Traffic Sign Recognition Using Extreme Learning Classifier with Deep Convolutional Features

Yujun Zeng, Xin Xu, Yuqiang Fang, Kun Zhao

College of Mechatronic Engineering and Automation
National University of Defense Technology, Changsha, China
{YujunZeng, XinXu, YuqiangFang}@nudt.edu.cn

Abstract. Traffic sign recognition is an important but challenging task, especially for automated driving and driver assistance. Its accuracy depends on two aspects: feature extractor and classifier. Current popular algorithms mainly use convolutional neural networks (CNN) to execute both feature extraction and classification. Such methods could achieve impressive results but usually on the basis of an extremely huge and complex network. What's more, since the fully-connected layers in CNN form a classical neural network classifier, which is trained by gradient descent-based implementations, the generalization ability is limited and sub-optimal. The performance could be further improved if other favorable classifiers are used and extreme learning machine (ELM) classifier is just the candidate. In this paper, ELM classifier equipped with deep convolutional features is utilized, which integrates the terrific discriminative capability of deep convolutional features learnt by CNN with the outstanding generalization performance of ELM classifier. Firstly CNN learns deep and robust features, followed by the removing of the fully-connected layers, which turns CNN to be the feature extractor. Then ELM fed with the CNN features is used as the classifier to conduct an excellent classification. Experiments on German traffic sign recognition benchmark (GTSRB) demonstrate that the proposed method can obtain competitive results with state-of-the-art algorithms with much less complexity.

Keywords: Traffic sign recognition, convolutional neural network, extreme learning machine.

1 Introduction

Traffic sign recognition is a promising subfield of object recognition with various applications, which could provide reliable safety precaution and guiding information to drivers in motorway, urban environment and the like. Nowadays, it is an indispensable opponent of the driver assistance system (DAS) and unmanned ground vehicle (UGV). Even though a lot of algorithms have been put forward, just as the German traffic sign recognition benchmark (GTSRB) [1] shows, there are still problems such as viewpoint changes, color distortion, motion blur, contrast degradation, occlusion, underexposure or overexposure, etc., which make it challenging to achieve a satisfying recognition accuracy.

Traditional methods for traffic sign recognition generally share a baseline that consists of hand-crafted features (e.g. HOG [2]) and regular classifiers (e.g. SVM [3], Random Forests [4], etc.), but Razavian Ali Sharif, et al. [5] state that hand-crafted features, which are also called shallow features, are not discriminative enough as databases become larger and larger and generic deep features should push the recognition performance even further. Convolutional neural networks (CNN) [6] is one of the deep neural networks (DNN) models and can learning robust and more discriminative features. The state-of-the-art high-rated traffic sign recognition algorithms are almost CNN-based and it works superiorly compared with the results of current best-performing methods using hand-crafted features.

There are a variety of CNN variants having been proposed. Pierre Sermanet and Yann LeCun [7] fed both the high-level and low-level features extracted by different convolutional layers to the fully-connected layers. This method combined global invariant features with the local detailed ones and the accuracy record was 99.17%. Dan Cireşan, et al. [8] preprocessed input images by contrast-limited adaptive histogram equalization (CLAHE) and boosted the CNN's performance with multilayer perceptrons (MLPs) trained with HOG feature descriptors. Its recognition accuracy reached 99.15% and was further increased to 99.46% through the ensemble of multiple CNNs [9] trained with data preprocessed with different contrast normalization methods.

Recently, more and more proofs indicate that the generalization ability of the fully-connected layers in CNN is limited and just applying CNN-learnt features to simple robust classifiers will be beneficial. Both [5] and [10] integrated the CNN-learnt representations to a linear support vector machine (SVM) classifier. After testing on many standard datasets (e.g. Oxford 102 flowers [11], Caltech-UCSD birds [12], MIT indoor scenes 67 [13], PASCAL VOC 2007 [14], etc.), all astonishingly produced the best performance. Nevertheless, cross-validation is usually inevitable so as to decide proper parameters for the training of SVM. The time cost will be unacceptable as the size of dataset grows and even that its generalization performance is not guaranteed to be optimal.

