Improving the transfer learning performances in the classification of the automotive traffic roads signs

Anass Barodi[1, *], *Abderrahim* Bajit[1], *Mohammed* Benbrahim[1], and *Ahmed* Tamtaoui[2]

[1]Laboratory of Advanced Systems Engineering ISA, National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco.
[2]National Institute of Posts and Telecommunications (INPT-Rabat), SC Department, Mohammed V University. Rabat, Morocco.

Abstract. This paper represents a study for the realization of a system based on Artificial Intelligence, which allows the recognition of traffic road signs in an intelligent way, and also demonstrates the performance of Transfer Learning for object classification in general. When systems are trained on the aspects of human visualization (HVS), which helps or generates the same decisions, the construct robust and efficient systems. This allows us to avoid many environmental risks, both for weather conditions, such as cloudy or rainy weather that causes obscured vision of signs, but the main objective is to avoid all road risks that are dangerous to achieve road safety, such as accidents due to non-compliance with traffic rules, both for vehicles and passengers. However, simply collecting road signs in different places does not solve the problem, an intelligent system for classifying road signs is needed to improve the safety of people in its environment. This study proposed a traffic road sign classification system that extracts visual characteristics from a Convolution Neural Network (CNN) classification model. This model aims to assign a class to the image of the road sign through the classifier with the most efficient optimized. Then the evaluation of its effectiveness according to several criteria, using the Confusion Matrix and the classification report, with an in-depth analysis of the results obtained by the images that are taken from the urban world. The results obtained by the system are encouraging in comparison with the systems developed in the scientific literature, for example, the Advanced Driving Assistance Systems (ADAS) of the sector automobile.

1 Introduction

Machine learning, which is one of the sub-domains of Artificial Intelligence, aims at automatically extracting and exploiting the information present in a dataset for the automobile sector [1]. It differs from traditional approaches and facilitates the use of machines or systems in building models from sample data to automate decision-making processes based on the data entered [2]. Learning algorithms can be categorized according to the type of learning they employ. Training a Convolutional Neural Network is very expensive. The more layers are stacked, the more convolutions and parameters to be optimized. The machine must be able to store several gigabytes of data and do the calculations efficiently. That's why hardware manufacturers are stepping up efforts to provide high-performance, GPU graphics processors that can quickly drive a Deep Neural Network by parallelizing the computations [3]. Transfer Learning allows you to do Deep Learning without having to spend a calculating.

The principle is to use the knowledge acquired by a Neural Network when solving a problem to solve another more or less similar one [4]. In this way, a transfer of knowledge is achieved. Like many things in Machine Learning, this technique is inspired by human behavior, for example, when the driver is driving the car in a very dangerous situation, so he has to react as quickly as possible, so as not to have very serious consequences. In addition to speeding up the training of the network, Transfer Learning also helps to avoid overfitting [5]. Indeed, when the collection of input images is small, it is strongly advised not to train the Neural Network from scratch. Nowadays, we can easily retrieve it, and especially in Deep Learning libraries, such as Keras [6]or Pytorch [7], we can exploit the pre-trained Neural Network in several ways, depending on the size of the input data set and its similarity with the one used during pre-training. This strategy consists of using the features of the pre-trained network to represent the images of the new problem. To do this, remove the last fully-connected layer and set all other parameters [8]. This truncated network will then calculate the representation of each input image from the features already learned during pre-training. A classifier, randomly initialized, is then trained on these representations to solve the new problem when the new image collection is small and similar to the pre-training images. Indeed, training the network on so few images is dangerous since the risk of overfitting, it’s very important point in Deep Learning. Also, if the new images look like the old ones, then they can be represented by the same features [9].

[*] Corresponding author: barodi.anass@uit.ac.ma https: //orcid.org/0000-0003-3022-4761

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (http://creativecommons.org/licenses/by/4.0/).

2 Strategy

In this system in Figure.1, the explore the fine-tuning technique, that allows us to exploit a pre-trained Convolutional Neural Network, and adapt it to a new classification case. The system is composed of two phases, each phase contains a module. The first phase is devoted to learning the model. This phase starts with an essential step which is the pre-processing of the image base. Then we obtain a new pre-processed image base, these images go through two other stages. The first step is to adapt the architecture of the pre-trained network to the new classification case. This is done by changing the network architecture. In this step, we explore two different Convolutional Neural Network architectures namely VGG-16, ResNet-34. The second stage consists of training the Convolutional Neural Network on a learning basis. During this step, the network parameters are adjusted to the new problem. Finally, after performance evaluation, the trained model is deployed.

