Balar Nancy – 13928122, Ekta Balhotra - 1379109

Computer Vision - Object Classification – German Traffic Sign Classification

***Abstract* — With the surge in demand for autonomous vehicles and driver assistance technologies, accurate traffic sign recognition has emerged as a critical component for ensuring road safety. In this research project, we focus on the development of an efficient object classification system specifically tailored for identifying German traffic signs, a crucial task in the realm of autonomous driving. Recognizing the real-time constraints inherent in autonomous driving systems, our approach prioritizes streamlined data pre-processing without compromising on accuracy. Leveraging deep convolutional neural networks (CNNs), we propose a robust classification methodology capable of handling diverse traffic sign scenarios encountered on roadways. Central to our study is the utilization of the German Traffic Sign Recognition Benchmark (GTSRB) [16-18] dataset, renowned for its comprehensive coverage and reliability. By training our model on this dataset, we aim to demonstrate its efficacy in real-world applications, particularly in advanced driver assistance systems (ADAS) and autonomous driving solutions. Our experimental results showcase the effectiveness of the proposed approach, achieving a competitive performance of 97.29% accuracy on the GTSRB dataset [23]. This outcome underscores the potential of our system to enhance road safety by providing accurate and timely recognition of German traffic signs, thereby facilitating smoother navigation and decision-making in autonomous vehicles. In conclusion, our research contributes to the ongoing efforts aimed at advancing the capabilities of autonomous driving systems, with a particular focus on improving object classification accuracy in the context of German traffic sign recognition.**

***Index Terms* — Advanced Driver Assistance Systems (ADAS), Autonomous Driving, Computer Vision, Convolutional Neural Networks (CNN), Deep Learning, Edge Computing, Image Processing, Mobile Devices, Object Detection, Real-Time Systems, TensorFlow, Traffic Sign Recognition, Urban Navigation**

I. INTRODUCTION

In recent years, advancements in deep learning have led to remarkable breakthroughs in computer vision applications. One fascinating area is real-time object detection, which finds applications in augmented reality and intelligent surveillance systems. This research aims to create and implement an effective solution for real-time object detection, specifically tailored for common mobile devices like laptops and smartphones. Our focus is on utilizing TensorFlow and Keras, a mobile and edge deployment solution from TensorFlow and Keras, for efficient and lightweight deployment of deep learning models.

The primary goal of this research is to develop a neural network capable of tracking, locating, and identifying objects or items in images, videos, or through a live webcam feed. TensorFlow and Keras, optimized for mobile and edge devices, appears promising for deploying such models in real-world scenarios with resource constraints. This research paper aims to provide a comprehensive overview of the entire process, covering aspects such as dataset creation, model architecture selection, training methodology, model conversion for TensorFlow and Keras, and the live execution of the model on local devices.

Through a live demonstration, we intend to showcase the practical implementation of the developed model, emphasizing its real-time capabilities and potential applications. The paper delves into the technical intricacies of the project while offering insights into the broader significance of real-time object detection on mobile devices. As we navigate through the research, we will unravel the challenges encountered, solutions devised, and the implications for future developments in the field of computer vision on resource-constrained platforms.

This research signifies a significant step forward in making state-of-the-art computer vision technology accessible for everyday applications, with a specific focus on the seamless integration of real-time object detection with mobile computing. As we elucidate the layers of our methodology, we invite the reader to delve into the intricacies of deploying deep learning models in real-time scenarios, contributing to the growing landscape of intelligent and portable systems.

Used version for this projects are: autokeras==1.0.20, keras==2.10.0, matplotlib==3.5.2, numpy==1.21.5, opencv_python==4.6.0.66, pandas==1.4.4, scikit_learn==1.1.3

II. RESEARCH AREA

This research is dedicated to developing an advanced pipeline for real-time object detection on common mobile devices, specifically leveraging Convolutional Neural Networks (CNN) and deep learning techniques. The primary objective of this project is to deploy a neural network capable of seamlessly detecting, localizing, and identifying objects in real-time scenarios, with applications extending to domains such as autonomous driving and urban navigation.