In [15] and [16], the random vector version of the functional-link (RVFL) net was justified theoretically that it could efficiently approximate continuous functions if the training set is bounded finite dimensional. As the single-hidden layer feedforward neural network (SLFN) being a special case of RVFL net, Chen, et al. [17] [18] proposed a rapid learning algorithm that treated SLFN as a linear system and formulated the training of nonlinear function approximator as the derivation of linear least-square solution. What's more, Schmidt, et al. [19] proved experimentally that the weights of most hidden neurons are not necessary to be tuned for high accuracy, which could be guaranteed if the weights of output layer are well-learnt. Motivated by the ideas aforementioned, Guang-bin Huang, et al. proposed a powerful SLFN-based learning algorithm called extreme learning machine (ELM) [20,21,22]. It randomly chose the hidden node parameters (i.e. input weights and biases) and determined the output weight by solving the smallest norm least-squares solution of SLFN. It was quite training-efficient and able to ensure an impressive classification result. In [23], ELM is trained with features learnt by stacked denoising autoencoder (SDA, another DNN model) and used to detect ships in remote sensing images. It outperformed SVM and other regular classifiers not only in detection performance but also time cost.

Meanwhile, Bernardete Ribeiro, et al. [24] and Wen-bing Huang, et al. [25] resorted to deep belief networks (DBN, also a DNN model) to build up a DBN-ELM architecture to undertake pattern recognition and attained competitive and often better results than other successful approaches, DBN-SVM included. Consequently, it is believed that ELM would bring in exciting performance when cooperating with CNN.

In this paper, a novel traffic sign recognition architecture is proposed, which is able to combine the terrific discriminative capability of deep convolutional features learnt by CNN with the outstanding generalization performance of ELM classifier. Firstly CNN is trained using raw traffic sign images. Then the fully-connected layers of the trained CNN are removed, resulting in a feature extractor that learns deep and robust features. ELM fed with the CNN features is then used as classifier to conduct an excellent classification. Experimental results clarify that the proposed method could reach comparable performance (99.40%) of the state-of-the-art approaches with less computation burden and shorter training time. The rest of this paper is organized as follows. Section 2 and 3 introduce the mechanism of CNN and ELM respectively. The proposed architecture method is described in details in section 4. Then experimental results and comparisons are shown in section 5. At last, the conclusion and future work are given in section 6.

2 Convolutional Neural Networks

CNN is one of the neural network models for deep learning, which is characterized by three specific characteristics, namely locally connected neurons, shared weight and spatial or temporal sub-sampling. Generally, CNN can be considered to be made up of two main parts (see Fig.1). The first contains alternating convolutional and maxpooling layers. The input of each layer is just the output of its previous layer. As a result, this forms a hierarchical feature extractor that maps the original input images into feature vectors. Then the extracted features vectors are classified by the second part, that is, the fully-connected layers, which is a typical feedforward neural network.

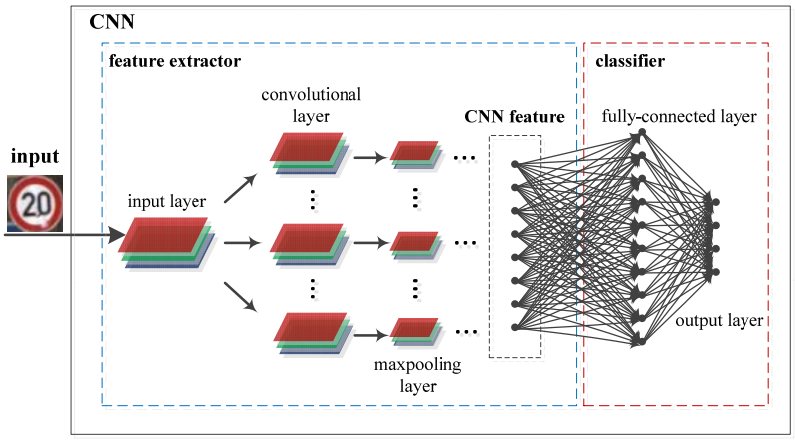

Fig. 1. CNN feature extractor architecture.

