

Gabor Convolutional Networks

Shangzhen Luan Baochang Zhang* Siyue Zhou
School of Automation Science and Electrical Engineering, Beihang University
bczhang@139.com; bczhang@buaa.edu.cn

Chen Chen
Center for Research
in Computer Vision (CRCV)
University of Central Florida

Jungong Han
School of Computing
and Communications,
Lancaster University

Wankou Yang Jianzhuang Liu
School of Automation, Southeast University, Nanjing Noah's Ark Lab, Huawei

Abstract

Steerable properties dominate the design of traditional filters, e.g., Gabor filters, and endow features the capability of dealing with spatial transformations. However, such excellent properties have not been well explored in the popular deep convolutional neural networks (DCNNs). In this paper, we propose a new deep model, termed Gabor Convolutional Networks (GCNs or Gabor CNNs), which incorporates Gabor filters into DCNNs to enhance the resistance of deep learned features to the orientation and scale changes. By only manipulating the basic element of DCNNs based on Gabor filters, i.e., the convolution operator, GCNs can be easily implemented and are compatible with any popular deep learning architecture. Experimental results demonstrate the super capability of our algorithm in recognizing objects, where the scale and rotation changes occur frequently. The proposed GCNs have much fewer learnable network parameters, and thus is easier to train with an end-to-end pipeline.

1. Introduction

Anisotropic filtering techniques have been widely used to extract robust image representation. Among them, the Gabor wavelets based on a sinusoidal plane wave with particular frequency and orientation can characterize the spatial frequency structure in images while preserving information of spatial relations, thus enabling to extract orientation-dependent frequency contents of patterns. Recently, deep convolutional neural networks (DCNNs) based on convolution filters have attracted significant attention in computer vision. This efficient, scalable and end-to-end model has

the amazing capability of learning powerful feature representations from raw image pixels, boosting performances of many computer vision tasks, such as image classification, object detection, and semantic segmentation. Unlike hand-crafted filters without any learning process, DCNNs-based feature extraction is a purely data-driven technique that can learn robust representations from data, but usually at the cost of expensive training and complex model parameters. Additionally, the capability of modeling geometric transformations mostly comes from extensive data augmentation, large models, and hand-crafted modules (e.g. max-pooling [2] for small translation-invariance). Therefore, DCNNs normally fail to handle large and unknown object transformations if the training data are not enough. And one reason originates from the way of filter designing [2, 36].

Fortunately, the need to enhance model capacity to transformations has been perceived by researchers and some attempts have been made in recent years. In [4], a deformable convolution filter was introduced to enhance DCNNs' capacity of modeling geometric transformations. It allows free form deformation of the sampling grid, whose offsets are learned from the preceding feature maps. However, the deformable filtering is complicated and associated with the Region of Interest (RoI) pooling technique originally designed for object detection [7]. In [36], **Actively Rotating Filters (ARFs)** were proposed to give DCNNs the generalization ability of rotation. However, such a filter rotation method is actually only suitable for small and simple filters, i.e. 1x1 and 3x3 filters. Although the authors claimed a general method to modulate the filters based on the Fourier transform, it was not implemented in [36], which is probably due to its computational complexity. Furthermore,

3D filters [13] are hardly modified by deformable filters or ARFs. In [11], by combining low level filters (Gaussian derivatives up to the 4-th order) with learned weight coefficients, the regularization over the filter function space is shown to improve the generalization ability but only when the set of training data is small.

In Fig. 1, the visualization of convolutional filters [16] indicates that filters are often redundantly learned, such as Alexnet filters trained on ImageNet¹, and some of the filters from shallow layers are similar to Gabor filters. It is known that the steerable properties of Gabor filters are widely adopted in the traditional filter design due to their enhanced capability of scale and orientation decomposition of signals, which is neglected in most of the prevailing convolutional filters in DCNNs. A few works have explored Gabor filters for DCNNs. However, they do not explicitly integrate Gabor filters into the convolution filters. Specifically, [10] simply employs Gabor filters to generate Gabor features and uses them as input to a CNN, and [23] only fixes the first or second convolution layer by Gabor filters, which mainly aims at reducing the training complexity of CNNs.

In this paper we propose to use traditional hand-crafted Gabor filters to modulate the learnable convolution filters, aiming to reduce the number of learnable network parameters and enhance the robustness of learned features to orientation and scale changes. Specifically, in each convolution layer, the convolution filters are modulated by Gabor filters with different orientations and scales to produce convolutional **Gabor orientation filters (GoFs)**, which endow the convolution filters the additional capability of capturing the visual properties such as spatial localization, orientation selectivity and spatial frequency selectivity in the output feature maps. GoFs are implemented on the basic element of CNNs, *i.e.* the convolution filter, and thus can be easily integrated into any deep architecture. **DCNNs with GoFs, referred to as GCNs, can learn more robust feature representations**, particularly for images with spatial transformations. In addition, since GoFs are generated based on a small set of learnable convolution filters, the proposed GCNs model is more compact and easier to train. The contributions of this paper are two-fold:

1) To the best of our knowledge, it is the first time Gabor filters are incorporated into the convolution filter to improve the robustness of DCNNs to image transformations such as transitions, scale changes and rotations.

