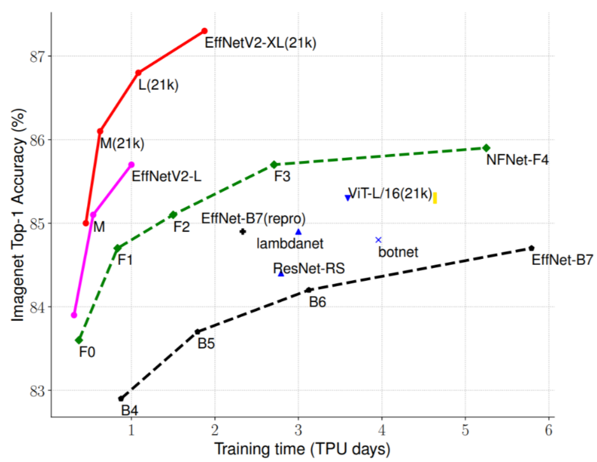

Figure 7: Comparison of Different Models over ImageNet Dataset

5.4.1 Applying Efficientnetv2l Model

The EfficientNetV2 family of image classification models outperforms earlier works in terms of parameter efficiency and training speed. Our EfficientNetV2 models, which are based on EfficientNetV1, use neural architecture search (NAS) to jointly maximize model size and training time. They are scaled up in a way that allows for faster training and interpretation speed.

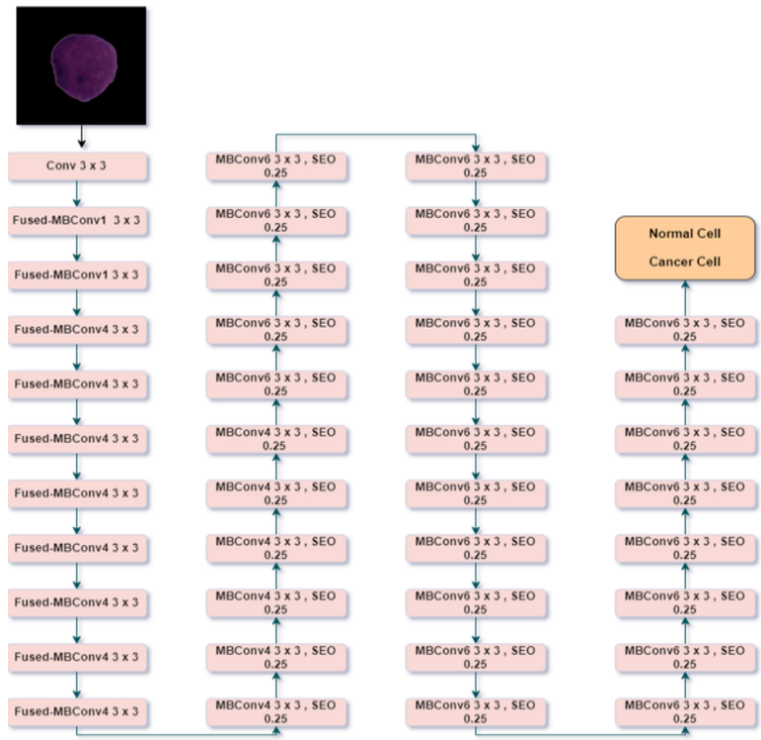

Figure 8: EfficientNetV2L Model Architecture

- In the initial layers, EfficientNetV2L makes considerable use of both MB-Conv and the recently added fused-MBConv.
- As smaller expansion ratios often have less memory access overhead, EfficientNetV2L favours smaller expansion ratios for MBConv.
- EfficientNetV2L prefers 3x3 kernel sizes that are smaller, but it also uses more layers to make up for the smaller receptive field.

5.4.2 Applying VGG19 Model

The name VGG stands for Visual Geometry Group. It consists of multiple layers over the standard Convolutional Neural Network. VGG19 is one of the variants in VGG models and contains 19 layers (16 convolution layers, 3 Fully connected layers, 5 Maxpooling layers and 1 softmax layer).