In convolutional layer, each neuron is connected locally to its inputs of the previous layer, which functions like a 2D convolution with certain filter, then its activation could be computed as the result of a nonlinear transformation (see eq.1)

$$a_{i,j} = \sigma(\mathbf{f} \otimes \mathbf{x}) = \sigma(\sum_{i'=1}^{H}\sum_{j'=1}^{W} f_{i',j'} \cdot x_{i+i',j+j'} + b) \tag{1}$$

where **f** is a $H \times W$ weight matrix of the convolutional filter, **x** refers to the activations of the input neurons connected with the neuron (i,j) in convolutional layer, $a_{i,j}$ is the corresponding activation. $\sigma(\bullet)$ is a nonlinear activation function (usually sigmoid or hyperbolic tangent), b is the bias. $\otimes$ is the convolution operator.

In maxpooling layer, the situation is similar but a maximum response operation is taken instead of convolution to the activations of the neurons in a non-overlapping rectangular region of the previous layer (see eq.2).

$$o_{i,j} = \max(\mathbf{y}) = \max_{1 \le i' \le H', 1 \le j' \le W'} y_{i+i',j+j'} \tag{2}$$

where **y** is the $H' \times W'$ neuron patch connected with the neuron (i,j) in maxpooling layer, $\max(\bullet)$ is the maximum response operator, $o_{i,j}$ is the output. Note that the maxpooling is non-overlapping and usually has a stride that equal to the size of the neuron patch, which ensures dimension reductions with the factor of H' and W' along each direction.

Different from other deep learning models, CNN is trained using back-propagation because the amount of parameters is diminished greatly due to its combination of the three architectural characteristics. Hence, all the parameters are jointly optimized through back-propagation.

3 Extreme Learning Machine

According to G.-B. Huang et al.[20], once input weights and hidden layer biases are chosen randomly, together with hidden nodes using infinitely differentiable activation functions, the SLFN classifier trained by ELM (i.e. ELM classifier) is able to possess an impressive generalization performance.

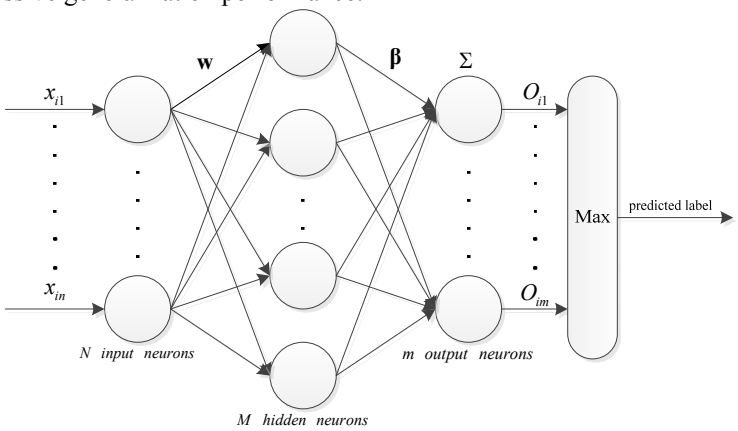

Fig. 2. ELM classifier architecture.

A standard ELM classifier (see Fig.2), whose M hidden nodes use infinitely differentiable activation functions, could approximate arbitrary samples with zero error [21], which means given a training set of instance-label pairs ($\mathbf{x}_i$, $\mathbf{l}_i$), $i=1,2,...,N$, where $\mathbf{x}_i = [x_{i1}, x_{i2},\ldots, x_{in}]^T \in \mathbf{R}^n$ and $\mathbf{l}_i = [l_{i1}, l_{i2},\ldots, l_{im}]^T \in \mathbf{R}^m$, there exist β_j, $\mathbf{w}_i$ and b_i that make eq.3 hold true.

$$\sum_{j=1}^{M} \beta_j f(\mathbf{w}_{ij} \cdot \mathbf{x}_i + b_j) = \mathbf{o}_i = \mathbf{l}_i \qquad i = 1,2,...,N \cdot \tag{3}$$

where β_j is the weight vector which connects the j th hidden node with the output nodes, $\mathbf{o}_i$ is the SLFN output vector for the i th sample, $\mathbf{l}_i$ is the label vector of i th sample, and $\mathbf{w}_{ij}$ is the weight vector connecting the i th sample and the j th hidden node, b_j is the bias of the j th hidden node. $f(\bullet)$ is the activation function.

