Annals of Telecommunications (2020) 75:369–379
https://doi.org/10.1007/s12243-019-00731-9

Lightweight deep network for traffic sign classification

Jianming Zhang [1,2] · Wei Wang [1,2] · Chaoquan Lu [1,2] · Jin Wang [1,2] · Arun Kumar Sangaiah [3]

Received: 15 December 2018 / Accepted: 18 July 2019 / Published online: 31 July 2019

Abstract
Deeper neural networks have achieved great results in the field of computer vision and have been successfully applied to tasks such as traffic sign recognition. However, as traffic sign recognition systems are often deployed in resource-constrained environments, it is critical for the network design to be slim and accurate in these instances. Accordingly, in this paper, we propose two novel lightweight networks that can obtain higher recognition precision while preserving less trainable parameters in the models. Knowledge distillation transfers the knowledge in a trained model, called the teacher network, to a smaller model, called the student network. Moreover, to improve the accuracy of traffic sign recognition, we also implement a new module in our teacher network that combines two streams of feature channels with dense connectivity. To enable easy deployment on mobile devices, our student network is a simple end-to-end architecture containing five convolutional layers and a fully connected layer. Furthermore, by referring to the values of batch normalization (BN) scaling factors towards zero to identify insignificant channels, we prune redundant channels from the student network, yielding a compact model with accuracy comparable to that of more complex models. Our teacher network exhibited an accuracy rate of 93.16% when trained and tested on the CIFAR-10 general dataset. Using the knowledge of our teacher network, we train the student network on the GTSRB and BTSC traffic sign datasets. Thus, our student model uses only 0.8 million parameters while still achieving accuracy of 99.61% and 99.13% respectively on both datasets. All experimental results show that our lightweight networks can be useful when deploying deep convolutional neural networks (CNNs) on mobile embedded devices.

Keywords Convolutional neural networks · Traffic sign classification · Knowledge distillation · Network pruning

1 Introduction

Traffic signs use characters, symbols, and colors to convey mandatory, prohibitory, and danger information. The recognition of traffic sign is an indispensable component of autonomous vehicles and advanced driver assistance systems (ADAS). Traffic sign recognition has high requirements as concern real-time processing, accuracy, and robustness; moreover, energy-efficient processing is also important in a mobile computing environment [1]. Traffic sign images or videos in natural scenes are collected by the camera installed on the vehicle, then inputted into the vehicle computer. The semantics of signs can be understood through types of processing including detection, location, tracking, classification, and so on. However, no real-time, accurate, and adaptable system has been introduced so far due to the existence of several challenges, such as the complex and diverse backgrounds in real scenes, different national traffic sign standards, illumination variations, diversified shooting angles, and real-time requirements.

Deep learning [2], a hot topic in machine learning methods of late, has been successfully applied to tasks including handwritten numeral recognition [3], classification [4, 5], detection [6], tracking [7–9], natural language processing [10], and intelligent question and answer systems [11] and has achieved accomplishments that go beyond the reach of traditional methods. The success of deep learning benefits not only from larger and deeper models with more parameters but also from large-scale annotated or unlabeled data provided by academia and industry. More specifically, the large model structure enhances the nonlinearity of deep learning, while the huge amount of training data enhances its generalizability.

✉ Arun Kumar Sangaiah
arunkumarsangaiah@gmail.com

[1] School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, Hunan Province, China

[2] Hunan Provincial Key Laboratory of Intelligent Processing of Big Data on Transportation, Changsha University of Science and Technology, Changsha 410114, China

[3] School of Computer Science and Engineering, Vellore Institute of Technology, Vellore 632014, India

Traditional traffic sign recognition methods typically use extreme learning machine (ELM) [12] or support vector machine (SVM) [13, 14] methods for feature classification, and use handcrafted features which may result in the loss of significant information. The recognition rate of traditional methods will decline when traffic signs are occluded or shaded. Recently, convolutional neural networks have been used for traffic sign recognition. MSCNN [15] extracts the features from different convolutional layers for traffic sign classification and achieves a recognition rate superior to that of traditional methods. MCDNN [16] proposes a multi-column network to classify traffic signs; the recognition ability of this network was improved through the application of expert voting. Moreover, after extracting features from CNNs, CNN-ELM [17] uses ELM as a classifier, combining the advantages of deep learning and traditional machine learning. CNN-HLSGD [18] trains a convolutional neural network with hinge loss, achieving a recognition rate on the GTSRB dataset better than that of most methods. Furthermore, in [19], the authors propose a novel approach called DP-KELM, which classifies the deep perceptual features using a kernel-based extreme learning machine (KELM) in the perceptual LAB color space, a method that can reduce the model's computational cost and yield an improved recognition rate.

Although deep neural networks perform well in traffic sign recognition experiments, they are still restricted by time and space in practical applications. As larger and deeper networks require more resources, graphics processing units (GPUs) [20] are commonly used to help speed up computation. The strong representation ability of convolutional neural networks arises as a result of their millions of trainable parameters; for example, AlexNet [21] has 60 M parameters, while VGG16 [22] has 138 M parameters. However, the reduced computing power and storage space of on-board equipment or wearable devices cannot support the operational resources required by these complex networks. As reported in [23], the parameters in CNNs exhibit a great deal of redundancy. As a result, many methods have been proposed to compress large CNNs. Deep network compression primarily encompasses five kind of methods: low-rank, pruning, quantization, knowledge distillation, and compact network design. Low-rank decomposition methods [24] such as singular value decomposition (SVD) or tensor train decomposition use a low-rank matrix to approximate a weight matrix in CNNs. Channel pruning methods refer to pruning those unimportant channels [25], making the selection of the correct pruning criterion a highly essential aspect of these methods. Quantization reduces model size by means of low-bit representation; for example, by quantizing weights into ternary values [26]. It is significant to use deep neural networks to identify traffic signs in real time with limited equipment resources. Therefore, in the present paper, we design two slim networks with fewer trainable parameters that do not require special software or hardware accelerators.