2) GCNs improve the widely used DCNNs architectures including conventional CNNs and ResNet [8], obtaining the state-of-the-art results on popular benchmarks.

¹For the illustration purpose, Alexnet is selected because its filters are of big sizes.

2. Related Work

2.1. Gabor filters

Gabor wavelets [6] were invented by Dennis Gabor using complex functions to serve as a basis for Fourier transforms in information theory applications. An important property of the wavelets is that the product of its standard deviations is minimized in both time and frequency domains. Gabor filters are widely used to model receptive fields of simple cells of the visual cortex. The **Gabor wavelets (kernels or filters)** are defined as follows [30, 33]:

$$\Psi_{u,v}(z) = \frac{\|k_{u,v}\|^2}{\sigma^2} e^{-(\|k_{u,v}\|^2 \|z\|^2 / 2\sigma^2)} [e^{ik_{u,v}z} - e^{-\sigma^2/2}], \quad (1)$$

where $k_{u,v} = k_v e^{ik_u}$, $k_v = (\pi/2)/\sqrt{2}^{(v-1)}$, $k_u = u\frac{\pi}{U}$, with $v = 0, \dots, V$ and $u = 0, \dots, U$ and v is the frequency and u is the orientation, and $\sigma = 2\pi$. Gabor filters are bounded as shown in [32], which provides a foundation for its extensive applications. In [35, 22], Gabor wavelets were used to initialize the deep models or serve as the input layer. However, we take a different approach by utilizing Gabor filters to modulate the learned convolution filters. Specifically, we **change the basic element of CNNs – convolution filters to GoFs** to enforce the impact of Gabor filters on each convolutional layer. Therefore, the steerable properties are inherited into the DCNNs to enhance the robustness to scale and orientation variations in feature representations.

2.2. Learning feature representations

Given rich and often redundant convolutional filters, data augmentation is used to achieve local/global transform invariance [26]. Despite the effectiveness of data augmentation, the main drawback lies in that learning all possible transformations usually requires a large number of network parameters, which significantly increases the training cost and the risk of over-fitting. Most recently, TI-Pooling [17] alleviates the drawback by using parallel network architectures for the transformation set and applying the transformation invariant pooling operator on the outputs before the top layer. Nevertheless, with a built-in data augmentation, TI-Pooling requires significantly more training and testing computational cost than a standard CNN.

Spatial Transformer Networks: To gain more robustness against spatial transformations, a new framework for spatial transformation termed **spatial transformer network (STN)** [12] is introduced by using an additional network module that can manipulate the feature maps according to the transform matrix estimated with a localization sub-CNN. However, STN does not provide a solution to precisely estimate complex transformation parameters.

Oriented Response Networks: By using **Actively Rotating Filters (ARFs)** to generate orientation-tensor feature maps, **Oriented Response Network (ORN)** [36] encodes

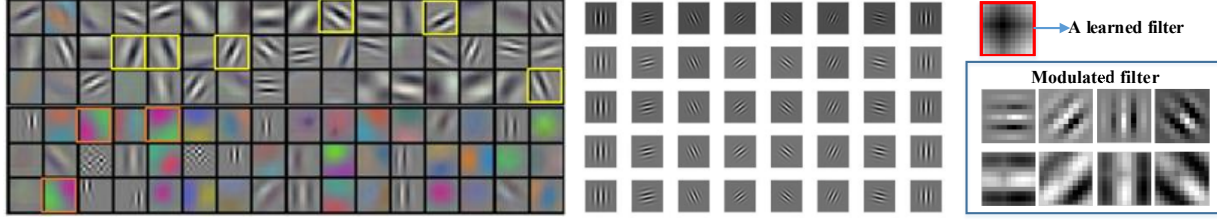


Figure 1. Left illustrates Alexnet filters. Middle shows Gabor filters. Right presents the convolution filters modulated by Gabor filters. Filters are often redundantly learned in CNN, and some of which are similar to Gabor filters. Based on this observation, we are motivated to manipulate the learned convolution filters using Gabor filters, in order to achieve a compressed deep model with reduced number of filter parameters. In the right column, a convolution filter is modulated by Gabor filters via Eq. 2 to enhance the orientation property.

hierarchical orientation responses of discriminative structures. With these responses, ORN can be used to either encode the orientation-invariant feature representation or estimate object orientations. However, ORN is more suitable for small size filters, *i.e.* 3×3 , whose orientation invariance property is not guaranteed by the ORAlign strategy based on their marginal performance improvement as compared with TI-Pooling.

