Partially Connected Neural Networks for an Efficient Classification of Traffic Signs

Bousarhane Btissam, Bouzidi Driss
Smart Systems Laboratory (SSL),
National School of Computer Science and Systems Analysis ENSIAS,
Mohammed V University,
Rabat, Morocco
ibtissam_bousarhane, driss.bouzidi@um5.ac.ma

Abstract—**Road signs recognition plays an important role in improving traffic safety for both drivers and pedestrians. To ensure this recognition, many approaches are proposed by researchers. To overcome the limitations of the existing methods, Deep Learning approaches are used. This type of approaches achieves high recognition performances, and is also less sensitive to real world adverse conditions. However, they are in contrast very computationally expensive due essentially to three main factors, which are more precisely, the size of input images, the type of used layers, and the number of used parameters. From this perspective, the objective of this work is to adopt an approach that aims to reduce this computational complexity, in order to ensure a fast and efficient classification of traffic signs, especially for low and limited resources environments. The adopted approach reaches good classification accuracies, and that by using BTSCD dataset.**

I. Introduction

Traffic accidents cause over two deaths per minute, and each year over 1.3 million people lose their lives on roads. This situation is due to many factors, as for example the use of unsafe vehicles, the quality of roads' infrastructure, etc. However, human factors, especially drivers' behavior represents one of the most principal causes of traffic incidents [1]. These behaviors include speeding, inattention, distraction, etc.

Where comes the crucial role of road signs recognition in improving traffic safety, specifically for Advanced Driver Assistance Systems (ADAS) and autonomous vehicles. To fulfil this important role, a recognition system should ensure, at the same time, an accurate precision and also a very quick response time, in order to guarantee a real time recognition of traffic signs.

In effect, the notion of real time depends tightly on the type of used applications. Hence, the response time is defined according to the nature of each task. Unlike soft real time tasks [2], which tolerate a certain delay after the pre-defined deadline, traffic signs recognition is considered instead as a hard real time task, especially for self-driving cars.

By consequence, if this recognition is produced after a certain deadline, that will certainly cause a huge number of dramatic and catastrophic accidents, and will presents a real threat for both drivers and pedestrians as well.

Knowing that traffic signs recognition includes two major stages, which are the detection and the classification, the latency required for each of these two stages should not then exceed few milliseconds maximum per frame.

From this perspective, the objective of our work is to propose a fast and efficient approach for traffic signs classification. Hence, the rest of our paper is organized as follow: Section 2 presents some related works. The proposed approach is presented in Section 3. Section 4 presents an overview of the experimental results, while Section 5 discusses the obtained results. The conclusion of our paper is presented in the last section.

II. Related works

Road signs recognition plays an important role in improving traffic safety, and in saving lives on roads. For that, many approaches are proposed by researchers in order to ensure this recognition [3-5]. These approaches include color-based & shape-based approaches, Machine Learning, and also Deep Learning approaches.

However, many limitations and challenges still face their implementation, especially for color and shape-based approaches. These limitations are essentially related to their sensitivity to real world adverse conditions, as for example occlusions, illumination, weather changes, etc. Additionally, edge-based techniques are also very time consuming.

In contrast, Machine Learning approaches like Support Vector Machines (SVM) achieve high recognition performances, in comparison to classical ones, but they need however many classifiers. Furthermore, this type of approaches is not suitable for Big Data [6].

To overcome these difficulties, the Neural Networks and especially Deep Learning (DL) approaches are proposed and used by many researchers [7-8], because they need instead less classifiers and big datasets are used to train this type of models. In addition to that, they are less sensitive to real world adverse conditions, which represents another important advantage, especially for computer vision and objects recognition systems, including traffic signs detection and classification.

In this context, Extreme Learning Machine (ELM) and Kernel ELM are used by Huang, Yu, Gu & Liu [9]. The proposed approach includes two stages, the first one consists on extracting features from the images, and that by using

Histogram of Oriented Gradient variant (HOG). While in the second stage a single-hidden-layer classifier is used to train the model. The random feature mapping is used to connect input and hidden layers, while hidden and output layers are connected using trained weights. The adopted approach achieves high performances, in terms of accuracy, with 98.62%.

Saouli, Aroussi & Fakhri [10] propose a similar approach for traffic sign recognition based, instead, on combining multiple features, which are more precisely HOG, Gabor and Compound Local Binary Pattern (CLBP). After features 'extraction, they have used also ELM for the classification, which is much faster than SLFNs (Single Layer Feedforward Neural Networks), and needs less number of tuning parameters. The used classifier includes 58 nodes with Sigmoid activation function. The obtained results show that ELM outperforms SVM and KNN (k-Nearest Neighbors) in terms of accuracy (98.30%) and inference time.

