Analysis on the Dropout Effect in Convolutional Neural Networks

Sungheon Park and Nojun Kwak(✉)

Graduate School of Convergence Science and Technology,
Seoul National University, Seoul, Korea
{sungheonpark,nojunk}@snu.ac.kr

Abstract. Regularizing neural networks is an important task to reduce overfitting. Dropout [1] has been a widely-used regularization trick for neural networks. In convolutional neural networks (CNNs), dropout is usually applied to the fully connected layers. Meanwhile, the regularization effect of dropout in the convolutional layers has not been thoroughly analyzed in the literature. In this paper, we analyze the effect of dropout in the convolutional layers, which is indeed proved as a powerful generalization method. We observed that dropout in CNNs regularizes the networks by adding noise to the output feature maps of each layer, yielding robustness to variations of images. Based on this observation, we propose a stochastic dropout whose drop ratio varies for each iteration. Furthermore, we propose a new regularization method which is inspired by behaviors of image filters. Rather than randomly drop the activation, we selectively drop the activations which have high values across the feature map or across the channels. Experimental results validate the regularization performance of selective max-drop and stochastic dropout is competitive to the dropout or spatial dropout [2].

1 Introduction

Convolutional neural networks (CNNs) have been widely used for many computer vision tasks such as image classification, segmentation, and detection in recent years, mainly due to their high representation power and superior performance. Since deep neural networks are involved with a large number of parameters, regularization is a critical task to reduce overfitting. Other than a weight decay term, many algorithms have been presented to regularize neural networks. Dropout [1] is the most commonly used technique for regularization. For CNNs, stochastic pooling [3] or maxout networks [4] are well known techniques to regularize convolutional layers. Though dropout has shown its effectiveness in convolutional layers in some cases [1,5,6], it is still rarely used with convolutional layers in practice. Moreover, the effect of dropout in convolutional layers has not been studied thoroughly. Different from the fully connected layers, convolutional layers have smaller number of parameters compared to the size of feature maps. Hence, it is believed that convolutional layers suffer less from overfitting.

In this paper, we analyze the effect of dropout in convolutional layers. We found that dropout in convolutional layers as well as the fully connected layers

S.-H. Lai et al. (Eds.): ACCV 2016, Part II, LNCS 10112, pp. 189–204, 2017.
DOI: 10.1007/978-3-319-54184-6_12

are effective for regularization. The generalization effect of dropout in convolutional layers is due to the enhanced robustness by adding noise to the inputs of convolutional layers, not due to the model averaging in the case of fully connected layers. Based on this observation, we propose two variants of dropout which is suited for convolutional layers of CNNs. Similar to dropout [1], the proposed methods turn off the activations of convolutional layers. While dropout turns off the activations randomly, the first variant, max-drop, selectively drops the activation which is the maximum value within each feature map or within the same spatial position of feature maps. Since the neurons with high activation values contain key information about the problem at hand, dropping the maximum activation probabilistically can grant generalization power to the networks. The other variant, stochastic dropout, varies the dropout probability based on the probability distribution which makes the network robust to inputs with different levels of noise. Experimental results show that the proposed method effectively regularizes convolutional layers and shows competitive performance against dropout. This result indicates that unlike dropout, only dropping a small portion of activations in the network can lead to a powerful generalization performance.

The rest of the papers will be presented as follows. Related works are introduced in Sect. 2, and we analyze the effect of dropout in convolutional layers in Sect. 3. Based on the analysis, two variants of dropout, max-drop and stochastic dropout, are proposed in Sects. 4 and 5 respectively. Experiments on various datasets are conducted to compare the generalization performance of proposed methods with dropout, and the results are illustrated in Sect. 6. Finally, conclusions are made in Sect. 7.

2 Related Work

Many efforts have been made for regularizing neural networks. Dropout [1] is the most popular method for network regularization. It randomly drops the pre-designed portion of activations at each iteration to regularize the network. Dropout can be viewed as cooperation of multiple models trained on different subsets of data. From similar inspiration, DropConnect [7] drops the connections of the network instead of activations. It showed comparable generalization performance with dropout.

Dropout works well in practice especially with fully connected layers. However, when applied to convolutional layers in a deep CNN, the performance of dropout has been thought to be questionable. It is argued that convolutional layers does not suffer from overfitting because the number of parameters for the convolutional layers is small relative to the number of activations. Nevertheless, dropout in convolutional layer is proven to improve generalization performance in some extent by adding noise to the activations [1]. Network-in-Network [8] efficiently integrated dropout in convolutional layer by using 1×1 convolutional layer followed by dropout, which enhances both representation and generalization power. On the other hand, spatial dropout [2] has been suggested to consider

the correlated activations in convolutional layers. The method drops the entire feature maps rather than individual activations. Since spatially close activations in the same feature map are tend to be correlated, the paper argues that dropout does not suitably applied to volumetric feature map since it assumes independence between the activations.

