Learn to Rectify Label Through Kernel Extreme Learning Machine

Qiang Cai[1,2,3], Fenghai Li[1,2,3(✉)], Haisheng Li[1,2,3], Jian Cao[1,2,3], and Shanshan Li[1,2,3]

[1] School of Computer, Beijing Technology and Business University, Beijing, China
[2] National Engineering Laboratory for Agri-product Quality Traceability, Beijing Technology and Business University, Beijing, China
[3] Beijing Key Laboratory of Big Data Technology for Food Safety, Beijing Technology and Business University, Beijing, China

Abstract. Recent studies attempt to construct complicated and redundant Convolutional Neural Networks (CNNs) to improve image classification performance. In this paper, instead of painstakingly designing a CNN's architecture, we consider promoting classification performance by revising CNN's classification results. We therefore propose a novel image classification approach that Learns to Rectify Label (LRL) through Kernel Extreme Learning Machine (KELM). It includes two phases: (1) Pre classification, we put images into a trained CNN to generate corresponding incomplete labels. (2) Label Rectification, the incomplete labels are rectified by the KELM's high-dimensional mapping, so final classification results are acquired. Extensive experiments conducted on public datasets demonstrate the effectiveness of our method. At the meantime, our method has well generalizability that can be integrated with many popular networks.

Keywords: Convolutional Neural Networks · Kernel extreme learning machine · Image classification

1 Introduction

Image classification is a fundamental task in computer vision, which aims to distinguish the image categories according to their semantic information. It is widely involved in many real-world application, including face recognition [5], traffic sign detection [29] and brain image analysis [2]. An early typical approach is using the handcraft feature (e.g.SIFT [24], HOG [4]) and feature description combined with classical classifiers (e.g.SVM [3]).

Convolutional neural networks (CNNs) have exhibited strong learning capability on image classification [19]. Subsequent works [25,27] build deeper CNNs by designing small convolutional kernels. In [8,11], shortcut connections build relation of different convolutional layers and alleviate vanishing gradient problem

X. Wu et al. (Eds.): QShine 2020, LNICST 381, pp. 260–269, 2021.
https://doi.org/10.1007/978-3-030-77569-8_19

in deep networks. SKNet [21] propose a dynamic selection mechanism in CNNs that allows each neuron to adaptively adjust its receptive field size based on multiple scales of input information. Res2Net [6] represents multi-scale features at a granular level and increases the range of receptive fields for each network layer. All of above-mentioned methods try to improve CNN's design to boost classification performance, but complicated architectures and enormous parameters often lead to redundant complicated loads and poorly trained models.

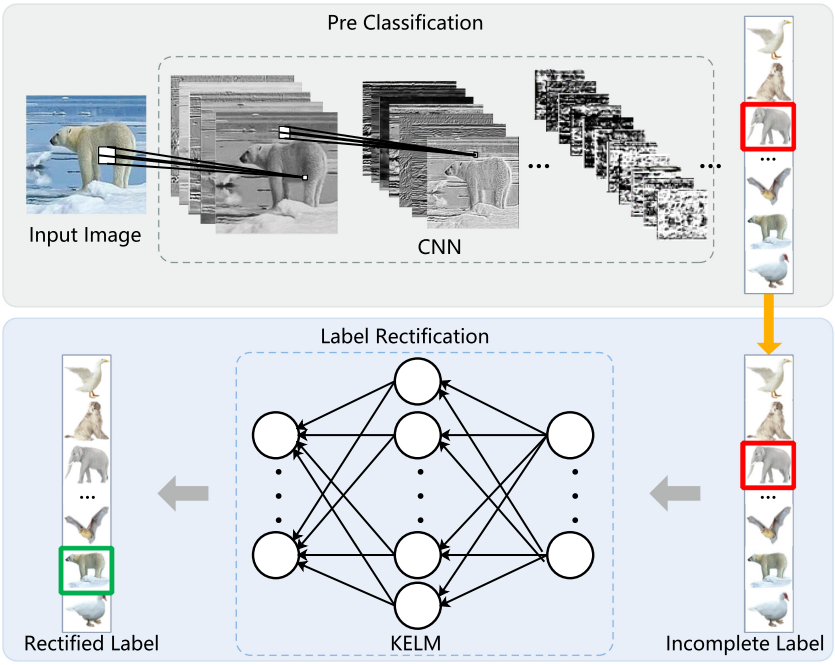

Fig. 1. Framework of the proposed approach.

