Dynamic Selection of Image Processing Filters for Convolutional Neural Networks

Yusei Kawai†, Shinichi Shirakawa†

Yokohama National University†

1 Introduction

Convolutional neural networks (CNNs) have shown the remarkable performance in computer vision tasks [1]. In image classification tasks, the CNN represents the mapping from an input image to the class probabilities, and the weight parameters in the CNN are optimized in the training phase. Although a raw image or a normalized image is usually input to the CNN, adding the pre-processed images as the input could contribute to improving the predictive performance.

In this report, we consider adding the images transformed by the typical image processing filters such as an edge detection filter as the input images. Since the computational cost of almost the image processing filters is much smaller than that of the convolution operation in CNN, it might help to reduce the number of the convolution layers and lead to reduce the overall computational cost of the CNNs without the performance deterioration.

As the appropriate filters must depend on the task, we propose the method for dynamically selecting the input images during the training based on the technique introduced in [2, 3]. In the proposed method, the binary vector is sampled from the multivariate Bernoulli distribution and decides the selection of the pre-processed images to be input. Considering to optimize the distribution parameters instead of directly optimizing the binary vector, we can update the weights and the distribution parameters by a gradient method at the same time. We apply the proposed method to the image classification tasks using several image datasets and show that the proposed method can improve the predictive performance compared to the method without image processing filter selection.

We note that the aim of our proposed method is different from the so-called data augmentation techniques [1, 4, 5] which transform the input images by randomly cropping and flipping to artificially increase the number of data. Our method can be regarded as attempting to increase the model complexity by adding the transformed images to inputs instead of adding the convolution layers.

2 Dynamic Selection of Image Processing Filters

The method for dynamically selecting the image processing filters is based on the embedded feature selection technique proposed in [3]. Assuming the d candidate input image planes transformed by the image processing filters, the selection of the filter can be represented by a d-dimensional binary vector $M = (m_1, ..., m_d)^{\mathrm{T}}$, $m_i \in \{0, 1\}$. Introducing the multivariate Bernoulli distribution for the random variable of M, the probability mass function is represented by $p_\theta(M) = \prod_{i=1}^{d} \theta_i^{m_i}(1-\theta_i)^{1-m_i}$, where $\theta = (\theta_1, ..., \theta_d)^{\mathrm{T}}$, $\theta_i \in [0, 1]$ indicates the parameters of the distribution. The value of θ_i can be regarded as the selection probability of the i-th filter. Figure 1 shows the conceptual illustration of the proposed CNN model with the filter selection.

We then represent the overall function consisting of the filter selection and CNN operation as $\phi(W, M)$ modeled by two parameter vectors, the weight vector $W \in \mathcal{W}$ in the CNN and the binary vector $M \in \mathcal{M}$ for the filter selection. Following [2], we consider to minimize the following expected loss $\mathcal{L}'(W, \theta)$ under $p_\theta(M)$, instead of directly optimizing the loss function $\mathcal{L}(W, M)$.

$$\mathcal{L}'(W, \theta) = \sum_{M \in \mathcal{M}} \mathcal{L}(W, M) p_\theta(M), \qquad (1)$$

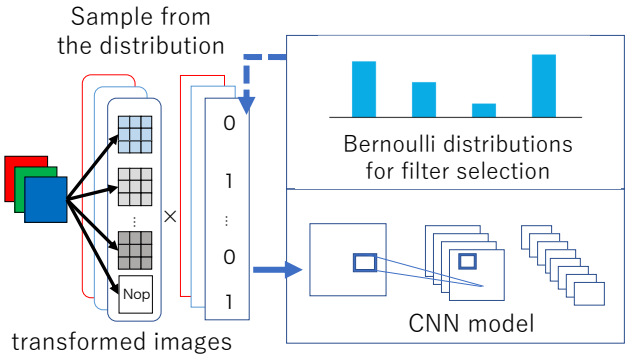

Fig. 1 Conceptual illustration of the proposed CNN model with the filter selection.

where $\mathcal{L}(W,M) = \int_{\mathcal{D}} l(x,W,M)p(x)\mathrm{d}x$, $\mathcal{D}$ indicates the dataset, and $l(x,W,M)$ is the loss function for a given data x. When we adopt the mini-batch for training, the loss $\mathcal{L}(W,M)$ is approximated using a mini-batch $\mathcal{Z}$ of N training data. The approximated loss function is given by $\bar{\mathcal{L}}(W,M) = N^{-1}\sum_{z\in\mathcal{Z}} l(z,W,M)$.