Various pooling methods have been proposed to regularize CNNs. Stochastic pooling [3] determines the elements to pool probabilistically based on the input activation values. Generalized pooling functions [9] learn parameters to combine average and max pooling. Strided convolution [5] can also be viewed as generalization of pooling operations. Wu and Gu [10] proposed probabilistic weighted pooling which combines dropout in convolutional layers and max pooling together.

Adding noise in the training step or to the activation function also helps enhancing generalization power. Neelakantan et al. [11] found that adding noise to the gradient during backpropagation helps deep networks converge faster and prevent overfitting. Audhkhasi et al. [12] and Gulcehre et al. [13] showed that adding carefully chosen noise to the activation can speed up training procedure. Maxout networks [4] regularize networks by propagating only maximum activations. Huang et al. [14] proposed the regularization techniques which combine maxout and dropout. Opposed to the maxout networks, our method prohibits maximum activations from forward and backward propagation.

Among the numerous regularization methods, dropout is still used in most neural networks due to its simplicity and reasonable performance. Though Srivastava et al. [1] empirically proved the effectiveness of dropout in the convolutional layers, dropout is not preferable to apply every layer in a deep convolutional neural network since the scale of backpropagated error drops whenever it passes the layer with dropout, which slows down the learning speed in the lower convolutional layers. Therefore, dropout has been applied only to the fully connected networks in most cases.

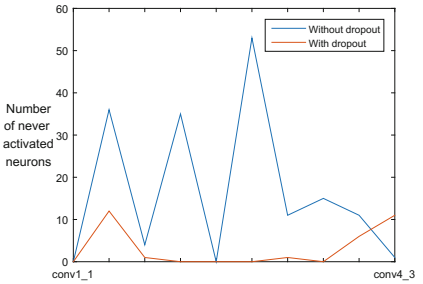
3 Effectiveness of Dropout in Convolutional Layer

We first investigate the effect of dropout in convolutional layers of CNNs. Dropout is interpreted as bagging of different models which is trained on different subsets of data. On the other hand, it is believed that the regularization effect of dropout in convolutional layers is mainly obtained from the robustness to noisy inputs. To analyze the characteristics of dropout that actually help generalizing convolutional layers, we scrutinized the distribution of activations in a CNN trained on the CIFAR-10 dataset [15] with and without dropout in convolutional layers. The network model used in this section consists of 10 convolutional layers and 4 pooling layers. All convolutional layers have kernels of 3×3 size, and inputs to the convolutional layers are padded by 1 pixels for both sides. All pooling layers use 2×2 max pooling with stride of 2 except the last layer for which we used 4×4 mean pooling. Dropout after *pool*4 with probability of 0.5 is applied regardless of using dropout in convolutional layers or not. The

number of filters is doubled after each pooling layer, which is a similar approach to the VGGnet [16]. Rectified linear unit (ReLU) is used as a activation function in all layers. Detailed configuration is illustrated in Fig. 1(a). While the CNN that does not use dropout achieved 83.16% accuracy, when dropout is applied to the output of every convolutional layer except the last *conv*4_3 layer with ratio of 0.1, the network achieved 87.78% accuracy. We analyzed the reason of accuracy improvement by looking into the behavior of the activated neurons in the convolutional layers.

name	filter size	channels
conv1_1	3×3	64
conv1_2	3×3	64
pool1	max 2×2 / 2	
conv2_1	3×3	128
conv2_2	3×3	128
pool2	max 2×2 / 2	
conv3_1	3×3	256
conv3_2	3×3	256
conv3_3	3×3	256
pool3	max 2×2 / 2	
conv4_1	3×3	512
conv4_2	3×3	512
conv4_3	3×3	512
pool4	mean 4×4	
fc-softmax	512×10	

(a)



(b)

Fig. 1. (a) Structure of CNN used in the experiments. (b) Number of neurons that never activated in each layer.

First, we investigated that every neuron in CNNs are working effectively, which means that the filters in CNNs do not learn redundant or useless information. One of the difficulties for training deep CNNs is that there exist dead neurons in the convolutional layer that are never activated. Using variants of ReLU activation functions such as leaky ReLU [17] or parametric ReLU [18] is one of the solutions to avoid dead neurons. We verify that dropout is also useful for avoiding dead neurons while the network still uses ReLU activation function. We counted the number of neurons that are not activated at the test time for each layer. The portions of never activated neurons with and without dropout are shown in Fig. 1(b). Large number of dead neurons are observed in most layers when dropout is not applied. On the other hand, when dropout is applied to the convolutional layers, almost all neurons are activated. Therefore, we verify that dropout in convolutional layers helps filters to learn informative features of images, which improves representation power of the network and classification performance as a consequence.

