Neural Networks 99 (2018) 158–165

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

Deep neural network for traffic sign recognition systems: An analysis of spatial transformers and stochastic optimisation methods

Álvaro Arcos-García *, Juan A. Álvarez-García, Luis M. Soria-Morillo

Dpto. de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, 41012, Sevilla, Spain

HIGHLIGHTS

- A Deep Neural Network that is top-1 ranked in the German traffic sign benchmark.
- Effectiveness analysis of Spatial Transformer Networks for traffic sign recognition.
- Quantitative comparison of several stochastic gradient descent optimisation methods.

ARTICLE INFO

Article history:
Received 25 July 2017
Received in revised form 15 November 2017
Accepted 18 January 2018
Available online 31 January 2018

Keywords:
Deep learning
Traffic sign
Spatial transformer network
Convolutional neural network

ABSTRACT

This paper presents a Deep Learning approach for traffic sign recognition systems. Several classification experiments are conducted over publicly available traffic sign datasets from Germany and Belgium using a Deep Neural Network which comprises Convolutional layers and Spatial Transformer Networks. Such trials are built to measure the impact of diverse factors with the end goal of designing a Convolutional Neural Network that can improve the state-of-the-art of traffic sign classification task. First, different adaptive and non-adaptive stochastic gradient descent optimisation algorithms such as SGD, SGD-Nesterov, RMSprop and Adam are evaluated. Subsequently, multiple combinations of Spatial Transformer Networks placed at distinct positions within the main neural network are analysed. The recognition rate of the proposed Convolutional Neural Network reports an accuracy of 99.71% in the German Traffic Sign Recognition Benchmark, outperforming previous state-of-the-art methods and also being more efficient in terms of memory requirements.

1. Introduction

Traffic sign recognition systems (TSRS) are essential in many real-world applications such as autonomous driving, traffic surveillance, driver safety and assistance, road network maintenance, and analysis of traffic scenes. Normally, a TSRS concerns two related subjects which are traffic sign detection (TSD) and traffic sign recognition (TSR). The former focuses on the localisation of the targets in the pictures while the latter performs a fine-grained classification to identify the type of targets detected (De La Escalera, Moreno, Salichs, & Armingol, 1997).

Traffic signs constitute a fundamental asset within the road network because their aim is to be easily noticeable by pedestrians and drivers in order to warn and guide them during both the day and night. The fact that signs are designed to be unique and to have distinguishable features such as simple shapes and uniform colours implies that their detection and recognition is a constrained problem. Nevertheless, the development of a robust real-time TSRS still presents a challenging task due to real-world variability, such as scale variations, bad viewpoints, motion-blur, faded colours, occlusions, and lightning conditions. On top of that, there are more than 300 different traffic sign categories defined by the Vienna Convention on Road Traffic (United Nations Economic Commission for Europe, 1968). This treaty has been signed by 63 countries, although a few minor visual variations of traffic sign pictographs still exist between countries, which can lead to complications in the automated recognition task. Any TSRS must cope well with such issues.

The main contributions of this work are four-fold: (1) A state-of-the-art traffic sign recognition system based on a Convolutional Neural Network (CNN) that includes Spatial Transformer Networks (STN) and outperforms previously published work related with the German Traffic Sign Recognition Benchmark (GTSRB) (Stallkamp, Schlipsing, Salmen, & Igel, 2011); (2) An insight into the proposed CNN capabilities along with the performance impact of spatial transformer layers within the network; (3) Analysis of the effect

* Corresponding author.
E-mail addresses: aarcos1@us.es (Á. Arcos-García), jaalvarez@us.es (J.A. Álvarez-García), lsoria@us.es (L.M. Soria-Morillo).

https://doi.org/10.1016/j.neunet.2018.01.005
0893-6080/

of diverse gradient descent optimisation algorithms on the CNN presented. (4) Multiple publicly available European traffic sign classification datasets are reviewed and evaluated by the CNN. These contributions lead to practical applications, such as self-driving cars and automated inventory and maintenance of vertical signage, since the CNN can perform fine-grained classification once the traffic sign has been detected. Moreover, as the CNN outperforms the human visual system, its inference time is low and can also be deployed as a stand-alone service, it can therefore be used in real-time applications.

The rest of the paper is organised as follows. Section 2 reviews related works of traffic sign recognition systems. Section 3 describes the experiments conducted to analyse the impact of both spatial transformers and stochastic optimisation algorithms on the proposed CNN. Recognition results are then shown in Section 4. Finally, conclusions are drawn and further work is proposed in Section 5.

2. Related work

Chronologically, approaches of published studies on traffic sign recognition systems have evolved from colour and shape-based methods to machine-learning-based methods. In recent times, Deep Neural Networks (DNN) have attracted attention in pattern recognition and computer vision research, and have been widely adopted for both object detection (Liu et al., 2016; Redmon & Farhadi, 2016; Ren, He, Girshick, & Sun, 2015) and recognition (Huang, Liu, Weinberger, & van der Maaten, 2016; Szegedy, Ioffe, Vanhoucke, & Alemi, 2017), thanks to the release of several publicly available datasets composed of millions of images (Everingham, Van Gool, Williams, Winn, & Zisserman, 2010; Krizhevsky, Sutskever, & Hinton, 2012; Lin et al., 2014). Moreover, DNNs have been applied in autonomous driving related challenges such as car (Huval et al., 2015), lane (Li, Mei, Prokhorov, & Tao, 2017), and pedestrian (Tian, Luo, Wang, & Tang, 2015) detection.

With regard to the traffic sign detection and classification problem domain, colour-based approaches are very common. These methods use different colour spaces for segmentation of the road image, such as RGB (Escalera, Moreno, Salichs, & Armingol, 1997), HIS (Maldonado-Bascon, Lafuente-Arroyo, Gil-Jimenez, Gomez-Moreno, & Lopez-Ferreras, 2007), and HSV (Shadeed, Abu-Al-Nadi, & Mismar, 2003). The shape-based method is another popular approach for traffic sign recognition and detection. Symmetry information of circular, triangular, square and octagonal shapes are used in Loy and Barnes (2004), a radial symmetry detector is proposed in Barnes, Zelinsky, and Fletcher (2008), Hough transforms are investigated in Barnes, Loy, and Shaw (2010) and a circular traffic sign recognition system is studied in Kaplan Berkaya, Gunduz, Ozsen, Akinlar, and Gunal (2016). Hence, neither colour nor shape-based techniques, need any prior knowledge of traffic signs and heavily depend on custom-designed algorithms and feature engineering.