On the other hand, some studies view CNNs as two main components, i.e. a feature extractor and a Softmax classifier. They consider improving discriminative ability of features by replacing Softmax with other machine learning algorithms [1,22]. e.g. Support Vector Machine (SVM) or Extreme Learning Machine (ELM). However, extracted high-dimensional features in matrix manipulation is time-consuming, which still remains a challenge.

In this paper, we consider a different view that tries to rectify labels output by CNN to a more correct distribution. We propose to learn to rectify label (LRL) through kernel extreme learning machine (KELM). Figure 1 illustrates our framework schematically. It involves two stages, i.e. pre classification and label rectification. In pre classification, we put images into a trained CNN and get corresponding classification results called incomplete labels. These labels may have large deviations with their ground truth. In label rectification, we aim

to exploit label-wise relation, so the incomplete labels are fed into a KELM. By random kernel mapping and linear combination, we can get final classification results. Notably, compared with KSVM, KELM is more appropriate for mutil-class classification and it is higher efficiency [13].

In summary, the main contributions of this paper can be concluded as follows:

- To the best of our knowledge, it is the first time that a label rectification method is proposed for image classification.
- We present a novel image classification framework (LRL) that combines CNN with KELM, and it has well generalizability for different CNN's architecture.
- Our experiments on public dataset also demonstrate our superiority on image classification task.

The rest of our paper is organized as follows: In Sect. 2, we introduce the pre classification. In Sect. 3, we introduce the way label rectification briefly. In Sect. 4, we show the experiments result. In Sect. 5, we draw a conclusion of this paper.

2 Pre Classification

CNNs have achieved a significant success in image classification. And it is widely believed that a CNN contains two components: a feature extractor and a Softmax classifier. The feature extractor can be constructed deeply, so as to obtain strong representative capability for input images. Most of existing efforts try to improve the architecture of feature extractor (increasing depth [25], multi-scale kernel size [27] and attention mechanism [26] etc.) for learning a more complicated mapping. Some methods notice the limitation of Softmax classifier in nonlinear conditions. So they substitute it with other machine learning models (SVM [1], ELM [22] etc.). However, all of these methods ignore to exploit predicted label information which is generally regarded as final classification results of CNNs. Moreover, according to existing observations, CNNs are sensitive for hyper-parameters, so well training a network becomes hard, and it lacks adaptiveness in real-world application.

Above-mentioned illustrations motivate us to capture label information output by CNN to boost classification performance. This paper denotes the labels as incomplete labels because we conjecture these labels still have potential to be improved. In our method, we consequently extract labels of CNN instead of features. It can be formulated as Formula (1):

$$L = F_\theta(I) \tag{1}$$

where F is a well-trained CNN, I is the input image, L is extracted labels, θ is the parameters of CNN. Compared with features in fully connected layers [22], extracted labels contain predicted scores of each image category, and they are relatively low dimension (It depends on the number of all categories), so it is more efficient in subsequent operations. In addition, for a pre-trained CNN, the proposed method can drastically improve their performance according to our experiments. Notably, we do not use Softmax function to normalize the extracted labels in our method.

3 Label Rectification

Given incomplete labels L, we aim to revise them by a model f. For efficiently training f, we utilize ELM to rectify incomplete labels. ELM was proposed by Huang [15], it's a single-hidden layer feedforward neural networks (SLFNs) which randomly chooses hidden nodes and analytically determines the output weight of SLFNs. ELM tends to provide good generalization performance at extremely fast learning speed and the hidden layer need not be tuned. ELM consists of three layers: input layer, hidden layer and output layer. The structure of it is shown in Fig. 2.