In effect, this type of approaches is generally based on a manual extraction of features. For that, many researchers opt for DL approaches to ensure instead an automatic extraction of the features. However, although their high performances in terms of recognition accuracy, DL approaches are in contrast very computationally expensive [11].

From this perspective, Convolutional Neural Networks (CNNs) are used by Jurišić, Filkovic & Kalafatic [12]. The proposed model is based on the work of Ciregan, Meier & Schmidhuber [13], but the approach proposes instead a multi-scale architecture inspired by the work of Sermanet & LeCun [14]. An accuracy of 98.17% is achieved, and that by adding a fully connected layer after each convolutional one, and concatenates then their outputs to reach high performances.

In the same context, we find the work of Arcos, Álvarez & Soria [15]. The adopted approach is based on a single CNN with three Spatial Transformer modules. The objective of these modules is to remove background and geometric noises from features maps, which helps the model to be spatially invariant to input data, and reach a high recognition accuracy with 98.87%.

A small Convolutional Neural Network is also used by Li & Zeng [16]. The proposed model (MyNet) is based on an accurate extraction of features (TS-Module), and uses the global average layer instead of the fully connected one to create one-dimensional vectors of the features. In order to reduce the number of used parameters, TS-Module combines three convolutional filters, where 1*3 kernel and 3*1 kernel are connected by a 1*1 kernel to achieve a 3*3 kernel. The classification rate of the model is 98.1%.

For Zaibi, Ladgham & Sakly [17], they have adopted instead a lightweight model based on LeNet-5. The Enhanced LeNet-5 uses instead two successive convolution layers each time to extract high-level features from input images. Contrary to the traditional LeNet-5, the enhanced model includes only one fully connected layer before the output layer, and LeakyReLU is added after each convolution one. This activation function is a variation of the ReLU that allows backward propagation even with negative input values. A recognition rate of 98.37% is reached by the adopted approach.

Although the high performances of DL approaches, hardware optimization is required to ensure a real time detection and classification of traffic signs. While algorithmic optimization is highly needed for low resources environments, like autonomous vehicles, mobiles, etc.

From this context, the main objective of the proposed approach is to reduce the computational complexity of this type of networks, in order to ensure a fast and efficient classification of traffic signs for such type of environments, as presented in the next section.

III. ADOPTED APPROACH

In the field of computer vision, and especially for traffic signs recognition, the computational cost of DL architectures is tightly related to three main factors, which are more precisely: the size of input images, the type of used layers and the number of used parameters. These parameters include generally the size and the number of layers, nodes and filters.

In effect, these three factors have a huge impact on increasing the computational cost of DL models. From this perspective, the objective of our work is to propose a Deep Learning approach that could deal effectively with these three main problems, in order to ensure a fast and efficient classification of traffic signs.

A. Input images size

The first important factor that contributes to the computational complexity of DL models is the size of input images. In effect, we find that in this field of research the minimum size of used images is generally 32*32. Hence, the objective of the adopted approach is to reduce even more the size of these images, and that without altering their quality, and also without losing important and meaningful details.

For this purpose, we find that many filters are used in order to smooth and enhance the quality of images. Among these filters, we find for example the Max, Min, Average, Median, etc.

For the Max filter (Maximum), the objective of this filter is to replace each pixel with the lightest one on its surrounding, and that based on a kernel. Hence, the Max filter is useful for finding the brightest points in images [18].

Contrary to the Max Filter, the Min Filter (Minimum) consists instead on replacing the pixels' value with the darkest pixels in the sliding windows.

Concerning the Average Filter, it is a sort of a convolutional filter, that takes into consideration the surrounding pixels on the images. That kernel replaces each pixel value of an image matrix with the average value of its neighbors [19]. Hence, the role of Average or Mean filter is to eliminate the unrepresentative pixels' value of their surroundings.

Like the Mean Filter, the Median is also used for noise reduction (Fig.1). However, instead of replacing each pixel with the average of its neighbors, this filter consists on sorting into numerical order the pixels' values from the surrounding neighborhood, and replacing after that the pixel being considered with the middle pixel value.

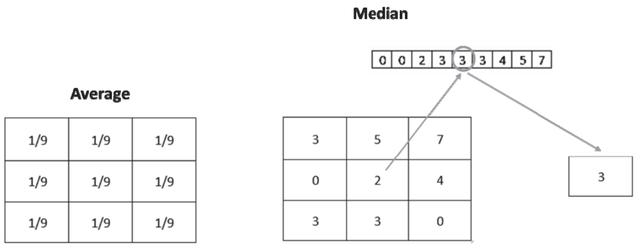

Fig.1. Average and Median filters using a 3*3 sliding window

The Median filter is efficient in removing noise, and it preserves at the same time the sharpness of an image's edges. We find that, this filter is better for noise reduction than the other filters, because it helps to preserve more details from images [20-21].