At the heart of this research is the strategic use of CNN, a specialized neural network architecture renowned for its effectiveness in image processing tasks. Deep learning methodologies are intricately incorporated into the project, aiming to leverage the necessary depth and complexity for training models with efficient object recognition capabilities. The integration of TensorFlow and Keras, a mobile and edge deployment solution, further optimizes the deployment of the CNN-based model on mobile platforms, ensuring efficient real-time performance despite inherent resource constraints.

The significance of this research area becomes evident in its capacity to address challenges associated with deploying sophisticated computer vision models on everyday mobile devices. The outcomes hold promise for contributing valuable insights into the intricate relationship between CNN, deep learning, and real-time object detection, pushing the boundaries of intelligent systems.

Applications of this research extend across various domains, with notable relevance in the context of autonomous driving.

Real-time object detection is crucial for ensuring safety and responsiveness in dynamic environments. By navigating through the complexities of dataset creation, model architecture, training processes, and live execution demonstrations, this research area aims to shed light on the path toward advancing the capabilities of intelligent systems in practical, real-world applications.

Development of data analysis, data preprocessing, and data augmentation methods for Traffic signs recognition pipeline. The next issue is to identify and apply a suitable deep learning methodology that can extract features from images for classification after the availability of labelled data for the selected use case. Nevertheless, there have been numerous suggestions for how to solve this issue. Following a thorough analysis of each technique, the one that best meets our requirements is chosen and put into practice for the specified use case. This leads to the project's second and primary research goal. This evaluation can aid in anticipating the need for additional approaches for the use case. The final research objective includes this requirement.

III. RELATED WORK

Object detection is a crucial aspect of computer vision, involving the identification and location of objects in images or videos. Various studies or experiments have been conducted in the past exploring the field of object detection in computer vision.

Firstly, the paper [26] by Joseph Redmon et al. is a significant milestone. This work introduces the YOLO algorithm, a real-time object detection system known for its outstanding accuracy and speed. YOLO stands out by using a single neural network for directly predicting bounding boxes and class probabilities, simplifying the process.

Next, a study done by Shaoqing Ren et al. introduces the Faster R-CNN algorithm [27]. This two-stage detection system includes a region proposal network (RPN) for generating object proposals, along with a Fast R-CNN network for subsequent classification and refinement. The innovative approach presented in this paper has significantly enhanced the real-time capabilities of object detection systems.

The research landscape unfolds further with the research done by Wei Liu et al. [28] This paper introduces the MultiBox Detector, a single-shot system that employs a neural network to directly predict object locations and class probabilities from full images in a single evaluation. The efficiency and simplicity of the MultiBox Detector make it a notable addition to object detection methodologies.

The paper by Andrew G. Howard et al, introduces the MobileNet architecture [29], a lightweight convolutional neural network meticulously designed for mobile and embedded vision applications. The optimization for low-latency and low-power consumption positions MobileNet as an ideal candidate for real-time object detection on resource-constrained mobile devices.

The paper by by Mingxing Tan, Ruoming Pang, and Quoc V. Le, proposes the EfficientDet algorithm [30], which is a family of object detection models that are both accurate and efficient. EfficientDet uses a compound scaling method to balance the trade-off between accuracy and efficiency, and achieves state-of-the-art performance on several object detection benchmarks Moving further, Xiaoyu Yue, Jiashi Feng, and Shuicheng Yan did a survey of object detection research over the past 20 years, covering traditional methods as well as deep learning-based methods. The paper also discusses the challenges and future directions of object detection research.

In summarizing these research findings, we establish a strong foundation for a comprehensive literature review on object detection, exploring the different studies done in order to enhance the task of object detection with the help of various technologies and algorithms.