Next, as discussed in [1], we compared the sparsity of the activations. It is verified from [1] that in the fully connected layer, the activations are sparser when dropout is used. To confirm that this statement also holds for the convolutional layers in both lower and higher layers, we calculated the mean activation of

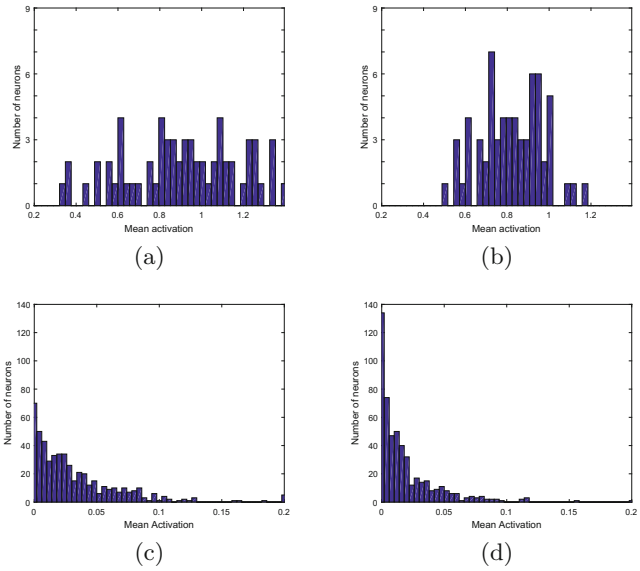


Fig. 2. Histogram of mean activation of (a) conv1_1 layer without dropout. (b) conv1_1 layer with dropout. (c) conv4_2 layer without dropout. (d) conv4_2 layer with dropout.

all neurons in each layer. The mean activation of lower convolutional layers, *conv*1_1 (Fig. 2(a) and (b)), and that of higher convolutional layers, *conv*4_2 (Fig. 2(c) and (d)) are shown. In the lower convolutional layers, the histogram is almost flat when dropout is not used while it is bell shape when dropout is applied. This indicates that some neurons are activated frequently or have larger activation values, and others are activated less frequently or have small activation values when dropout is not applied. Meanwhile, with dropout, every neuron has similar mean activation value, which means that every neuron is similarly activated. Since lower layers in CNN usually captures common features, it is preferable behavior that neurons have similar mean activation values. In the higher convolutional layer, we could verify the sparsity of activations with dropout. A high peak near zero value is observed, which implies that the mean activation of most neurons are concentrated at small values when dropout is applied.

Based on these observations, we conclude that dropout in convolutional layers helps filters to learn informative features. However, when dropout is applied to every convolutional layers in deep CNNs, training process can be slow since activation signals are dropped exponentially as dropout is applied repeatedly. If higher drop probability such as 0.5 is applied in convolutional layers, CNNs perform poor or cannot be trained at all. In the next sections, we propose two variants of dropout to deal with this problem while maintaining the competitive generalization power with dropout.

4 Max-Drop Layer

In this section, we will explain a new regularization method named as max-drop. Based on the information from Sect. 3, we note that neurons with high activation contain important information in the network. Max-drop layer selectively drops only the maximum activations. While dropout is motivated by model averaging, max-drop layers originate from different inspiration from CNNs. Different images of the same class often do not share the same features due to the occlusion, viewpoint changes, illumination variation, and so on. For instance, human face images may contain one eye or two eyes depending on the viewpoint. Therefore, a feature which plays an important role in an image may not appear in different images of the same class. Max-drop aims to simulate these cases by dropping high activations deterministically, rather than randomly select activations to drop off. In the lower layer of convolutional layers, this procedure of dropping the maximum activations simulates the case that important features are not present due to occlusion or other types of variations. In the higher layer, each feature map learns more abstracted and class-specific information [19]. Therefore, turning off high activations helps other neurons to learn the class-specific characteristics. It is intuitively uncertain that dropped high activations on the higher convolutional layers give generalization power, but we empirically prove that max-drop layers effectively regularize higher convolutional layers similar to dropout.