Although the fact that median filtering has a good noise reducing effect, however it is time consuming [22]. While, we find in contrast that average filtering preserves the main structures and details of images [19], and it is instead much faster and doesn't take longer to compute.

In addition to that, unlike other image processing techniques, for the Neural Networks the presence of some noise is very important to enhance the model performances and ameliorate its generalizability [23], especially during the training process.

For these specific reasons, and in order to accentuate the unique features of input images, we have added the Mean Filter as the input layer for our proposed model. Generally, convolutional and fully connected layers are usually used as input layers in Neural Networks. While in our approach we have added a Mean layer as the first layer. The objective of this layer is to reduce the size of input images, and that without deteriorating their quality.

To test the performances of the adopted approach, we have created four Neural Networks with deep and light architectures: one fully connected (FCN), two convolutional ones (CNN) and a Partially Connected Network (PCN). The obtained results are presented in Table I.

Adopting the proposed approach, we have added a mean layer to the FCN as an input layer, while changing each time the size of the filter. The used network (FCN) includes more than 5 million parameters. Table I shows that adding mean layer, with a filter size of (2*1), improves the accuracy from 90.91% to 91.94%, while it reduces the number of used parameters by almost 30%.

For the CNN-1, we find that applying a mean filter of (2*1), in the first layer, increases the performances of the model (93.37% to 94.44%), and it reduces also the parameters by more than 50%. Almost the same results are obtained for the CNN-2, where the accuracy is improved from 96.47% to 97.18%, while the parameters are reduced by almost 60%.

Finally, for the PCN model [24], instead of using fully connected layers after the convolutional ones, we have added 2 layers based on partial or local connections, which helps to improve enormously the obtained results with 98.33%.

Furthermore, Table I shows that by adding a mean layer to this PCN model the obtained performances have increased with 98.49%, and the used parameters have reduced by more than 67%.

TABLE I. ADDING MEAN KERNEL TO INPUT LAYERS IN NEURAL NETWORKS

Network	Architecture	First layer	Parameters	Accuracy
FCN	• 2 Fully 1024 & 2048	Fully	5 372 990	90.91%
	• Mean (1*2) • 2 Fully 1024 & 2048	Mean	3 800 126	91.74%
	• Mean (2*1) • 2 Fully 1024 & 2048	Mean	3 800 126	91.94%
	• Mean (2*2) • 2 Fully 1024 & 2048	Mean	3 013 694	91.62%
	• Mean (3*3) • 2 Fully 1024 & 2048	Mean	2 534 462	91.82%
CNN-1	• 2 Conv 60 & 120 (4*4) • 2 Fully 240 & 480	Conv	4 382 402	93.37%
	• Mean (2*1) • 2 Conv 60 & 120 (4*4) • 2 Fully 240 & 480	Mean	1 848 002	94.44%
CNN-2	• 2 Conv 20 (4*4) • 2 Fully 128 & 256	Conv	1 075 050	96.47%
	• Mean (2*1) • 2 Conv 20 (4*4) • 2 Fully 128 & 256	Mean	440 170	97.18%
PCN	• 2 Conv 20 (4*4) • 2 Partially 8 (3*2) & 7 (2*1)	Conv	72 430	98.33%
	• Mean (2*1) • 2 Conv 20 (4*4) • 2 Partially 8 (3*2) & 7 (2*1)	Mean	23 758	98.49%

From the obtained results, we can conclude that using a mean filtering of (2*1) as input layer improves the performances of Neural Networks, and reduces at the same time the number of used parameters, which helps to reduce enormously the computational cost of these networks.

To take into consideration more relationships between neighboring pixels, and to extract more unique features that represent the best the group, we have used an overlapping Mean filtering (Over-Mean) in our approach as input layer instead of using the average filtering.

The objective of this first layer (Over-Mean) is to enhance input images by reducing the changes that could affect their aspect in real world situations, as variation in illumination, intensity, contrast, etc.

Knowing that larger kernels are quite time consuming, and are more useful when more severe smoothing is required [25]. We have applied a sliding window of (2*1) to input images, which reduces enormously the required number of additions. Hence, on each iteration of the kernel, the value of the pixels that corresponds to the center of the window is changed. Then the value of each pixel is replaced with the average of this pixel and its neighboring pixels.

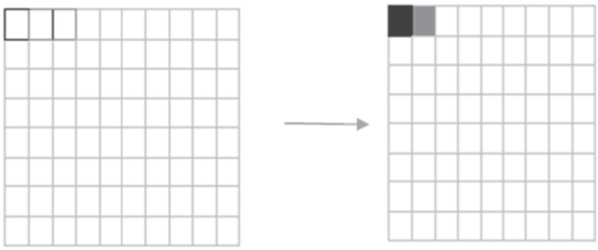

Fig.2. Overlapping Average with 2*1 sliding window

As presented in Fig.2, the Overlapping Mean filter calculates the sum of each two neighboring pixels, and then divides this sum by two, where the filtering or sliding window is moving one pixel at a time.