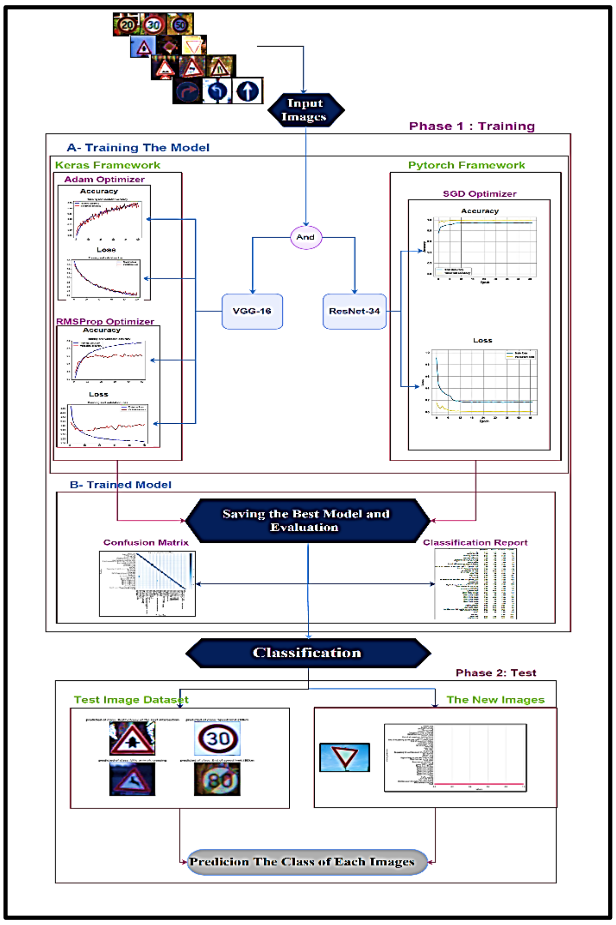


Fig. 1. The approach for the realization of the system.

When the new test images arrive, the second phase of the system is triggered to process the request following the same steps of the pre-processing previously applied during training. This is followed by the prediction and classification of each image. The architecture of the approach proposed in this work is based on the Transfer Learning technique. From a set of learning images, the system aims to deduce a link between the characteristics of each image and its class.

3 Methodology

Machines learn employing a loss function. This is a method for evaluating how well a specific algorithm model the data provided. If the predictions deviate too much from the actual results, the loss function would cover a very large number of them. Gradually, using a non-convex optimization function, the loss function learns how to reduce the forecast error [10]. There are several loss functions in addition to Categorical Cross-Entropy implemented in the model, and the learning rate is probably the most important aspect of gradient descent, along with other optimizers, and it comes down to experimentation and intuition [11]. The choice of the optimization algorithm can make the difference between achieving good or high accuracy in hours or days. Among these algorithms.

3.1 Gradient Descent Optimizer

This technique has been declined in many algorithms with their specificities, gradient descent by batch, stochastic (SGD), and mini-batch. In practice, gradient descent consists of calculating the gradient of the cost function with the derivative of the error concerning. The AI parameters that can be updated, then updating the parameters, for example, the weights an Artificial Neural Network, in the direction that will decrease the error, it is fast, efficient, robust, and above, all flexible in the sense, that it is used in many algorithms, from Deep Learning with Neural Networks to the more traditional Machine Learning [12]. Stochastic Gradient Descent is the simplest optimization algorithm to find parameters that minimize, the given cost function. SGD (the Stochastic Gradient Descent) updates the parameters for each example in the data set $x^{(i)}$ and label $y^{(i)}$.

$$\theta = \theta - \eta . \nabla_{\theta} . J\left(\theta;\ x^{(i)};\ y^{(i)}\right) \qquad (1)$$

When lowering the learning rate η , SGD shows the same convergence as the batch gradient goes down.

3.2 Adaptative Momentum estimation Optimizer

ADAM this method calculates adaptive learning rates, for each parameter, the keeps an exponentially decreasing average of squared and past gradients [13]. These two averages are estimates of the first moment (the mean), the second moment (the uncentered variance) of the gradients. This is an update the optimizer RMSProp, this optimizer is based essentially on the current averages of gradients and gradient moments, with the parameters, are w^twhich is linked to a loss function L^t, where t indexes the current training iteration indexed to 0, the parameters of Adam is given by these equations.

$$m_{\omega}^{(t+1)} \longleftarrow \beta_1 m_{\omega}^{(t)} + (1-\beta_1)\nabla_{\omega} L^{(t)} \qquad (2)$$

$$v_{\omega}^{(t+1)} \longleftarrow \beta_2 v_{\omega}^{(t)} + (1-\beta_2)(\nabla_{\omega} L^{(t)})^2 \qquad (3)$$

$$\hat{m}_{\omega} = \frac{m_{\omega}^{(t+1)}}{1-\beta_1}\ , \hat{v}_{\omega} = \frac{v_{\omega}^{(t+1)}}{1-\beta_2}\ , \omega^{(t)} \longleftarrow \omega^{(t+1)} - \eta \frac{\hat{m}_{\omega}}{\sqrt{\hat{v}_{\omega}}+\varepsilon} \qquad (4)$$

where ε is a small scalar (e.g. 10^{-8} used to prevent division by 0, and β_1 (e.g. 0.9) and β_2(e.g. 0.999), without forgetting its factors of gradients and gradient moments which serves the squaring and squaring is done by each element

3.3 Resilient Momentums prop Optimizer

RMSProp also tries to dampen the oscillations, but differently than the amount of movement. RMSprop also removes the need to adjust the learning rate and does so automatically. This RMSprop algorithm is an alternative version of Adagrad and looks like going down a gradient with momentum, the difference is in the mathematical calculation of the gradient. RMSProp is a method with a learning rate that adapts to each parameter. Dividing the learning rate for the weight, by a moving average the amplitudes of the recent gradients for this weight, with the moving average are calculated in terms of the square averages, the good adaptation of learning rate demonstrated in its following equations.

$$v(w,t) = \gamma v(w,t-1) + (1-\gamma)\left(\nabla Q_i(w)\right)^2 \qquad (5)$$

where γ is the factor of forgetting with updated parameters.

$$\omega_t = \omega_{t-1} - \frac{\eta}{\sqrt{v(\omega,t)}}\nabla Q_i(\omega) \qquad (6)$$

3.3 VGG-16 Architecture Description

Table 1. VGG-16 ARCHITECTURE.