Deformable convolutional network: Deformable convolution and deformable RoI pooling are introduced in [4] to enhance the transformation modeling capacity of CNNs, making the network robust to geometric transformations. However, the deformable filters also prefer operating on small-sized filters.

Scattering Networks: In wavelet scattering network [3, 24], expressing receptive fields in CNNs as a weighted sum over a fixed basis allows the new structured receptive field networks to increase the performance considerably over unstructured CNNs for small and medium datasets. In contrast to the scattering networks, our GCNs are based on Gabor filters to change the convolution filters in a steerable way.

3. Gabor Convolutional Networks

Gabor Convolutional Networks (GCNs) are deep convolutional neural networks using Gabor orientation filters (GoFs). A GoF is a steerable filter, created by manipulating the learned convolution filters via Gabor filter banks, to produce the enhanced feature maps. With GoFs, GCNs not only have significant fewer filter parameters to learn, but also lead to enhanced deep models.

In what follows, we address three issues in implementing GoFs in DCNNs. First, we give the details on obtaining GoFs through Gabor filters. Second, we describe convolutions that use GoFs to produce feature maps with scale and orientation information enhanced. Third, we show how GoFs are learned during the back-propagation update stage.

3.1. Convolutional Gabor orientation Filters (GoFs)

Gabor filters are of U directions and V scales. To incorporate the steerable properties into the GCNs, the orientation information is encoded in the learned filters, and at the same time the scale information is embedded into different layers. Due to the orientation and scale information captured by Gabor filters in GoFs, the corresponding convolution features are enhanced.

Before being modulated by Gabor filters, the convolution filters in standard CNNs are learned by back propagation (BP) algorithm, which are denoted as learned filters. Let a learned filter be of size $N \times W \times W$, where $W \times W$ is the size of 2D filter (N channels). For implementation convenience, N is chosen to be U , which is the number of orientations of the Gabor filters that will be used to modulate this learned filter. A GoF is obtained based on a modulated process using U Gabor filters on the learned filters for a given scale v . The details concerning the filter modulation are shown in Eq. 2 and Fig. 2. For the v th scale, we define:

$$C_{i,u}^v = C_{i,o} \circ G(u, v), \quad (2)$$

where $C_{i,o}$ is a learned filter, and \circ is an element-by-element product operation between $G(u, v)^2$ and each 2D filter of $C_{i,o}$. $C_{i,u}^v$ is the modulated filter of $C_{i,o}$ by the v -scale Gabor filter $G(u, v)$. Then a GoF is defined as:

$$C_i^v = (C_{i,1}^v, \dots, C_{i,U}^v). \quad (3)$$

Thus, the i th GoF C_i^v is actually U 3D filters (see Fig. 2, where $U = 4$). In GoFs, the value of v increases with increasing layers, which means that scales of Gabor filters in GoFs are changed based on layers. At each scale, the size of a GoF is $U \times N \times W \times W$. However, we only save $N \times W \times W$ learned filters because the Gabor filters are given, which means that we can obtain enhanced features by this modulation without increasing the number of parameters. To simplify the description of the learning process, v is omitted in the next section.

²The real parts of Gabor filters are used.

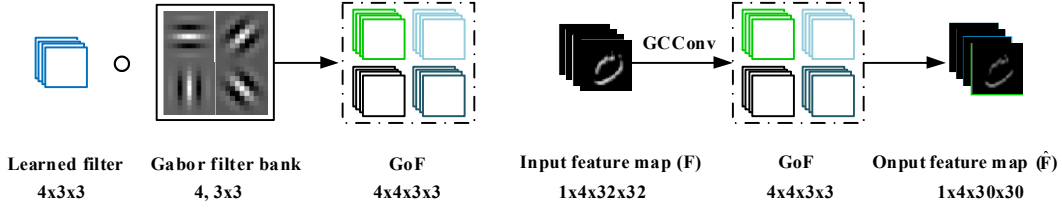


Figure 2. Left shows modulation process of GoFs. Right illustrates an example of GCN convolution with 4 channels. In a GoF, the number of channels is set to be the number of Gabor orientations U for implementation convenience.

3.2. GCN convolution

In GCNs, GoFs are used to produce feature maps, which explicitly enhance the scale and orientation information in deep features. A output feature map \hat{F} in GCNs is denoted as:

$$\hat{F} = GCconv(F, C_i), \quad (4)$$

where C_i is the i th GoF and F is the input feature map as shown in Fig. 2. The channels of \hat{F} are obtained by the following convolution:

$$\hat{F}_{i,k} = \sum_{n=1}^N F^{(n)} \otimes C_{i,u=k}^{(n)}, \quad (5)$$

where (n) refers to the n th channel of F and $C_{i,u}$, and $\hat{F}_{i,k}$ is the k th orientation response of \hat{F} . For example as shown in Fig. 2, let the size of the input feature map be $1 \times 4 \times 32 \times 32$, the size of corresponding output will be $1 \times 4 \times 30 \times 30$ (channel, orientation channel, H, W) after convolving with a GoF with 4 Gabor orientations, and if there are 20 such GoFs, the size of the output feature map will be $20 \times 4 \times 30 \times 30$ (no padding).