B. Type of layers

The second important factor that contributes to the computational complexity of Deep Learning approaches is the type of used layers. In effect, we find that CNNs present the best performances in terms of accuracy [26] for images classification, and that in comparison to other types of techniques, like Machine Learning [27], color-based and shape-based approaches.

However, convolutional layers are very time consuming. This computational complexity of convolutional layers comes from the huge number of multiplications needed to transform each input image. Where comes the need and the importance of reducing the size of these input images, as presented in the previous section.

For the convolutional layers, the three channels of RGB images are mixed together to get the output features maps. Which means that, the number of computed multiplications depends on the number, size and depth of each kernel (1).

$$f_l^k(p,q) = \sum_c \sum_{x,y} i_c(x,y)\, e_l^k(u,v) \qquad (1)$$

Where $i_c(x,y)$ is an element of the input image, $e_l^k(u,v)$ an element kernel of a layer and $f_l^k(p,q)$ represents an element of the feature map. The number of generated features maps or output channels depends also on the number of used filters.

To overcome the complexity of this type of layers, we have use instead three different types of layers in our approach. For the first layer (Sep-Conv), it consists on separating each channel of the input image (32x32 RGB image) before applying a convolution of shape 1x2x1 using 3 kernels (Fig.3).

In this stage, each 2x1x1 kernel iterates just one channel of the image, which generates a 31x32x1 one. The final output is then a 31x32x3 image. The three channels are then mixed and the number of output channels is multiplied using 10 kernels of 1x1x3 through the 31x32x3 image. By separating the channels, the complexity of the model as well as the training and the inference time are considerably reduced.

Concerning the second type of layers, we have used after the first convolution an Overlapping Mean filtering of 2x1 to the image, as discussed in the previous section, to get a transformed features map of 30x32x10.

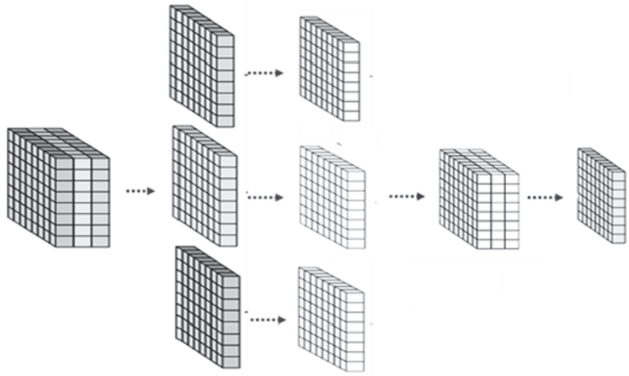

Fig.3. Applying a convolutional filter on each of the three channels

Additionally, a third type of layers is applied, which is more precisely partially connected layers (Part-1 & Part-2), as used in our previous model Mean-LC4 [28], presented in Fig.4. Contrary to convolutional ones, this type of layers is based on unshared weights, and apply instead different sets of kernels to each location in the images. Which helps to extract more representative features in a deeper way, and enhance even more the performances of the model.

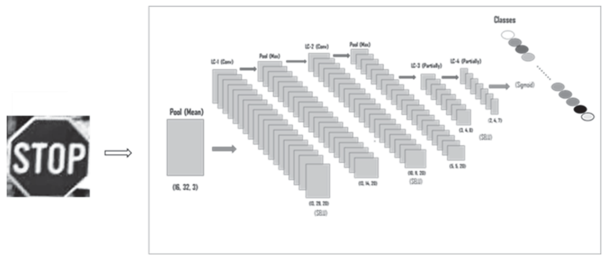

Fig.4. Mean-LC4 for traffic signs classification

However, knowing that unshared weights involve the increase of used parameters, we have added two convolutional layers (Conv-1 & Conv-2) based on sharing weights between the second and the third type of used layers. The objective of these two layers is to reduce the size of the generated features maps before applying the partially connected convolutions.

For our model Mean-LC4, it uses mean filtering as input layer. Hence, to test the performances of the adopted approach, we have added instead an Overlapping Mean layer to this model too. The obtained results are presented in the section of experimental results.

C. Number of parameters

Concerning the third important factor related to the depth, Table I shows that light architectures give better results, which means that increasing the number of used parameters it is not the best way to enhance the performances of Deep Learning approaches.

According to these results, our approach is based on using the minimum number of parameters needed to reach satisfactory results. These parameters include: the size and the number of filters, layers, etc.