Eq.4 is the compact version of eq.1, where $\mathbf{H}_{\mathbf{w},\mathbf{b},\mathbf{x}}$ is named the hidden layer output matrix (see eq.5).

$$\mathbf{H}_{\mathbf{w},\mathbf{b},\mathbf{x}}\boldsymbol{\beta} = \mathbf{L}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_M^T \end{bmatrix}_{M\times m}, \quad \mathbf{L} = \begin{bmatrix} \mathbf{l}_1^T \\ \vdots \\ \mathbf{l}_N^T \end{bmatrix}_{N\times m}. \tag{4}$$

$$\mathbf{H}_{\mathbf{w},\mathbf{b},\mathbf{x}} = \begin{bmatrix} f(\mathbf{w}_1 \cdot \mathbf{x}_1 + \mathbf{b}_1) & \ldots & f(\mathbf{w}_M \cdot \mathbf{x}_1 + \mathbf{b}_M) \\ \vdots & \ldots & \vdots \\ f(\mathbf{w}_1 \cdot \mathbf{x}_N + \mathbf{b}_1) & \ldots & f(\mathbf{w}_M \cdot \mathbf{x}_N + \mathbf{b}_M) \end{bmatrix}_{N\times M} \cdot \tag{5}$$

Since eq.4 represents a general linear system, the smallest training error could be achieved by computing the corresponding least-squares solution $\boldsymbol{\beta} = \mathbf{H}^{\dagger}_{\mathbf{w},\mathbf{b},\mathbf{x}}\mathbf{L}$, where $\mathbf{H}^{\dagger}_{\mathbf{w},\mathbf{b},\mathbf{x}}$ is the Moore–Penrose generalized inverse of $\mathbf{H}_{\mathbf{w},\mathbf{b},\mathbf{x}}$.

Altogether, the ELM training algorithm consists of the following three steps.

- Randomly assign hidden node parameters $\mathbf{w}_i$ and b_j, $j = 1,2,\ldots, N$.
- Calculate the hidden-layer output matrix $\mathbf{H}_{\mathbf{w},\mathbf{b},\mathbf{x}}$ and its Moore–Penrose generalized inverse $\mathbf{H}^{\dagger}_{\mathbf{w},\mathbf{b},\mathbf{x}}$.
- Calculate the output weight $\boldsymbol{\beta}$.

4 The Proposed Architecture

Current popular algorithms mainly use convolutional neural networks to execute both feature extraction and classification. Such these methods could achieve impressive results but often on the basis of an extremely huge and complex network or ensemble learning, together with over-massive data. For the purpose of making full use of the advantages of both CNN and ELM, we propose a novel traffic sign recognition architecture (see Fig.3). Before sent to CNN for feature extraction, the average image of the traffic signs is subtracted to ensure illumination invariance to some extent. In

contrast to SVM or conventional neural networks that are trained using back-propagation gradient descent, ELM could ensure a prominent classification performance, so that CNN works as the feature extractor, which means that only the convolutional and maxpooling layers are retained when the training of the whole CNN is completed. ELM is fed with features extracted by CNN and trained as the classifier. In the following subsections, more details are described.

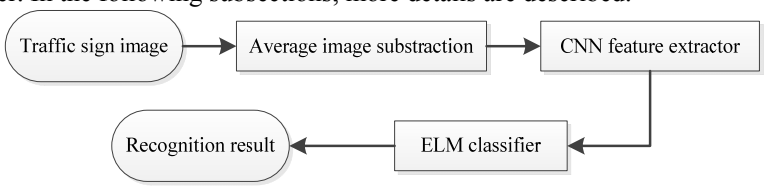

Fig. 3. Flowchart of the proposed architecture for traffic sign recognition.

4.1 Feature Extraction Using CNN Extractor

In order to indicate that the deep convolutional features learnt by plain CNN is discriminative enough for traffic sign recognition. Here we refer to the original and simple structure proposed in [8] to build up the CNN. The difference is that an extra convolutional layer with 200 feature maps of 1×1 neuron is added before the fully-connected layer, Table 1 shows the whole setting of the CNN architecture. The max pooling layer here is non-overlapping and no rectification or inner-layer normalization operation is used.

Table 1. CNN architecture parameters.