Layer Types		Output Shapes	Parameters
Input Layer		(None, 32, 32, 3)	0
Block1	Conv2D	(None, 32, 32, 64)	1792
	Conv2D		36928
	MaxPooling2D	(None, 16, 16, 64)	0
Block2	Conv2D	(None, 16, 16, 128)	73856
	Conv2D		147584
	MaxPooling2D	(None, 8, 8, 128)	0
Block3	Conv2D	(None, 8, 8, 256)	295168
	Conv2D		590080
	Conv2D		590080
Block3	Conv2D	(None, 8, 8, 256)	590080
	MaxPooling2D	(None, 4, 4, 256)	0
Block4	Conv2D	(None, 4, 4, 512)	1180160
	Conv2D		2359808
	Conv2D		2359808
	Conv2D		2359808
	MaxPooling2D	(None, 2, 2, 512)	0
Block5	Conv2D	(None, 2, 2, 512)	2359808
	Conv2D		2359808
	Conv2D		2359808
	Conv2D		2359808
	MaxPooling2D	(None, 1, 1, 512)	0
Total parameters: 20,024,384/ Trainable parameters: 20,024,384 Non-trainable parameters: 0			

VGG-16 consists of several layers, including 13 convolution layers and 3 fully-connected layers. So, it has to learn the weights of 16 layers. It takes a 224 × 224 px color images as input and classifies it into one of 1000 classes. It then returns a vector of size 1000, which contains the probabilities of belonging to each of the classes[14]. So, we will spend some time studying the configuration of the different layers of VGG-16. VGG-16 is made of several layers, including 13 convolution layers and 3 fully-connected layers, in our case, we take a 32x32 px color image and classifies it in one of the 43 classes. The architecture of VGG-16 is illustrated by the Table. 1 below. Now that we've mastered the VGG-16 architecture, the fun part is the implementation. Implementing a Neural Network with Keras means creating a sequential model and enriching it with the corresponding layers in the right order. The most difficult step is to correctly define the parameters of each layer, hence the importance is understanding the network architecture. Finally, the ReLU layer is used just after the convolution layer, the argument activation ReLU is a very important activation function in Deep Learning. For example in Table.1, for the first convolution layer, the network has to learn 64 color filters thus depth 3 of size 3; as well as a bias parameter for each filter. This makes a total of (3*3*3) *64 + 64 = 1792 parameters.

3.4 ResNet34 Architecture Description

ResNet-34 Unlike traditional sequential network architectures such as VGG. ResNet is rather a form of exotic architecture based on micro-architecture modules, also called network within network architectures. The term micro-architecture refers to the set of building blocks used to construct the network. A set of micro-architecture building blocks with the standard layers Convolution, Pooling, etc. leads to the macro-architecture, i.e. the end network itself. The papers describe these in detail of each step, that ResNet consists of an orange convolution and pooling step followed by 4 layers of similar behavior. Each layer follows the same pattern. The perform a 3x3 convolution with a fixed function map dimension (F) 64, 128, 256, 512 respectively, bypassing the input every 2 convolutions. Besides, the width (W) and height (H) dimensions remain constant throughout the layer. The dotted line is there precisely because the input volume dimension has changed, a reduction because of the convolution [15]. Note that this reduction between layers is obtained by increasing the stride, from 1 to 2, at the first convolution of each layer; instead of a clustering operation, what are used to seeing as samplers. The ResNet-34 architecture is illustrated in the diagrams below in figure.2.

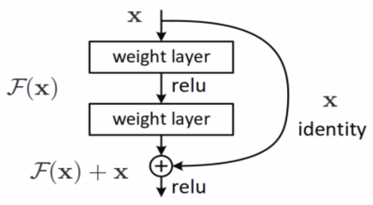

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig. 2. Residual connection.

This makes it easier to understand the mechanism of a particular model, to adapt it to our particular needs. The next section shows how to simply change the strengths of the dataset to change the architecture of the whole model. Also, trying to follow the notation close to the official implementation of PyTorch to facilitate its later implementation on Python. All the operations of convolutions, effected with batch normalization and ReLU activation to an input, except for the last operation in a block, which does not have the ReLU, we are faced with a base now.

4 Related work

The experimentation is done in the Keras and PyTorch Frameworks, which are very intuitive libraries of Deep Learning in Python. The focus of our study is VGG-16, a version for implementing Transfer Learning in ASUSTEK Computer i7-7500U GPU NVIDIA GEFORCE 920M. Python offers several libraries and has an active community. Regarding data processing, the Numpy, Pandas, and Matplotlib are libraries for matrix calculations, data analysis, and visualization. For Deep Learning, python has working environments such as Keras and Pytorch. These work environments are very useful because they allow easy management of the backpropagation algorithm for large Neural Networks, VGG-16, and ResNet-34-. They also ensure the parallelization of calculations on the GPU.

4.1 Model Transfer Learning Keras

The data are limited, then the proposition is to take a VGG network and to train only the top classifier part of the network, indeed the 2D convolution part already does a huge work of feature extraction on an already-trained network, and is already able to classify a lot of objects. The adaptation of VGG-16 to our needs by replacing only the top layers, in the situation has very few training images. Also artificially inflate the training set by reusing the same images several times in Keras. The VGG-16 is trained on ImageNet, which offers in our case 43 classes in output. The first step is to remove the top layers, both fully connected and softmax, from a pre-trained VGG-16 and replace them with our grading network. Considering the architecture, VGG-16 will simply be charged, not freezing the fully connected layers (FC). Redefining the network allows us to test different architectures that can add only the softmax or change the number of neurons in the FC. This model is the result of different experiments in Table. 2.

Table 2. VGG-16 Architecture update.