Consider the incomplete labels L is passed through an ELM network, the hidden layer can map it to large dimensionality that increase the the universal infinite approximation ability of the ELM. The output function of ELM for generalized SLFNs can be described as Formula (2):

$$f_l(L) = \sum_{i=1}^{l} \beta_i h_i(L) = h(L)\beta \tag{2}$$

where $\beta = [\beta_1, ..., \beta_l]^T$ is output weights vector of hidden layer $h(L) = [h_1(L), ..., h_l(L)]$, and $h_i(L)$ is the output of the ith hidden node output, $h(L)$ maps the d-dimensional label L to the l-dimensional hidden layer feature. And the output functions of hidden nodes may not be unique. Different output functions may be used in different hidden neurons. In real applications, $h_i(L)$ can be defined as:

$$h_i(L) = G(a_i, b_i, L), a_i \in R^d, b_i \in R \tag{3}$$

where $G(a, b, L)$ is a nonlinear piecewise continuous function satisfying ELM universal approximation capability theorems and $(a_i, b_i)_{i=1}^{L}$ are randomly generated according to any continuous probability distribution. In our algorithm, we use the Sigmoid (4) which is used in feed forward networks and Gaussian kernel (5) function [14] which is used in RBF networks. In this paper, the sigmoid function is applied in ELM, and the Gaussian kernel is applied in KELM.

$$G(a, b, L) = \frac{1}{1 + exp(-(\alpha L + b))} \tag{4}$$

$$G(a, b, L) = exp(-b\|L - a\|^2) \tag{5}$$

ELM is to minimize the training error as well as the norm of the output weights. The ELM learning function use the minimal norm least square method, it can be formulated as:

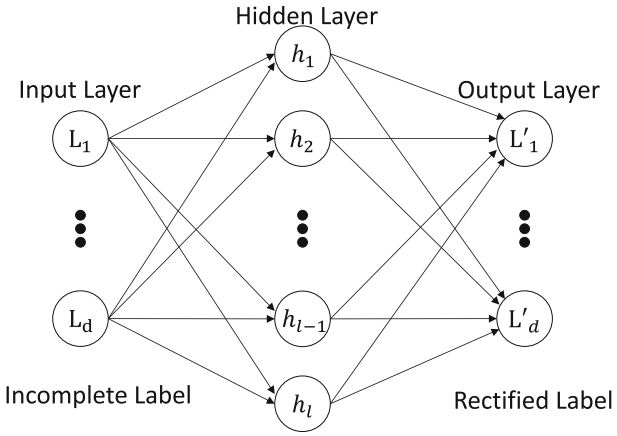

Fig. 2. Architecture of the ELM.

$$\arg\min_{\beta} \quad \|H\beta - Y\|. \tag{6}$$

where H is the hidden layer output matrix, Y is training data target matrix.

The minimal norm least square method was used in ELM, it can provide the solution of β:

$$\hat{\beta} = H^{\dagger}Y \tag{7}$$

where $H^{\dagger}$ is the Moore–Penrose generalized inverse of matrix H. The orthogonal projection method can be used to calculate $H\dagger$: when $H^T H$ is nonsingular, $H^{\dagger}$ can be defined as:

$$H^{\dagger} = (H^T H)^{-1} H^T \tag{8}$$

or when HH^T is nonsingular, $H^{\dagger}$ can be defined as:

$$H^{\dagger} = H^T (HH^T)^{-1} \tag{9}$$

As far as we get the value of $\hat{\beta}$, we can pass through the incomplete labels L_{test} to well-trained ELM or KELM, and the rectified label feature matrix L' can be defined as:

$$L' = h(L_{test})\hat{\beta} \tag{10}$$

4 Experiments

4.1 Label Rectified Evaluation

To evaluate the effectiveness of the proposed, we conduct experiments on CIFAR-10 and CIFAR-100 [18]. Those two datasets are widely used in image classification benchmark and consist of colored natural scene images. The size of all images

is 32×32 pixel. The training and test sets contain 50k and 10k images respectively. For adequately evaluating our method, we utilize two classical CNNs, i.e. VGG19 [25] and ResNet50 [8]. We set two baselines [12,23] for comparisons. Table 1 shows our experimental results.