The main contributions of this paper can be summarized as follows:

1. To alleviate the abovementioned problems, a new training strategy and two lightweight convolutional neural networks are proposed. These two networks work as a teacher model and a student model respectively. We design a new module in our teacher network that combines two streams of feature channels with dense connectivity to make the network deeper. Our student network is a simple, end-to-end architecture with five convolutional layers and a fully connected layer.
2. The teacher model can assist the training of the student model by means of knowledge distillation, while the student model can obtain a better traffic sign recognition rate than the teacher model. Finally, according to the values of BN scaling factors towards zero to identify insignificant channels, our student model is pruned to reduce the number of parameters and the computational costs.
3. Compared with some existing traffic sign classification algorithms, the proposed lightweight network has fewer parameters while still obtaining the same high recognition rate; moreover, the input data does not require extra preprocessing operations, thus enabling the implementation of a simple and efficient end-to-end network.

Our proposed lightweight networks can achieve an accuracy loss of between 0.33 and 0.63% while parameters can be reduced to one tenth or even 1% of those employed by the compared algorithms. The rest of this paper is organized as follows: In Sect. 2, our proposed method is described in detail. The performance on two traffic sign datasets is presented in Sect. 3. Finally, the conclusion is provided in Sect. 4.

2 Proposed methodology

In this paper, our work is divided into three steps. Firstly, we design a teacher network and a student network. After the teacher model converges on the traffic sign classification training set, knowledge distillation is utilized in order to improve the precision of the student model. Finally, the student model's channels are pruned, reducing the overall computational cost.

2.1 Knowledge distillation

Knowledge distillation [27] can help to train the shallower student network through the softened output of the teacher

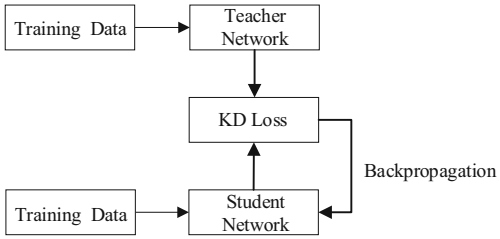


Fig. 1 The knowledge distillation procedure

network on the target datasets. The training sets are as follows: $D = \{X = \{x_1, x_2..., x_N\}, Y = \{y_1, y_2..., y_N\}\}$, where x and y represent an input and a target output respectively. The output of the teacher model is $t = \text{teacher}(x)$; likewise, the output of the student model is $s = \text{student}(x)$. We train the student model to minimize the following loss function:

$$L_{KD} = (1-\alpha)L_{CE}(y, \sigma(s)) + 2T^2\alpha L_{CE}\left(\sigma\left(\frac{t}{T}\right), \sigma\left(\frac{s}{T}\right)\right) \quad (1)$$

where T and α are hyperparameters, α controls the ratio of the two terms, T is a temperature parameter, and $\sigma()$ is the softmax function. L_{CE} denotes a standard cross-entropy loss that penalizes the student network when it classifies the target incorrectly. If there is only a tiny difference between the outputs of the teacher model and the student model, the second term is minimized.

A student network trained on softened outputs is significantly better than one that learns directly from the original training data. This is because what the model learns directly from the traffic sign training set is one-hot class label. For example, we train a network to classify a specific object as either a car, dog, or cat, [0.05, 0.9, 0.6]; this is a softened output, which is the most likely prediction of an image. Since dogs are more similar to cats than cars, the difference between the second and the third probabilities is smaller than the difference between the first and the second. In cases where the initial prediction information in [0.05, 0.9, 0.6] makes only a minimal contribution to the weights update, increasing the temperature parameter T can help transfer the knowledge to the student model.

The knowledge distillation procedure is illustrated in Fig. 1. The teacher network and the student network use the same training set. The teacher network is first trained to converge on the traffic sign training set, the parameters of which will not be updated during the training of the student network, as they only play the role of guiding the updating of the student network parameters. We thus need to find the appropriate hyperparameters T and α. When the accuracy of the teacher network on the target dataset is low, it is hard to guide the student model to update parameters; accordingly, it is easy for the student network to fall into the local optimum value. Aimed at addressing this problem, our proposed teacher network achieves an accuracy of 99.23% and 98.89% on the GTSRB and BTSC datasets respectively, meaning that it can be an effective means of helping to improve the traffic sign recognition rate of the student model.

2.2 Teacher network

Our proposed teacher network is based on an important observation result: namely, that low-level convolutional features can better represent the target's texture information, while high-level convolutional features contain more semantic information. We connect each layer by means of dense connectivity [28], while the feature maps are input to all subsequent layers. A high utilization rate of feature maps is achieved; moreover, the low-level and high-level feature maps are combined effectively, enabling improved learning of the features (Fig. 2).