Layer Types	Output Shapes	Parameters
Model VGG16	(None, 512)	14714688
Dense		262656
Dropout		0
Dense		262656
Dropout		0
Dense	(None, 43)	22059
Total parameters: 15,262,059 Trainable parameters: 547,371 Non-trainable parameters: 14,714,688		

4.2 Implementation with Adam optimizer

This section is for the implementation of the VGG16 net architecture with optimizer Adam. In practice, this is the most used optimization especially for Deep Learning in Neural Networks, because of its efficiency and stability, the Transfer Learning, even if, as we will see, it is not perfect, also can see how to optimize our classifier. The majority of scientific researchers recommend us when the learning rate is reduced, this affects the training by given game, in this case, the value in Table.3 without forgetting Adam's betas.

Table 3. ADAM acceleration by betas.

Optimizer	Lr	Beta_1	Beta_2
Adam	0.0001	0.9	0.999

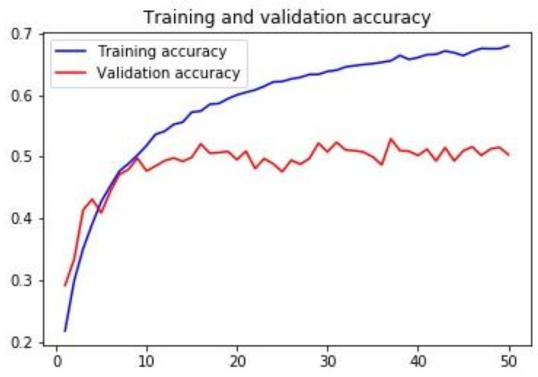


Fig. 3. Accuracy function of VGG-16 with ADAM.

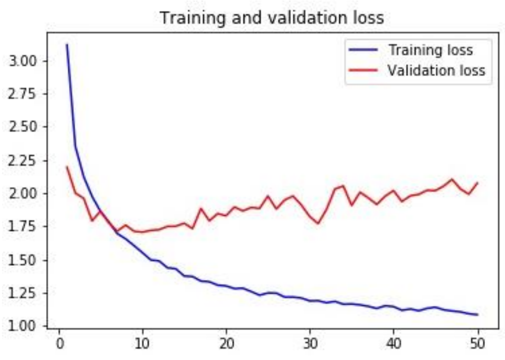


Fig. 4. The loss function of VGG-16 with ADAM.

For Adam optimizer for Figure.3 and Figure.4. Note that Adam combines the ideas of Adagrad and RMSprop. Thus, in addition to the classifier, the new images are trained on the unfixed layers, which generally correspond to the highest layers of the network. This strategy is used when the new image collection is small but very different from the pre-training images. On the one hand, as there are few training images, training the whole network is not feasible, so the caused the overfitting. On the other hand, it also eliminates, since the new images have very little in common with the old ones, using the optimizer Adam is not a perfect idea, but remember that the features of the lower layers are simple and generic so can be found in two very different images, while those of the upper layers are complex and specific to the problem. So, the strategy of setting, change the optimizer, and training the classifier.

4.3 Implementation with RMSprop optimizer

This part starts by using the architecture of VGG16 net with optimizer RMSprop, the model is the same, in the same direction as the previous ones, with a reduction in the learning rate, and defined the essential parameters of the optimizer, are rho and epsilon defined all in the Table. 4, at which we go down the curve of accuracy and loss for visualization the effect of changing these updates parameters.

Table 4. RMSprop acceleration by Rho.

Optimizer	Lr	Rho	Epsilon
RMSprop	2e-3	0.9	1e-07

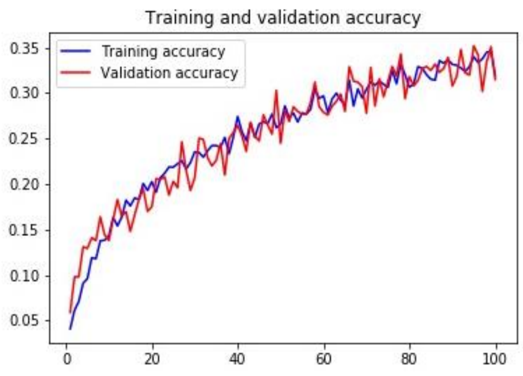


Fig. 5. Accuracy function of VGG-16 with RMSprop.

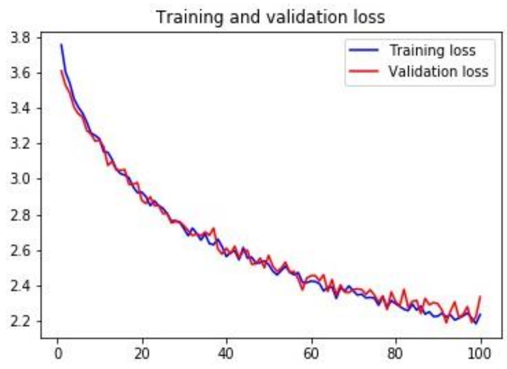


Fig. 6. The loss function of VGG-16 with RMSprop.

The RMSProp optimizer takes the stress out of choosing the right learning speed. Now that we know a bit more about optimizers, let's take a look at RMSProp, the evolution that corrects Adam's main weakness, his strong dependence on learning speed. Indeed, as shown in Figure.5 and Figure.6, AI precision learning curves on the validation data with Adam, can conclude that final results depend strongly on the choice of the learning rate of algorithm. Its far from the risk of overfitting, while there are some spikes in the validation accuracy and loss, overall, that it is much closer to the training accuracy, with the loss indicating obtained of model, that generalizes much better as compared to previous model. But the remark can tack at the high score of accuracy is 35%, it's not better, because it doesn't take time enough. The idea now, to change this model by using the model profound which is residual in Pytorch Framework, and evaluate the performances on test dataset.