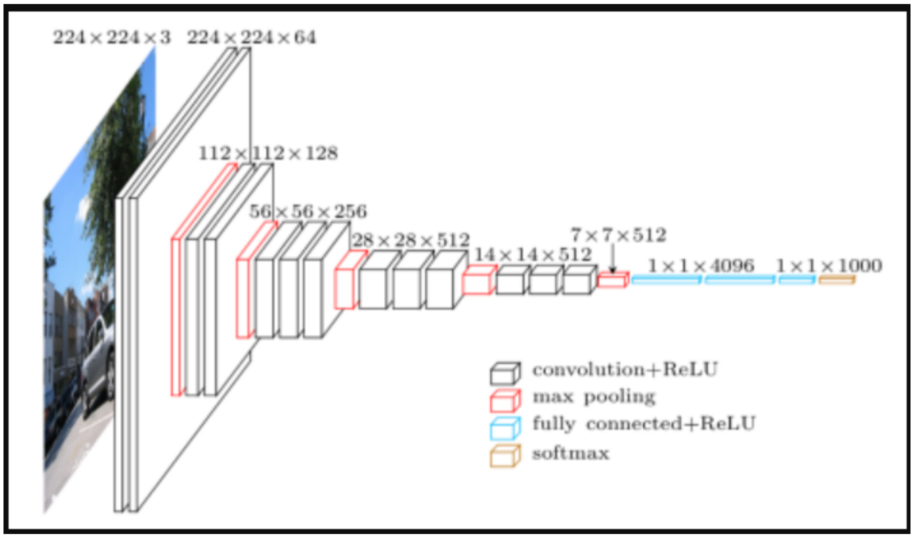

Figure 9: VGG19 Model Architecture

- The input to this network was a fixed-size (224 * 224) RGB picture, indicating that the matrix was shaped (224,224,3).
- The sole processing carried out was to subtract the mean RGB value from each pixel over the whole training set.
- They were able to cover the entirety of the image by using kernels that were (3 * 3) in size and had a stride size of 1 pixel.
- To maintain the image's spatial resolution, spatial padding was applied.
- With the help of SRIDE 2, max pooling was carried out over a 2 * 2 pixel window.
- Rectified linear unit (ReLu) was used as a follow-up to introduce non-linearity and speed up computing by improving model classification.
- Implemented three completely linked layers, the first two of which were 4096 in size. The third layer, a softmax function, followed by a layer with 1000 channels is a 1000-way ILSVRC classification

5.5 Model Output

The classification model is utilized to forecast the input photos from the testing folder to produce a confusion matrix using the EfficientNetV2L model and the transfer learning phenomena. These outcomes demonstrate how well our model performed. We obtained the test accuracy after testing the model using test photos from the test dataset. The findings section highlights the output of the probabilities as a confusion matrix.

6 Evaluation

Confusion matrix and classification report are used to analyse the findings and evaluate our research. As my research was carried out against VGG19 and EfficientNetV2L models, it achieved better accuracy of 93% with EfficientNetV2L. In order to assess the performance of the EfficientNetV2L approach for recognizing traffic signs, precision, recall, and the F1-Score are chosen in classification report.

Classification Report

Precision - It is the ratio of correctly predicted positive observations to the total predicted positive observations.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}$$

Recall - It is the ratio of correctly predicted positive observations to all the observations in the current class.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$$

F1-Score - It is the weighted average of Precision and Recall.

$$\text{F1} = \frac{\text{2* Recall * Precision}}{\text{Recall + Precision}}$$

where TP stands for the number of items that were accurately predicted; FP for the number of items that were mistakenly identified, and FN for the number of objects that were actually identified.