3.3. Updating GoF

In the back-propagation (BP) process, only the leaned filer $C_{i,o}$ needs to be updated. And we have:

$$\delta = \frac{\partial L}{\partial C_{i,0}} = \sum_{u=1}^U \frac{\partial L}{\partial C_{i,u}} \circ G(u, v) \quad (6)$$

$$C_{i,o} = C_{i,o} - \eta \delta, \quad (7)$$

where L is the loss function. From the above equations, it can be seen that the BP process is easily implemented and is very different from ORNs and deformable kernels that usually require a relatively complicated procedure. By only updating the learned convolution filters $C_{i,o}$, the GCNs model is more compact and efficient, and also is more robust to orientation and scale variations.

Table 1. Results (error rate (%) on MNIST) vs. Gabor filter scales.

kernel	5x5	5x5	3x3	3x3
# params (M)	1.86	0.51	0.78	0.25
$V = 1$ scale	0.48	0.57	0.51	0.7
$V = 4$ scales	0.48	0.56	0.49	0.63

Table 2. Results (error rate (%) on MNIST) vs. Gabor filter orientations.

U	2	3	4	5	6	7
5x5	0.52	0.51	0.48	0.49	0.49	0.52
3x3	0.68	0.58	0.56	0.59	0.56	0.6

4. Implementation and Experiments

In this section, we present the details of the GCNs implementation based on conventional DCNNs architectures. Afterwards, we evaluate GCNs on the MNIST digit recognition dataset [19, 18] as well as its rotated version MNIST-rot used in ORNs, which is generated by rotating each sample in the MNIST dataset by a random angle between $[0, 2\pi]$. To further evaluate the performance of GCNs, the experiments on the SVHN dataset [21], CIFAR-10 and CIFAR-100 [15], as well as a 100-class ImageNet2012 [5] subset are also provided. We have two GPU platforms used in our experiments, NVIDIA GeForce GTX 1070 and GeForce GTX TITAN X(2).

4.1. MNIST

For the MNIST dataset, we randomly select 10,000 samples from the training set for validation and the remaining 50,000 samples for training. Adadelta optimization algorithm [29] is used during the training process, with the batch size as 128, initial learning rate as 0.001 (η) and weight decay as 0.00005. The learning rate is reduced to half per 25 epochs. We report the performance of our algorithm on a test set after 200 epochs based on the average over 5 runs. The state-of-the-art STN [12], TI-Pooling [17], ResNet [8] and ORNs [36] are employed for comparison. Among them, STN is more robust to spatial transformation than the baseline CNNs, due to a spatial transform

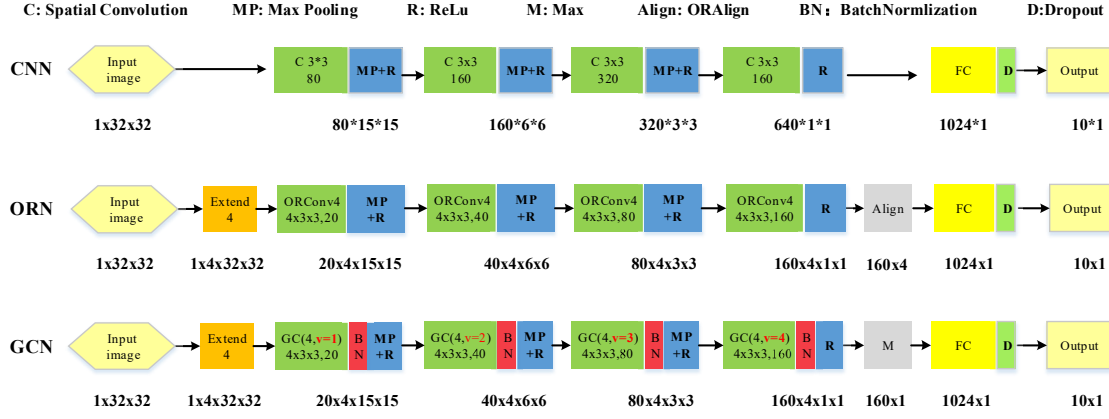


Figure 3. Network structures of CNNs, ORNs and GCNs.