We propose a novel module, as shown in Fig. 3. Our teacher network is constructed as follows: (a) two 1×1 convolutional filters are used to reduce the number of channels of the input feature maps. Compared with the 3×3 kernels, the parameters are reduced one ninth when 1×1 kernels are used. The nonlinearity of the network can be greatly increased while the size of the feature maps remains unchanged, which makes the network deeper. (b) The 1×1 kernels and the 3×3 kernels execute convolution operations in parallel and splice all output results, as shown in Fig. 4. Different convolution operations can obtain different information being obtained about the input image, while the feature maps following parallel operation exhibit a stronger feature representation ability. (c) Six cells are used to establish the direct connection between different layers, making full use of the feature maps of each layer, as well as integrating the characteristics of each channel in order to alleviate the gradient disappearance problem. At

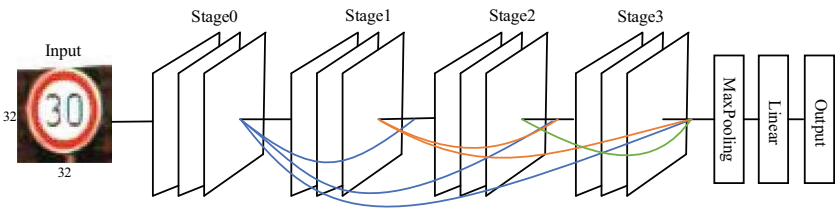
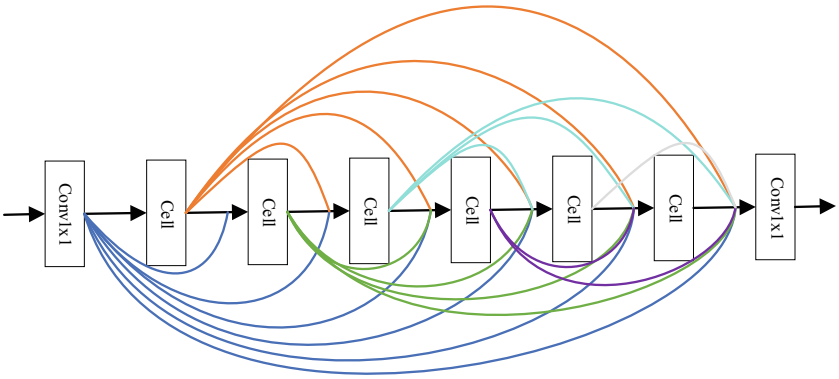


Fig. 2 The architecture of our teacher network

Fig. 3 One Stage module



the same time, the batch normalization [29] layer and ReLU [30] function are used in each layer to further prevent gradient disappearance and gradient explosion, increasing the degree of network nonlinearity.

The teacher network proposed in this paper can integrate the features between different layers, obtain more detail information (such as the texture features and the edge features), and increase the network's ability to recognize traffic signs. By using our novel module, we can stack multiple convolutional layers in order to obtain a deeper network structure. As the number of network layers increases, the representation ability of the extracted features grows stronger; finally, only a fully connected layer is used to transform the feature vector into a probability vector for traffic sign classification.

2.3 Student network

CNNs can automatically extract features in an unsupervised manner. As shown in Table 1, the student network is an end-to-end structure consisting of five convolutional layers and a fully connected layer. The input layer loads the input data, and the input images do not require the use of data augmentation. RGB three-color channels are used to retain the original information of traffic signs. Convolutional layer is used for learning features, and each type of convolutional filter corresponds to the extraction of a specific feature in the image. We add a BN layer and ReLU layer following each convolutional layer in order to increase the network nonlinearity. The pooling layer can prevent overfitting and reduce the dimensionality of feature maps, while average-pooling can better preserve the information of the input feature maps, such as the background information. Moreover, max-pooling can obtain textural features, inhibiting the attenuation of the reverse gradient. The fully connected layer can also work as a classifier to transfer the feature vectors into target class probability.

The Adam [31] algorithm is used to update the weights and bias during the training of the student network. Let m_t be the exponential moving averages of the gradient, which estimate the first moment of the gradient. v_t denotes the squared gradient, which estimates the second raw moment of the gradient. The exponential decay rates of these moving averages are controlled by the hyperparameters β_1, $\beta_2 \in [0, 1)$. Finally, g_t indicates the gradient at timestep t:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{2}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{3}$$

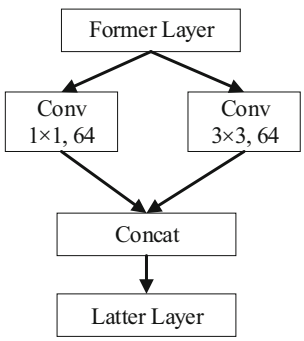


Fig. 4 One Cell block

Table 1 Description of our student network

Type	Kernel size	Stride	Output size
Input			32 × 32
Convolutional	3 × 3	1	32$_v$32
Convolutional	3 × 3	1	32 × 32
Max-pooling	2 × 2	2	16 × 16
Convolutional	3 × 3	1	16 × 16
Convolutional	3 × 3	1	16 × 16
Max-pooling	2 × 2	2	8 × 8
Convolutional	3 × 3	1	8 × 8
Average-pooling	2 × 2	2	4 × 4
Fully connected			4096
Output			43/62

Fig. 5 Sample images from the GTSRB dataset

The bias-corrected first moment estimate and the bias-corrected second raw moment estimate are computed as follows:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{4}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{5}$$

Finally, the updated formula of the parameter is as outlined below; here, η is the learning rate, while θ_0 denotes the initial parameter vector. Moreover, ϵ is set to 10^{-8} to avoid division by zero.