IV. METHODS

This study employs a dual-methodology approach to investigate the effectiveness of Convolutional Neural Networks (CNNs). Qualitatively, the focus is on unraveling the interpretability and decision-making processes of the CNN model by examining intricate patterns and features. On the quantitative front, the methodology involves statistical analyses, meticulous data pre-processing, and augmentation techniques. The implementation incorporates code snippets and algorithms, as applicable, to ensure transparency and reproducibility. A key emphasis is placed on the training phase of the neural network architecture, optimizing model parameters for improved performance. This combined methodology offers a comprehensive exploration of CNN capabilities, integrating both qualitative insights and quantitative rigor.

1. Qualitative

a) Deep Learning

The qualitative methodology adopts deep learning, a subset of artificial intelligence. Deep learning employs artificial neural networks (ANNs) with multiple layers, mimicking the human brain's neurons [9]. The basic unit, an artificial neuron (Fig. 1), processes inputs through weighted connections, biases, and activation functions.

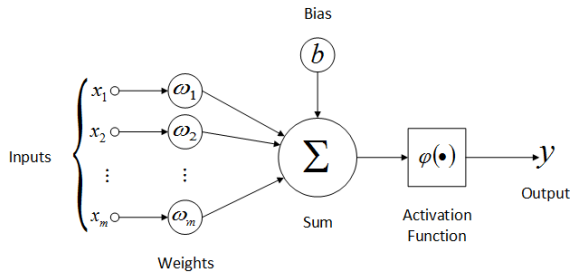

Fig. 1 Mathematical model of neuron [9]

The architecture extends to multi-layer perceptrons [9] Figure (2), and learning involves forward and backpropagation. Forward propagation yields predictions, and the error is quantified using a loss function. Backpropagation updates weights and biases iteratively through an optimization process, guided by the learning rate (η).

Balar Nancy – 13928122, Ekta Balhotra - 1379109

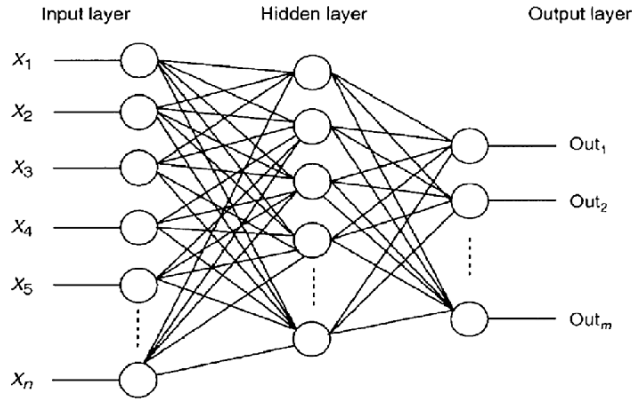

Fig. 2 Schematic View of multi-layer perceptron [9]

This deep dive into deep learning fundamentals lays the groundwork for the subsequent discussion on the tailored baseline architecture for image-related tasks in the research.

b) Convolutional Neural Network (CNN)

The methodology for this project on traffic sign recognition centers around the utilization of Convolutional Neural Networks (CNNs), also known as ConvNets. CNNs are multilayer neural networks designed for spatial architectures [9], particularly effective in computer vision tasks. The architecture comprises three key layers:

Convolutional Layer: The main objective of convolutional layers is to extract features from the image as depicts in the Figure (3), when filter f *f (f = 4) is convolved over image n*n (n = 6), it performs element wise multiplication followed by addition. This filter will slide over the image with same output size as the input size. In this case, zero-padding rows and columns are added on each side of the image (referred to as the same padding). It is general practice to choose odd numbered filter sizes to have symmetric padding.