One of the main problems before the year 2011 was the lack of publicly available traffic sign datasets. The Belgian Traffic Sign Dataset (BTSD) (Timofte, Zimmermann, & Van Gool, 2011), the German Traffic Sign Recognition and Detection Benchmark (GTSRB and GTSDB) (Stallkamp et al., 2011), the Croatian traffic sign dataset (rMASTIF) (Jurisic, Filkovic, & Kalafatic, 2015), the Dataset of Italian Traffic Signs (DITS) (Youssef, Albani, Nardi, & Bloisi, 2016) and the Tsinghua-Tencent 100 K benchmark (Zhu et al., 2016) solved this issue and boosted research into TSRS since several of these datasets are commonly used to evaluate the performance of computer vision algorithms for traffic sign detection and recognition. These kinds of datasets are crucial to generate robust machine learning and deep learning models as they contain a huge amount of traffic sign samples of multiple categories, taken by cameras with various weather and lighting conditions, occlusions, bad viewpoints, etc.

More recently, machine learning has started to play a key role in the traffic sign classification task. Mathias, Timofte, Benenson, and Van Gool (2013) propose fine-grained classification by applying different methods through a pipeline of three stages: feature extraction, dimensionality reduction and classification. On GTSRB, they reach 98.53% accuracy by merging grey-scale values of traffic sign images and features based on the Histogram of Oriented Gradients (HOG), reducing the dimensionality through Iterative Nearest Neighbours-based Linear Projections (INNLP) and finally classifying with Iterative Nearest Neighbours (INNC) (Timofte & Van Gool, 2015). Although other machine learning algorithms such as Support Vector Machines (SVM) (Salti, Petrelli, Tombari, Fioraio, & Di Stefano, 2015), Random Forests (Zaklouta, Stanciulescu, & Hamdoun, 2011) and Nearest Neighbours (Gudigar, Chokkadi, Raghavendra, & Acharya, 2017) have been widely used to recognise traffic sign images, Convolutional Neural Networks (Lecun, Bottou, Bengio, & Haffner, 1998), also known as ConvNets or CNNs, showed particularly higher classification accuracies in the competition. Neural networks are data driven self-adaptive methods because they can adjust themselves to the data without any explicit specification of functional or distributional form for the underlying model (Huang, 1996). In addition, there are universal functional approximators in the neural networks that can approximate any function with arbitrary accuracy (Huang, 1999; Huang & Du, 2008). Cireşan, Meier, Masci, and Schmidhuber (2012) won the GTSRB contest (Stallkamp, Schlipsing, Salmen, & Igel, 2012) with 99.46% accuracy thanks to a committee of 25 CNNs by using data augmentation and jittering. Sermanet and LeCun (2011) used a multi-scale CNN and achieved an accuracy of 98.31%, thereby granting them second place in the GTSRB challenge. Later, Jin, Fu, and Zhang (2014) proposed a hinge loss stochastic gradient descent method to train an ensemble of 20 CNNs that resulted in 99.65% accuracy and offered a faster and more stable convergence than previous work. However, these approaches can still be improved through the avoidance of the use of hand-crafted data augmentation and of the application of multiple CNNs in an ensemble or via a committee for the reason that these normally lead to higher memory and computation costs.

3. Methodology

In this work, we propose a traffic sign recognition system that carries out fine-grained classification of traffic sign images through a CNN whose main blocks are convolutional and spatial transformer modules. In order to find an accurate and efficient CNN for such a purpose, the effect of using several STNs and different stochastic gradient descent optimisation methods are researched and discussed.

3.1. Dataset and data pre-processing

Several publicly available traffic sign datasets have been gathered in countries such as the United States (Mogelmose, Trivedi, & Moeslund, 2012), Belgium (Timofte et al., 2011), Germany (Stallkamp et al., 2011), Croatia (Jurisic et al., 2015), Italy (Youssef et al., 2016), Sweden (Larsson & Felsberg, 2011), and China (Zhu et al., 2016).

This paper focuses on both the spatial transformer effectiveness and cost function optimisation experiments on the GTSRB (Stallkamp et al., 2011) dataset. There are multiple reasons for choosing this dataset over the others, including the fact that it is highly accepted and is used for comparing traffic sign recognition approaches in the literature. Moreover, its authors and the organisation behind them held a public competition whereby scientists from different fields contributed with their results and tested the GSTRB dataset. Nowadays, a GTSRB website is maintained where

Fig. 1. GTSRB dataset pre-processed.

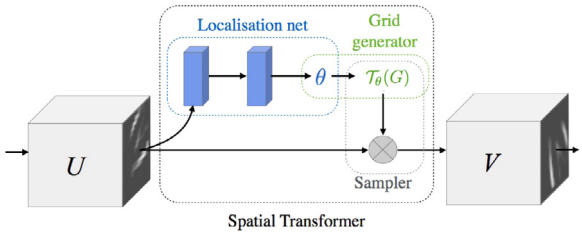


Fig. 2. Spatial transformer network components (Jaderberg et al., 2015).

submissions of results are still accepted, processed and shown in a leaderboard. Such ranking helps to find out which are the state-of-the-art methodologies utilised for the task of traffic sign classification. Last but not least, the GTSRB dataset contains traffic sign samples with different resolutions and image distortions that were extracted from 1-second video sequences. These samples each belong to one of the 43 existing classes. Its ground truth data is reliable due to its semi-automatic annotation, the training set has 39,209 images, and the validation set consists of 12,630 images, which are used to measure the performance of the algorithms. Traffic sign samples are raw RGB images whose size varies from 15×15 to 250×250 pixels.

During the pre-processing stage, all the samples are down-sampled or up-sampled to 48×48 pixels, and both global normalisation and local contrast normalisation with Gaussians kernels (Jarrett, Kavukcuoglu, Ranzato, & LeCun, 2009) are computed for the purpose of centring each input image around its mean value as well as for the enhancement of the edges (Fig. 1).