As done in [2, 3], we simultaneously update the parameters W and θ along the gradient directions. The gradient of W is approximated using λ samples as

$$\nabla_W \mathcal{L}'(W,\theta) \approx \frac{1}{\lambda}\sum_{i=1}^{\lambda} \nabla_W \bar{\mathcal{L}}(W,M_i). \tag{2}$$

The gradient $\nabla_W \bar{\mathcal{L}}(W,M_i)$ can be computed by the back-propagation, and we can use any stochastic gradient optimizer for optimizing W. Also, the natural gradient [6] of θ is approximated as

$$\tilde{\nabla}_\theta \mathcal{L}'(W,\theta) \approx \frac{1}{\lambda}\sum_{i=1}^{\lambda} \bar{\mathcal{L}}(W,M_i)\tilde{\nabla}_\theta \ln p_\theta(M_i), \tag{3}$$

where $\tilde{\nabla}_\theta \ln p_\theta(M) = F^{-1}(\theta)\nabla_\theta \ln p_\theta(M)$ is the natural gradient of the log-likelihood $\ln p_\theta(M)$, and $F(\theta)$ is the Fisher information matrix of $p_\theta(M)$. In the case of the Bernoulli distribution we consider here, the natural gradient is given by $\tilde{\nabla}_\theta \ln p_\theta(M) = M - \theta$. Also, we transform the loss value into the ranking-based utility as done in [2, 3] as follows:

$$\bar{\mathcal{L}}(W,M_i) \mapsto u_i = \begin{cases} 1 & (\text{best } \lceil \lambda/4 \rceil \text{ samples}) \\ -1 & (\text{worst } \lceil \lambda/4 \rceil \text{ samples}) \\ 0 & (\text{otherwise}) \end{cases}. \tag{4}$$

With this utility transformation, the θ update becomes

$$\theta^{t+1} = \theta + \frac{\eta_\theta}{\lambda}\sum_{i=1}^{\lambda} u_i(M_i - \theta^t), \tag{5}$$

where η_θ is the learning rate for θ. The update rule of Eq. (5) is the same as the population-based incremental learning (PBIL) [7], but we simultaneously update W with a different mechanism. We use $\lambda = 2$ and $\eta_\theta = 1/(d\sum|u_i|)$ for all experiments. Also, we restrict the range of θ_i within $[1/d, 1-1/d]$ to keep the possibility of generating any binary vector.

Following the literature [3], although the binary vector is sampled from the distribution in the training phase, it is fixed in the testing phase as $m_i = 0$ if $\theta_i < 0.5$ and $m_i = 1$ if $\theta \geq 0.5$ to obtain the deterministic prediction.

3 Experiments and Results

We evaluate the proposed dynamic filter selection method on three image classification tasks using the CIFAR-10, SVHN, and MIT scene image datasets. The information of each dataset is summarized in Table 1. We prepare the 12 image processing filters including raw images (no operation) shown in Table 2. We apply the image processing filters to each input channel of RGB and obtain 36 candidate input image planes (i.e., $d = 36$).

We adopt the ResNet [8] as the CNN model in the proposed method. The ResNet is one of the popular CNNs for image classification and has the skip connection between the layers. The ResNet stacks the residual building blocks consisting of two convolution layers and the shortcut connection. We can construct the deeper CNN architecture by increasing the building blocks. To investigate the effect of the image processing filters, we vary the depth of the ResNet. For the CIFAR-10 and SVHN datasets, we use the 8, 14, and

Table 1 Summary of the datasets used in the experiments.

Dataset	No. of data (training / test)	No. of class	Image size (width × depth)
CIFAR-10[†]	50000 / 10000	10	32 × 32
SVHN[††]	73257 / 26032	10	32 × 32
MIT Scene Image[‡]	2366 / 338	8	256 × 256

Table 2 List of the image processing filters used in the experiments.