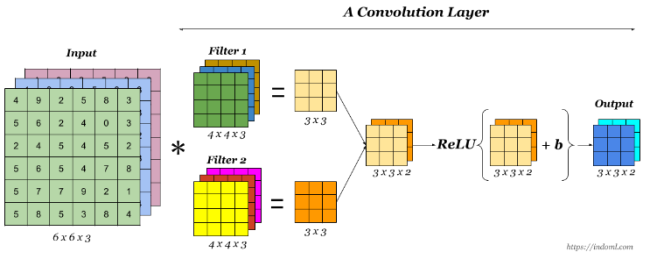

Fig. 3 Convolutional layer [9]

Pooling Layer: The pooling layer is in charge of reducing the convolution feature map's spatial size, which quickens computations and expands the receptive field for subsequent layers. The feature map is to be down sampled, which is the main goal of the pooling operation. In essence, Max-pooling and Average pooling are the two most common types of pooling processes. The input feature map is separated into sub squares, as seen in Figure (4), and based on pooling, either the maximum value or the average value is returned. The majority of CNNs employ max-pooling procedures as opposed to average pooling. While there are no parameters to be learned from the pooling layers, the values of the filters are the weights of CNNs.

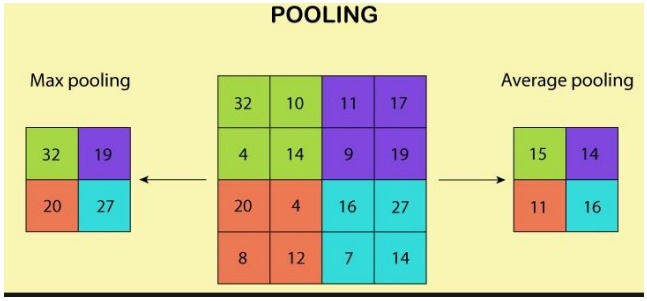

Fig. 4 Pooling layer [9]

Fully Connected Layer: Similar to a traditional multi-layer neural network, it receives flattened outputs from the last pooling layer. It facilitates classification by providing probabilities for each class, often using softmax activation [9].

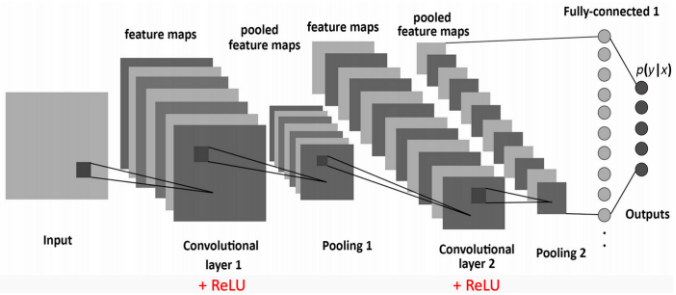

Fig. 5 Fully Connected Layer [9]

CNNs excel in feature learning for computer vision, generating invariant features through convolutional layers and subsequent down-sampling in pooling layers. The final features are occlusion-invariant and progressively abstract, capturing elements like edges in early layers and more complex features in later layers. In this project, CNNs are applied to extract crucial information from traffic sign images, reducing size for subsequent fully connected layers.

2. Quantitative

Deep learning systems require effective data pre-processing and augmentation to enhance model performance. Pre-processing involves structuring unstructured data for better model comprehension, such as resizing data to fit the input layer and normalizing or removing noise. Accurate pre-processing can expedite model training and inference. For image datasets, data augmentation is crucial to generate additional training instances, a popular method being used in computer vision problems.

Data Pre-Processing: Our dataset [23] includes actual imagers with highlighted backgrounds, which adds more features that are unnecessary for training our network and make classification more challenging. In order to facilitate the classification of these images by our network, we can preprocess them in two stages. First, colorful images are turned into grayscale versions. There are two key reasons why this grayscale conversion is significant. The first is that color is not a particularly important aspect to look for when differentiating distinct traffic signs because the lighting on our photographs fluctuates and many of these signals have similar colors and the edges, curves, and internal shapes of the signs are crucial

aspects for our classification task, thus that is what the network should concentrate on. As a result, when we convert these images, we decrease the color depth from 3 to 1. Because the input shape will only have a depth of 1 channel, our network will need less data and less processing time to categorize our data.

Data Augmentation: Data augmentation involves creating new images from existing ones with minor edits. In our case, we apply augmentation to expand the dataset [23]: Rotation (10 Degrees): Images are rotated by 10 degrees to avoid inaccurate representations. Figure (6) depicts rotated images.