Finding and choosing these parameters are based on our previous work [29], while Fig.5 presents the architecture of the proposed model (Mean-LC5), the type and the number of used layers, in addition to the size of the input and output of each of these layers.

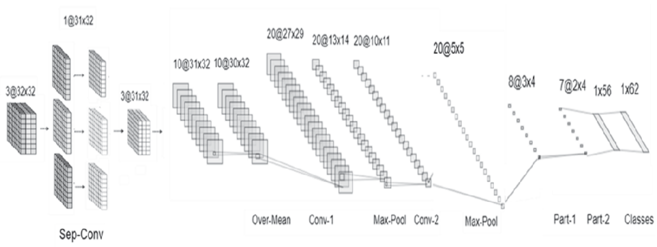

Fig.1. Mean-LC5 for traffic signs classification

The proposed model includes three types of layers with a minimal number of parameters: one separable, two convolutional, and two partially or locally connected. The separation of the three channels reduces enormously the complexity of the convolution operation, while the Overlapping Mean filtering helps to extract more representative features with a reduced dimension. Finally, more representative features are extracted in a deeper way by applying different sets of filters using the Partially Connected layers. The objective of these layers is to extract the features that represent the best the group, in order to ensure the performances of the model for unseen data.

IV. EXPERIMENTAL RESULTS

To evaluate the performances of the proposed approach, we have used two public datasets, which are more precisely CURE-TSR (Challenging Unreal and Real Environments for Traffic Sign Recognition) and Belgium Traffic Sign Classification Dataset (BTSCD).

For CURE-TSR, this database includes 14 classes of real-world and simulator images [30-32], and it covers many challenging conditions that vary from the least to the most severe ones. The used images are processed with state-of-the-art visual effect software to simulate challenging conditions. These challenges include decolorization, lens blur, codec error, darkening, dirty lens, exposure, gaussian blur, noise, rain, shadow, snow and haze.

Concerning BTSCD [33], it is a widely used dataset for the benchmark of traffic signs classification approaches. Additionally, the dataset contains a large variety of traffic signs types (62 classes) and samples, with more than 4500 and 2500 images for the training and the testing process respectively.

Hence, to train and test our models, we have used BTSCD and the dataset extracted from CURE-TSR used to evaluate and test our previous approaches [24], as shown in Table II. The objective of the extracted dataset from CURE-TSR is to study the impact of reducing, the minimum, the number of training data on the generalizability of the model for unseen data (testing samples).

TABLE II. TRAINING & TESTING DATASETS

Datasets	Classes	Training samples	Testing samples
Extracted dataset from CURE-TSR	11	1331	4039
Belgium dataset (BTSCD)	62	4575	2520

The extracted set from CURE-TSR includes 11 classes: no overtaking, no stopping, no parking, stop, no entry, speed limit, hump, no left, no right, priority to and yield as shown in Fig.6.

Fig.2. Types of traffic signs used for the training (CURE-TSR)

Fig.7 shows the number and the type of traffic signs in each of the 62 classes of BTSCD. While the balance between the different classes of the two datasets is presented in Fig.8.

Fig.3. Types of traffic signs in BTSCD

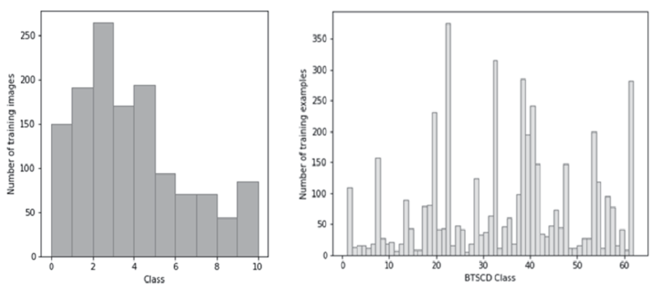

Fig.8. Balance between classes in CURE-TSR & BTSCD

To evaluate the performances of the two models (Mean-LC4 & Mean-LC5) to unseen data during the training process, we have divided the two datasets to a training (90%) and a validation set (10%). The validation set is created using cross validation, to ensure the randomness of the selected sets.