5 Discussion

The observation that it was logical to state that the deeper, the better concerning Convolutional Neural Networks. This makes sense, models should be more capable, their flexibility to adapt to any space increases, a larger space of parameters to explore. However, it has been noticed that after a certain depth, performance degrades. It was one of VGG's bottlenecks. The couldn't get as far as to want, because starting to lose the ability to generalize with SGD, this method is faster but too frequent parameter updates cause the objective function to oscillate, these oscillations, on the one hand, allow landing in potentially better local minima, but on the other hand, make convergence more difficult.

5.1 Amelioration of SGD optimizer

SGD has difficulty in areas where the surface is much more curved in one dimension than another, and their common around local minima. In these cases, SGD wobbles through the slopes of these regions and makes only hesitant progress towards local optimums. One method that can help the network get out of these traps is to use the Momentum Coefficient.

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta) \quad with \ \theta = \theta - v_t \qquad (7)$$

or $\gamma \in [0; 1]$ represents the Momentum coefficient.

When using Momentum, one pushes a ball down a hill. The ball accumulates Momentum as it rolls downhill becoming faster and faster. The same thing happens when updating the settings. Momentum increases for dimensions whose gradient points in the same direction and reduces updates whose gradient changes direction [16]. The result is faster convergence and reduced oscillations, based on the most relevant new parameters, increases the Learning rate with Momentum defined in Table. 5.

Table 5. SGD acceleration by Momentum.

Optimizer	Lr	Momentum
SGD	0.001	0.9

5.2 Model Transfer Learning Pytorch

ResNet allows the learning of very deep networks more than 150 layers. One of the difficulties in learning such deep networks is related to gradient backpropagation. The deeper the network is the lower, the gradient is for updating the weights of the lower layers the first layers. Thus, an architecture that is too deep does not update these layers. The idea developed in ResNet is the use of residual connections allowing better optimization of very deep networks. A residual connection allows us to pass the input in two convolution filters, but also to pass this input directly to the next layers. This is done, by adding the result of the two convolution layers and the input as shown in Figure 2. With this architecture, the authors demonstrate the interest in learning very deep networks due to their performances and propose a way to learn them efficiently. It also uses global AVG pooling instead of PMC at the end. The error rate in ImageNet visual recognition challenge, Deep Learning exceeds human performance.

Table 6. ResNet-34 Architecture update.

Layer (type)	Output Shape	Param #
model_1(Resnet34)	[-1, 512, 7, 7]	21,284,672
AdaptiveAvgPool2d-123	[-1, 512, 1, 1]	0
Linear-124	[-1, 43]	22,059
Total params: 21,306,731 Trainable params: 21,306,731 Non-trainable params: 0		
Input size (MB): 0.57 Forward/backward pass size (MB): 96.28 Params size (MB): 81.28 Estimated Total Size (MB): 178.13		

The main flaw of VGG-16 is the imposing size of the network about 500 MB in Table.2, compared to other networks such as ResNet-34 about 178 MB. Also, the computing power needed for one pass is important, as shown in the Table. 6. However, it is a robust and relatively readable architecture for a deep network. It is not the best model, but it is a good reference for the first estimation of possible performance on classification as it is easy and quick to implement. ResNet on paper is mainly explained for the ImageNet dataset. The experiments with sets of ResNets, to do it on traffic road signs. Since the input images of the panels are (32x32), we trying to make transformations and dimensionality compatible with the architecture used which admits (224x224). To have control over modifications can be applied for the ResNet, the requires to understand the details. The majority of researchers change the dimension but make the efficiency of the network to solve the problem posed low as shown previously for VGG-16.

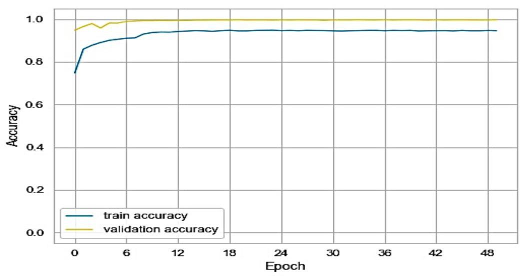


Fig. 7. Accuracy function of ResNet-34 with SGD.

The problem is solved by ResNet-34, which is the famous known disappearing gradient for accuracy and loss in Figure. 7 with a Figure. 8. Indeed, when the network is too deep, the gradients from which the loss function is calculated, easily reduced nearly to zero after several applications of the chain rule. This result on the weights never updates its values, and therefore no learning is done, with ResNet-34, the gradients can pass directly through the jump back connections of the subsequent layers to the initial filters. The pre-trained model is so good, that got very high accuracy and low loss after 50 epochs. Unfortunately, the validation set is too small to get some meaningful metrics from it in the Figure. 8. The AI convergence over time as a function of the chosen learning rate, and the optimizer SGD thus for a learning speed of 0.001, the reach in just over 50 epochs 97% accuracy, but when using Adam, 70 % accuracy with Learning rate 0.0001 and RMSprop 35% accuracy with learning rate 0.0002. So, when a change model has improved the work in the Framework of Pytorch, for SGD optimizer, we would have the algorithm converges to the same accuracy. It is then longer necessary to perform the work in future the AI for real training.