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{6}$$

2.4 Network pruning

There are some common methods used to convert networks into more compact networks with fewer trainable parameters. Weight quantization [26] uses low-bit weights and activations for model compression; however, these methods require a suitable data structure to be utilized in order to store quantized parameters. Low-rank [32] can decompose a convolutional layer into several efficient ones, although it is not efficient for the current network with a 1×1 convolutional layer. Network pruning is still a hot topic in the context of small accuracy drops and efficient structured pruning.

Network pruning mainly reduces the computation required by the model by removing redundant parameters in the network. The most time-consuming aspect of a convolutional neural network is the convolutional layer; moreover, the fully connected layer contains the network's main parameters. Minimizing the difference in accuracy between the full and pruned models depends on the criterion used to identify the "least important" parameters. Reasonable criteria of this kind include minimum weight, activation value, and mutual information. The method outlined in [33] prunes redundant connections by learning only the important connections. In [34], moreover, the model is pruned on the fully connected layer.

Network slimming [35] employs γ parameters in batch normalization layers as the scaling factors for channel pruning. The smaller the value, the lower the importance of the channels and the easier they are to prune. $B = \{x_1, x_2..., x_m\}$ denotes the current mini-batch:

$$\hat{x}_t = \frac{x_t - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{7}$$

Here, μ_B and σ_B are the mean and standard deviation values, Z_{out} is the output of a batch normalization layer, and γ and β are trainable affine transformation parameters, such that

$$Z_{out} = \gamma \hat{x}_t + \beta \tag{8}$$

This loss function imposes the L1-norm on the scaling factors of each channel in order to make the values of

Fig. 6 Sample images from the BTSC dataset

unimportant channels smaller. Finally, those channels with small scaling factors are pruned, after which the pruned network is retrained to resume the recognition rate accuracy on a target dataset. Minimizing the train objective loss:

$$L = \sum_{(x,y)} l(f(x, W), y) + \lambda \sum_{y \in \Gamma} g(\gamma) \tag{9}$$

Table 2 Performance comparison of our teacher model and the general model on the CIFAR-10 dataset

Model	Parameters (millions)	Convolutional layers	Accuracy
AlexNet [21]	60	5	74.74%
VGG16 [22]	138.3	13	92.64%
GoogleNet [39]	6.7	22	92.57%
ResNet110 [40]	1.7	109	93.57%
Our teacher model	7.9	32	93.16%

Table 3 Different hyperparameters for knowledge distillation

α	T	Accuracy
0.85	1	99.34%
0.85	10	99.11%
0.85	20	99.36%
0.85	30	99.44%
0.90	1	99.48%
0.90	10	99.24%
0.90	20	99.50%
0.90	30	99.35%
0.95	1	98.96%
0.95	10	98.87%
0.95	20	98.97%
0.95	30	99.39%
0.99	1	98.42%
0.99	10	98.33%
0.99	20	98.76%
0.99	30	98.71%

