2021 International Conference on Recent Advances in Mathematics and Informatics (ICRAMI) | 978-1-6654-4171-1/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICRAMI52622.2021.9585952

An Analytical Study of Efficient CNNs Tuning and Scaling for Traffic Signs Recognition

Imene Bouderbal, Abdenour Amamra and Mohamed Akrem Benatia
Ecole Militaire Polytechnique
Bordj El-Bahri BP 17, Algiers, Algeria
Email: imene.bouderbal@yahoo.com,

***Abstract*—Deep learning-based traffic sign recognition has been a very active area of research in autonomous driving since the appearance of Convolutional Neural Networks (CNN) as a substitute for classical machine learning algorithms. However, a good traffic sign recognition system (TSR) should inclusively fulfill accuracy, and response time compromise to be palatable in self-driving applications. Besides, the considerable computational load remains a burden to the adaptation and the design of CNN architectures for real-time applications. This paper aims to investigate the relationship between accuracy, efficiency, and computational complexity for the classification of traffic signs. MobileNetV2 and EfficientNet architectures were evaluated as they are specifically designed to be computationally efficient. When most of the contributed work in the literature focuses on accuracy, we rather focus on the choice of the most efficient model (best accuracy/model complexity ratio). The results support the intuitive idea that performance remains proportional to network size up to a given level beyond which it saturates.**

***Index Terms*—Traffic Sign Recognition, CNNs, Model Complexity, Hyper-parameters Tuning, Autonomous and Assistive Driving.**

I. Introduction

Ground vehicles remain the primary means of transportation used today. Due to the increase in the number of vehicles, drivers are experiencing several road-traffic risks leading to an increasing number of accidents. The latter (accidents) are mainly due to drivers' inattention while driving. Traffic signs are a crucial element for road traffic regulation, whose role is to assist the driver by providing valuable information for safe driving. The signs inform the driver of speed regulations and give warnings in advance before striking hazards. Nevertheless, missing or damaged signs, bad weather conditions, or weak driver attention may contribute to the inability of the driver to see a critical traffic sign, and consequently to a mortal accident. Therefore, efficient traffic sign recognition is necessary for both semi-autonomous (assistive) and autonomous driving.

Computer vision techniques have been used for traffic sign recognition since the early 90s [1]. Traffic signs recognition systems can be divided into two classes: *detection* and *classification* sub-systems. The former is responsible for deciding whether the acquired camera frame (the camera is mounted on the front of the vehicle) contains a traffic sign; if so it proceeds to the localization of the region containing the visual clues of the sign. The classification or recognition stage is responsible for classifying the detected sign into one of several known categories. In this work, we are interested in the classification stage; particularly, in deep learning-based traffic sign recognition. We start by exploring two efficient CNN architectures, from which we carry on hyper-parameters tuning, scaling, and study of complexity impact on the classification task. The impact of data augmentation is also investigated in the present work.

The remainder of the paper is organized as follows. In Section II, we present a brief review of the existing methods used in traffic sign recognition and the publicly available traffic signs datasets. In Section III, we describe our technical methodology. The obtained results are shown and discussed in Section IV. Finally, we conclude the paper in Section V, where we summarize our contributions and give some future works.

II. Related works

A. Traffic Signs Recognition

Several methods have been used in both traffic sign detection and recognition. Traditionally hand-crafted features like the histogram of oriented gradients (HOG), scale-invariant feature transform (SIFT), local binary patterns (LBP) or integral channel features [2] were used. Later, a wide range of machine learning methods have been proposed, ranging from support vector machine (SVM), logistic regression (LR), random forests (RF), to artificial neural networks [3]. However, the former methods (SVM, LR, RF) suffer from a limited feature description capability and they are strongly dependent on expert's knowledge in feature engineering. Consequently, they cannot be discriminating classes as their number increases. To overcome this limitation deep neural networks were necessary.

Traffic signs recognition has benefited from the deep learning renaissance, owing to the recent progress in modern convolutional neural architectures and graphic processing units (GPU). Many classical object classification networks have been developed, ranging from LeNet [4] to MobileNet [5], ShuffleNet [6], and many others.