3.2. Convolutional neural network architecture

Inspired by the approach by Cireşan et al. (2012), the proposed method for the recognition of traffic signs is a single CNN that combines several types of layers: convolutional, spatial transformer (Jaderberg, Simonyan, Zisserman, et al., 2015), Rectified Linear Units (ReLU) (Nair & Hinton, 2010), local contrast normalisation (Jarrett et al., 2009) and max-pooling (Scherer, Müller, & Behnke, 2010). These layers act as a feature extractor that maps raw pixel information of the input image to a tensor which is classified later into a particular traffic sign category by two fully connected layers. All variable parameters of these layers are optimised together through the minimisation of the misclassification error over the GTSRB training set.

The convolutional layers carry out a 2-dimensional convolution of their $n-1$ input maps with a filter of size $F_x^n \times F_y^n$, where x and y represent the size of each dimension. Each convolutional layer is composed of neurons which have learnable biases and weights. During the feed-forward process of the neural network, each filter is convolved across the height and width of the input map, and a dot product is performed that produces a 2-dimensional activation map of that filter. The resulting activations of the n output maps are given by the sum of the $n-1$ convolutional responses, which are passed through a non-linear activation function f, that is computed by a ReLU layer in our case, where n is the convolutional layer, i and j represent the input map and the output map respectively, a indicates a map of size $x \times y$, the weights w_{ij} are represented as a filter of size $F_x \times F_y$ which connects the input map with the output map, and b_j is the bias of the output map (Eq. (1)).

$$a_j^n = \sum_{i=1}^{n-1} a_i^{n-1} * w_{ij}^n + b_j^n. \tag{1}$$

ReLU layers (Nair & Hinton, 2010) are made up of neurons that apply the activation function $f(x) = max(0, x)$, where x is the input to a neuron. These layers enhance the non-linear properties of the network, including the decision function, without affecting the learnable parameters of the convolutional layer.

Local contrast normalisation layers (Jarrett et al., 2009) normalise the contrast of an input map through subtractive local normalisation and divisive local normalisation. Both operations use a Gaussian kernel, and are computed in local spatial regions of the input map on a per-feature basis.

Max-pooling layers (Scherer et al., 2010) progressively reduce the spatial size of the feature maps, by directly decreasing the amount of parameters along with computation costs. Moreover, these layers control overfitting by selecting superior invariant features and generalisation is improved. The output of this layer is given by the maximum activation over non-overlapping regions of filter size $F_x \times F_y$, where the input map is downsampled by a factor of F_x and F_y along both width and height, although depth dimension remains unchanged.

Fully connected layer neurons have full connections to all activations in the previous layer and therefore they combine the outputs of the previous layer into a 1-dimensional feature vector. The last fully-connected layer of the network performs the classification task since it has one output neuron per class, followed by a logarithmic softmax activation function.

Spatial transformer units (Jaderberg et al., 2015) aim to perform a geometric transformation on an input map so that CNNs are provided with the ability to be spatially invariant to the input data in a computationally efficient manner. Thanks to such transformations, there is no need for extra training supervision, hand-crafted data augmentation (such as rotation, translation, scaling, skewing, cropping), or dataset normalisation techniques. This differentiable module can be inserted into existing CNN architectures since the parameters of the transformation that are applied to feature maps are learnt by means of a back-propagation algorithm. Spatial transformer networks consist of 3 elements: the localisation network, the grid generator and the sampler (Fig. 2).

The localisation network $f_{loc}()$ takes an input feature map $U \in R^{H \times W \times C}$, where H, W and C are the height, width and channels respectively, and outputs the parameters θ of the transformation T_θ to be applied to the feature map $\theta = f_{loc}(U)$. The dimension of θ depends on the transformation type T_θ that is being parameterised: this is 6-dimensional in our proposed network since it performs a 2D affine transformation A_θ, which allows translation, cropping, rotation, scaling, and skewing. The localisation network can comprise any number of convolutional and fully connected layers and must have at least one final regression layer with 6 output neurons in order to generate the transformation parameters θ. It should be borne in mind that this final output layer is initialised with the identity transformation matrix. Such parameters are used by the grid generator to create a sampling grid, which is a set of points where the input map has to be sampled to obtain the desired transformed output. Finally, the sampler uses the sampling grid and the input feature map U as inputs in order to perform a bilinear sampling, which produces the transformed output feature map

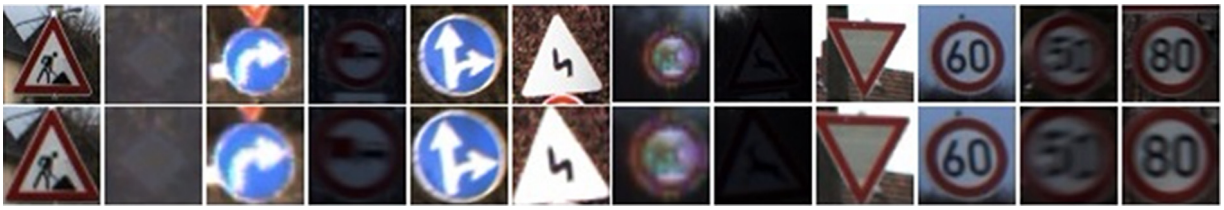


Fig. 3. Spatial transformer network. Input images above and output images below after computing affine transformations.

$V \in R^{H' \times W' \times C}$, where H', W' are the height and width of the sampling grid respectively.

For source coordinates in the input feature map (x_i^s, y_i^s) and a learnt 2D affine transformation matrix A_θ, the target coordinates of the regular grid in the output feature map (x_i^t, y_i^t) are given as follows (Eq. (2)):

$$\begin{pmatrix} x_i^s \\ y_i^s \end{pmatrix} = A_\theta \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} = \begin{bmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{bmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix}. \quad (2)$$

As regards traffic sign recognition, spatial transformer networks learn to focus on the traffic sign by gradually removing geometric noise and background so that only the interesting zones of the input are forwarded to the next layers of the network (Fig. 3). To the best of our knowledge, no peer review work has been published that has included the spatial transformer unit in a CNN for the traffic sign recognition task.