Table 3. Results comparison on MNIST

Method	error (%)				
	# network stage kernels	# params (M)	time (s)	MNIST	MNIST-rot
Baseline CNN	80-160-320-640	3.08	6.50	0.73	2.82
STN	80-160-320-640	3.20	7.33	0.61	2.52
TIpooling _(×8)	(80-160-320-640) _{×8}	3.08	50.21	0.97	not permitted
ORN4(ORAlign)	10-20-40-80	0.49	9.21	0.57	1.69
ORN8(ORAlign)	10-20-40-80	0.96	16.01	0.59	1.42
ORN4(ORPooling)	10-20-40-80	0.25	4.60	0.59	1.84
ORN8(ORPooling)	10-20-40-80	0.39	6.56	0.66	1.37
GCN4(with 3 × 3)	10-20-40-80	0.25	3.45	0.63	1.45
GCN4(with 3 × 3)	20-40-80-160	0.78	6.67	0.56	1.28
GCN4(with 5 × 5)	10-20-40-80	0.51	10.45	0.49	1.26
GCN4(with 5 × 5)	20-40-80-160	1.86	23.85	0.48	1.10
GCN4(with 7 × 7)	10-20-40-80	0.92	10.80	0.46	1.33
GCN4(with 7 × 7)	20-40-80-160	3.17	25.17	0.42	1.20

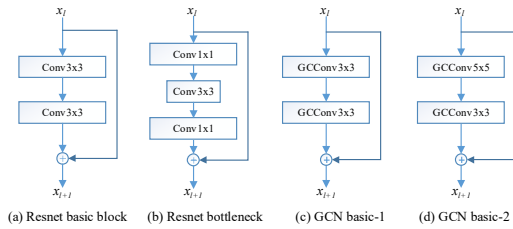


Figure 4. The residual block. (a) and (b) are for ResNet. (c) Small kernel and (d) large kernel are for GCNs.

layer prior to the first convolution layer. TI-Pooling adopts a transform-invariant pooling layer to get the response of main direction, resulting in rotation robust features. ORNs capture the response of each direction by rotating the original convolution kernel spatially.

Fig. 3 shows the network structures of CNNs, ORNs and GCNs ($U = 4$), which are used in this experiment. For all

models, we adopt **Max-pooling** and **ReLU** after convolution layers, and a **dropout layer** [9] after the fully connected (FC) layer to avoid over-fitting. To compare with other CNNs in a similar model size, we reduce the width of layer³ by a certain proportion as done in ORNs, *i.e.* 1/8 [36].

We evaluate different scales for different GCNs layers (*i.e.* $V = 4, V = 1$), where larger scale Gabor filters are used in shallow layers or a single scale is used in all layers. It should be noted that in the following experiments, we also use $V = 4$ for deeper networks (ResNet). As shown in Table 1, the results of $V = 4$ in terms of error rate are better than those when a single scale ($V = 1$) is used in all layers and $U = 4$. We also test different orientations as shown in Table 2. The results indicate that GCNs perform better using 3 to 6 orientations when $V = 4$, which is more flexible than ORNs. In comparison, ORNs use a complicated interpolation process via ARFs besides 4- and 8-pixel rotations.

³The number of convolution kernels per layer.

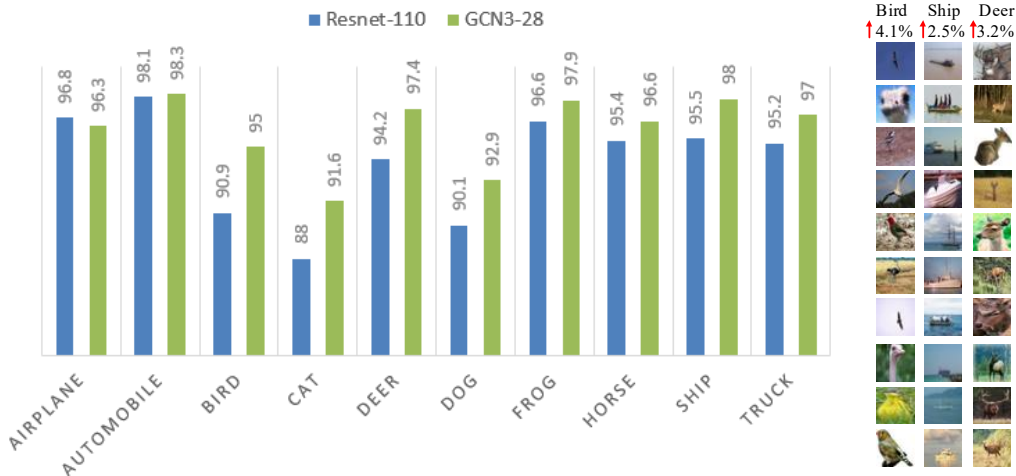


Figure 5. Recognition results of different categories on CIFAR10. Compared with ResNet-110, GCNs perform significantly better on the categories with large scale variations.