B. Efficient CNNs

Modern neural networks have become very deep and difficult to tune on insufficiently large datasets [7]. Besides,

designing efficient (small and performant) neural networks has become a new research focus for several domains particularly those requiring real-time performance. According to the literature, there are four methods to build an efficient network:

- *Model Compression*: In this class of techniques, the original model is modified in a few ways like :
  - Pruning: Parameters that do not help accuracy are removed systematically.
  - Quantization: Model parameters are often stored as 32-bit floating-point numbers but these values are usually not uniformly distributed over the 32-bit space. Model quantization techniques examine the distribution of parameters and store the 32-bit numbers in a smaller number of bits without much loss of precision.
- *Hand-crafting new networks*: Often smaller networks are designed from scratch. SqueezeNets [8], MobileNets [5, 9], and ShuffleNets [6] are examples of networks designed with a small architectures.
- *Network Search*: Another class of methods that is gaining popularity is an automated search for finding the best network depth, width, size of convolution kernels, etc. MnasNet [10] is an example of this approach.
- *Model Scaling* :
  - Simple Model Scaling: Model scaling is about scaling the existing model in terms of model depth or model width, and less popular input image resolution to improve the performance of the model. The model can either be scaled up (i.e. use larger parameters) or down (i.e. use fewer parameters).
  - Compound Model Scaling: Recently in [11] the authors proposed a new scaling method. This method suggests that instead of scaling only one model attribute such as depth, width, or resolution; strategically scaling all three of them together delivers better results. Also, the authors of [11], proposed neural architecture search, designed a new baseline network and scale it up using their proposed compound scaling method, and obtain a family of models called EfficientNets [11].

  These models achieve much better accuracy and efficiency than previous CNNs.

In this work, two efficient convolutional neural networks will be explored, namely MobileNetV2 [5] and EfficientNet [11]. The first is a small hand-crafted CNN, and the second is obtained using neural architecture search. Both offer the possibility of scaling as follows:

- MobileNetV2: width scaling;
- EfficientNet: compound scaling.

Thus, we aim to set the right hyperparameter values to define an upper limit to the computational load required for efficient traffic signs recognition.

C. Traffic Signs Recognition Datasets

Since a lot of parameters need to be learned, CNN-based methods require large training data and specialized computation hardware [1]. Public datasets provide benchmarks for comparison. The work in [12] provides a summary of these publicly available datasets.

In this study, the German Traffic Sign Recognition Benchmark (GTSRB) [13] is used for evaluation purposes. It includes a performance reference in the form of classification results from 32 test persons. The resulting human classification accuracy is 98.84% where only a few classifiers performing better than humans. In the original competition, [14] surpass human performance using a committee of CNNs with a classification accuracy of 99.46%. Later work by [15] further improves the results, with a classification accuracy of 99.71%.

III. Methodology

In this section, we describe :

1) the dataset used;
2) the preprocessing techniques and data augmentations;
3) the architectural specifications;
4) the training procedure.

A. Traffic Signs Dataset

GTSRB is a comprehensive dataset that has been widely used in training and testing different traffic sign recognition algorithms. The dataset includes 43 classes and with a total of 39209 training samples and 12630 test instances.

Fig. 1. Example image for each class of road sign.

B. Image data augmentation

After analyzing the dataset, we found that it is imbalanced. In order to mitigate overfitting, a new balanced dataset will be artificially augmented with basic image transformations (contrast, brightness, rotation, image sharpening, blurring, translation).

C. Preprocessing

In this section, we will describe the several preprocessing techniques applied to the images. Images were resized to a fixed size. An image size of 32×32 is used to better emulate the small size of traffic signs detection in real-life scenarios.

1) *Shuffling*: Since the images were taken progressively by the camera in each scene, learning data is mixed up for better generalization;

2) *Contrast Limited Adaptive Histogram Equalization (CLAHE)*: in order to enhance fine details within images, CLAHE [16] is also used as a preprocessing technique.
3) *Normalization*: Image data is be normalized so that it has a zero mean and unit variance.

D. Architecture details

MobileNet [5, 9] defines a class of CNNs that are parameterized by a parameter α that controls the network complexity. The complexity parameter is a scaling factor for the number of channels in each of the convolutional layers of the model. This architecture uses depthwise separable convolutions followed by pointwise convolution (normal convolution with kernel size 1×1), which can be seen as a factorized version of normal convolution [9](see figure 2. This reduces the number of parameters and computational load, compared to a normal convolution layer [9], at a little performance loss. This architecture uses Batch-Normalization after every convolutional layer to accelerate the training process. Unlike MobileNetV1, MobileNetV2 uses a different building block containing a linear layer and a skip connection (figure 3).