Fig. 6 Images after 10 degree rotation

More significant rotations could distort traffic signs. Sample data augmentation involves generating 50000 augmented images in batches of 500 during training iterations. Techniques include horizontal and vertical translation, shear, and zoom, creating diverse training instances.

By incorporating these pre-processing and augmentation techniques, our deep learning model is better equipped to handle real-world variations in input data, ultimately improving its performance in classifying traffic signs.

3. Training the Neural Network Architecture

TensorFlow and Keras frameworks are used for training and evaluation of the CNN model. A complete open-source machine learning platform called TensorFlow has a vast, adaptable ecosystem of tools, libraries, and community resources. Keras serves as an interface for the TensorFlow library and reduces the number of user actions needed for common use cases.

First Lenet [19] CNN architecture is used for German traffic signs recognition but because of its poor performance (around 90% accuracy) we did tunning of its structural parameters as well as hyperparameters. A deep learning CNN model that is considered to be a part of this study is shown in Table (1) four convolutional layers and two fully connected layers make up the CNN. After the convolutional layer, max pooling is used to produce an abstracted form of the representation and prevent over-fitting. Each step of the small-scale CNN employs a rectified linear (ReLU) activation function. The softmax probabilities for each of the 43 classes of traffic signs are provided in the top layer. There are 378,023 total trainable parameters in our CNN model.

Keras applications API is used for construction of the encoder network. First, The CNN network is loaded after fully connected layer is added. After making changes to accommodate the data processing explained in above section, the model is trained on CPU. Adam optimizer is used for training with an initial learning rate of 0.001. The hyperparameters are tuned by evaluating the performance of the network on validation data set with different settings.

Table 1: CNN architecture

Layer (type)	Output Shape	Parameters
conv2d_4 (Conv2D)	(None, 28, 28, 60)	1560
conv2d_5 (Conv2D)	(None, 24, 24, 60)	90060
max_pooling2d_2 (MaxPooling 2D)	(None, 12, 12, 60)	0
conv2d_6 (Conv2D)	(None, 10, 10, 30)	16230
conv2d_7 (Conv2D)	(None, 8, 8, 30)	8130
max_pooling2d_3 (MaxPooling 2D)	(None, 4, 4, 30)	0
flatten_1 (Flatten)	(None, 480)	0
dense_2 (Dense)	(None, 500)	240500
dropout_1 (Dropout)	(None, 500)	0
dense_3 (Dense)	(None, 43)	21543

Table 2 : CNN parameters

Total parameters: 378,023
Trainable parameters: 378,023
Non-trainable parameters: 0

Categorical cross entropy was used in the suggested CNN model. The fact that only one solution to our classification problem can be correct is the main motivation for using categorical cross entropy. The model is trained using the following loss function.

$$\text{Loss} = -\sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i \quad \textbf{(1)}$$

Here, $\hat{y}$ denotes the predicted value. The distributions of the predictions and the actual distribution will be compared using categorical cross entropy. Likelihood of the true class is 1 in this case, while probability of the other classes is 0.VI.

V. RESULTS AND EVALUATION

The network is trained for 10 epochs and training is stopped as there was no further improvement in validation loss. Figure (7) and (8) displays the training and validation loss plots with respect to epochs of LeNet architecture [19] model and fine tunned model. It is evident from the graph that both the training and validation losses are decreasing with epochs. This shows a good convergence behavior of the fine-tuned model shown in Figure (8). The evaluation and test results of this network are shown and discussed in following:

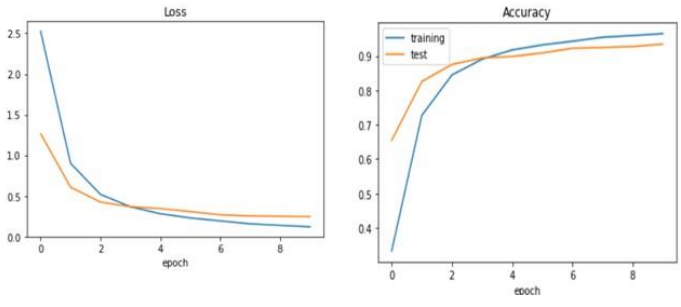

Fig. 7 Loss and accuracy plot of LeNet architecture

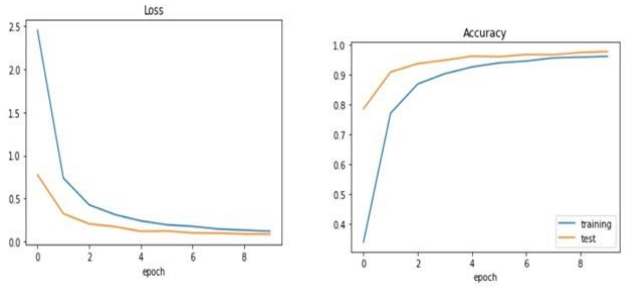

Fig. 8 Model loss and accuracy

In this experiment, after doing fine tuning of model by changing learning rate, adding CNN and dropout layers, training accuracy of 96.53% is accomplished with a loss reduction of 0.1210. With a reduced loss of 0.0867, testing accuracy of 97.60% was obtained. The dataset's [23] testing accuracy score is 95.13%. Similar to our CNN design employed in this study, the LeNet architecture [19] has seven layers. Both architectures use inputs that are 32*32 in size. 90% testing accuracy is attained utilizing the LeNet architecture [19] and a 32*32 input size.

After that we have tried k fold cross validation for evaluation of our model. In applied deep learning, cross-validation is generally used to evaluate a deep learning model's performance on untrained data. That is, to use a small sample to assess how the model will generally perform when used to provide predictions on data that was not utilized during the model's training. Table 3 depicts the results in form of loss and accuracy of our model. Here we can see that average accuracy is 97.29 % and loss is 0.095 respectively.

Table 3 : Loss and accuracy of each fold

Folds	Loss	Accuracy
1	0.1106	96.91
2	0.0934	97.32
3	0.09651	97.55
4	0.0794	97.76
5	0.06934	98.18
6	0.08286	97.68
7	0.1435	95.37
8	0.0992	97.20
9	0.0883	97.89
10	0.0942	97.12

Average scores for all folds:

Accuracy: 92.29932(+-0.7408159)

Loss: 0.0957514

VI. DISCUSSION AND CONCLUSION

Referring back to the objectives laid out in Abstract, we can derive the following conclusions:

- In driverless cars and autonomous cars, it can be difficult to recognize traffic signs. The project research work focuses on a review of recent work for traffic sign recognition. The ADAS-based system uses the GTSRB dataset [23] in conjunction with the suggested CNN model. The accuracy levels attained for testing, training, and validation are 95.13%, 97.29%, and 98.60%, respectively.
- When compared to the LeNet [19] deep learning model, this finding significantly improves testing accuracy. Additionally, the training process for our CNN model only requires about 10 seconds per epoch. Due to its minimal computation time and optimal computing resources, this model can be employed for a variety of real-time applications.

VII. FUTURE WORK AND LIMITATIONS

Future study could look into ways to enhance performance in low-light environments, such as dawn, dusk, twilight, or evening, as current work focuses on traffic sign recognition under typical lighting conditions. This would include gathering more information in these circumstances and creating cutting-edge algorithms designed to manage difficult lighting situations.

The speed of the ego vehicle, which can affect the relative motion between the car and traffic signs when driving, is currently not taken into consideration by the classification method. Subsequent research attempts may investigate techniques for integrating vehicle speed ego into the classification procedure, so enhancing the precision of traffic sign identification and detection.