LRL can obviously promote CNN's classification performance. On CIFAR-10, ResNet50 achieves 90.79% accuracy, while ResNet50-LRL (KELM) achieves 92.28% accuracy. On CIFAR-100, VGG19-LRL (KELM) outperforms VGG19 by 1.39% accuracy. From these comparisons, we can conclude the proposed method can effectively rectify incomplete labels output by CNNs. Compared with LRL (ELM), LRL (KELM) has more potential for image classification task, and it demonstrates RBF kernel is more appropriate to exploit label relations.

Table 1. Accuracy (%) on the CIFAR-10 and CIFAR-100 datasets.

Model	CIFAR-10	CIFAR-100
Gao et al. [12]	88.34	62.20
ResNet [9]	89.44	–
FractalNet [20]	89.92	64.66
Network in network [23]	89.59	64.32
VGG19	90.49	63.83
VGG19-LRL (ELM)	90.64	65.01
VGG19-LRL (KELM)	90.86	65.22
ResNet50	90.79	68.52
ResNet50-LRL (ELM)	92.11	71.44
ResNet50-LRL (KELM)	**92.28**	**72.27**

4.2 Compare with the State-of-the-arts

In this section, we compare our method with sate-of-the-art methods. we run experiments on the Caltech-256 dataset [7]. It contains more classes and less samples than CIFAR, with 256 classes and total of 30607 images. Each category has a minimum of 80 images. Following [22], we utilize 60 randomly selected images from per class as the training dataset, the rest as the testing dataset. We resize all images to 256×256 pixel. VGG19 [25], ResNet18 [8], Zhu et al. [28], SqueezeNet [16], VGG19-BN [17], Inception-V3 [10] are introduced for comparison, and we finally adopt ResNet50 for pre classification. Table 2 shows our experimental results.

Notably, ResNet50-LRL (KELM) also works well on Caltech-256 and outperforms all other state-of-the-art approaches. ResNet50 achieves to 80.40% Top-1 and 92.95% Top-5 accuracy. Compared with Zhu.et al. [28], our method outperforms it by 11.96% Top-1 and 2.4% Top-5 accuracy, ResNet50-LRL (KELM) outperforms base model by 1.46% Top-1 and 0.32% Top-5 accuracy. Those experimental results demonstrate the superiority of our approach on image classification.

Table 2. Accuracy (%) on the Caltech-256 dataset compared with other state-of-the-art methods.

Model	Top-1	Top-5
VGG19 [25]	70.54	87.81
ResNet18 [8]	73.22	86.96
SqueezeNet [16]	66.00	85.00
Zhu et al. [28]	70.00	89.00
Inception-V3 (with WCD) [10]	80.61	–
VGG19-BN [17]	74.83	89.85
ResNet50 [8]	80.40	92.95
ResNet50-LRL (KELM)	**81.96**	**93.27**

4.3 Classifier Comparisons

In our method, we adopt KELM to rectify incomplete labels. In order to evaluate it, we compare it with other different classifiers. Linear SVM, Kernel SVM (KSVM), Neural Network (NN) and Sigmoid function ELM (ELM). Besides, we introduce a method of Li et al. [22], which utilizes the KELM replace the Softmax classifier, that may cause two problem. First, features may contain more redundant information than labels. Second, the dimension of features is larger than labels, which consumes more times for training. In our method, we extract the labels whose dimension equal the images classes. We train the ResNet50 as pre classification. Table 3 exhibits our experimental results.