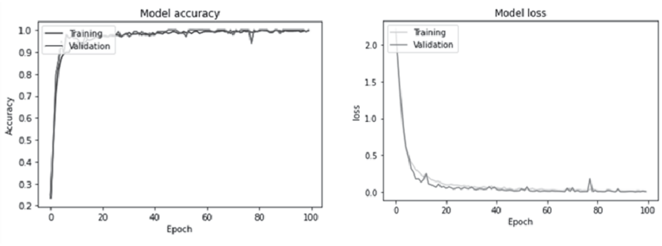

Fig.4. Training accuracy & loss for CURE-TSR (Mean-LC5)

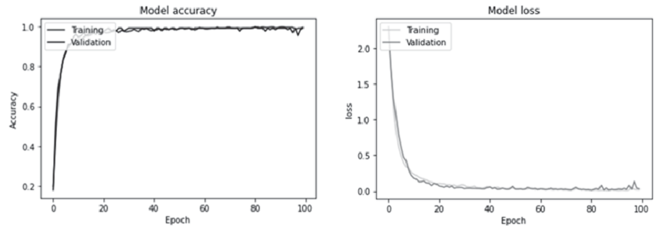

Fig.5. Training accuracy & loss for CURE-TSR (Mean-LC4)

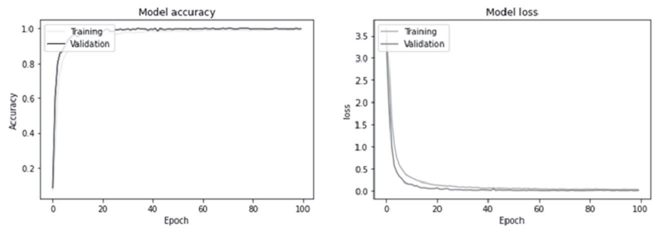

Fig.6. Training accuracy & loss for BTSCD (Mean-LC5)

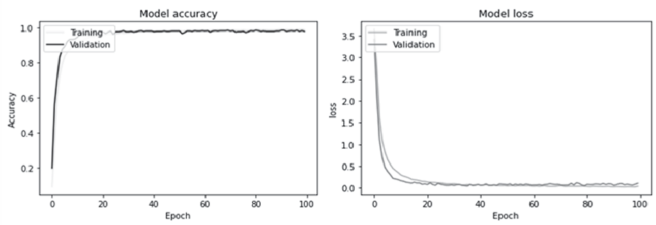

Fig.7. Training accuracy & loss for BTSCD (Mean-LC4)

From Fig.9, Fig.10, Fig.11 & Fig.12, we find that the adopted models converge very well for the training and the validation processes, and that using the two datasets.

For the testing process, we have used 4039 images extracted from CURE-TSR, and the 2520 images from BTSCD (Table II). Table III shows that the accuracy obtained by Mean-LC4 is 98.36% and 98.53% for CURE-TSR and BTSCD respectively. While the results obtained by Mean-LC5 are 98.09% for CURE-TSR, and 98.93% for BTSCD.

TABLE III. TESTING RESULTS

Model	**CURE-TSR**	**BTSCD**
Mean-LC4	98.36%	98.53%
Mean-LC5	98.09%	98.93%

V. DISCUSSION

As presented in the previous section, the adopted approach reaches high classification accuracies using the two public datasets. Additionally, it ensures also a fast training, validation and inference process. Table IV shows a comparison between the two adopted models and some existing real-time approaches for traffic signs classification. All the presented methods use BTSCD to evaluate their classification performances.

TABLE IV. COMPARISON BETWEEN OUR APPROACH AND SOME REAL-TIME CLASSIFICATION METHODS (BTSCD)

Reference	Approach	Accuracy	Time	Parameters	Configuration
Our 2nd approach	MEAN LC5	98.93%	0.2ms	25 244	2.40 GHz, i7-5500U CPU, 6GB RAM
[15]	Single CNN with 3 STNs	98.87%	4280ms	14 629 801	i7-6700k CPU, 16 GB RAM Nvidia GeForce GTX 1070 GPU
[9]	Kernel ELM	98.62%	1.46ms	-	i7-4790 CPU
Our 1st approach	MEAN LC4	98.53%	0.2ms	23 758	2.40 GHz, i7-5500U CPU, 6GB RAM
[9]	ELM	98.38%	1.42ms	-	i7-4790 CPU
[17]	Enhanced LeNet-5	98.37%	-	0.38 million	
[10]	ELM	98.30%	30ms	-	i5 microprocessor, 4 cores, 4GB RAM
[12]	OneCNN	98.17%	2ms	-	GPU, GeForce GTX 970 CUDA
[16]	MyNet	98.1%	705.10ms	-	i7-6700K CPU, 32GB RAM NVIDIA-GTX1070Ti GPU
[10]	SVM	97.15%	2220ms	-	i5 microprocessor, 4 cores, 4GB RAM
[10]	KNN	96.22%	2100ms	-	i5 microprocessor, 4 cores, 4GB RAM

From Table IV, we find that the approach adopted by Arcos, Álvarez & Soria [15] achieves a high accuracy of 98.87% using BTSCD. The inference time is 4.28s per image, and that by using i7-6700k CPU and Nvidia GeForce GTX 1070 discrete GPU. The number of used parameters is almost 14 million.