In order to measure the performance of spatial transformer layers for traffic sign recognition, we set the main CNN architecture shown in Table 1, which contains no STN. This CNN progressively increases the number of feature maps extracted from the input image through convolutional layers. At the same time, the input image's dimension is reduced by max-pooling layers and therefore the network is able to extract features on different scales. Finally, a fully-connected layer performs the classification of the traffic sign fed into the network. The stride of each convolutional layers is set to 1 in order to leave all spatial down-sampling computation to max-pooling layers, and zero-padding is set to 2. Regarding max-pooling layers, their stride is set to 2 and zero-padding to 0. Input and output feature maps of convolutional layers, as well as kernel sizes, are fixed.[1]

Due to the possibility of combining up to 3 STNs in different parts of the CNN, several network architectures were set in order to measure their influence in the final result. Note that no more than three STNs are included in the analysis since the size of output feature maps of the subsequent network's layers could not be further decreased. Progressively, spatial transformer modules are added immediately before the convolutional layers of the main network. The localisation network of the three spatial transformer layers is built with a max-pooling layer followed by two blocks of convolutional, ReLU and max-pooling, and finally, two fully-connected layers joined by a ReLU unit. The output of the last fully-connected layer consists of 6 neurons, which correspond to the parameters of the affine transformation matrix. Detailed architectures of localisation networks are drawn in Table 2. Analogous to the configuration of convolutional layers, kernel sizes and the number of input and output feature maps are fixed.

In total, there are eight different CNN architectures as a result of the possible combinations described. To denote such configurations, on one hand, c refers to a convolutional block which includes convolutional, ReLU, max-pooling and local contrast normalisation layers. On the other hand, s_i indicates the ith configuration of a spatial transformer module. For instance, a network with only one spatial transformer at the beginning is expressed as $s_1\_c\_c\_c$. Note that s_1 can only be placed before the first convolutional layer, s_2 ahead of the second convolutional unit, and s_3 preceding the third convolutional module.

[1] https://github.com/aarcosg/tsr-torch.

Table 1
Main CNN architecture without spatial transformer modules.

Layer	Type	# Maps & neurons	Kernel
0	Input	3 m. of 48 × 48 n.	
1	Convolutional	200 m. of 46 × 46 n.	7 × 7
2	ReLU	200 m. of 46 × 46 n.	
3	Max-Pooling	200 m. of 23 × 23 n.	2 × 2
4	Local Contrast Norm.	200 m. of 23 × 23 n.	
5	Convolutional	250 m. of 24 × 24 n.	4 × 4
6	ReLU	250 m. of 24 × 24 n.	
7	Max-Pooling	250 m. of 12 × 12 n.	2 × 2
8	Local Contrast Norm.	250 m. of 12 × 12 n.	
9	Convolutional	350 m. of 13 × 13 n.	4 × 4
10	ReLU	350 m. of 13 × 13 n.	
11	Max-Pooling	350 m. of 6 × 6 n.	2 × 2
12	Local Contrast Norm.	350 m. of 6 × 6 n.	
13	Fully connected	400 neurons	1 × 1
14	ReLU	400 neurons	
15	Fully connected	43 neurons	1 × 1
16	Softmax	43 neurons	

Table 2
Localisation network details of spatial transformers used in the basic CNN. Kernel size of convolutional layers is set to 5 × 5 and max-pooling layers to 2 × 2. The annotation shown in the table is simplified, for instance, 3 of 48 × 48 stand for 3 feature maps of 48 × 48 neurons each.

Layer/Type	Loc. net of ST 1	Loc. net of ST 2	Loc. net of ST 3
0/Input	3 of 48 × 48	200 of 23 × 23	250 of 12 × 12
1/Max-Pool	3 of 24 × 24	200 of 11 × 11	250 of 6 × 6
2/Conv	250 of 24 × 24	150 of 11 × 11	150 of 6 × 6
3/ReLU	250 of 24 × 24	150 of 11 × 11	150 of 6 × 6
4/Max-Pool	250 of 12 × 12	150 of 5 × 5	150 of 3 × 3
5/Conv	250 of 12 × 12	200 of 5 × 5	200 of 3 × 3
6/ReLU	250 of 12 × 12	200 of 5 × 5	200 of 3 × 3
7/Max-Pool	250 of 6 × 6	200 of 2 × 2	200 of 1 × 1
8/Fc	250 neurons	300 neurons	300 neurons
9/ReLU	250 neurons	300 neurons	300 neurons
10/Fc	6 neurons	6 neurons	6 neurons

3.3. Stochastic gradient descent optimisation algorithms

Optimisation is the process of finding the set of parameters w that minimise the loss function L. The loss function L quantifies the quality of a particular set of parameters w based on how well the inferred scores match the ground truth labels in the training data. In this work, the last layer of the CNNs proposed is a softmax classifier that uses the cross-entropy loss function (Eq. (3)), where i enumerates the different classes, y is the predicted probability distribution, and y' is the true distribution represented as a one-hot vector. The softmax function (Eq. (4)) is employed to compute y. It takes a K-dimensional vector of arbitrary real-valued scores z and squashes it to a K-dimensional vector $f(z)$ of values in the range (0, 1] that add up to 1, where j represents the jth element of

Table 3
Configuration parameters of stochastic gradient descent optimisation algorithms.

SGD w/o momentum	SGD with Nesterov
Momentum = 0 Weight decay = 0 Learning rate = 1e−2	Momentum = 0.9 Weight decay = 1e−4 Nesterov Learning rate = 1e−3
RMSprop	**Adam**
$\alpha = 0.99$ $\epsilon = 1e{-}8$ Weight decay = 0 Learning rate = 1e−5	$\beta 1 = 0.9$ $\beta 2 = 0.999$ $\epsilon = 1e{-}8$ Weight decay = 0 Learning rate = 1e−4

the vector f.

$$H_{y'}(y) = -\sum_i y'_i \log(y_i)$$
$$y_i \in (0, 1) : \sum_i y_i = 1 \forall i \tag{3}$$

$$f_j(z) = \frac{e^{z_j}}{\sum_{j=1}^{K} e^{z_k}}. \tag{4}$$

Gradient descent is the most common and established algorithm for the optimisation of the neural network's loss function. Iteratively, it computes the gradient of the objective function L with respect to the model's parameters w and then updates them. One of its variants is the mini-batch gradient descent that can be written as follows:

$$w_{k+1} = w_k - \eta_k \check{\nabla} L(w_k). \tag{5}$$

This computes the gradient $\check{\nabla} L(w_k) := \nabla L(w_k; x_k^{(i:i+n)}; y_k^{(i:i+n)})$ of the loss function L and performs an update for every mini-batch of n training examples $x^{(i)}$ and labels $y^{(i)}$, where η represents the learning rate.