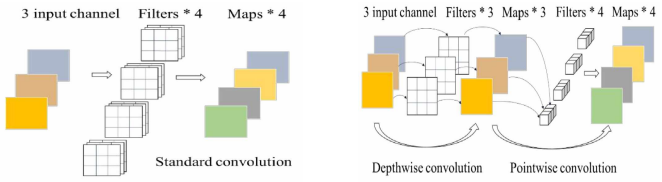

Fig. 2. Illustration of : (left) a standard convolution and (right) a depthwise followed by a pointwise convolution.

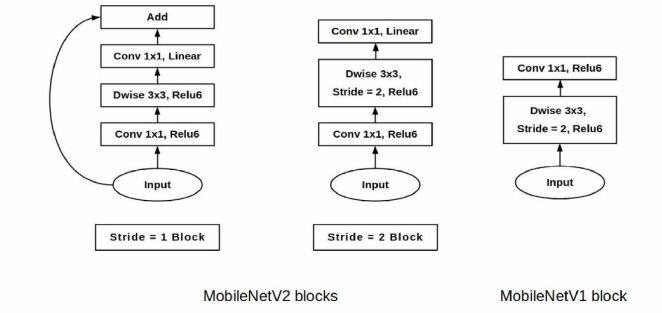

Fig. 3. Convolutional building blocks of MobileNetV1 and MobileNetV2.

EfficientNet base network was obtained by doing a Neural Architecture Search (NAS) that optimizes for both accuracy and frame rate [11]. The main building block of this network consists of MBConv to which squeeze-and-excitation optimization is added. MBConv is similar to the inverted residual blocks used in MobileNetV2. Hence, yielding a shortcut connection between the beginning and end of a convolutional block. The input activation maps are first expanded using 1×1 convolutions to increase the depth of the feature maps. This is followed by 3×3 Depth-wise convolutions and Pointwise convolutions that reduce the number of channels in the output feature map. The shortcut connections connect the narrow layers whilst the wider layers are present between the skip connections. Such a structure helps in decreasing the overall number of operations required as well as the model size. Figure 4 illustrates the difference between the compound scaling method used with the EfficientNet family of models and simple model scaling.

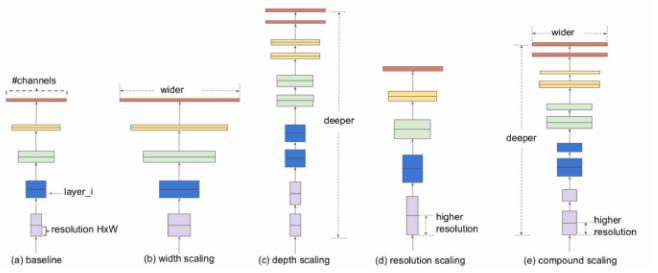

Fig. 4. Compound Model Scaling [11]. (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is the proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

E. Training procedure

1) MobileNetV2 :
   - the complexity parameter α of the network architecture is varied. The following values for α are investigated: $0.2, 0.5, 1.0, 1.5, 2.0$;
   - MobileNetV2 based models are trained using the SGD optimizer with learning rate = 0.001 and momentum = 0.9, since it results in faster convergence and slightly better results on the validation set.
2) EfficientNets : the models ranging from B0 to B7 are used.
   - After hyper-parameters fine tuning, the EfficientNet models are trained using ADAM algorithm with $learningrate = 0.0001$.

- All models are trained for 50 epochs, with no early stopping to ensure a fair test comparison between their results;
- For all models, the input images have the fixed shape of $32 \times 32 \times 3$, since we are interested in investigating network size and complexity on the classification task, rather than image resolution;
- During training, the initial learning rate is reduced dynamically based on a monitored validation metric using a "reduce-on-plateau" strategy;

IV. Results and Discussion

The preliminary results obtained after hyper-parameters tuning of the two selected models for this study show that EfficientNet baseline model (B0) performs better than the baseline MobileNetV2 model. It's worth mentioning that EfficientNetB0 is twice as big as MobileNetV2. Based on the results in tables I and II, we conclude that the performance of

models can be improved by scaling at the expanse of computational load. Also, the compound scaling method seems to give better results than using simple scaling with MobileNetV2.