	precision	recall	f1-score	support
0	0.96	0.90	0.93	60
1	0.86	1.00	0.92	720
2	0.96	1.00	0.98	750
3	0.91	0.94	0.92	450
4	0.93	0.99	0.96	660
5	0.84	0.99	0.91	630
6	0.97	0.99	0.98	150
7	1.00	0.90	0.95	450
8	0.98	0.98	0.98	450
9	0.95	1.00	0.97	480
10	1.00	0.89	0.94	660
11	0.97	0.93	0.95	420
12	0.92	0.96	0.94	690
13	1.00	0.99	1.00	720
14	0.82	0.87	0.84	270
15	0.99	0.46	0.63	210
16	1.00	1.00	1.00	150
17	0.95	0.95	0.95	360
18	0.97	0.72	0.83	390
19	0.97	0.98	0.98	60
20	0.99	1.00	0.99	90
21	0.98	0.98	0.98	90
22	1.00	0.75	0.86	120
23	0.99	0.90	0.94	150
24	0.87	1.00	0.93	90
25	0.96	0.97	0.96	480
26	0.60	0.92	0.72	180
27	0.81	1.00	0.90	60
28	0.95	0.97	0.96	150
29	0.87	1.00	0.93	90
30	0.79	0.84	0.81	150
31	1.00	1.00	1.00	270
32	0.90	1.00	0.94	60
33	0.99	0.81	0.90	210
34	0.75	0.97	0.85	120
35	0.95	0.95	0.95	390
36	1.00	0.88	0.94	120
37	0.80	0.98	0.88	60
38	0.99	0.80	0.88	690
39	0.82	0.67	0.74	90
40	0.87	0.86	0.86	90
41	1.00	0.75	0.86	60
42	0.86	1.00	0.92	90
accuracy			0.93	12630
macro avg	0.92	0.92	0.91	12630
weighted avg	0.94	0.93	0.93	12630

Figure 10: Classification Report

Confusion Matrix

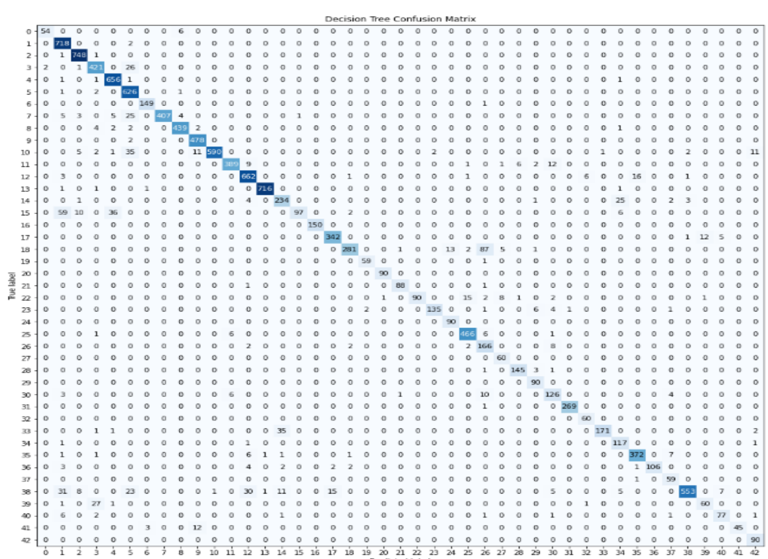

Figure 11: Confusion Matrix

7 Conclusion and Future Work

In this research, I implemented transfer learning models like VGG19 and EfficientNetV2L for classification of traffic signs in over 43 different classes. For this, I used GTSRB dataset, which contains all types of images, like low brightness, distorted, low clarity and so on, achieving an accuracy of 74% by applying the brightness range(0.0, 1.0) to the existing data. Post implementing the EfficientNetV2l model, with data augmentation like increasing the brightness range (1.0,2.0) it achieves 93% accuracy. I carried out the experiment in Kaggle over 20 epochs with a batch size of 64 along with GPU T4*2 computing resources. For future work, I would like to work with various data augmentation techniques from over 100 epochs with high computing resources.