Layer	Type	Number of maps and neurons	Kernel size	Stride
1	input	1 or 3 maps of 48×48 neurons	—	—
2	convolutional	100 maps of 46×46 neurons	3×3	1
3	max pooling	100 maps of 23×23 neurons	2×2	2
4	convolutional	150 maps of 20×20 neurons	4×4	1
5	max pooling	150 maps of 10×10 neurons	2×2	2
6	convolutional	250 maps of 8×8 neurons	3×3	1
7	max pooling	250 maps of 4×4 neurons	2×2	2
8	convolutional	200 maps of 1×1 neuron	4×4	1
9	fully-connected	43 neurons	—	—

Considering that the traffic signs images are relatively invariable in shape and the size of the samples in GTSRB dataset varies from 15×15 to 250×250, here we assume that the influence coming from cropping and wrapping is considered neglectable. Thus, only images in bounding boxes given by the annotations are cropped and resized to 48×48 uniformly. Note that data augmentation is not used, which means

that random deformation (translation, rotation, scaling, etc.) is not applied to the training set.

Since CNN is used to extract deep features rather than conduct classification, the first eight layers of the CNN are taken out as a feature extractor while the fully-connected layers are removed when training is done.

4.2 Classification Using ELM with Deep Convolutional Features

The accuracy of traffic sign recognition depends on two aspects: feature extractor and classifier. The more discriminative features are and the more powerful classifier is, the higher recognition rate will be. The fully-connected layers are equivalent to a general SLFN classifier and are trained through back-propagation. For one thing, back-propagation is sensitive to local minima of training error surface. For the other, SLFN would probably be over-trained and thus gain non-ideal generalization performance when back-propagation learning is used. In other words, the generalization performance of the fully-connected layers in conventional CNN may be sub-optimal and not suited to the discriminative deep convolutional features.

Fortunately, when SLFN is trained using ELM, that is, the resulting ELM classifier will own better generalization performance, because the solution computed by $\boldsymbol{\beta} = \mathbf{H}^{\dagger}_{\mathbf{w},\mathbf{b},\mathbf{x}}\mathbf{L}$ is the smallest norm least square solution, which means that $\|\boldsymbol{\beta}\| = \left\|\mathbf{H}^{\dagger}_{\mathbf{w},\mathbf{b},\mathbf{x}}\mathbf{L}\right\| \le \left\|\hat{\boldsymbol{\beta}}\right\|$ ($\hat{\boldsymbol{\beta}}$ is an arbitrary least square solution of eq.4. Thus, it is reasonable that ELM classifier is more preferable for deep convolutional features so that the $\mathbf{H}_{\mathbf{w},\mathbf{b},\mathbf{x}}$ will be turned into $\mathbf{H}_{\mathbf{w},\mathbf{b},\mathbf{x}_c}$, where $\mathbf{x}_c$ is the feature vector extracted by CNN feature extractor, so it is with $\mathbf{H}^{\dagger}_{\mathbf{w},\mathbf{b},\mathbf{x}}$.

The only parameter to be determined is the number of hidden nodes. Since there is no instructive criterion that states how many hidden nodes should be used for a specific task like traffic sign recognition and the training of ELM classifier is extremely fast, here we decide the reasonable number experimentally without the time-consuming cross-validation.

5 Experiments and Analysis

We use the MatConvNet toolbox [26] to train the CNN feature extractor. Matlab 2014a is used to conduct all the operations, running on a system with 8 Intel(R) Xeon(R) E5-2643 CPUs (3.30GHz), 12 GB DDR4. The CNN training ends once the default training epoch (e.g. 50) is reached. Initial weights of the CNN are drawn from a uniform random distribution in the range [−0.01, 0.01]. Each neuron's activation function is sigmoid. For ELM, it is fed with CNN features and different numbers of hidden nodes are chosen and the corresponding recognition rate is recorded in Table 2.

Apparently, the recognition rate generally increases as there are more hidden nodes used. It reaches 99.23% with 4000 hidden nodes, which is already better than that of many methods. So far the highest recognition rate is 99.40% with 12000 hidden nodes.

In addition, considering that the ELM is trained with the hidden node parameters set randomly, we take five trials and the average recognition rate is adopted as the final result.

Table 2. ELM performance with different numbers of hidden nodes.