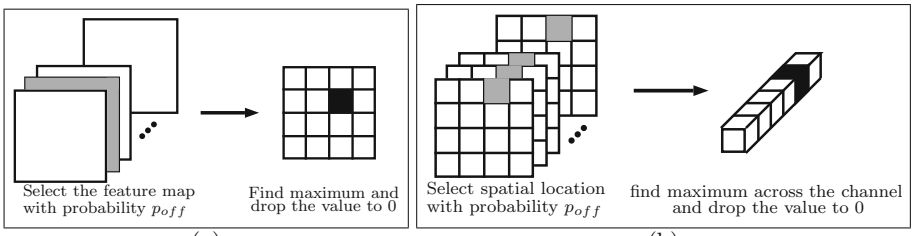


Fig. 3. Illustration of max-drop layer. Two different ways to find maximum value is proposed in this paper: (a) feature-wise max-drop finds maximum value within each feature map and drops the maximum values with probability p_{off}. (b) Channel-wise max-drop finds maximum value across each channel in the same spatial position and drops the maximum values with probability p_{off}.

In the max-drop layer, maximum element is found in the activations of convolutional layers and the maximum activation is dropped to zero with the probability of p_{off}. Max-drop can be applied to both outputs of convolutional layers or pooling layers as in the case of dropout. We propose two different strategies to find maximum value which is illustrated in Fig. 3. The first strategy is to find maximum within each feature map, which will be called as feature-wise max-drop. This scheme turns off the most informative feature within the feature map and drop the value to 0 with the probability p_{off}. The portion of dropped activations with respect to the entire activations of the convolutional layer is calculated as

$$p_f = \frac{1}{n_w \times n_f} p_{off}, \tag{1}$$

where n_w, n_h is the width and height of the feature map respectively. For instance, if convolutional layer outputs 4×4 size feature map, then the maximum probability of drop will be 0.0625, which is smaller than the typical dropout ratio.

Another strategy is to find maximum across the channels in the same position of feature map, which is denoted as channel-wise max-drop. This scheme prevents the highest activation to be propagated to the next layer on a certain spatial position of the feature map. The actual drop probability for the channel-wise max-drop is

$$p_c = \frac{1}{n_c} p_{off}, \tag{2}$$

where n_c is the number of channels in the convolutional layer outputs. With the same drop rate, channel-wise max-drop will drop smaller number of activations than the feature-wise max-drop in higher layers where the size of feature map is much smaller than the number of channels, and vice versa in the lower layers.

Dropping small number of neurons has an advantage over conventional dropout. Max-drop does not suffer from slow training since gradients are propagated through all activations except the maximum activations that are selected to turn off. Empirically, when max-drop is applied to every convolutional layer, test error decreases faster in the early stage of training than the case when dropout is applied. Moreover, with the same learning rate, the network with max-drop can be trained when high p_{off} (larger than 0.5) is used while the network with dropout usually failed to be trained when the drop probability exceeds 0.2.

5 Stochastic Dropout

If we interpret the effect of dropout as gaining robustness by putting in noisy inputs, giving different degrees of noise might be helpful. Also, it is hard to determine an appropriate drop rate for the convolutional layers in most cases. If dropout ratio is determined differently for every iteration, we believe that CNN can be learned to handle different amount of information. Based on this idea, we propose a stochastic dropout in which dropout ratio is determined from probability distribution. In stochastic dropout, probability of dropping neurons is drawn from the uniform distribution or normal distribution, i.e.,

$$p_{off} \sim N(\mu, \sigma) \quad \text{or} \quad p_{off} \sim U(a, b) \tag{3}$$

where $N(\mu, \sigma)$ is the normal distribution with mean μ and standard deviation σ, and $U(a, b)$ is the uniform distribution whose range is $[a, b]$. For our implementation, if $\mu = 0$ for normal distribution, we used the absolute value of drawn probability as p_{off}. When μ is non-zero, we set $p_{off} = 0$ if negative number is drawn.

We implemented max-drop and stochastic dropout using Caffe framework [20]. Like the dropout implementation, the activations scale up by inverse of drop probability when max-drop or stochastic dropout is applied.[1] Note that the scale factor is almost 1 for max-drop since the actual drop probability is near 0. We also found that the performance is almost the same for max-drop regardless of scale factor multiplication. GPU implementation of max-drop showed similar computation time for one iteration of backpropagation with dropout.

6 Experimental Results

We examined the regularization performance of various algorithms using MNIST [21], CIFAR-10, CIFAR-100 [15], and the street view house numbers (SVHN) [22] dataset. Max-drop and stochastic dropout are compared to dropout [1] and spatial dropout [2] to validate the generalization performance of the proposed methods against the conventional algorithms. To verify the regularization effect on the recently proposed very deep neural networks, we also conducted an experiment with ResNet [23] on CIFAR-10 dataset.