In order to accelerate training, certain techniques, such as Nesterov's Accelerated Gradient method (NAG) (Nesterov, 1983), and Polyak's heavy-ball method (HB) (Polyak, 1964), have been widely used. These can be categorised as stochastic momentum methods.

Adaptive optimisation methods constitute another family of gradient descent algorithms. In contrast to non-adaptive methods, they perform local optimisation by choosing a local distance measure constructed from the history of iterates $w_1, \ldots, w_k$. Examples in this category include the Adaptive Gradient algorithm (AdaGrad) (Duchi, Hazan, & Singer, 2011), Root Mean Square Propagation (RMSprop) (Tieleman & Hinton, 2012), and Adaptive Moment Estimation (Adam) (Kingma & Ba, 2015).

In this paper, we compare the effectiveness of four mini-batch gradient descent optimisation algorithms applied to the CNNs proposed in Section 3.2: Stochastic Gradient Descent (SGD) without momentum (Qian, 1999), SGD with Nesterov's accelerated gradient, RMSprop, and Adam.

For hyper-parameter tuning, several networks were trained for several epochs in order to find an adequate initial learning rate value that reaches model convergence. We observed that a high learning rate such as 0.01 fails to work well in the cases of RMSprop and Adam, since it achieves low accuracy scores. The main reason could be that, unlike SGD where the learning rate is fixed and it can optionally follow an annealing schedule, RMSprop and Adam calculate adaptive learning rates for each model's parameter based on the history of iterates. Consequently, a lower learning rate is set for such methods in order to prevent loss values becoming stuck at bad spots in the optimisation landscape. The initial parameters of these algorithms are shown in Table 3.

Table 4
Recognition rate accuracy achieved by CNNs configurations described in Section 3.2 using different loss function optimisers: SGD without momentum (SGD), SGD with Nesterov accelerated gradient (SGD-N), Root Mean Square Propagation (RMSprop) and Adaptive Moment Estimation (Adam). c refers to convolutional block and s to spatial transformer module. Experiments were run for 15 epochs.

CNN/Optimiser	SGD	SGD-N	RMSprop	Adam	# Parameters
c_c_c	98.31	98.33	98.66	98.81	7,303,883
s_1_c_c_c	99.09	99.15	99.37	99.20	11,137,389
c_s_2_c_c	99.22	99.13	99.28	99.15	9,046,339
c_c_s_3_c	99.02	99.04	99.11	99.39	9,053,839
s_1_c_s_2_c_c	99.31	99.30	99.38	99.23	12,879,845
s_1_c_c_s_3_c	99.21	99.25	99.32	99.32	12,887,345
c_s_2_c_s_3_c	99.34	99.23	99.45	99.28	10,796,295
s_1_c_s_2_c_s_3_c	**99.49**	99.43	99.40	99.42	14,629,801

4. Results

Having described the CNN architectures and the loss function optimisers, 32 experiments were run on a computer built with an Intel Core i7-6700k CPU, 16 GB of RAM, and a Nvidia Geforce GTX 1070 discrete GPU which has 1920 CUDA cores and 8 GB of RAM, whereby the Torch scientific computer framework (Collobert, Kavukcuoglu, & Farabet, 2011) and an implementation of spatial transformer networks for Torch (Oquab, 2017) were applied as development tools. The objective is to identify the best places to add the STNs within the CNN at the same time as adding the best stochastic gradient descent optimiser. With a mini-batch size of 50, each experiment is a two-stage process that trains the neural network with the GTSRB training set and then tests it with the GTSRB validation set for 15 epochs. The results presented in Table 4 show the maximum accuracy percentage achieved by each CNN model over the validation set. The best configuration found contains three spatial transformer modules (s_1_c_s_2_c_s_3_c) and the computed loss value is optimised by means of SGD without momentum algorithm. On the other hand, the worst results are obtained by the CNN that includes no spatial transformer (c_c_c) regardless of the optimiser, and the second-worse results are given by the CNN with a spatial transformer located immediately before the last convolutional layer (c_c_s_3_c). It should be borne in mind that the winning configuration contains double the number of the model parameters of the worst CNN. As a consequence, for the SGD without momentum algorithm, the training time per epoch of the CNN with three spatial transformers is 355.05 ± 0.8 s while the CNN with no spatial transformer takes 212.12 ± 0.1 s. To sum up, the inclusion of spatial transformer units into the main CNN leads to superior classification performance, especially when they are added between at least the first layers. This improvement in performance is due to the fact that the spatial transformer scale-normalises and crops out the appropriate traffic sign region, thereby simplifying the subsequent classification task.

By considering such results and choosing the CNN s_1_c_s_2_c_s_3_c (Fig. 4), certain insights related with the comparison of the optimisation algorithms were revealed. Firstly, the solutions obtained by adaptive methods (RMSprop, Adam) generalise worse than those attained by non-adaptive methods (SGD, SGD-N). Early on in training, all four methods achieve nearly perfect training accuracy; however, during testing time, non-adaptive methods outperform adaptive methods in terms of accuracy and they display a more stable behaviour as shown in Fig. 5(b). Secondly, the adaptive methods achieve similar training loss values and lower testing loss values than non-adaptive methods. Nevertheless, their testing performance is worse, which again leads to the idea that non-adaptive algorithms generalise better than adaptive algorithms. Finally, it should be emphasised that Adam and RMSprop required the initial learning rate to be tuned, as detailed in previous section, since