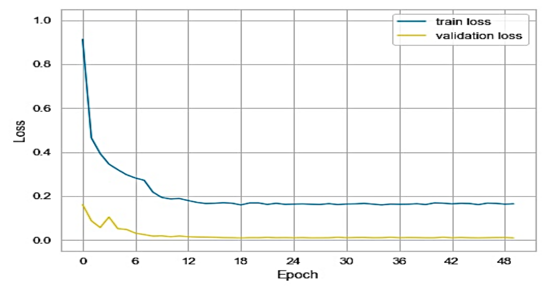


Fig. 8. The loss function of ResNet-34 with SGD.

6 Results and Experiments

6.1 Evaluation performance of a better model

The objective of this step is to make the global test of the learning model built in the previous steps. Before going on to the learning, the image data is divided into three subsets, one for learning (60%), one for testing (20%), and one for validation (20%). The learning is done through the images that make up the main part of the dataset, and since this is a supervised method. For the evaluation model, it receives the test images as input, it gives as output a vector of probabilities belonging to the images to each class. In this section, present the Confusion Matrix of Transfer Learning. According to Figure.9, the ResNet-34 model had classified all test images correctly, set to some classes. We observe confusion between the Low and Medium class. ResNet-34, the matrix shows the performance improvement with cross-validation. This is shown by the diagonal of the matrix which proves that all images were correctly classified. The encouragement to keep in mind the possible strategies introduced in the previous fine-tuning, for feature extraction.

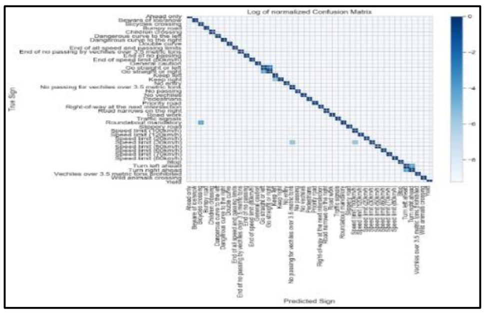


Fig. 9. Confusion Matrix of ResNet-34.

6.2 Classification report

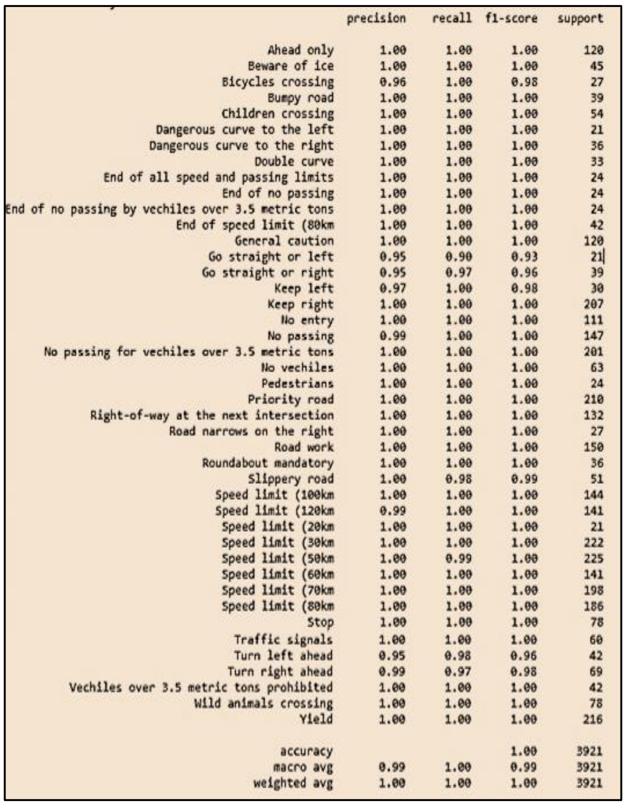
The classification report shows us that the ResNet-34 model is perfect, can see in Figure.10, that the accuracy of the model reaches 99.74%. Moreover, to check the rate agreement or concordance, also can calculate the KAPPA coefficient [17] based on this equation below.

$$KAPPA = \frac{(P_0 - P_e)}{(1 - P_e)} \qquad (8)$$

$$with\ P_0 = \sum_{i=1}^{r} p_{ii} \quad P_e = \sum_{i=1}^{r} p_i . p_i$$

r : The number of ways of judging.
P_0: The proportion of agreement observed.
P_e: The proportion of random agreement or concordance is expected under the assumption of independence of judgments.