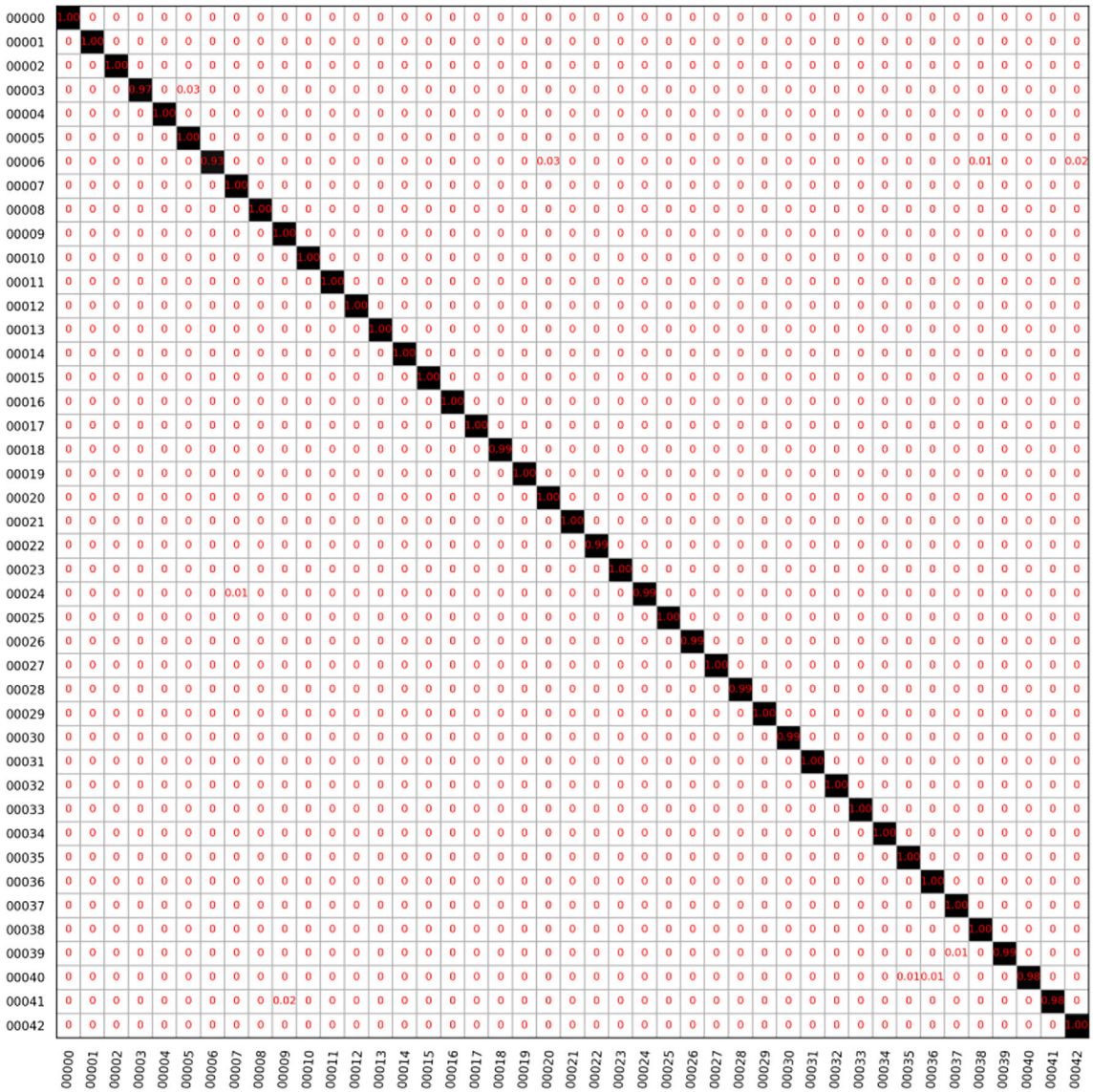


Fig. 7 CM of student network on GTSRB dataset

Here, (x, y) represents the input and target output respectively. Let W denote the trainable parameters, while g() depicts an L1 regularization on channel scaling factors; the two terms are balanced by λ.

Table 4 Performance comparison on GTSRB Dataset

Method [Referenced]	Parameters	Accuracy
Single CNN with 3 STNs [41]	1.4×10^7	99.71%
CNN-HLSGD-ensemble [18]	2.3×10^7	99.65%
Our student network	7.3×10^5	99.61%
DP-KELM [19]	1.4×10^6	99.54%
MCDNN [16]	$\sim 9 \times 10^7$	99.46%
Our teacher network	7.9×10^6	99.23%
CNN-HLSGD-single [18]	1.2×10^6	99.16%

3 Experimental results

3.1 Dataset and hyperparameters

3.1.1 Dataset

The German Traffic Sign Recognition Benchmark (GTSRB) [36] dataset is a multi-class, single-image dataset that poses a challenge in traffic sign classification tasks. The dataset consists of 51,839 samples, ranging in size from 15 × 15 to 250 × 250, and not all of them are square. This dataset has 43 categories, each of which comprises 100~1000 images including prohibitory signs, danger signs, and mandatory signs. The training set contains 39,209 images; the remaining 12,630 images are selected as the testing set. Due to perspective change, shade, color degradation, lighting conditions, and so

Table 5 Pruning 50% of channels for the student network

Layer/Type	Kernel size	Numbers (pruned)	Stride	Padding	Output size
1/Conv	3 × 3	32(−32)	1	1	32 × 32
2/Conv	3 × 3	32(−32)	1	1	32 × 32
3/Max-pool	2 × 2	1	2	0	16 × 16
4/Conv	3 × 3	64(−64)	1	1	16 × 16
5/Conv	3 × 3	64(−64)	1	1	16 × 16
6/Max-pool	2 × 2	1	2	0	8 × 8
7/Conv	3 × 3	128(−128)	1	1	8 × 8
8/Avg-pool	2 × 2	1	2	0	4 × 4

on, it can be difficult even for humans to recognize many of these signs (Fig. 5).

Moreover, there are 4533 training images and 2562 testing images in the Belgian Traffic Sign Classification dataset (BTSC) [37], which is divided into 62 traffic sign types. The images in the BTSC dataset are often distorted due to weather change, occlusions, etc. Compared with the GTSRB dataset, the BTSC dataset contains a larger number of different types but less training samples of traffic signs, which increases the difficulty associated with correct classification (Fig. 6).

3.1.2 Training teacher network

Our experiments are performed with PyTorch on a Linux PC with an Intel® Xeon(R), CPU E5-2670 v3 @ 2.30 GHz×24 and an NVIDIA TITAN X, 12 GB RAM. We first train our teacher network on a general dataset. The CIFAR-10 dataset [38] contains ten classes, each of which contains 6000 RGB images. The ten classes are as follows: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. The training set contains 50,000 images, while the testing set contains 10,000 images. We train using a batch size of 128 for 300 epochs. The learning rate is initially set to 0.001, and individual adaptive learning rates are computed using the Adam method. We use a weight decay of 10^{-5}. The converged model can achieve comparable performance with fewer trainable parameters. The results in Table 2 demonstrate the generality of our teacher network.

3.1.3 Hyperparameters for knowledge distillation

To find the appropriate hyperparameters T and α, we set out 16 kinds of hyperparameter configurations, each of which is trained for 500 epochs on the GTSRB dataset. The results are listed in Table 3. To achieve knowledge distillation, we set α to 0.9 and use a temperature of 20, which can facilitate better experimental results.