The baseline model structure of the CNN is the same as the model described in Fig. 1(a) except the MNIST dataset in Sect. 6.1 and ResNet [23] experiment in Sect. 6.2. For the MNIST dataset, the number of channels for the network is reduced from 64, 128, 256, 512 to 64, 96, 128, 256. Also, *pool*3 layer has 3×3 kernels with a stride of 2, and *pool*4 has 3×3 kernels to deal with the 28×28 input size. For ResNet experiment, we used the same 32-layer model suggested in [23] except that the number of feature maps in every convolutional layer is doubled. Mean substraction is the only preprocessing for the whole experiments.

For fair comparison, we searched the best parameter (e.g. drop probability) for each method. To ease the parameter tuning process, we applied the regularization algorithms for every convolutional layer with the same parameters. For all models in the experiments, dropout with probability of 0.5 is applied after *pool*4 and before the softmax. Dropout, spatial dropout, max-drop, or stochastic dropout is applied after every convolution layers except the last *conv*4_3 layer. When batch normalization [24] is applied, dropout after *pool*4 is removed and the regularization methods are applied after *conv*4_3. Since drop probability of max-drop has large values, we searched the parameter for max-drop with the interval of 0.1, ranging from 0.1 to 0.9, and we used the interval of 0.05 for dropout and spatial dropout, ranging from 0.05 to 0.5. We reported the selected parameter together with the regularization method.

6.1 MNIST Dataset

As a sanity check, we experimented the proposed max-drop and stochastic dropout on the MNIST dataset. MNIST has 60,000 training images and 10,000

[1] Caffe implementation of dropout scales up the activations at training time instead of scaling down them at test time unlike the original dropout paper [1].

test images with 28×28 size. We trained CNNs for 60 epochs with the initial learning rate of 0.01 and the batch size of 128. The learning rate is decreased by 0.1 for every 20 epochs. Since MNIST classification is an easy task, and the performance is highly saturated, we conducted training 5 times for each model. We reported the average classification error for each method with the standard deviation as well as the classification of the ensemble classification error by averaging the predictions of 5 models. The results are shown in Table 1.

Table 1. Classification error on MNIST

Method	Classification error (%)	
	Average of 5 models	Ensemble of 5 models
Baseline (without dropout)	0.604 ± 0.0829	0.57
Dropout ($p = 0.2$)	0.430 ± 0.0212	**0.38**
Spatial dropout ($p = 0.1$)	0.504 ± 0.0493	0.42
Feature-wise max-drop ($p = 0.2$)	0.488 ± 0.0657	0.42
Channel-wise max-drop ($p = 0.5$)	0.502 ± 0.0148	0.40
Stochastic dropout ($N(0.2, 0.05)$)	$\mathbf{0.410 \pm 0.0122}$	**0.38**
Stochastic dropout ($U(0.1, 0.3)$)	0.448 ± 0.0363	0.42

It is shown that all regularization methods significantly improve the performance of the baseline. Dropout has higher improvement on classification accuracy than max-drop. For average performance, stochastic dropout also showed the best performance.

We also analyzed the effect of regularization methods with small amount of training data. We randomly select 20% of the training images from the MNIST dataset and trained with the small dataset. The performance is illustrated in Table 2, which shows similar tendency to Table 1.

Table 2. Classification error on MNIST with 20% of training data.

Method	Classification error (%)	
	Average of 5 models	Ensemble of 5 models
Baseline	1.126 ± 0.0802	0.92
Dropout ($p = 0.2$)	0.808 ± 0.0740	0.76
Spatial dropout ($p = 0.1$)	0.872 ± 0.0335	0.78
Feature-wise max-drop ($p = 0.4$)	0.882 ± 0.0676	0.83
Channel-wise max-drop ($p = 0.5$)	0.888 ± 0.0638	0.79
Stochastic dropout ($N(0.2, 0.05)$)	0.810 ± 0.0534	0.80
Stochastic dropout ($U(0.1, 0.3)$)	$\mathbf{0.802 \pm 0.0444}$	**0.75**

In general, regularization methods reduced the classification error by 20 $\sim$ 30%. Also, standard deviation has smaller values when regularization methods are applied, which means that regularization in convolutional layers provides stable results. Dropout shows superior performance to max-drop in MNIST dataset. Though stochastic dropout works slightly better than dropout with fixed probability, it seems that giving different levels of noise does not take much advantage against dropout. Spatial dropout showed inferior performance, which indicates that independence between feature map does not play an important role in regularization of convolutional layers.

6.2 CIFAR-10 and CIFAR-100 Dataset

CIFAR-10 and CIFAR-100 datasets are image classification dataset which consist of 10 and 100 classes respectively. Each dataset has 50,000 training images and 10,000 test images with 32×32 size. For the CIFAR datasets, we reported the classification error of a single model for each method. To ensure convergence of models, we trained CNNs for 250 epochs with the initial learning rate of 0.02 and the batch size of 128. The learning rate is decreased by 0.5 for every 25 epochs.