Based on the results in Figure 7, the error rate can be reduced quite significantly from 31.54% to 8.6% by increasing the complexity for the MobileNetV2 models. This relative error rate reduction of 22.94% comes at the cost of 8.6 million more parameters. For the EfficientNet family of models, the scaling method permitted to reduce the error rate from 5.22% to 1.85% and this by adding almost 40 million more parameters for a reduction of only 3.37%. In general, the error rate drops when increasing the complexity of the network, however, except EfficientNetB3. A possible explanation for such a performance drop could be that hyper-parameters tuned for a baseline model don't match those of a scaled model, and thus each resulting model should have its hyper-parameters tuned separately.

In Figure 7, the best performing network is EfficientNetB7 with an error rate of 1.79%. This is almost on par with the average human performance of 1.16% [13]. It is, however, not on par with the best performing CNN on the GTSRB dataset with an error rate of 0.29% [15]. For comparison, the CNN in [15] uses 14.6 million trainable parameters, whereas the best model in this study is at 65.1 million trainable parameters. Thus, scaling models for better performance isn't always a good answer. For the MobileNetV2 models, even though the scaling parameter α increased, the performance of the model seems to saturate (table II).

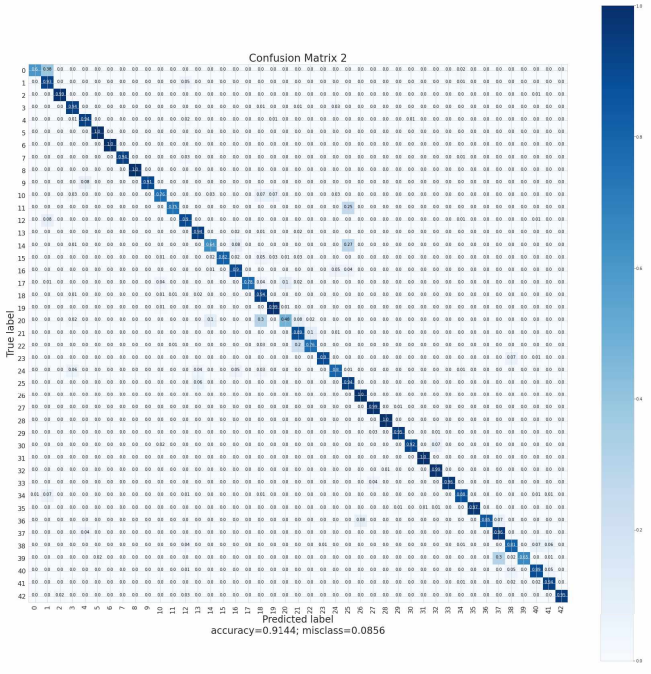

Fig. 5. Confusion matrix for MobileNetV2 scaled model with α=2

Figure 5 show the confusion matrix for the scaled MobileNetV2 model. It can be seen that even with 19 million parameters (for α = 3) the model undergoes miss classifications for several classes. After inspection of the miss-classified samples, some erroneous cases were identified to be due to low resolution, blur, overexposure, occlusion, and clutter.

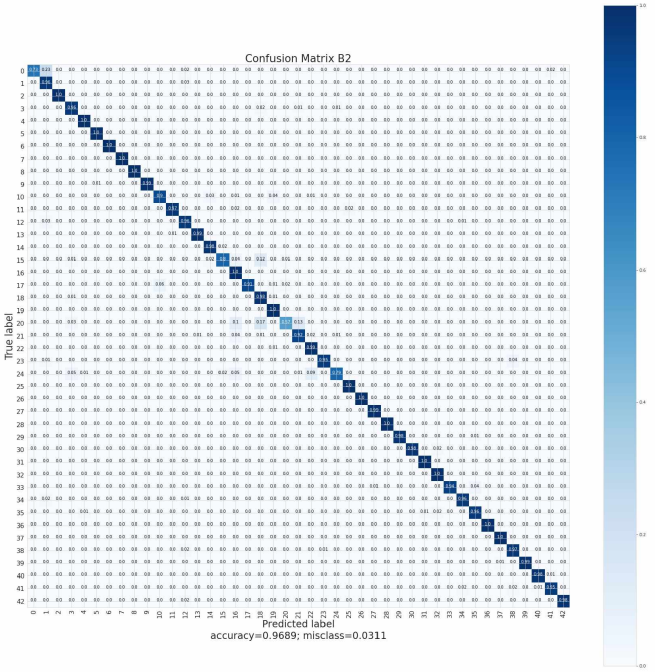

Fig. 6. Confusion matrix for EfficientNetB2 model

For EffiecientNet models, and by inspecting the diagonal of the different confusion matrices, we found that most of the classes have high class-specific accuracy, but that there are a few classes for which the accuracy is significantly worse (less than 95%). However, the error rates are acceptable for all models (in the range of [0.05,0.01]). Confusion matrix for EfficientNetB2 is illustrated in figure 6. The erroneous cases are due to the same reasons mentioned above