A high recognition accuracy is also achieved by using Extreme Learning Machine (ELM) and Kernel ELM, with 98.62% and 98.38% respectively [9]. The proposed approaches are computationally efficient with 1.42ms and 1.46ms per image respectively, using i7-4790 CPU.

The ELM used for the classification by Saouli, Aroussi & Fakhri [10] outperforms SVM and KNN in terms of accuracy by 98.30%, and an inference time of 0.03s. The approach uses an i5 microprocessor with 4 cores and 4 GB of RAM.

For the lightweight model based on LeNet-5 proposed by Zaibi, Ladgham & Sakly [17]. The network reaches an accuracy of 98.37%, using just 0.38 million parameters.

The network (OneCNN) proposed by Jurišić, Filkovic & Kalafatic [12] achieves a high recognition accuracy of 98.17%, while the inference time is about 2ms on average, using CUDA implementation of Caffe framework, on a GeForce GTX 970.

While the small convolutional neural network used by Li & Zeng [16] reaches a good classification accuracy of 98.1% with an inference time of 705.10ms, using i7-6700K CPU, 32GB RAM and NVIDIA-GTX1070Ti GPU.

In comparison to the presented methods (Table IV), we find that our approach is efficient in terms of classification performances, response time, hardware and memory requirements. The adopted approach achieves high recognition accuracies with 98.93% and 98.53% for the second and the first model respectively. In addition to that, the two models have also a very quick response time, that doesn't exceed 0.2 millisecond per image, and that using 2.40 GHz, i7-5500U CPU, 6GB RAM.

VII. Conclusion

Traffic signs recognition systems represent an important component in ADAS and self-driving cars. Their main goal is to ensure the safety of both drivers and pedestrians. To fulfil this crucial role many approaches are adopted by researchers, especially Deep Learning approaches.

However, although their high performances, DL approaches are very computationally expensive, which limits their use in real-time recognition applications.

From this perspective, our DL model (Mean-LC5) is proposed. The adopted model reaches high classification accuracy using CURE-TSR and Belgium Traffic Sign Classification Database (98.93%). Furthermore, the proposed approach ensures a fast training and validation process, using a very limited number of parameters.

On the other hand, the approach has a quick response time (0.2ms/image), and could be implemented in real-time systems with low resources environments.

To ensure the generalizability of the obtained results regardless of the used datasets, we will work on adopting a scaling method based on state-of-the-art approaches, especially EfficientNet proposed by Tan & Le [34] in order to ameliorate the performances of our model.

References

[1] World Health Organization, *Global Status Report on Road Safety*. Geneva, 2018.

[2] A. Kejariwal and F. Orsini, "On the definition of real-time: applications and systems", *in Proc. IEEE Trustcom/BigDataSE/ISPA Conf.*, 2016, pp. 2213-2220.

[3] M. Romadi, R. Faizi, and R. Chiheb, "A shape-based approach for detecting and recognizing traffic signs in a video stream", *in Proc. IEEE 4th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2017.

[4] K. Sumi and K. Arun, "Detection and recognition of road traffic signs: a survey", *International Journal of Computer Applications*, 160(3), 2017.

[5] B. Sanyal, R. K. Mohapatra, and R. Dash, "Traffic sign recognition: a survey", *in Proc. International Conference on Artificial Intelligence and Signal Processing (AISP)*, 2020, pp. 1-6.

[6] B. Bousarhane, D. Bouzidi, and S. Bensiali, "Road signs recognition: state-of-the-art and perspectives", *International Journal of Data Analysis Techniques and Strategies*, vol.13, No.1/2, 2021, pp. 128-150.

[7] D. Zang, W. Zhihua, M. Bao, J. Cheng, D. Zhang, K. Tang, and X. Li, "Deep learning-based traffic sign recognition for unmanned autonomous vehicles", *in Proc. (IMechE) Systems and Control Engineering Conf.*, 2018.

[8] V. Hoang, M. Le, T. Tran, and V. Pham, "Improving traffic signs recognition-based region proposal and deep neural networks", *in Proc. (S. I. AG) ACIIDS: Intelligent Information and Database Systems Conf.*, 2018, pp. 604- 613.

[9] Z. Huang, Y. Yu, J. Gu, and H. Liu, "An efficient method for traffic sign recognition based on extreme learning machine", *in Proc. IEEE Transactions on Cybernetics Conf.*, vol.47, no.4., 2017, pp. 920-933.

[10] A. Saouli, M. Aroussi, and Y. Fakhri, "Traffic sign recognition based on multi-feature fusion and ELM classifier", *Procedia Computer Science*, 127, 2018, pp. 146-153.

[11] Z. Qin and W. Yann, "Traffic-sign recognition using deep learning. geometry and vision", *in Proc. First International Symposium, ISGV Conf.*, 2021.