Number of hidden nodes	1000	2000	3000	4000	5000	6000
Recognition rate (avg.) [%]	98.70	99.03	99.15	99.23	99.29	99.32
Number of hidden nodes	7000	8000	9000	10000	11000	12000
Recognition rate (avg.) [%]	99.34	99.37	99.35	99.36	99.38	**99.40**

Fig.4 shows the comparison of the recognition rate between the proposed architecture method and other recently reported results. Obviously, compared with Multi-scale CNN [7], CNNaug/MLP [8], CNN-SVM and plain CNN, the proposed method performs superiorly and from this it is believed that the CNN-learnt features are discriminative enough but need a classifier with more powerful generalization ability and ELM is preferable. Even though the proposed method works a little inferior to the method of the best performance MCDNN [9] (99.46%), its complexity of model structure is much lower for the usage of single CNN and no need of data augmentation, which will relieve the training procedure a lot. In terms of training time, the proposed architecture method takes 5~6 hours, including 28 minutes for the training of ELM. It is much faster than MCDNN, which needs to train up to 25 DNNs and costs 37 hours even when the GPU parallelization is carried out.

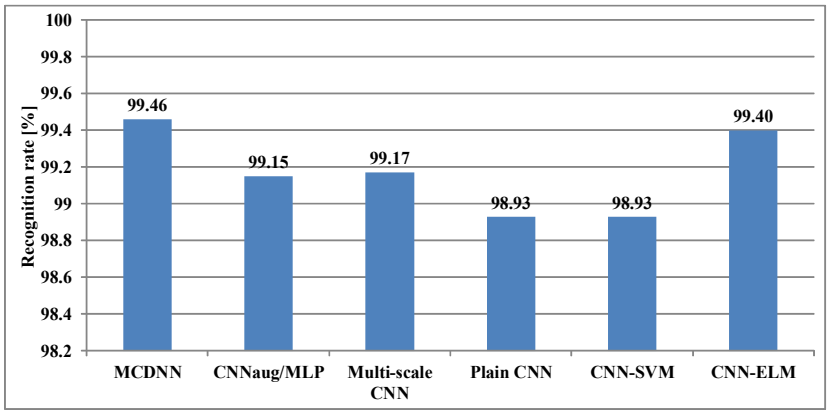

Fig. 4. Recognition accuracy comparison of MCDNN, CNNaug/MLP, Multi-scale CNN, Plain CNN, CNN-SVM, and the proposed architecture, where Plain CNN retains the fully-connected layers to implement classification and CNN-SVM is the combination of CNN feature extractor and SVM classifier.

6 Conclusion

In this paper, a novel architecture for traffic sign recognition is proposed, where CNN acts as a feature extractor and ELM trained on CNN-learnt features as the classifier,

so that the discriminative deep convolutional features could match well with the generalization performance of ELM classifier, leading to a satisfactory recognition accuracy without using more complex CNNs, ensemble features, or data augmentation. In contrast with state-of-the-art methods, the proposed method could achieve competitive results (99.40%, without any data augmentation and preprocessing like contrast normalization) with a much simpler architecture that relieves the time-consuming training procedure a lot.

Yet, the fact that the most errors are mainly due to motion blur implies that the performance may be further improved if the CNN is equipped with some layers that could learn blur-invariant features. Once new and larger traffic sign recognition databases are available, the portability of the proposed method is still to be confirmed. Additionally, the recognition accuracy might be further improved if operations like overlapping maxpooling, cross-channel normalization and ReLu [27] are taken and kernel-based ELM [28] is utilized instead of the standard one. The training time could be further cut down if a GPU implementation is accessible.

Acknowledgments. This work is supported by the National Natural Science Foundation of China under Grant 61075072, & 91220301, and the New Century Excellent Talent Plan under Grant NCET-10-0901.