The KAPPA coefficient in our case is 99.73%, the score is higher than 81%, is an excellent degree in the agreement but more importantly, the accuracy in identifying images, that are traffic road signs is what makes it a reliable removal filter.



```
                                                precision    recall  f1-score   support

                                    Ahead only       1.00      1.00      1.00       120
                                 Beware of ice       1.00      1.00      1.00        45
                             Bicycles crossing       0.96      1.00      0.98        27
                                    Bumpy road       1.00      1.00      1.00        39
                             Children crossing       1.00      1.00      1.00        54
                  Dangerous curve to the left        1.00      1.00      1.00        21
                 Dangerous curve to the right        1.00      1.00      1.00        36
                                  Double curve       1.00      1.00      1.00        33
           End of all speed and passing limits       1.00      1.00      1.00        24
                             End of no passing       1.00      1.00      1.00        24
End of no passing by vechiles over 3.5 metric tons       1.00      1.00      1.00        24
                       End of speed limit (80km       1.00      1.00      1.00        42
                               General caution       1.00      1.00      1.00       120
                           Go straight or left       0.95      0.90      0.93        21
                          Go straight or right       0.95      0.97      0.96        39
                                     Keep left       0.97      1.00      0.98        30
                                    Keep right       1.00      1.00      1.00       207
                                      No entry       1.00      1.00      1.00       111
                                    No passing       0.99      1.00      1.00       147
    No passing for vechiles over 3.5 metric tons       1.00      1.00      1.00       201
                                   No vechiles       1.00      1.00      1.00        63
                                   Pedestrians       1.00      1.00      1.00        24
                                 Priority road       1.00      1.00      1.00       210
         Right-of-way at the next intersection       1.00      1.00      1.00       132
                      Road narrows on the right       1.00      1.00      1.00        27
                                     Road work       1.00      1.00      1.00       150
                          Roundabout mandatory       1.00      1.00      1.00        36
                                 Slippery road       1.00      0.98      0.99        51
                            Speed limit (100km       1.00      1.00      1.00       144
                            Speed limit (120km       0.99      1.00      1.00       141
                             Speed limit (20km       1.00      1.00      1.00        21
                             Speed limit (30km       1.00      1.00      1.00       222
                             Speed limit (50km       1.00      0.99      1.00       225
                             Speed limit (60km       1.00      1.00      1.00       141
                             Speed limit (70km       1.00      1.00      1.00       198
                             Speed limit (80km       1.00      1.00      1.00       186
                                          Stop       1.00      1.00      1.00        78
                               Traffic signals       1.00      1.00      1.00        60
                               Turn left ahead       0.95      0.98      0.96        42
                              Turn right ahead       0.99      0.97      0.98        69
     Vechiles over 3.5 metric tons prohibited       1.00      1.00      1.00        42
                         Wild animals crossing       1.00      1.00      1.00        78
                                         Yield       1.00      1.00      1.00       216

                                      accuracy                           1.00      3921
                                     macro avg       0.99      1.00      0.99      3921
                                  weighted avg       1.00      1.00      1.00      3921
```

Fig. 10. Report classification for each class.

6.4 Tests results with images of the test dataset

Visualization performance of the model in classifications images test dataset in Figure.11, the results prediction the model, it was able to correctly guess 4 of the traffic road signs, which gives an accuracy of 100%. This compares favorably to the accuracy on the test set of 99.73%.

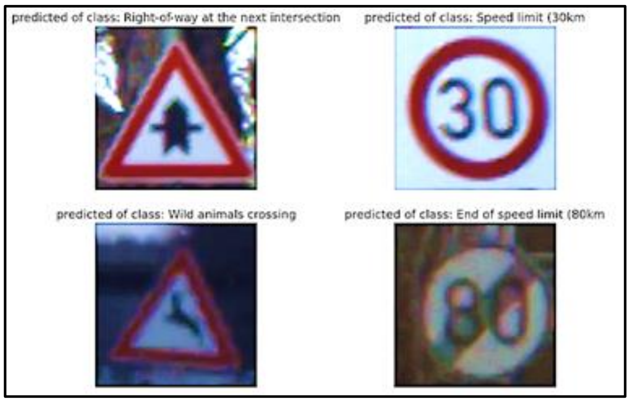
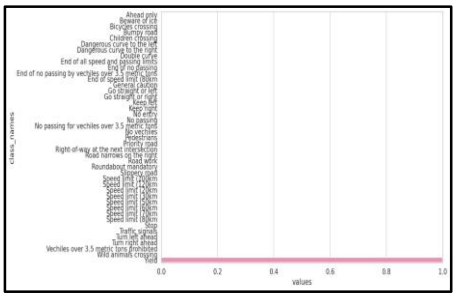
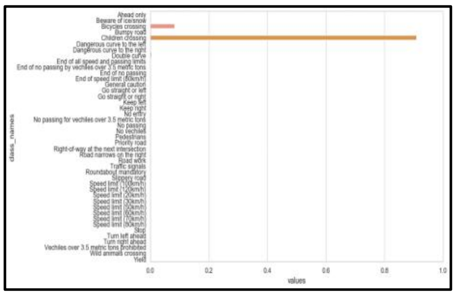

predicted of class: Right-of-way at the next intersection
predicted of class: Speed limit (30km
predicted of class: Wild animals crossing
predicted of class: End of speed limit (80km

Fig. 11. Classification of test images.

6.4 Test results with new images

In this part, shows certain the model is when predicting each of the new images taking from the real world or real-time, by looking at the softmax probabilities for each prediction class. To evaluate the model confidence in classifying traffic road signs, the new images set, are the "Yield" and "Crossing Children" traffic road signs.

Table 7. Classification of new images.

Image	Prediction
	
	

This result is expected given in Table.7, the new images used, in this test has some of the noise, so it is very hard to classify correctly. For this, that improves the confidence for recognition. The model is performing very well, for classification "Yield" signs in the prediction in the correct traffic signs by the accuracy of 100%. For the "Children Crossing" sign we have nearly 90% of prediction correctly and 10% by confusion with a class of "Bicycle Crossing" class, this could be corrected by using a deeper residual architecture, which contains more layers like ResNet-101 or ResNet-150, which perform better with a more powerful GPU to get very satisfying results.

6 Conclusion

This research work is part of the classification, as well as image classification techniques. In the paper, the dealt with an issue that affects the whole world, which is a great challenge facing all societies and research laboratories today. The adopted a research methodology that consists of designing and implementing a large system of automatic image classifications including an automatic prediction perspective. Indeed, the system is based on an approach that exploits the Transfer Learning technique for automatic learning, which extracts low-level visual characteristics from the image of traffic road signs in different environments. This module aims to create, train, and evaluate a classifier model with residual architecture. Therefore, conclude that image classification with deep supervised learning methods is an important avenue of research.