For approximately the same number of parameters, we have :

- EfficientNeB0 and MobileNetV2(α=1.5) : classification accuracy of 94.78% against 87.23% respectively;
- EfficientNeB2 and MobileNetV2(α=2) : classification accuracy of 96.86% against 91.40%respectively;
- EfficientNetB4 and MobileNetV2(α=3) : classification accuracy of 97.19% against 91.66% respectively.

EfficientNets models outperform MobileNets models possibly due to EfficientNet baseline model being searched rather than designed.

A. Discussion

- First, the preliminary results obtained from EfficientNetB0 and MobileNetV2 confirm that simple architectures achieve reasonable classification scores (Figure 8). Reason for which researchers are designing more efficient architectures to increase generalization performance.
- Second, regularization techniques such as dropout, batch normalization, etc, have been developed to extend deep

TABLE I
CLASSIFICATION RESULTS OF GTSRB [13] OBTAINED WITH EFFICIENTNETS [11].

Model	Number of parameters	Test Accuracy	Test Loss
EfficientNetB0	4,784,327	0.9478	0.2004
EfficientNetB1	7,309,995	0.9629	0.1410
EfficientNetB2	8,569,373	0.9686	0.1073
EfficientNetB3	11,650,387	0.9676	0.1331
EfficientNetB4	18,672,771	0.9719	0.1101
EfficientNetB5	29,644,571	0.9793	0.0836
EfficientNetB6	42,223,283	0.9815	0.0829
EfficientNetB7	65,492,923	0.9821	0.0731

TABLE II
CLASSIFICATION RESULTS OF GTSRB [13] OBTAINED WITH DIFFERENT VALUES OF α, FOR MOBILENETV2 [5]

Model	Value of α	Number of parameters	Test Accuracy	Test Loss
MobileNetV2α4	0.2	255,419	0.6846	1.0326
MobileNetV2α3	0.5	761,307	0.8128	0.6605
MobileNetV2 (original)	1	2,313,067	0.8721	0.4863
MobileNetV2α1	1.5	5,071,131	0.8723	0.5235
MobileNetV2α2	2	8,875,435	0.9140	0.3420
MobileNetV2α5	3	19,687,147	0.9166	0.3662

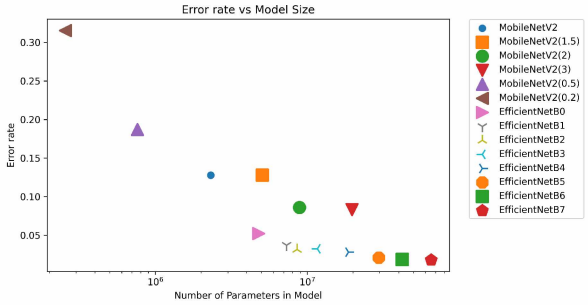

Fig. 7. Average classwise error rate vs model parameters for the evaluated networks

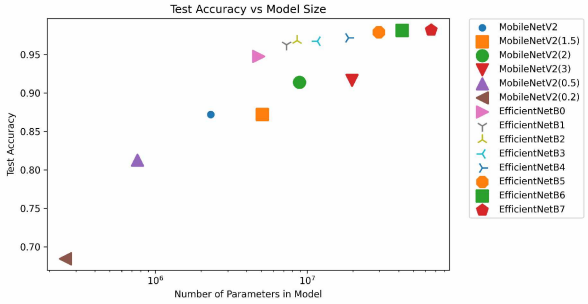

Fig. 8. Classification accuracy vs model parameters for the evaluated networks

neural networks' capability in dealing with overfitting and learning from small datasets.