The classification accuracy is illustrated in Table 3. In CIFAR-10, channel-wise max-drop showed better result than dropout. Note that despite the high drop probability, the actual drop probability of channel-wise max-drop is very small, about 0.01 for the first convolutional layer and about 0.001 for the last convolutional layer. The result verifies that dropping only high activations results in similar regularization effect to random drop. Also, unlike MNIST experiment, stochastic dropout of normal distribution with zero mean showed best performance. One possible interpretation is that giving different levels of noise to the input of the convolutional layers makes the layers robust to intra-class variations, thus obtaining enhanced generalization power.

Table 3. Classification error on CIFAR-10 dataset.

Method	Classification error (%)
Baseline	16.84%
Dropout ($p = 0.1$)	12.22%
Spatial dropout ($p = 0.05$)	13.78%
Feature-wise max-drop ($p = 0.2$)	12.55%
Channel-wise max-drop ($p = 0.7$)	12.00%
Stochastic dropout ($N(0.0, 0.2)$)	**11.79%**
Stochastic dropout ($U(0.0, 0.4)$)	12.86%

Next, we compared the progress of training for baseline, dropout, and channel-wise max-drop models. The losses on the training set and the test set, and the accuracies on the test set is illustrated in Fig. 4. The training losses are

plotted in log-scale (Fig. 4(a)). It is shown that the training loss of dropout and max-drop fluctuates heavily compared to the baseline model since each layer in those models takes noisy inputs. Dropout has larger variations than max-drop since the number of dropped activations is larger. These fluctuations does not affect the test loss or accuracy, as shown in Fig. 4(b). It is interesting that the test loss of max-drop is even higher than the baseline model while maintaining similar accuracy with dropout. Since max-drop drops the highest activation which contains important information, the model is learned to classify an image with less informative feature. This will increase the uncertainty of the prediction, which leads to high softmax loss values.

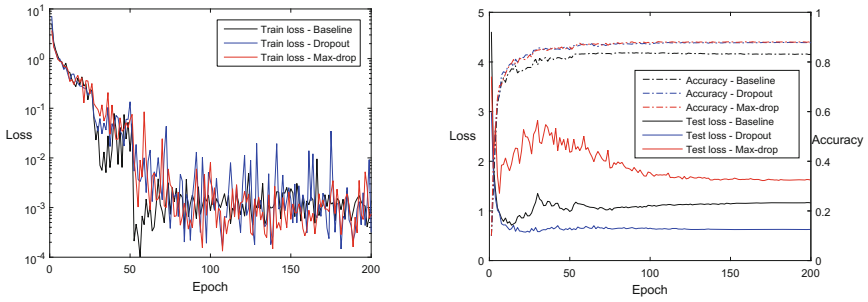


Fig. 4. (a) Training error of baseline, dropout ($p = 0.1$), and channel-wise max-drop ($p = 0.7$). (b) Test error and test set accuracy of the models.

We also analyzed the activation behavior of max-drop following the analysis of dropout in Sect. 3. The number of neurons that are never activated at the test time are counted for each regularization method and reported in Fig. 5. Similar to dropout, all regularization methods have little number of never activated neurons for all layers. Thus, it is verified that max-drop helps neurons to learn discriminative features as in the case of dropout.

The histogram of mean activation in the lower and higher convolutional layers for max-drop models are shown in Fig. 6. As observed in Fig. 6(a) and (b), the histogram is bell-shaped in the lower convolutional layer like dropout, which indicates that max-drop also make neurons evenly activated. Meanwhile, for the higher convolutional layer, number of neurons that has mean activation near zero is small unlike either dropout or no regularization case. Max-drop pushes neurons to have similar mean activations, but it does not prefer sparse activations.

To investigate the usefulness of the regularization methods in the specific layers, we trained the model by applying dropout and max-drop only to the lower layers (*conv*1_1 and *conv*1_2) and only to the higher layers (*conv*4_1 and *conv*4_2). The classification errors for both cases are shown in Table 4. Regularization methods improves the network in both lower and higher layers, but the regularization effect is more powerful in the higher layers. We found that high

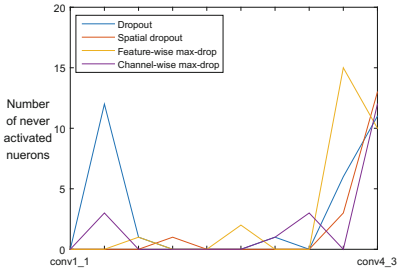


Fig. 5. Number of never activated neurons in the models with dropout, max-drop, and spatial dropout.