[12] F. Jurišić, I. Filkovic, and Z. Kalafatic, "Multiple-dataset traffic sign classification with oneCNN", *in Proc. 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, 2015, pp. 614-618.

[13] D. Ciregan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification", *in Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3642-3649.

[14] P. Sermanet and Y. LeCun. "Traffic sign recognition with multi-scale Convolutional Networks. *in Proc. International Joint Conference on Neural Networks*, 2011, pp. 2809-2813.

[15] Á. Arcos-García, JA. Álvarez-García, and LM. Soria-Morillo, "Deep neural network for traffic sign recognition systems: an analysis of spatial transformers and stochastic optimization methods", *Neural Networks*, vol. 99, 2018, pp. 158-165.

[16] W. Li, D. Li, and S. Zeng, "Traffic sign recognition with a small Convolutional Neural Network", *in Proc. IOP Conf. Series: Materials Science and Engineering,* vol.688, issue 4, 2019.

[17] A., Zaibi, A. Ladgham, and A. Sakly, "A lightweight model for traffic sign classification based on enhanced LeNet-5 network". *Journal of Sensors*, 2021, 13 pages.

[18] B. Shinde, "Noise detection and removal filtering techniques in medical images", *International Journal of Engineering Research and Applications (IJERA)*, vol.2, no.4, 2012, pp.311-316.

[19] I. Sazanita, S. Noraini, M. Muzaimi and D. Sailudin, "Evaluating denoising performances of fundamental filters for T2-weighted MRI images", *Procedia Computer Science*, vol.60, 2015, pp. 760-768.

[20] H. P. Singh, A. Nigam, and A. K. Gautam, "Noise reduction in images using enhanced average filter", *in Proc. IJCA International Conference on Advances in Computer Engineering and Applications ICACEA(4)*, 2014, pp. 25-28.

[21] V. Malik and M. Gahlan, "Comparative study of image smoothing using mean and median filtering on digital images", *IJESC*, 2016, pp. 2042-2045.

[22] Y. Zhu and C. Huang, "An improved median filtering algorithm for image noise reduction", *Physics Procedia*, 25, 2012, pp. 609-616.

[23] Z. You, J. Ye, K. Li, Z. Xu, and P. Wang, "Adversarial noise layer: regularize neural network by adding noise", *in Proc. IEEE International Conference on Image Processing (ICIP)*, 2019, pp. 909-913.

[24] B. Bousarhane and D. Bouzidi, "Traffic signs recognition challenges, *International Journal of Computer Science and Applications*, in press.

[25] S. Rakshit, A. Ghosh, and B. U. Shankar, "Fast mean filtering technique (FMFT)", *Pattern Recognition*, vol.40, no.3, 2007, pp. 890–897.

[26] Y. Sun, P. Ge, and D. Liu, "Traffic sign detection and recognition based on Convolutional Neural Network", *in Proc. Chinese Automation Congress (CAC) Conf.*, 2019, pp. 2851-2854.

[27] M. T. Islam, "Traffic sign detection and recognition based on Convolutional Neural Networks", *in Proc. International Conference on Advances in Computing, Communication and Control (ICAC3)*, 2019, pp. 1-6.

[28] B. Bousarhane and D. Bouzidi, "A new deep learning architecture for improving the accuracy and the inference time of traffic signs classification", *in Proc. Fifth International Conference on Big Data and Internet of Things (BDIoT'21)*, in press.

[29] B. Bousarhane and D. Bouzidi, "Convolutional Neural Networks for traffic signs recognition", *in Proc. International Conference on Advanced Communication Systems and Information Security (ACOSIS 2019)*, 2020, pp. 73-91.

[30] D. Temel, G. Kwon, M. Prabhushankar, and G. AlRegib, “CURE-TSR: Challenging Unreal and Real Environments for Traffic Sign Recognition”, *in Proc. Neural Information Processing Systems (NIPS) Workshop on Machine Learning for Intelligent Transportation Systems, U.S*, 2017.
[31] D. Temel and G. AlRegib, “Traffic signs in the wild: highlights from the IEEE video and image processing cup 2017 student competition [sp competitions]”, *IEEE Signal Processing Magazine*, vol.35, no.2, 2018, pp. 154-161.
[32] D. Temel, M-H. Chen, T. Alshawi, and G. AlRegib, “Challenging environments for traffic sign detection: reliability assessment under inclement conditions, arXiv:1902.06857.
[33] R. Timofte, M. Mathias, R. Benenson, and L. Van Gool, “Traffic sign recognition - how far are we from the solution?”, *in Proc. International Joint Conference on Neural Networks (IJCNN)*, 2013.
[34] M. Tan and Q. Le, "EfficientNet: rethinking model scaling for convolutional neural networks," *in Proc. 36th Int. Conf. Mach. Learn.,* 2019, pp. 6105–6114.