3.2 Performance on GTSRB dataset

The best recognition rate achieved by our teacher model is 99.23%, while the average recognition rate of our student model increased to 99.61%. The confusion matrix (CM) is one of the most widely used evaluation metrics. We construct our own confusion matrix in order to further analyze the effect of the proposed lightweight network, as shown in Fig. 7. CM is an evaluation criterion that can measure an algorithm's performance in a visual way. When using a CM, each row and column represent the actual categories and the predicted value respectively. Accordingly, the question of whether these multiple categories are confused or not can be intuitively seen in Fig. 7.

To demonstrate the performances of our teacher network and student network, we first evaluate the classification task

Table 6 Pruning 70% of channels for the student network

Layer/Type	Kernel size	Numbers (pruned)	Stride	Padding	Output size
1/Conv	3 × 3	21(− 43)	1	1	32 × 32
2/Conv	3 × 3	44(− 20)	1	1	32 × 32
3/Max-pool	2 × 2	1	2	0	16 × 16
4/Conv	3 × 3	54(− 74)	1	1	16 × 16
5/Conv	3 × 3	29(− 99)	1	1	16 × 16
6/Max-pool	2 × 2	1	2	0	8 × 8
7/Conv	3 × 3	43(− 213)	1	1	8 × 8
8/Avg-pool	2 × 2	1	2	0	4 × 4

Table 7 Performance comparison of the original and pruned student models on the GTSRB dataset

Model	Trainable parameters	Accuracy
Our student model	732,139	99.61%
Student model (50% pruned)	227,851	99.38%
Student model (70% pruned)	85,593	99.08%

on the GTSRB testing images. Table 4 presents the recognition rate and trainable parameters of the currently typical CNN model on the GTSRB dataset. Our teacher network achieves a competitive result with significantly improved computational efficiency. The batch size is 128. The learning rate is initially set to 0.001, while epochs are set to 300. As CNN-HLSGD [18] needs to train 20 same networks, the number of parameters it uses can be as large as 2.3×10^7, which is nearly 31 times the number used by our student network. Compared with DP-KELM [19], moreover, the student network uses half as many parameters; our student network is also an end-to-end system without augmentation.

Table 5 shows the results of pruning the student network following fine-tuning on the GTSRB dataset. We use the values of scaling factors, which approximate to zero, to identify insignificant channels, then prune those channels via thresholding; finally, we retrain the pruned network until convergence on the target task is achieved. As shown in Table 6, we prune the filters of each convolutional layer in order to reduce the number of redundant parameters. For example, as 32(− 32) in Conv1 means 64 filters in the first convolutional layer, the number of remaining filters following channel pruning is 32.

Table 7 presents the recognition rate of the pruned network and the full student network. When the model is pruned of 50% of its channels, the recognition rate of the student network falls to 99.38%, but the number of parameters is reduced by 70%. Moreover, when compared with DP-KELM [19], our pruned student network utilizes 16% of the parameters used by this method, while losing only a small amount of accuracy. When 70% channels are pruned, moreover, the number of parameters used by the student network is only 85,593; it can thus be deployed on mobile devices with limited power budgets.

3.3 Performance on BTSC dataset

As shown in Fig. 8, the best recognition rate achieved by our teacher network is 98.89%, while the average training loss value is 0.628 over 30 epochs. One cross-entropy loss value is calculated at each epoch, after which the mean of the loss