Table 4. Results comparison on SVHN. No additional training set is used for training

Method	VGG	ResNet-110	ResNet-172	GCN4-40	GCN4-28	ORN4-40	ORN4-28
# params	20.3M	1.7M	2.7M	2.2M	1.4M	2.2M	1.4M
Accuracy (%)	95.66	95.8	95.88	96.9	96.86	96.35	96.19

In Table 3, the second column refers to the width of each layer, and a similar notation is also used in [28]. Considering a GoF has multiple channels (N), we decrease the width of layer (*i.e.* the number of GoFs per layer) to reduce the model size to facilitate a fair comparison. The parameter size of GCNs is linear with channel (N) but quadratic with width of layer. Therefore, the GCNs complexity is reduced as compared with CNNs (see the third column of Table 3). In the fourth column, we compare the computation time (s) for training epoch of different methods using GTX 1070, which clearly shows that GCNs are more efficient than other state-of-the-art models. The performance comparison is shown in the last two columns in terms of error rate. By comparing with baseline CNNs, GCNs achieved much better performance with 3x3 kernel but only using 1/12, 1/4 parameters of CNNs. It is observed from the experiments that GCNs with 5x5 and 7x7 kernels achieve test errors of 1.10% on MNIST-rot and 0.42% on MNIST, respectively, which are better than those of ORNs. This can be explained by the fact that the kernels with larger size carry more information of Gabor orientation, and thus capture better orientation response features. Table 3 also demonstrates that a larger GCNs model can result in better performance. In addition, on the MNIST-rot datasets, the performance of baseline CNNs model is greatly affected by rotation, while ORNs and GCNs can capture orientation features and achieve better results. Again, GCNs outperform ORNs, which confirms that Gabor modulation indeed helps

to gain the robustness to rotation variations. This improvement is attributed to the enhanced deep feature representations of GCNs based on the steerable filters. In contrast, ORNs only actively rotate the filters and lack a feature enhancement process.

4.2. SVHN

The Street View House Numbers (SVHN) dataset [21] is a real-world image dataset taken from Google Street View images. SVHN contains MNIST-like 32x32 images centered around a single character, which however include a plethora of challenges like illumination changes, rotations and complex backgrounds. The dataset consists of 600000 digit images: 73257 digits for training, 26032 digits for testing, and 531131 additional images. Note that the additional images are not used for all methods in this experiment. For this large scale dataset, we implement GCNs based on ResNet. Specifically, we replace the spatial convolution layers with our GoFs based GCConv layers, leading to GCN-ResNet. The bottleneck structure is not used since the 1x1 kernel does not propagate any Gabor filter information. ResNet divides the whole network into 4 stages, and the width of stage (the number of convolution kernels per layer) is set as 16, 16, 32, and 64, respectively. We make appropriate adjustments to the network depth and width to ensure our GCNs method has a similar model size as compared with VGG [25] and ResNet. We set up 40-layer and 28-layer GCN-ResNets with basic block-(c)(Fig. 4), using

Table 5. Results comparison on CIFAR-10 and CIFAR-100

Method			error (%)	
			CIFAR-10	CIFAR-100
NIN			8.81	35.67
VGG			6.32	28.49
	# network stage kernels	# params Fig.4(c)/Fig.4(d)		
ResNet-110	16-16-32-64	1.7M	6.43	25.16
ResNet-1202	16-16-32-64	10.2M	7.83	27.82
GCN2-110	12-12-24-45	1.7M/3.3M	6.34/5.62	26.14/25.3
GCN2-110	16-16-32-64	3.4M/6.5M	5.65/4.96	
GCN4-110	8-8-16-32	1.7M	6.19	24.23
GCN2-40	16-32-64-128	4.5M	4.95	
GCN4-40	16-16-32-64	2.2M	5.34	25.65
WRN-40	64-64-128-256	8.9M	4.53	21.18
WRN-28	160-160-320-640	36.5M	4.00	19.25
GCN2-40	16-64-128-256	17.9M	4.41	20.73
GCN4-40	16-32-64-128	8.9M	4.65	21.75
GCN3-28	64-64-128-256	17.6M	3.88	20.13

the same hyper-parameters as ResNet. The network stage is also set as 16-16-32-64. The results are listed in Table 4. Compared to VGG model, GCNs have much smaller parameter size, yet obtain a better performance with 1.2% improvement. With a similar parameter size, the GCN-ResNet achieves better results (1.1%, 0.66%) than ResNet and ORNs respectively, which further validates the superiority of GCNs for real-world problems.

4.3. Natural Image Classification

For the natural image classification task, we use the CIFAR datasets including CIFAR-10 and CIFAR-100 [15]. The CIFAR datasets consist of 60000 color images of size 32x32 in 10 or 100 classes, with 6000 or 600 images per class. There are 50000 training images and 10000 test images.