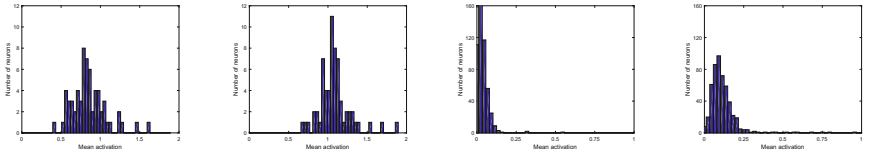


Fig. 6. Histogram of mean activation of (a) conv1_1 layer with feature-wise max-drop. (b) conv1_1 layer with channel-wise max-drop. (c) conv4_2 layer with feature-wise max-drop. (d) conv4_2 layer with channel-wise max-drop.

drop ratio is preferable for higher layers, while low drop ratio showed better performance in lower layers. Feature-wise max-drop in higher layers and channel-wise max-drop in lower layers showed better regularization performance. Spatial dropout also proved its effectiveness in higher layers.

Table 4. Effect of regularization in lower and higher convolutional layers.

Method	*conv*1 regularization		*conv*4 regularization	
	Parameter	Classification err. (%)	Parameter	Classification err. (%)
Dropout	$p = 0.05$	15.69	$p = 0.3$	15.14
Spatial dropout	$p = 0.05$	16.16	$p = 0.25$	**14.48**
Feature-wise max-drop	$p = 0.1$	15.93	$p = 0.7$	15.06
Channel-wise max-drop	$p = 0.1$	**15.02**	$p = 0.4$	15.47

Next, we combined the regularization methods with other methods that improves generalization performance. Batch normalization [24] improves training speed and the performance of network by normalizing the activations of each layer in neural networks. We applied the regularization methods after batch normalization is performed. Data augmentation is also a simple way to grant gen-

eralization power to neural networks. Following the previous works [4,8,25], we applied data augmentation to training data by padding 4 pixels on all sides of images and by flipping images horizontally. The classification errors are shown in Table 5. With batch normalization, dropout showed the best performance. After activations are normalized, it seems that the importance of maximum value is decreased, which leads to the poor generalization performance of max-drop compared to dropout or spatial dropout. With data augmentation, spatial dropout showed the smallest error, but the improvement of all regularization methods from the baseline is very small. The result indicates that data augmentation imposes generalization power to the CNNs which make the regularization methods less effective.

Table 5. Effect of regularization when combined with batch normalization and data augmentation.

Method	With batch normalization		With data augmentation	
	Parameter	Classification err. (%)	Parameter	Classification err. (%)
Baseline	-	12.10	-	8.01
Dropout	$p = 0.1$	**9.85**	$p = 0.05$	8.54
Spatial dropout	$p = 0.15$	10.69	$p = 0.05$	**7.17**
Feature-wise max-drop	$p = 0.2$	10.67	$p = 0.2$	7.73
Channel-wise max-drop	$p = 0.2$	11.15	$p = 0.4$	7.49

Recently, deep residual learning [23] enabled training of very deep networks. To investigate the regularization performance in the very deep CNNs, we trained 32-layer ResNet on CIFAR-10 dataset. We followed the training procedure and hyper parameters selection from [23] without data augmentation. The result is illustrated in Table 6. All of the tested methods showed superior performance over the baseline with a margin of $2 \sim 4\%$ except spatial dropout. This indicates that dropout and max-drop is still effective for regularizing very deep networks.

Table 6. Classification error on CIFAR-10 dataset using ResNet-32.

Method	Classification error (%)
Baseline	12.84%
Dropout ($p = 0.1$)	9.14%
Spatial dropout ($p = 0.1$)	16.33%
Feature-wise max-drop ($p = 0.2$)	10.72%
Channel-wise max-drop ($p = 0.1$)	11.15%

Lastly, we evaluated the regularization methods on CIFAR-100 dataset. The dataset has much less training samples for each class than CIFAR-10. The classification errors without data augmentation are shown in Table 7. Regularization effect is much stronger than CIFAR-10 mainly due to the small amount of training samples, which reduced the classification error up to 15%. Without batch normalization, max-drop methods outperforms dropout. When batch normalization is used, dropout shows more improvements.

Table 7. Classification errors on CIFAR-100 dataset.

Method	W/O batch normalization		W/batch normalization	
	Parameter	Classification err. (%)	Parameter	Classification err. (%)
Baseline	-	50.26	-	38.84
Dropout	$p = 0.3$	37.23	$p = 0.15$	**32.46**
Spatial dropout	$p = 0.15$	42.07	$p = 0.1$	35.28
Feature-wise max-drop	$p = 0.4$	36.22	$p = 0.2$	34.27
Channel-wise max-drop	$p = 0.7$	**35.33**	$p = 0.3$	34.71

Table 8. Classification errors on SVHN dataset.