Apparently, Li et al. and LRL (KELM) are efficient in improving the performance of image classification. But other classifiers provide negative effect on improving accuracy. Li et al. can improve the accuracy of base model about 1.12%, and LRL (KELM) can improve about 1.07%. But, in contrast, experiment shows that the LRL (KELM) has advantage of high training speed. In conclusion, the LRL (KELM) achieve better balance between accuracy and training time.

4.4 Training Observation

In our training phase, we apply a pre-trained network (on ImageNet) into specific datasets (e.g. Caltech-256), so we can obtain a well-trained CNN for pre classification. But badly trained CNNs should be considered. So we apply LRL method into each ResNet50's training epoch. We plot the Top-1 accuracy of ResNet50, ResNet50-LRL (ELM) and ResNet50-LRL (KELM). The visual comparisons are provided in Fig. 3.

Obviously, KELM based LRL (green line) can substantially promote ResNet50 (blue line) performance in very early epochs. It manifests our method is effective for unwell-trained CNNs. Although the disparity of them becomes small, KELM based LRL still outperforms ResNet50. Besides, we notice the

Table 3. Accuracy (%) and training times on the Caltech-256 with different methods.

Classifier	Promotion (%)	Time (s)
Li et al. [22]	+1.12	18.02
Linear SVM	−5.06	16.26
KSVM	−3.03	47.07
NN	−5.70	120.31
ELM	−2.06	2.81
KELM	+1.07	4.58

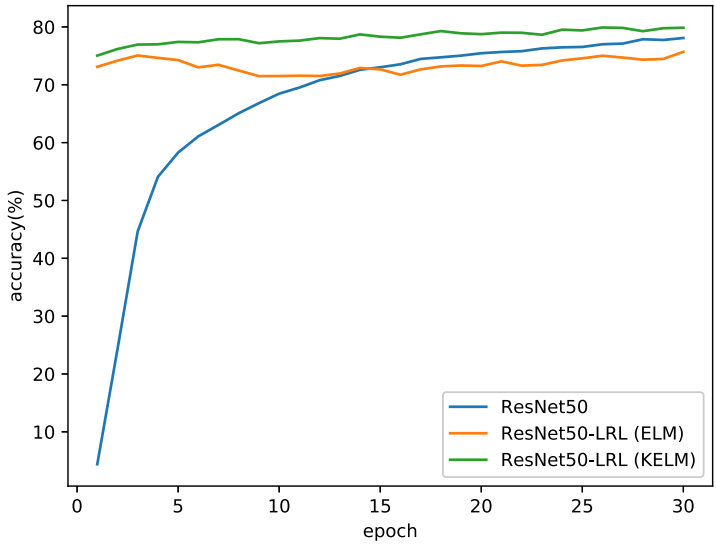

Fig. 3. Training observation of ResNet50 on Caltech-256 dataset.

ELM based LRL (orange line) cannot improve ResNet50 performance after the 13-th epoch.

4.5 Implementation Details

For Caltech-256 dataset, we modify the last layer of VGG19 and ResNet50 to 257 outputs. These CNNs are trained using a batch size of 32 for 50 epochs and the learning rate is set to 10^{-4}. For the CIFAR-10 and CIFAR-100 datasets, we replaced the fully-connected layer of ResNet50 to 11 and 101 outputs. ResNet50 are trained using a batch size of 128 for 300 epochs and the initial learning is set to 0.001 and is divided by 10 at 30 % and 75 % of the total number of training epochs. All CNNs are trained using stochastic gradient descent. And we use a weight decay of 0.01 and Nesterov moumentum of 0.9. In all experiments, the images are randomly flipped and cropped before passing into the networks. In

all our simulations on ELM with Sigmoid additive hidden node and RBF hidden node, $l = 1000$. All the hidden node parameters are randomly generated based on uniform distribution. Experiments are performed on a NVIDIA Titan Xp GPU.