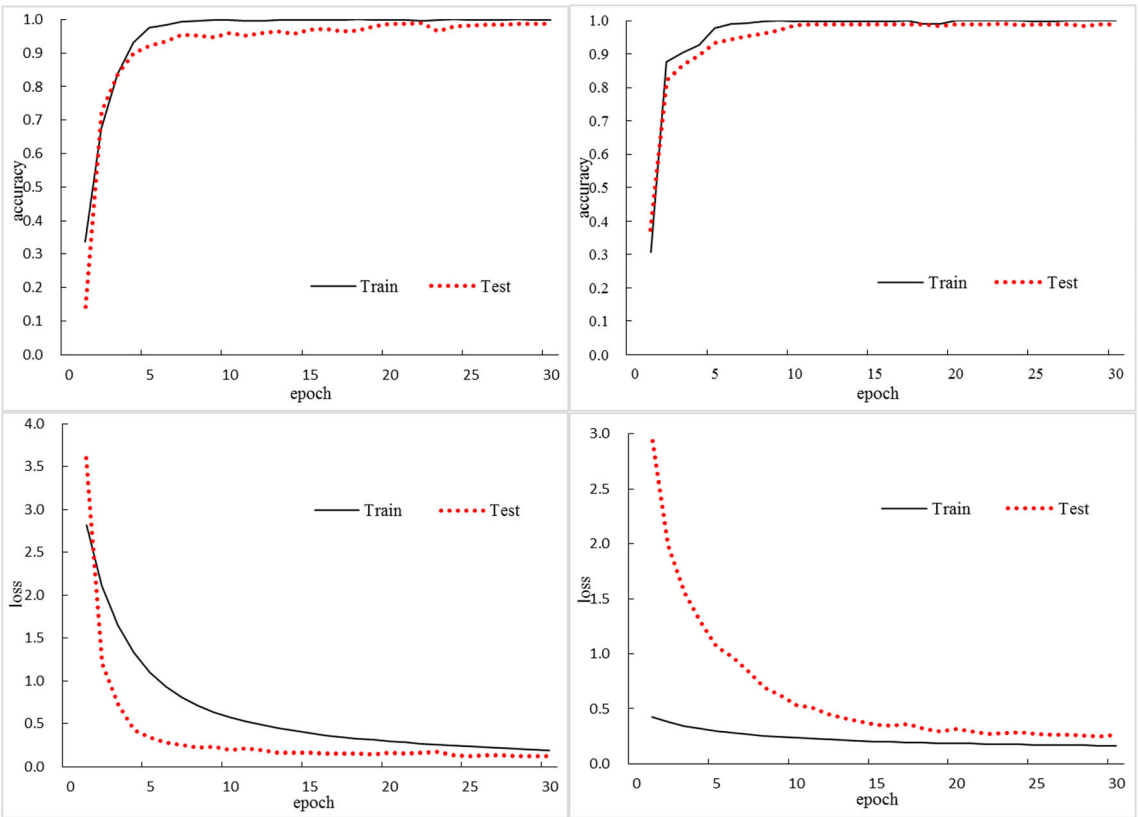


Fig. 8 Comparison of the accuracy and loss of the teacher network (two pictures on the left) and the student network (two pictures on the right)

Table 8 Performance comparison on BTSC dataset

Method [Referenced]	Accuracy
Our student network	99.13
GDBM [42]	98.92
Our teacher network	98.89
Our student network (50% pruned)	98.89
Single CNN with 3 STNs [41]	98.87
OneCNN [43]	98.17 ± 0.22
INNLP + SRC(PI) [37]	97.83

values across all epochs is defined as the average loss value. We use a standard cross-entropy loss to optimize the traffic sign classification task; here, the batch size is 128 and the initial learning rate is 0.001. Subsequently, the average recognition rate of our student network increases to 99.13%. Compared with the teacher network, the student network achieves better progress using knowledge distillation.

As shown in Table 8, the best recognition rate achieved by our teacher network is 98.89%, while that of our student network is 99.13%. The number of trainable parameters of a single CNN with 3 STNs [41] is 14,629,801; by contrast, the number used by our student network is only 809,982. After filters are pruned by half, the number of parameters used by our student network is 266,782; under these conditions, the obtained recognition rate is 98.89% on the BTSC dataset.

4 Conclusion

In this paper, we propose two lightweight networks for traffic sign classification. We implement a new module in our first model, referred to as the teacher network, which uses 1×1 convolutional layers and dense connectivity to learn features through parallel channels. Due to the large size of the neural networks involved, many models are difficult to deploy on mobile devices (which have limited power budgets) in traffic sign recognition systems. The second model, referred to as the student network, is a simple end-to-end architecture comprising only six layers. The performance of our method illustrates that our lightweight network is able to reduce the number of redundant parameters while retaining comparable accuracy. Moreover, we also prune channels for the student network, which yields a compact model. In conclusion, our lightweight network can provide an effective solution to deploying CNN for traffic sign classification in a resource-limited setting. In our future work, we aim to find a novel pruning criterion that can prune channels while producing a lower accuracy loss. We also plan to accelerate both the inference time and training procedure by implementing a compact model.

Funding information This work was supported in part by the National Natural Science Foundation of China under Grant 61772454, Grant 61811530332, and Grant 61811540410; in part by the Scientific Research Fund of Hunan Provincial Education Department under Grant 16A008; in part by the "Double First-class" International Cooperation and Development Scientific Research Project of Changsha University of Science and Technology under Grant 2019IC34; in part by the Postgraduate Scientific Research Innovation Fund of Hunan Province under Grant CX2018B565; in part by the Postgraduate Training Innovation Base Construction Project of Hunan Province under Grant 2017-451-30; and in part by the Postgraduate Course Construction Fund of CSUST under Grant KC201611.

References

1. Wang J, Ju C, Gao Y, Sangaiah AK, Kim G (2018) A PSO based energy efficient coverage control algorithm for wireless sensor networks. Comput Mater Continua 56(3):433–446
2. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
3. Plamondon R, Srihari SN (2000) Online and off-line handwriting recognition: a comprehensive survey. IEEE Trans Pattern Anal Mach Intell 22(1):63–84
4. Tu Y, Lin Y, Wang J, Kim JU (2018) Semi-supervised learning with generative adversarial networks on digital signal modulation classification. Comput Mater Continua 55(2):243–254
5. Zhang J, Lu C, Li X, Kim HJ, Wang J (2019) A full convolutional network based on DenseNet for remote sensing scene classification. Math Biosci Eng 16(5):3345–3367
6. Girshick R (2015) Fast R-CNN. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp 1440–1448
7. Zhang J, Jin K, Sun J, Wang J, Sangaiah AK (2018) Spatial and semantic convolutional features for robust visual object tracking. Multimed Tools Appl. https://doi.org/10.1007/s11042-018-6562-8. Accessed 19 Dec 2018
8. Wang N, Yeung DY (2013) Learning a deep compact image representation for visual tracking. In: Proceedings of the advances in neural information processing systems (NIPS), pp 809–817
9. Zhang J, Jin X, Sun J, Wang J, Li K (2019) Dual model learning combined with multiple feature selection for accurate visual tracking. IEEE Access 7:43956–43969
10. Zeng D, Dai Y, Li F, Sherratt RS, Wang J (2018) Adversarial learning for distant supervised relation extraction. Comput Mater Continua 55:121–136
11. Severyn A, Moschitti A (2015) Learning to rank short text pairs with convolutional deep neural networks. In: Proceedings of the 38th International ACM Conference on Research and Development in information retrieval, pp 373–382
12. Wang G, Ren G, Wu Z, Zhao Y, Jiang LH (2013) A robust, coarse-to-fine traffic sign detection method. In: Proceedings of the 2013 International Joint Conference on Neural Networks (IJCNN), pp 754–758
13. Chen Y, Xu W, Zuo J, Yang K (2018) The fire recognition algorithm using dynamic feature fusion and IV-SVM classifier. Clust Comput. https://doi.org/10.1007/s10586-018-2368-8. Accessed 19 Dec 2018
14. Chen Y, Xiong J, Xu W, Zuo J (2018) A novel online incremental and decremental learning algorithm based on variable support vector machine. Clust Comput. https://doi.org/10.1007/s10586-018-1772-4. Accessed 19 Dec 2018
15. Sermanet P, LeCun Y (2011) Traffic sign recognition with multi-scale convolutional networks. In: Proceedings of the 2011