Method	W/O batch normalization		W/batch normalization	
	Parameter	Classification err. (%)	Parameter	Classification err. (%)
Baseline	-	2.46	-	2.34
Dropout	$p = 0.25$	2.46	$p = 0.1$	**2.02**
Spatial dropout	$p = 0.05$	2.58	$p = 0.15$	2.07
Feature-wise max-drop	$p = 0.4$	**2.29**	$p = 0.2$	2.14
Channel-wise max-drop	$p = 0.4$	2.30	$p = 0.7$	2.28

6.3 SVHN Dataset

SVHN dataset contains much more training samples than the previous datasets. The dataset consists of over 600,000 training images and 26,032 test images. We trained CNN for 15 epochs with the initial learning rate of 0.01 and the batch size of 128 for the experiments on SVHN dataset. The learning rate is decreased by 0.1 for every 5 epochs. Data augmentation is not applied. The classification errors are reported in Table 8. Huge number of training samples weakens the effect of regularization. Without batch normalization, max-drop methods showed a small improvement, while dropout and spatial dropout worsen the performance of the network. Dropout showed the best performance when batch normalization is applied.

7 Conclusion

We have investigated and verified the usefulness of dropout-like methods in convolutional layers. Usage of dropout in convolutional layers is justified by looking into the activation behavior of neurons. Regularization effect in the convolutional layers is strong when training samples are small and when data augmentation is not used. Also, newly-proposed max-drop and stochastic dropout methods showed competitive results to the conventional dropout, which implies that these methods can substitute dropout in convolutional layers of CNNs. Max-drop layer can be generalized such as dropping largest k activations or suppress the activations by multiplying constant value instead of dropping them to zero. We expect that carefully adjusted parameters may increase the performance.

Acknowledgement. This research was supported by Basic Research Program through the National Research Foundation of Korea (NRF-2016R1A1A1A05005442).

References

1. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**, 1929–1958 (2014)
2. Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C.: Efficient object localization using convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 648–656 (2015)
3. Zeiler, M.D., Fergus, R.: Stochastic pooling for regularization of deep convolutional neural networks. arXiv preprint arXiv:1301.3557 (2013)
4. Goodfellow, I., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y.: Maxout networks. In: Proceedings of the 30th International Conference on Machine Learning, pp. 1319–1327 (2013)
5. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: the all convolutional net. arXiv preprint arXiv:1412.6806 (2014)
6. Graham, B.: Spatially-sparse convolutional neural networks. arXiv preprint arXiv:1409.6070 (2014)
7. Wan, L., Zeiler, M., Zhang, S., Cun, Y.L., Fergus, R.: Regularization of neural networks using dropconnect. In: Proceedings of the 30th International Conference on Machine Learning (ICML 2013), pp. 1058–1066 (2013)
8. Lin, M., Chen, Q., Yan, S.: Network in network. arXiv preprint arXiv:1312.4400 (2013)
9. Lee, C.Y., Gallagher, P.W., Tu, Z.: Generalizing pooling functions in convolutional neural networks: mixed, gated, and tree. arXiv preprint arXiv:1509.08985 (2015)
10. Wu, H., Gu, X.: Towards dropout training for convolutional neural networks. Neural Netw. **71**, 1–10 (2015)
11. Neelakantan, A., Vilnis, L., Le, Q.V., Sutskever, I., Kaiser, L., Kurach, K., Martens, J.: Adding gradient noise improves learning for very deep networks. arXiv preprint arXiv:1511.06807 (2015)
12. Audhkhasi, K., Osoba, O., Kosko, B.: Noise-enhanced convolutional neural networks. Neural Netw. **78**, 15–23 (2016)

13. Gulcehre, C., Moczulski, M., Denil, M., Bengio, Y.: Noisy activation functions. arXiv preprint arXiv:1603.00391 (2016)
14. Huang, Y., Sun, X., Lu, M., Xu, M.: Channel-max, channel-drop and stochastic max-pooling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 9–17 (2015)
15. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images (2009)
16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
17. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of the ICML, vol. 30, p. 1 (2013)
18. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1026–1034 (2015)
19. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10590-1_53
20. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: CAFFE: convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)
21. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**, 2278–2324 (1998)
22. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning. In: NIPS Workshop on Deep Learning and Unsupervised Feature Learning, Granada, Spain, vol. 2011, p. 4 (2011)
23. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
24. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on Machine Learning, pp. 448–456 (2015)
25. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.: Deeply-supervised nets. In: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, pp. 562–570 (2015)