CIFAR datasets contain a wide variety of categories with object scale and orientation variations. Similar to SVHN, we test GCN-ResNet on CIFAR datasets. Experiments are conducted to compare our method with the state-of-the-art networks (*i.e.* NIN [20], VGG [25], and ResNet [8]). On CIFAR-10, Table 5 shows that GCNs consistently improve the performance regardless of the number of parameters or kernels as compared with the baseline ResNet. We further compare GCNs with the Wide Residue network (WRN) [28], and again GCNs achieve a better result (3.88% vs. 4% error rate) when our model is half the size of WRN, indicating significant advantage of GCNs in terms of model efficiency. Similar to CIFAR-10, one can also observe the performance improvement on CIFAR-100, with similar parameter sizes. Moreover, when using different kernel size configurations (from 3×3 to 5×5 as shown in Fig. 4(c)

and Fig. 4(d)), the model size is increased but with a performance (error rate) improvement from 6.34% to 5.62%. We notice that some top improved classes in CIFAR10 are bird (4.1% higher than baseline ResNet), and deer (3.2%), which exhibit significant within class scale variations. This implies that the Gabor filter modulation in CNNs enhances the capability of handling scale variations (see Fig. 5). The convergence curves of training loss and test error are shown in Fig.6.

4.4. Large Size Image Classification

The previous experiments are conducted on datasets with small size images (*e.g.* 32×32 for the SVHN dataset). To further show the effectiveness of the proposed GCNs method, we evaluate it on the ImageNet [5] dataset. Different from MNIST, SVHN and CIFAR, ImageNet consists of images with a much higher resolution. In addition, the images usually contain more than one attribute per image, which may have a large impact on the classification accuracy. Since this experiment is only to validate the effectiveness of GCNs on large size images, we don't use the full ImageNet dataset because it will take a significant time to train a deep model on such large scale set. Alternatively, we choose a 100-class ImageNet2012 [5] subset in this experiment. The 100 classes are selected from the full ImageNet dataset at a step of 10. Similar subset is also applied in [14, 1, 27].

For the ImageNet-100 experiment, we train a 34-layer GCN with 4-orientation channels, and the scale setting is the same as previous experiments. A ResNet-101 model is set as the baseline. Both GCNs and ResNet are trained after 120 epochs. The learning rate is initialized as 0.1 and de-

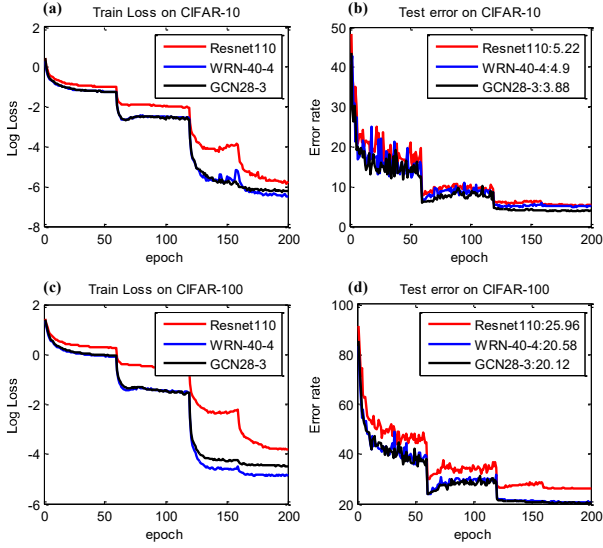


Figure 6. Training loss and test error curves on CIFAR dataset. (a),(b) for CIFAR-10, (c),(d) for CIFAR-100. Compared with baseline ResNet-110, GCN achieved a faster convergence speed and lower test error. WRN and GCN achieved similar performance, but GCN had lower error rate on the test set.

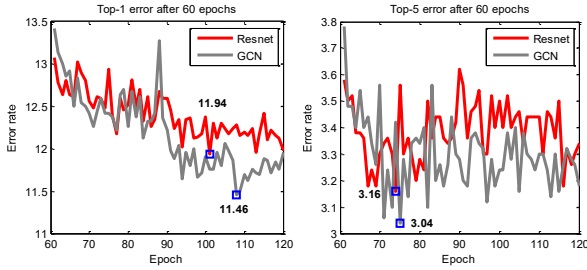


Figure 7. Test error curve for the ImageNet experiment. The first 60 epochs are omitted for clarity.

creases to 1/10 times per 30 epochs. Top-1 and Top-5 errors are used as evaluation metrics. The convergence of ResNet is better at the early stage. However, it tends to be saturated quickly. GCNs have a slightly slower convergence speed, but show better performance in later epochs. The test error curve is depicted in Fig. 7. As compared to the baseline, our GCNs achieve better classification performances (*i.e.* Top-5 error: 3.04% vs. 3.16%, Top-1 error: 11.46% vs. 11.94%) when using fewer parameters (35.85M vs. 44.54M).