VIII. REFERENCES

[1] Islam, K. T., & Raj, R. G. (2017). Real-time (vision-based) road sign recognition using an artificial neural network. Sensors, 17(4), 853.
[2] Haloi, M. (2015). Traffic sign classification using deep inception based convolutional networks. arXiv preprint arXiv:1511.02992.
[3] Tabernik, D., & Skočaj, D. (2019). Deep learning for large-scale traffic-sign detection and recognition. IEEE transactions on intelligent transportation systems, 21(4), 1427-1440.
[4] Ouyang, W., Wang, X., Zeng, X., Qiu, S., Luo, P., Tian, Y., ... & Tang, X. (2015). Deepid-net: Deformable deep convolutional neural networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2403-2412).
[5] https://www.quantamagazine.org/how-computationally-complex-is-a-single-neuron 20210902/
[6] Kumar, A. D. (2018). Novel deep learning model for traffic sign detection using capsule networks. arXiv preprint arXiv:1805.04424.
[7] Hinton, G. E., Sabour, S., & Frosst, N. (2018, February). Matrix capsules with EM routing. In International conference on learning representations.
[8] Behrendt, Karsten, Libor Novak, and Rami Botros. "*A deep learning approach to traffic lights: Detection, tracking, and classification*." 2017 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2017.
[9] Aggarwal, C. C. (2018). Neural networks and deep learning. Springer, 10, 978-3.
[10] Iyer, B., Patil, N. IoT enabled tracking and monitoring sensor for military applications. Int J Syst Assur Eng Manag 9, 1294–1301 (2018). https://doi.org/10.1007/s13198-018-0727-8
[11] Dalal, Navneet, and Bill Triggs. "*Histograms of oriented gradients for human detection.*" 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). Vol. 1. IEEE, 2005.

[12] Dibua, Ohiremen, Aman Sinha, and John Subosits. "*Data-Driven Modeling and Control of an Autonomous Race Car."*
[13] Bahlmann, Claus, et al. *"A system for traffic sign detection, tracking, and recognition using color, shape, and motion information."* IEEE Proceedings. Intelligent Vehicles Symposium, 2005. IEEE, 2005.
[14] T. Hibi, *"Vision based extraction and recognition of road sign region from natural color image, by using HSL and coordinates transformation,"* presented at 29th Inter. Symposium on Automotive Technology and Automation, Robotics, Motion and Machine Vision in the Automotive Industries, ISATA, 1996.
[15] Wu, Yiqiang, et al. *"Real-time traffic sign detection and classification towards real traffic scene."* Multimedia Tools and Applications (2020): 1-19. 19
[16] Shustanov, Alexander, and Pavel Yakimov. *"CNN design for real-time traffic sign recognition."* Procedia engineering 201 (2017): 718-725.
[17] Stallkamp, Johannes, et al. "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition." Neural networks 32 (2012): 323-332.
[18] Houben, Sebastian, et al. *"Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark."* The 2013 international joint conference on neural networks (IJCNN). IEEE, 2013.
[19] Kuo, C. C. J. (2016). Understanding convolutional neural networks with a mathematical model. Journal of Visual Communication and Image Representation, 41, 406-413
[20] Dillon, J. V., Langmore, I., Tran, D., Brevdo, E., Vasudevan, S., Moore, D., ... & Saurous, R. A. (2017). Tensorflow distributions. arXiv preprint arXiv:1711.10604.
[21] Ketkar, N. (2017). Introduction to keras. In Deep learning with Python (pp. 97-111). Apress, Berkeley, CA.
[22] Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2011, July). The German traffic sign recognition benchmark: a multi-class classification competition. In The 2011 international joint conference on neural networks (pp. 1453-1460). IEEE.
[23] Zoph, B., & Le, Q. V. (2016). Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578.
[24] Kuo, C. C. J. (2016). Understanding convolutional neural networks with a mathematical model. Journal of Visual Communication and Image Representation, 41, 406-413.
[25] https://arxiv.org/pdf/1506.02640.pdf
[26] https://arxiv.org/pdf/1506.01497.pdf
[27] https://arxiv.org/pdf/1512.02325.pdf
[28] https://arxiv.org/pdf/1801.04381.pdf
[29] https://arxiv.org/pdf/1911.09070.pdf
[30] https://arxiv.org/pdf/1905.05055.pdf