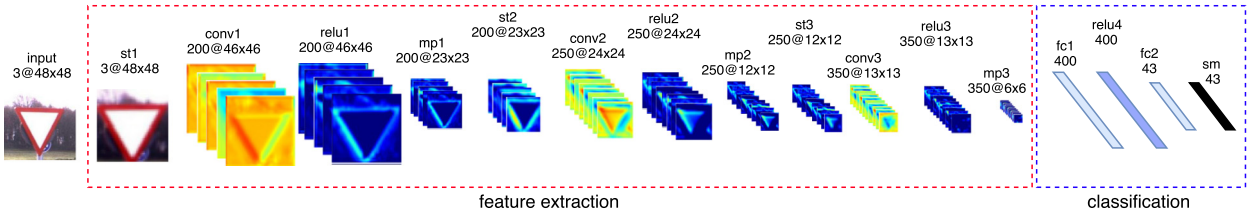
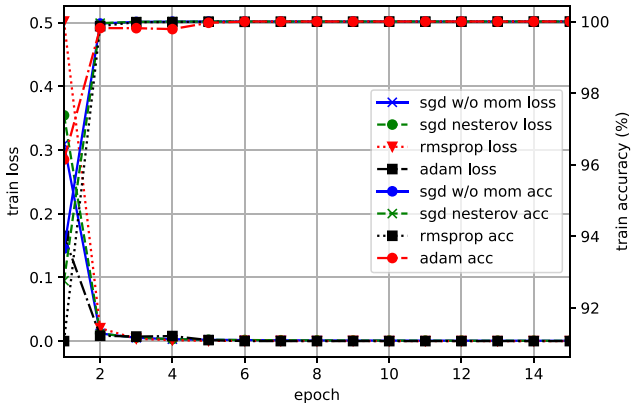
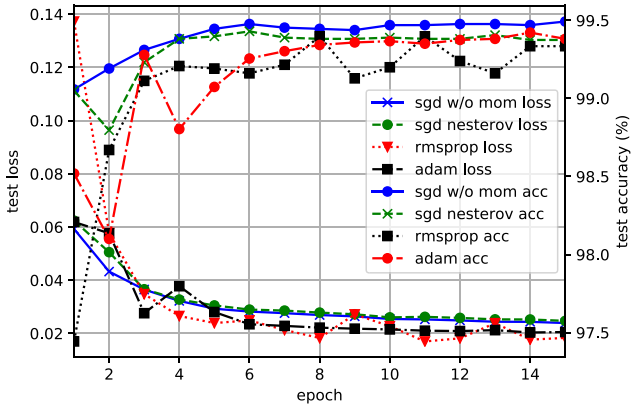


Fig. 4. CNN for traffic sign recognition. Local contrast normalisation layers and the localisation network of spatial transformers have been omitted in the figure above to simplify its visualisation. The *st* layers refer to spatial transformer networks, *conv* to convolutional layers, *mp* to max-pooling layers, *fc* to fully-connected layers, and *sm* to the softmax layer.



(a) GTSRB (Train).



(b) GTSRB (Test).

Fig. 5. Comparison of training loss and testing loss versus accuracy for four different loss function optimisers on applying the CNN model with 3 STNs $s_1\_c\_s_2\_c\_s_3\_c$.

with default settings, they achieved very low accuracy scores in comparison with those of non-adaptive methods. Although these insights should be studied in greater depth using other kinds of deep neural network architectures and datasets, they do coincide with the authors' findings and with the results of a recent research (Wilson, Roelofs, Stern, Srebro, & Recht, 2017).

Therefore, henceforth, the CNN $s_1\_c\_s_2\_c\_s_3\_c$ along with the SGD without momentum algorithm constitutes our proposed method for traffic sign classification, whose processing times for training and for testing one sample are 11.18 ± 0.02 μs and 4.28 ± 0.02 μs, respectively.

The following subsections describe the German and Belgian traffic sign datasets along with the classification results attained. The structure of each dataset is shown in Table 5 together with the overall recognition results. Note that these datasets are highly imbalanced, as can be observed in Fig. 6.

Table 5
European traffic sign classification datasets with their precision, recall and f1-score recognition results.

Dataset	Training images	Testing images	Classes
Germany	39,209	12,630	43
Belgium	4,533	2,562	62
	Precision (%)	Recall (%)	F1 score (%)
Germany	99.71	99.71	99.71
Belgium	98.95	98.87	98.86

Table 6
Recognition-rate accuracy of various methods on GTSRB.

Paper	Method	Accuracy (%)
Ours	Single CNN with 3 STNs	99.71
Jin et al. (2014)	HLSGD (20 CNNs ensemble)	99.65
Cireşan et al. (2012)	MCDNN (25 CNNs committee)	99.46
Yu et al. (2016)	GDBM	99.34
Stallkamp et al. (2011)	Human performance (best)	99.22
Jurisic et al. (2015)	OneCNN	99.11 ± 0.10

Table 7
Number of learneable parameters of our proposed CNN compared with that of previous state-of-the-art approaches.

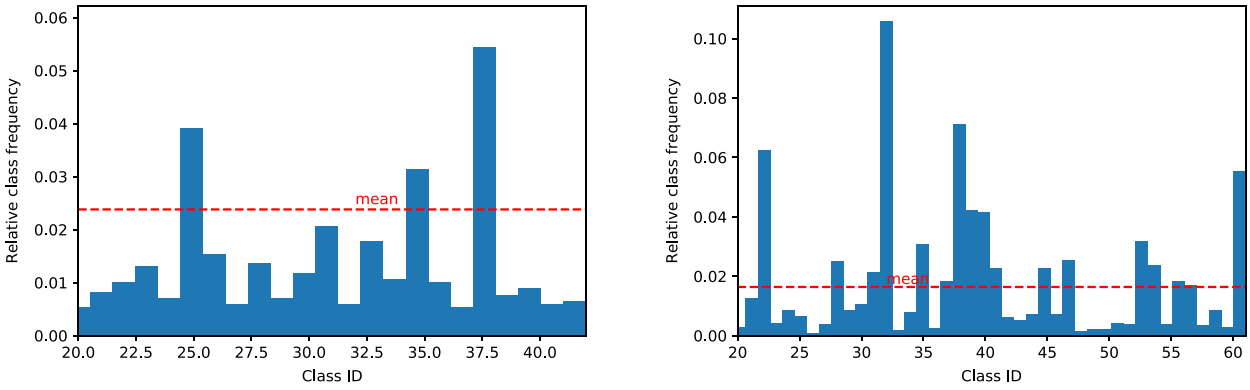
Paper	Data augment. or jittering	# trainable parameters	# ConvNets
Ours	No	14,629,801	1
Jin et al. (2014)	Yes	∼23 million	20 (ensemble)
Cireşan et al. (2012)	Yes	∼90 million	25 (committee)