International Joint Conference on Neural Network (IJCNN), pp 2809–2813
16. Ciregan D, Meier U, Masci J, Schmidhuber J (2012) Multi-column deep neural network for traffic sign classification. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR), pp 3642–3649
17. Zeng Y, Xu X, Fang Y, Zhao K (2015) Traffic sign recognition using deep convolutional networks and extreme learning machine. In: Proceedings of International Conference on Intelligence Science and Big Data Engineering (IScIDE), pp 272–280
18. Jin J, Fu K, Zhang C (2014) Traffic sign recognition with hinge loss trained convolutional neural networks. IEEE Trans Intell Transp Syst 15:1991–2000
19. Zeng Y, Xu X, Shen D, Fang Y (2017) Traffic sign recognition using kernel extreme learning machines with deep perceptual features. IEEE Trans Intell Transp Syst 18(6):1647–1653
20. Ngiam J, Coates A, Lahiri A, Prochnow B, Ng A (2011) On optimization methods for deep learning. In: Proceedings of the 28th International Conference on Machine Learning (ICML), pp 265–272
21. Krizhevsky A, Sutskever I, Hinton G (2012) Imagenet classification with deep convolutional neural networks. In: Proceedings of the advances in neural information processing systems (NIPS), pp 1097–1105
22. Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: Proceedings of the 3th International Conference on Learning Representations (ICLR)
23. Denil M, Shakibi B, Dinh L, Ranzato M, De Freitas N (2013) Predicting parameters in deep learning. In: Proceedings of the advances in neural information processing systems (NIPS), pp 2148–2156
24. Denton E, Zaremba W, Bruna J, Lecun Y, Fergus R (2014) Exploiting linear structure within convolutional networks for efficient evaluation. In: Proceedings of the advances in neural information processing systems (NIPS), pp 1269–1277
25. Molchanov P, Tyree S, Karras T, Aila T, Kautz J (2016) Pruning convolutional neural networks for resource efficient inference. In: Proceedings of the 5th International Conference on Learning Representations (ICLR), pp 1–17
26. Zhu C, Han S, Mao H, Dally WJ (2017) Trained ternary quantization. In: Proceedings of the 5th international conference on learning representations (ICLR)
27. Hinton G, Vinyals O, Dean J (2014) Distilling the knowledge in a neural network. In: Proceedings of the advances in neural information processing systems (NIPS), pp 2644−2652
28. Huang G, Liu Z, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 2261–2269
29. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the international conference on machine learning (ICML), pp 448–456
30. Nair V, Hinton G (2010) Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML), pp 807–814
31. Kingma DP, Ba JL (2015) Adam: a method for stochastic optimization. In: Proceedings of international conference on learning representations, pp 1–15
32. Tai C, Xiao T, Zhang Y, Wang X, Weinan E (2015) Convolutional neural networks with low-rank regularization. https://arxiv.org/abs/1511.06067. Accessed 19 Dec 2018
33. He Y, Zhang X, Sun J (2017) Channel pruning for accelerating very deep neural networks. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 1389–1397
34. Hu H, Peng R, Tai YW (2017) Network trimming: a data-driven neuron pruning approach towards efficient deep architectures. In: Proceedings of international conference on learning representations (ICLR), pp 214–222
35. Liu Z, Li J, Shen Z, Huang G, Yan S, Zhang C (2017) Learning efficient convolutional networks through network slimming. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 2755–2763
36. Matsubara S, Isayana H (2011) The German traffic sign recognition benchmark: a multi-class classification competition. In: Proceedings of International Joint Conference on Neural Networks (IJCNN), pp 1453–1460
37. Mathias M, Timofte R, Benenson R, Van GL (2013) Traffic sign recognition-how far are we from the solution? In: Proceedings of the 2013 international joint conference on neural networks (IJCNN), pp 1–8
38. Krizhevsky A (2009) Learning multiple layers of features from tiny images. Technical Report TR–2009, University of Toronto. http://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf. Accessed 19 Dec 2018
39. Szegedy C, Liu W, Jia Y, et al (2015) Going deeper with convolutions. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp 1–9
40. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of IEEE conference on computer vision and pattern recognition (CVPR)
41. Arcos-Garcia A, Alvarez-Garcia J, Soria-Morillo LM (2018) Deep neural network for traffic sign recognition systems: an analysis of spatial transformers and stochastic optimisation methods. Neural Netw 99:158–165
42. 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 J Photogramm Remote Sens 113:106–123
43. Jurisic F, Filkovic I, Kalafatic Z (2011) Multiple-dataset traffic sign classification with OneCNN. In: Proceedings of 2015 3rd IAPR Asian conference on pattern recognition, Kuala Lumpur, pp 614–618