References

1. J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel.: The German traffic sign recognition benchmark: a multi-class classification competition, in: Proceedings of International Joint Conference on Neural Networks, 2011, pp. 1453–1460.
2. J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel.: Man vs. computer: benchmarking machine learning algorithms for traffic sign recognition, Neural Networks 32 (August) (2012) 323–332.
3. A. Ruta, Y.M. Li, X.H. Liu.: Robust class similarity measure for traffic sign recognition, IEEE Trans. Intell. Transp. Syst. 11 (December (4)) (2010) 847–855
4. Zaklouta, Fatin, Bogdan Stanciulescu, and Omar Hamdoun.: Traffic sign classification using kd trees and random forests. Neural Networks (IJCNN), The 2011 International Joint Conference on. IEEE, 2011.
5. Razavian Ali Sharif, et al.: CNN Features off-the-shelf: an Astounding Baseline for Recognition. arXiv preprint arXiv:1403.6382 (2014).
6. LeCun, Yann, Koray Kavukcuoglu, and Clément Farabet.: Convolutional networks and applications in vision. Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on. IEEE, 2010.
7. Sermanet, Pierre, and Yann LeCun.: Traffic sign recognition with multi-scale convolutional networks. Neural Networks (IJCNN), The 2011 International Joint Conference on. IEEE, 2011.
8. Cireşan Dan, et al.: A committee of neural networks for traffic sign classification. Neural Networks (IJCNN), The 2011 International Joint Conference on. IEEE, 2011.
9. Cireşan Dan, et al.: Multi-column deep neural network for traffic sign classification. Neural Networks 32 (2012): 333-338.
10. Azizpour, Hossein, et al.: From generic to specific deep representations for visual recognition. arXiv preprint arXiv:1406.5774 (2014).

11. Nilsback, M-E., and Andrew Zisserman.: Automated flower classification over a large number of classes. Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on. IEEE, 2008.
12. C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie.: The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology (2011).
13. Quattoni, Ariadna, and Antonio Torralba.: Recognizing indoor scenes. In CVPR, 2009.
14. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman.: The PASCAL Visual Object Classes Challenge 2012(VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.
15. Igelnik, Boris, and Y-H. Pao. "Stochastic choice of basis functions in adaptive function approximation and the functional-link net." Neural Networks, IEEE Transactions on 6.6 (1995): 1320-1329.
16. Pao, Yoh-Han, Gwang-Hoon Park, and Dejan J. Sobajic. "Learning and generalization characteristics of the random vector functional-link net." Neurocomputing 6.2 (1994): 163-180.
17. Chen, CL Philip, and John Z. Wan. "A rapid learning and dynamic stepwise updating algorithm for flat neural networks and the application to time-series prediction." Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 29.1 (1999): 62-72.
18. Chen, CL Philip. "A rapid supervised learning neural network for function interpolation and approximation." Neural Networks, IEEE Transactions on 7.5 (1996): 1220-1230.
19. Schmidt, Wouter F., Martin A. Kraaijveld, and Robert PW Duin. "Feedforward neural networks with random weights." Pattern Recognition, 1992. Vol. II. Conference B: Pattern Recognition Methodology and Systems, Proceedings., 11th IAPR International Conference on. IEEE, 1992.
20. Huang, Guang-Bin, Qin-Yu Zhu, and Chee-Kheong Siew.: Extreme learning machine: a new learning scheme of feedforward neural networks. Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on. Vol. 2. IEEE, 2004.
21. Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew.: Extreme learning machine: theory and applications. Neurocomputing 70.1 (2006): 489-501.
22. Guang-Bin Huang, et al.: Extreme learning machine for regression and multiclass classification. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on 42.2 (2012): 513-529.
23. J. Tang, C. Deng, G.-B. Huang, and B. Zhao.: Compressed-Domain Ship Detection on Spaceborne Optical Image Using Deep Neural Network and Extreme Learning Machine. IEEE Transactions on Geoscience and Remote Sensing, 2014.
24. Ribeiro, Bernardete, and Noel Lopes.: Extreme Learning Classifier with Deep Concepts. Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications. Springer Berlin Heidelberg, 2013. 182-189.
25. Huang, Wen-bing, and Fu-chun Sun.: A Deep and Stable Extreme Learning Approach for Classification and Regression. Proceedings of ELM-2014 Volume 1. Springer International Publishing, 2015. 141-150.
26. MatConvNet toolbox.: https://github.com/almazan/matconvnet.git.
27. Dahl, George E., Tara N. Sainath, and Geoffrey E. Hinton.: Improving deep neural networks for LVCSR using rectified linear units and dropout. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013.
28. Huang Guang-Bin, and Chee-Kheong Siew.: Extreme learning machine with randomly assigned RBF kernels. International Journal of Information Technology 11.1 (2005): 16-24.