4.1. GTSRB dataset results

The GSTRB dataset was introduced in Section 3.1. Our proposed CNN with three spatial transformer layers and SGD without momentum as the loss function optimiser achieves an accuracy of 99.71% at the 21st epoch (6 more than in the previous experiment). At the time of writing this paper, our method is top-1 ranked in the GTSRB and outperforms all previously published approaches (Table 6). In addition, the total number of parameters learnt by this CNN is 14,629,801, which is much lower than in other CNNs proposed for traffic sign recognition systems (Table 7), thereby leading to the further advantages of lower memory consumption, lower computational cost, and a simpler pipeline.

4.2. BTSC dataset results

The Belgian traffic sign classification dataset (BTSC) (Mathias et al., 2013) has 4533 training images and 2562 validation images split into 62 traffic sign types. In comparison with the GTSRB dataset, this dataset has different traffic sign pictograms, lighting conditions, occlusions, image resolutions, etc. Moreover, it contains categories that cluster different types of traffic signs (e.g. 50-speed-limit sign and 70-speed-limit sign), thereby raising the difficulty in the recognition task. By using the SGD without momentum loss optimiser algorithm and the CNN with three spatial



(a) Germany. (b) Belgium.

Fig. 6. European dataset category distribution.

Table 8
Recognition-rate accuracy of various methods on BTSC.

Paper	Method	Accuracy (%)
Yu et al. (2016)	GDBM	98.92
Ours	Single CNN with 3 STNs	98.87
Jurisic et al. (2015)	OneCNN	98.17 ± 0.22
Mathias et al. (2013)	INNLP+SRC(PI)	97.83

transformer layers, the model obtains an accuracy of 98.87% in the 13th epoch (Table 8).

5. Conclusions and future work

In this paper, a method for automatic fine-grained recognition of traffic signs is presented. The classification process is carried out by using a single CNN that alternates convolutional and spatial transformer modules. To find out the best CNN architecture, several empirical experiments are conducted in order to investigate the impact of multiple spatial transformer network configurations within the CNN, together with the effectiveness of four stochastic gradient descent optimisation algorithms. The CNN model outperforms all previous state-of-the-art methods and achieves a recognition rate accuracy of 99.71% in the GTSRB, and it is therefore currently top-1 ranked. Furthermore, our proposed approach needs no hand-crafted data augmentation and jittering used in prior work (Cireşan et al., 2012; Jin et al., 2014; Sermanet & LeCun, 2011). Moreover, there are fewer memory requirements and the network has a lower number of parameters to learn compared with existing methods since the use of several CNNs in a committee or in an ensemble is avoided.

Although our method is ranked in the top positions of the German and Belgian datasets, there have been several recent releases of publicly available traffic sign recognition datasets: these have not yet been tested since they are less established than previous datasets. Nevertheless, to the best of our knowledge, no other scientific paper analyses the use of several STNs and the comparison of stochastic gradient descent optimisers in the traffic sign classification problem domain. These experiments and their results can help other researchers to apply this new proposal to these new datasets.

Future work should study how to build a single deep neural network that could provide top-notch traffic sign recognition-rate accuracy in every country whose traffic sign pictographs are similar, which is the case of Europe, for which no particular dataset for any of the member countries is needed. Finally, we encourage researchers and companies to build traffic sign classifiers which are robust to those adversarial examples that could pose security concerns that may cause negative effects, such as in the use of self-driving cars, and consequently, may endanger other drivers and pedestrians alike.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness and FEDER R&D through the projects “Hermes-Smart Citizen” and “VICTORY” (Grants Nos.: TIN2013-46801-C4-1-R and TIN2017-82113-C2-1-R).

References

Barnes, N., Loy, G., & Shaw, D. (2010). The regular polygon detector. *Pattern Recognition*, *43*(3), 592–602.

Barnes, N., Zelinsky, A., & Fletcher, L. S. (2008). Real-time speed sign detection using the radial symmetry detector. *IEEE Transactions on Intelligent Transportation Systems*, *9*(2), 322–332.

Cireşan, D., Meier, U., Masci, J., & Schmidhuber, J. (2012). Multi-column deep neural network for traffic sign classification. *Neural Networks*, *32*, 333–338.

Collobert, R., Kavukcuoglu, K., & Farabet, C. (2011). Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, EPFL-CONF-192376.

De La Escalera, A., Moreno, L. E., Salichs, M. A., & Armingol, J. M. (1997). Road traffic sign detection and classification. *IEEE Transactions on Industrial Electronics*, *44*(6), 848–859.

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research (JMLR)*, *12*, 2121–2159.

Escalera, A. D. L., Moreno, L., Salichs, M., & Armingol, J. (1997). Road traffic sign detection and classification. *IEEE Transactions on Industrial Electronics*, *44*(6), 848–859.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., & Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, *88*(2), 303–338.

Gudigar, A., Chokkadi, S., Raghavendra, U., & Acharya, U. R. (2017). Local texture patterns for traffic sign recognition using higher order spectra. *Pattern Recognition Letters*, *94*, 202–210.

Huang, D.-S. (1996). *Systematic theory of neural networks for pattern recognition*. Publishing House of Electronic Industry of China, Beijing, Vol. 201.

Huang, D.-S. (1999). Radial basis probabilistic neural networks: Model and application. *International Journal of Pattern Recognition and Artificial Intelligence*, *13*(07), 1083–1101.

Huang, D.-S., & Du, J.-X. (2008). A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks. *IEEE Transactions on Neural Networks*, *19*(12), 2099–2115.

Huang, G., Liu, Z., Weinberger, K. Q., & van der Maaten, L. (2016). Densely connected convolutional networks. arXiv preprint arXiv:1608.06993.