Our work opens scientific perspectives in the short and long term. In the following, the highlight perspectives that believe are relevant to the evolution of the systems developed in this project. Firstly, it would be interesting to work with a broader and more variant image base. This work is completed by implementation and improves the performance of Deep Learning by creation a CNN model, second perspective appears is to highlight the execution time, and detection and recognize Traffic Roads Signs in Real-Time with high precision [18].

References

1. J. Stilgoe, ‘Machine learning, social learning and the governance of self-driving cars’, Social Studies of Science, p. 32.
2. A. Barodi, A. Bajit, M. Benbrahim, and A. Tamtaoui, ‘An Enhanced Approach in Detecting Object Applied to Automotive Traffic Roads Signs’, in 2020 IEEE 6th International Conference on Optimization and Applications (ICOA), Beni Mellal, Morocco, Apr. 2020, pp. 1–6, doi: 10.1109/ICOA49421.2020.9094457.
3. S. Shi, Q. Wang, P. Xu, and X. Chu, ‘Benchmarking State-of-the-Art Deep Learning Software Tools’, in 2016 7th International Conference on Cloud Computing and Big Data (CCBD), Macau, China, Nov. 2016, pp. 99–104, doi: 10.1109/CCBD.2016.029.
4. J. Chi, E. Walia, P. Babyn, J. Wang, G. Groot, and M. Eramian, ‘Thyroid Nodule Classification in Ultrasound Images by Fine-Tuning Deep Convolutional Neural Network’, J Digit Imaging, vol. 30, no. 4, pp. 477–486, Aug. 2017, doi: 10.1007/s10278-017-9997-y.
5. G. Liang and L. Zheng, ‘A transfer learning method with deep residual network for pediatric pneumonia diagnosis’, Computer Methods and Programs in Biomedicine, vol. 187, p. 104964, Apr. 2020, doi: 10.1016/j.cmpb.2019.06.023.
6. K. Gopalakrishnan, S. K. Khaitan, A. Choudhary, and A. Agrawal, ‘Deep Convolutional Neural Networks with transfer learning for computer vision-based data-driven pavement distress detection’, Construction and Building Materials, vol. 157, pp. 322–330, Dec. 2017, doi: 10.1016/j.conbuildmat.2017.09.110.
7. A. Paszke et al., ‘PyTorch: An Imperative Style, High-Performance Deep Learning Library’, p. 12.
8. A. Bajit, M. Nahid, A. Tamtaoui, and M. Benbrahim, ‘A Psychovisual Optimization of Wavelet Foveation-Based Image Coding and Quality Assessment Based on Human Quality Criterions’, Adv. sci. technol. eng. syst. j., vol. 5, no. 2, pp. 225–234, 2020, doi: 10.25046/aj050229.
9. Abderrahim. Bajit, Mohammed. Najid, Ahmed. Tamtaoui, and Abdellah. Lassioui, ‘A Perceptually Optimized Embedded Image Coder and Quality Assessor Based Both on Visual Tools’, Adv. sci. technol. eng. syst. j., vol. 4, no. 4, 2019, doi: 10.25046/aj040428.
10. S. De, A. Mukherjee, and E. Ullah, ‘Convergence guarantees for RMSProp and ADAM in non-convex optimization and an empirical comparison to Nesterov acceleration’, arXiv:1807.06766 [cs, math, stat], Nov. 2018, Accessed: Aug. 08, 2020. [Online].
11. M. Anthimopoulos, S. Christodoulidis, L. Ebner, A. Christe, and S. Mougiakakou, ‘Lung Pattern Classification for Interstitial Lung Diseases Using a Deep Convolutional Neural Network’, IEEE Trans. Med. Imaging, vol. 35, no. 5, pp. 1207–1216, May 2016, doi: 10.1109/TMI.2016.2535865.
12. A. W. Senior et al., ‘Improved protein structure prediction using potentials from deep learning’, Nature, vol. 577, no. 7792, pp. 706–710, Jan. 2020, doi: 10.1038/s41586-019-1923-7.
13. D. P. Kingma and J. Ba, ‘Adam: A Method for Stochastic Optimization’, arXiv:1412.6980 [cs], Jan. 2017, Accessed: Aug. 08, 2020. [Online]. Available: http://arxiv.org/abs/1412.6980.
14. G. Pandey, A. Baranwal, and A. Semenov, ‘Identifying Images with Ladders Using Deep CNN Transfer Learning’, in Intelligent Decision Technologies 2019, vol. 142, I. Czarnowski, R. J. Howlett, and L. C. Jain, Eds. Singapore: Springer Singapore, 2020, pp. 143–153.
15. K. He, X. Zhang, S. Ren, and J. Sun, ‘Deep Residual Learning for Image Recognition’, arXiv:1512.03385 [cs], Dec. 2015, Accessed: Aug. 08, 2020. [Online]. Available: http://arxiv.org/abs/1512.03385.
16. D. Li and Q. Chen, ‘Deep Reinforced Attention Learning for Quality-Aware Visual Recognition’, arXiv:2007.06156 [cs], Jul. 2020, Accessed: Aug. 08, 2020.
17. J. Cohen, ‘A Coefficient of Agreement for Nominal Scales’, Educational and Psychological Measurement, vol. 20, no. 1, pp. 37–46, Apr. 1960, doi: 10.1177/001316446002000104.
18. A. Barodi, A. Bajit, M. Benbrahim and A. Tamtaoui "Applying Real-Time Object Shapes Detection To Automotive Traffic Roads Signs.," 2020 International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Morocco, Kenitra, 2020, pp. 1-6*, --Proceeding*