4.5. Automatic Modulation Classification

GCNs are genral, which can also be used in the field of automatic modulation classification (AMC). It benefits the communication reconfiguration and electromagnetic environment analysis, and plays an essential role in obtaining digital base band information from the signal. We collect

communication signal data sets, which make the transmitted wireless channel approximate to the actual facts on the basis of the actual geographical environment. Training and test sets including 1100 samples for 11 classes repectively. The experimental results show that GCNs achieved a better performance than conventional wavelet+SVM, *i.e.*, 86.4% vs. 85.2% when the signal-to-noise ratio (SNR) is 20 dB.

5. Conclusion

This paper has presented a new end-to-end deep model by incorporating Gabor filters to DCNNs, aiming to enhance the deep feature representations with steerable orientation and scale capacities. The proposed Gabor Convolutional Networks (GCNs) improve DCNNs on the generalization ability of rotation and scale variations by introducing extra functional modules on the basic element of DCNNs, *i.e.* the convolution filters. GCNs can be easily implemented using popular architectures. The extensive experiments show that GCNs significantly improved baselines, resulting in the state-of-the-art performance over several benchmarks. The future work will focus on action recognition and image restoration [34, 31].

Acknowledgement

The work was supported by the Natural Science Foundation of China under Contract 61672079 and 61473086. This work is supported by the Open Projects Program of National Laboratory of Pattern Recognition, and shenzhen peacock plan.

References

- [1] A. Banerjee and V. Iyer. Cs231n project report-tiny imagenet challenge.
- [2] Y.-L. Boureau, J. Ponce, and Y. LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML)*, pages 111–118, 2010.
- [3] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013.
- [4] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. *arXiv preprint arXiv:1703.06211*, 2017.
- [5] J. Deng, W. Dong, R. Socher, and L. J. Li. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255, 2009.
- [6] D. Gabor. Theory of communication. part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, 93(26):429–441, 1946.
- [7] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

- [8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.
- [9] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.
- [10] Y. Hu, C. Li, D. Hu, and W. Yu. Gabor feature based convolutional neural network for object recognition in natural scene. In *International Conference on Information Science and Control Engineering*, pages 386–390, 2016.
- [11] J.-H. Jacobsen, J. van Gemert, Z. Lou, and A. W. Smeulders. Structured receptive fields in cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2610–2619, 2016.
- [12] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015.
- [13] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 35(1):221–231, 2013.
- [14] F. Juefei-Xu, V. N. Boddeti, and M. Savvides. Local binary convolutional neural networks. *arXiv preprint arXiv:1608.06049*, 2016.
- [15] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. 2009.
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
- [17] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys. Tipooling: transformation-invariant pooling for feature learning in convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 289–297, 2016.
- [18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- [19] Y. LeCun, C. Cortes, and C. J. Burges. The mnist database of handwritten digits, 1998.
- [20] M. Lin, Q. Chen, and S. Yan. Network in network. in *Proc. of ICLR*, 2014.
- [21] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011.
- [22] W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2056–2063, 2013.
- [23] S. S. Sarwar, P. Panda, and K. Roy. Gabor filter assisted energy efficient fast learning convolutional neural networks. 2017.
- [24] L. Sifre and S. Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1233–1240, 2013.
- [25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [26] D. A. Van Dyk and X.-L. Meng. The art of data augmentation. *Journal of Computational and Graphical Statistics*, 10(1):1–50, 2001.
- [27] L. Yao and J. Miller. Tiny imagenet classification with convolutional neural networks. *CS 231N*, 2015.
- [28] S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.
- [29] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.
- [30] B. Zhang, Y. Gao, S. Zhao, and J. Liu. Local derivative pattern versus local binary pattern: Face recognition with high-order local pattern descriptor. *IEEE Transactions on Image processing*, 19(2):533–544, 2010.
- [31] B. Zhang, J. Gu, C. Chen, J. Han, X. Su, X. Cao, and J. Liu. One-two-one network for compression artifacts reduction in remote sensing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2018.
- [32] B. Zhang, A. Perina, Z. Li, V. Murino, J. Liu, and R. Ji. Bounding multiple gaussians uncertainty with application to object tracking. *International Journal of Computer Vision*, 118(3):364–379, 2016.
- [33] B. Zhang, S. Shan, X. Chen, and W. Gao. Histogram of gabor phase patterns (hgpp): A novel object representation approach for face recognition. *IEEE Transactions on Image processing*, 16(1):57–68, 2007.
- [34] B. Zhang, Y. Yang, C. Chen, Y. Linlin, J. Han, and L. Shao. Action recognition using 3d histograms of texture and a multi-class boosting classifier. *IEEE Transactions on Image processing*, 26(10):4648–4660, 2017.
- [35] Z. Zhong, L. Jin, and Z. Xie. High performance offline handwritten chinese character recognition using googlenet and directional feature maps. In *13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 846–850. IEEE, 2015.
- [36] Y. Zhou, Q. Ye, Q. Qiu, and J. Jiao. Oriented response networks. *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, 2017.