Huval, B., Wang, T., Tandon, S., Kiske, J., Song, W., & Pazhayampallil, J. et al., (2015). An empirical evaluation of deep learning on highway driving, arXiv preprint arXiv:1504.01716.

Jaderberg, M., Simonyan, K., Zisserman, A., et al. (2015). Spatial transformer networks. In *Advances in neural information processing systems* (pp. 2017–2025).

Jarrett, K., Kavukcuoglu, K., Ranzato, M. A., & LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th international conference on computer vision* (pp. 2146–2153).

Jin, J., Fu, K., & Zhang, C. (2014). Traffic sign recognition with hinge loss trained convolutional neural networks. *IEEE Transactions on Intelligent Transportation Systems*, *15*(5), 1991–2000.

Jurisic, F., Filkovic, I., & Kalafatic, Z. (2015). Multiple-dataset traffic sign classification with OneCNN. In *2015 3rd IAPR Asian conference on pattern recognition* (pp. 614–618).

Kaplan Berkaya, S., Gunduz, H., Ozsen, O., Akinlar, C., & Gunal, S. (2016). On circular traffic sign detection and recognition. *Expert Systems with Applications*, *48*, 67–75.

Kingma, D. P., & Ba, J. L. (2015). Adam: a method for stochastic optimization. In *International conference on learning representations 2015* (pp. 1–15).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097–1105).

Larsson, F., & Felsberg, M. (2011). Using fourier descriptors and spatial models for traffic sign recognition. In *Image analysis lecture notes in computer science, Vol. 11* (pp. 238–249).

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324.

Li, J., Mei, X., Prokhorov, D., & Tao, D. (2017). Deep neural network for structural prediction and lane detection in traffic scene. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(3), 690–703.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., et al. (2014). Microsoft coco: Common objects in context. In *European conference on computer vision* (pp. 740–755). Springer.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., et al. (2016). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21–37). Springer.

Loy, G., & Barnes, N. (2004). Fast shape-based road sign detection for a driver assistance system. In *2004 IEEE/RSJ international conference on intelligent robots and systems, 2004. Proceedings. Vol. 1* (pp. 70–75).

Maldonado-Bascon, S., Lafuente-Arroyo, S., Gil-Jimenez, P., Gomez-Moreno, H., & Lopez-Ferreras, F. (2007). Road-sign detection and recognition based on support vector machines. *IEEE Transactions on Intelligent Transportation Systems*, *8*, 264–278.

Mathias, M., Timofte, R., Benenson, R., & Van Gool, L. (2013). Traffic sign recognition How far are we from the solution? In *The 2013 international joint conference on neural networks* (pp. 1–8).

Mogelmose, A., Trivedi, M. M., & Moeslund, T. B. (2012). Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. *IEEE Transactions on Intelligent Transportation Systems*, *13*(4), 1484–1497.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning* (pp. 807–814).

Nesterov, Y. (1983). A method of solving a convex programming problem with convergence rate O (1/k2). In *Soviet mathematics doklady, Vol. 27* (pp. 372–376).

Oquab, M. (2017). stnbhwd.

Polyak, B. T. (1964). Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, *4*(5), 1–17.

Qian, N. (1999). On the momentum term in gradient descent learning algorithms. *Neural Networks*, *12*(1), 145–151.

Redmon, J., & Farhadi, A. (2016). YOLO9000: better, faster, stronger. arXiv preprint arXiv:1612.08242.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91–99).

Salti, S., Petrelli, A., Tombari, F., Fioraio, N., & Di Stefano, L. (2015). Traffic sign detection via interest region extraction. *Pattern Recognition*, *48*(4), 1039–1049.

Scherer, D., Müller, A., & Behnke, S. (2010). *Lecture notes in computer science. Evaluation of pooling operations in convolutional architectures for object recognition* (pp. 92–101).

Sermanet, P., & LeCun, Y. (2011). Traffic sign recognition with multi-scale convolutional networks. In *The 2011 international joint conference on neural networks* (pp. 2809–2813).

Shadeed, W., Abu-Al-Nadi, D., & Mismar, M. (2003). Road traffic sign detection in color images. In *10th IEEE international conference on electronics, circuits and systems, 2003. Proceedings of the 2003, Vol. 2* (pp. 890–893).

Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2011). The German traffic sign recognition benchmark: A multi-class classification competition. In *The 2011 international joint conference on neural networks* (pp. 1453–1460).

Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, *32*, 323–332.

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. A. (2017). Inception-v4, Inception-resnet and the impact of residual connections on learning. In *AAAI* (pp. 4278–4284).

Tian, Y., Luo, P., Wang, X., & Tang, X. (2015). Pedestrian detection aided by deep learning semantic tasks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5079–5087).

Tieleman, T., & Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*.

Timofte, R., & Van Gool, L. (2015). Iterative nearest neighbors. *Pattern Recognition*, *48*(1), 60–72.

Timofte, R., Zimmermann, K., & Van Gool, L. (2011). Multi-view traffic sign detection, recognition, and 3D localisation. *Machine Vision and Applications*, *25*(3), 633–647.

United Nations Economic Commission for Europe (1968). Convention on road signs and signals.

Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., & Recht, B. (2017). The Marginal Value of Adaptive Gradient Methods in Machine Learning. arXiv preprint arXiv:1705.08292.

Youssef, A., Albani, D., Nardi, D., & Bloisi, D. D. (2016). Fast traffic sign recognition using color segmentation and deep convolutional networks. In *Advanced concepts for intelligent vision systems: 17th international conference, Lecce, Italy, October 24-27, 2016, proceedings* (pp. 205–216). Springer International Publishing.

Yu, Y., Li, J., Wen, C., Guan, H., Luo, H., & Wang, C. (2016). Bag-of-visual-phrases and hierarchical deep models for traffic sign detection and recognition in mobile laser scanning data. *ISPRS Journal of Photogrammetry and Remote Sensing*, *113*, 106–123.

Zaklouta, F., Stanciulescu, B., & Hamdoun, O. (2011). Traffic sign classification using K-d trees and Random Forests. In *The 2011 international joint conference on neural networks* (pp. 2151–2155).

Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, S. (2016). Traffic-sign detection and classification in the wild. In *The IEEE conference on computer vision and pattern recognition* (pp. 2110–2118).