- The results obtained in this study are following the findings in [17], which state that it is generally common that bigger datasets result in better deep neural architectures. Augmentation allowed us to improve the accuracy of the classifier by first balancing the dataset, and second augmenting it during training.
- Another approach that could be used to tackle the challenges related to large amount of training data is the use of synthetically generated images [18].
- Model scaling typically improves CNNs' performances, however at a high computational cost.
- The improvement obtained using model scaling generally saturates after a certain threshold (or becomes small but at a very high cost) (Figures 7 and 8).
- Even though a fixed image size of $32 \times 32 \times 3$ is used, both scaling methods used with the two baseline models improved classification accuracy and reduced error rate.
- Image resolution and quality are often overlooked in the design of deep neural networks. A previous work of ours [19] shows that CNNs are resilient to image compression given that its level is sufficient. So, designing architectures that exploit directly compressed data will lead to more efficiency.
- Carrying out a Neural Architecture Search (NAS) that optimizes for performance requirements and scale with hardware capabilities is more interesting to investigate than hand-crafting new CNN architectures.
- These results can be used as a decision aid for computational complexity and performance

V. CONCLUSION AND PERSPECTIVES

We presented a study on traffic sign classification using scalable efficient CNNs. The main goal was to determine and test the impact of the architecture, regularization techniques, and dataset augmentation on traffic sign classification. The obtained results aligned with the results of the literature as follows:

- Simple architectures permit to achieve reasonably good classification accuracy;
- Hyper-parameters tuning plays an important role in model performances;

- Regularization techniques and data augmentation contributed in improving classification performance;
- Model scaling increases performance at the cost of computational load.

As further work, we will conduct in-depth research on lightweight models and explore more efficient CNN implementations. We'll explore Neural Architecture Search and CNN scaling for embedded applications with limited resources. Finally, we'll investigate image classification and recognition techniques based on small sample datasets.

REFERENCES

[1] Abdul Mannan et al. "Classification of Degraded Traffic Signs Using Flexible Mixture Model and Transfer Learning". In: *IEEE Access* 7 (2019), pp. 148800–148813.

[2] Ayoub Ellahyani et al. "Traffic sign detection and recognition using features combination and random forests". In: *International Journal of Advanced Computer Science and Applications* 7.1 (2016), pp. 686–693.

[3] Zhiyong Huang et al. "An efficient method for traffic sign recognition based on extreme learning machine". In: *IEEE transactions on cybernetics* 47.4 (2016), pp. 920–933.

[4] Yann LeCun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[5] Mark Sandler et al. "Mobilenetv2: Inverted residuals and linear bottlenecks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 4510–4520.

[6] Ningning Ma et al. "Shufflenet v2: Practical guidelines for efficient cnn architecture design". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 116–131.

[7] Xie Bangquan and Weng Xiao Xiong. "Real-time embedded traffic sign recognition using efficient convolutional neural network". In: *IEEE Access* 7 (2019), pp. 53330–53346.

[8] Forrest N Iandola et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and¡ 0.5 MB model size". In: *arXiv preprint arXiv:1602.07360* (2016).

[9] Andrew G Howard et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications". In: *arXiv preprint arXiv:1704.04861* (2017).

[10] Mingxing Tan et al. "Mnasnet: Platform-aware neural architecture search for mobile". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 2820–2828.

[11] Mingxing Tan and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 6105–6114.

[12] Chunsheng Liu et al. "Machine vision based traffic sign detection methods: Review, analyses and perspectives". In: *IEEE Access* 7 (2019), pp. 86578–86596.

[13] Johannes Stallkamp et al. "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition". In: *Neural networks* 32 (2012), pp. 323–332.

[14] D Ciresan et al. "Multi-column deep neural network for traffic sign classification. Neural Networks". In: *The International Joint Conference on Neural Network, IDSIA-USI-SUPSI— Galleria*. Vol. 2. 2012.

[15] Álvaro Arcos-García, Juan A Alvarez-Garcia, and Luis M Soria-Morillo. "Deep neural network for traffic sign recognition systems: An analysis of spatial transformers and stochastic optimisation methods". In: *Neural Networks* 99 (2018), pp. 158–165.

[16] Ankit Vidyarthi et al. "Classification of Breast Microscopic Imaging using Hybrid CLAHE-CNN Deep Architecture". In: *2019 Twelfth International Conference on Contemporary Computing (IC3)*. IEEE. 2019, pp. 1–5.

[17] Chen Sun et al. "Revisiting unreasonable effectiveness of data in deep learning era". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 843–852.

[18] Oualid Araar et al. "Traffic Sign Recognition Using a Synthetic Data Training Approach". In: *International Journal on Artificial Intelligence Tools* 29.05 (2020), p. 2050013.

[19] Imene Bouderbal, Abdenour Amamra, and Mohamed Akrem Benatia. "How Would Image Down-Sampling and Compression Impact Object Detection in the Context of Self-driving Vehicles?" In: *CSA*. 2020, pp. 25–37.