Acknowledgement

I would like to thank my lecturer, Aaloka Anant, for all of his assistance and guidance. Aaloka Anant was continuously given assistance and was enthusiastic and willing to assist in any way he could during the entire study implementation process.

References

Ahmed, S., Kamal, U. & Hasan, M. K. (2021), 'Dfr-tsd: A deep learning based framework for robust traffic sign detection under challenging weather conditions', *IEEE Transactions on Intelligent Transportation Systems* .

Bhatt, N., Laldas, P. & Lobo, V. B. (2022), A real-time traffic sign detection and recognition system on hybrid dataset using cnn, *in* '2022 7th International Conference on Communication and Electronics Systems (ICCES)', IEEE, pp. 1354–1358.

De Martino, M., Bertone, A., Albertoni, R., Hauska, H., Demšar, U. & Dunkars, M. (2002), 'Information visualisation for site planning'.

Fan, S. (2021), Road sign detection and recognition system based on multi-layers convolutional neural network model trained with german traffic sign recognition benchmark, *in* '2021 4th World Conference on Mechanical Engineering and Intelligent Manufacturing (WCMEIM)', IEEE, pp. 395–399.

Fernando, W. & Sotheeswaran, S. (2021), Automatic road traffic signs detection and recognition using 'you only look once'version 4 (yolov4), *in* '2021 International Research Conference on Smart Computing and Systems Engineering (SCSE)', Vol. 4, IEEE, pp. 38–43.

Güney, E. & BAYILMIŞ, C. (2022), 'An implementation of traffic signs and road objects detection using faster r-cnn', *Sakarya University Journal of Computer and Information Sciences* **5**(2), 216–224.

Güney, E., Bayilmiş, C. & Çakan, B. (2022), 'An implementation of real-time traffic signs and road objects detection based on mobile gpu platforms', *IEEE Access* **10**, 86191–86203.

KhabiriKhatiri, R., Abd Latiff, I. & Mohamad, A. S. (2021), Road traffic sign detection and recognition using adaptive color segmentation and deep learning, *in* '2021 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)', IEEE, pp. 134–139.

Kumagai, K. & Goto, T. (2022), Improving accuracy of traffic sign detection using learning method, *in* '2022 IEEE 4th Global Conference on Life Sciences and Technologies (LifeTech)', IEEE, pp. 318–319.

Li, Z., Chen, M., He, Y., Xie, L. & Su, H. (2022), An efficient framework for detection and recognition of numerical traffic signs, *in* 'ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)', IEEE, pp. 2235–2239.

Ma, X., Li, X., Tang, X., Zhang, B., Yao, R. & Lu, J. (2021), 'Deconvolution feature fusion for traffic signs detection in 5g driven unmanned vehicle', *Physical Communication* **47**, 101375.

Nieroda, B., Wojakowski, T., Skruch, P. & Szelest, M. (2022), A heatmap-based approach for analyzing traffic sign recognition and lane detection algorithms, *in* '2022 26th International Conference on Methods and Models in Automation and Robotics (MMAR)', IEEE, pp. 217–221.

Reddy, M. P., Mohiuddin, M. F., Budde, S., Jayanth, G., Prasad, C. R. & Yalabaka, S. (2022), A deep learning model for traffic sign detection and recognition using convolution neural network, *in* '2022 2nd International Conference on Intelligent Technologies (CONIT)', IEEE, pp. 1–5.

Swapna, Y., Reddy, M. S., Sai, J. V., Krishna, N. S. S. & Teja, M. V. (2021), Deep learning based road traffic sign detection and recognition, *in* '2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA)', IEEE, pp. 1–4.

Yuan, X., Kuerban, A., Chen, Y. & Lin, W. (2022), 'Faster light detection algorithm of traffic signs based on yolov5s-a2', *IEEE Access* .