5 Conclusion

In this paper, instead of designing complicated and redundant CNNs, we explore the label-wise relation for label rectification, then propose a method (LRL) to learn label rectify through kernel extreme learning machine to improve accuracy of image classification. Compared with features, labels contain less redundant information and dimension is smaller. LRL can achieve similar accuracy but save more times for training. In addition, by training observation, we found that LRL shows strong advantages in rectifying the labels of pre-trained models. Extensive experiments conducted on public datasets demonstrate the effectiveness of LRL (KELM). To our best knowledge, this is the first labels rectification approach for image classification. In future, we will exploit a method to learn label rectify from source dataset to target dataset directly.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (No. 61877002), Beijing Municipal Commission of Education PXM2019_014213_000007, Beijing Natural Science Foundation, Fengtai Rail Transit Frontier Research Joint Fund 19L00005, and Postgraduate Research Capacity Improvement Program from Beijing Technology and Business University in 2020.

References

1. Agarap, A.F.: An architecture combining convolutional neural network (CNN) and support vector machine (SVM) for image classification. arXiv preprint arXiv:1712.03541 (2017)
2. Bernal, J., et al.: Deep convolutional neural networks for brain image analysis on magnetic resonance imaging: a review. Artif. Intell. Med. **95**, 64–81 (2019)
3. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
4. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection (2005)
5. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: Arcface: additive angular margin loss for deep face recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4690–4699 (2019)
6. Gao, S.H., Cheng, M.M., Zhao, K., Zhang, X.Y., Yang, M.H., Torr, P.: Res2net: a new multi-scale backbone architecture. arXiv preprint arXiv:1904.01169 (2019)
7. Griffin, G., Holub, A., Perona, P.: Caltech-256 object category dataset (2007)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
9. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 630–645. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_38

10. Hou, S., Wang, Z.: Weighted channel dropout for regularization of deep convolutional neural network. In: AAAI (2019)
11. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2017)
12. Huang, G., Sun, Yu., Liu, Z., Sedra, D., Weinberger, K.Q.: Deep networks with stochastic depth. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 646–661. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_39
13. Huang, G.B.: An insight into extreme learning machines: random neurons, random features and kernels. Cogn. Comput. **6**(3), 376–390 (2014)
14. Huang, G.B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst. Man Cybern. Part B (Cybern.) **42**(2), 513–529 (2011)
15. Huang, G.B., Zhu, Q.Y., Siew, C.K.: Extreme learning machine: theory and applications. Neurocomputing **70**(1–3), 489–501 (2006)
16. Iandola, F.N., Han, S., Moskewicz, M.W., Ashraf, K., Dally, W.J., Keutzer, K.: Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5 mb model size. arXiv preprint arXiv:1602.07360 (2016)
17. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
18. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images. Technical Report, Citeseer (2009)
19. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, pp. 1097–1105 (2012)
20. Larsson, G., Maire, M., Shakhnarovich, G.: Fractalnet: ultra-deep neural networks without residuals. arXiv preprint arXiv:1605.07648 (2016)
21. Li, X., Wang, W., Hu, X., Yang, J.: Selective kernel networks (2019)
22. Li, Z., Zhu, X., Wang, L., Guo, P.: Image classification using convolutional neural networks and kernel extreme learning machines. In: 2018 25th IEEE International Conference on Image Processing (ICIP), pp. 3009–3013. IEEE (2018)
23. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint arXiv:1312.4400 (2013)
24. Lowe, D.G., et al.: Object recognition from local scale-invariant features. In: ICCV, vol. 99, pp. 1150–1157 (1999)
25. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
26. Sutskever, I., Vinyals, O., Le, Q.: Sequence to sequence learning with neural networks. In: Advances in NIPS (2014)
27. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
28. Zhu, C., Han, S., Mao, H., Dally, W.J.: Trained ternary quantization. arXiv preprint arXiv:1612.01064 (2016)
29. Zhu, Y., Zhang, C., Zhou, D., Wang, X., Bai, X., Liu, W.: Traffic sign detection and recognition using fully convolutional network guided proposals. Neurocomputing **214**, 758–766 (2016)