Name	Description
Mean3	Mean filter with 3 × 3 window
Mean5	Mean filter with 5 × 5 window
SobH	Sobel filter (horizontal direction)
SobV	Sobel filter (vertical direction)
Lap3	Laplacian filter with 3 × 3 window
Lap5	Laplacian filter with 5 × 5 window
Gau3	Gaussian filter with 3 × 3 window
Gau5	Gaussian filter with 5 × 5 window
Bil	Bilateral filter
Sha4	Sharpening filter for 4 direction
Sha8	Sharpening filter for 8 direction
Nop	output raw image (no operation)

Table 3 Mean test error (%) over 5 trials for the CIFAR-10 dataset.

	Proposed	All Select	No filters
ResNet8	30.5	31.1	35.6
ResNet14	28.4	28.2	31.2
ResNet20	27.1	27.2	30.1

Table 4 Mean test error (%) over 5 trials for the SVHN dataset.

	Proposed	All Select	No filters
ResNet8	15.3	16.1	26.5
ResNet14	13.9	16.3	21.1
ResNet20	13.5	15.5	18.5

20 layered ResNets, denoted ResNet8, ResNet14, and ResNet20, respectively. These models contain four down sampling convolution operations (i.e., the convolution operation with stride 2). For the MIT scene image dataset, we use the 18, 34, and 50 layered ResNets, denoted ResNet18, ResNet34, and ResNet50, respectively. These models contain five down sampling convolution operations.

In all experiments, we use the SGD with the Nesterov momentum of 0.9 and the weight decay of 10^{-4} to train the weight parameters. The weight parameters are initialized by He's initialization[9]. The number of the training iterations is set to $700,000$, and the learning rate for W is divided by 10 at 1/2 and 3/4 of the maximum number of iterations. This setting is based on the literature [8]. Note that we use the raw image data without any pre-processing (e.g., scaling) and do not apply any data augmentation technique.

We compare the proposed method with the following two baseline algorithms.

All Select the algorithm using all filtered images as inputs

No filters the algorithm using the only raw image as inputs (i.e., usual CNN)

Table 3 to 5 show the test error of each method for the CIFAR-10, SVHN, MIT scene image datasets, respectively. For the CIFAR-10 and SVHN datasets, we observe that the proposed method and the algorithm using all filters outperform the algorithm without filters. We also observe that the proposed method can improve the performance as the number of layers increases. These results show that adding the transformed images as the inputs has the possibility to reduce the test errors. Especially in the SVHN dataset, the proposed method succeeded to improve the prediction performance compared with the algorithm using all filters. Therefore, we can see the dynamical selection of the filters works efficiently in this case. On the other hand, the performance improvement of the proposed method is not magnificent on the MIT scene image dataset, but the performance does not deteriorate.

[†] https://www.cs.toronto.edu/~kriz/cifar.html
[††] http://ufldl.stanford.edu/housenumbers/
[‡] http://cvcl.mit.edu/database.html

Table 5 Mean test error (%) over 5 trials for the MIT scene image dataset.

	Proposed	All Select	No filters
ResNet18	16.4	18.2	16.5
ResNet34	14.3	15.8	15.1
ResNet50	14.1	14.5	14.2

Table 6 An example of the optimized selection probabilities for each filter (i.e., the value of θ) when using ResNet8 for the SVHN dataset. The symbols, R, G and B, mean the red, green, and blue channels, respectively.

Filter name	R	G	B
Mean3	0.92	0.03	0.40
Mean5	0.92	0.28	0.39
SobH	0.25	0.93	0.29
SobV	0.03	0.03	0.89
Lap3	0.94	0.06	0.06
Lap5	0.53	0.94	0.03
Gau3	0.25	0.51	0.03
Gau5	0.97	0.67	0.19
Bil	0.86	0.54	0.71
Sha4	0.38	0.19	0.07
Sha8	0.81	0.04	0.88
Nop	0.96	0.97	0.97

Table 6 displays the selection probabilities for each filter (i.e., the value of θ) after the typical training when ResNet8 is used for the SVHN dataset. We observe that the raw images (Nop) are selected with high-probability and 17 filters are used to predict the test data. We can observe the similar tendency for the values of θ after the training in other trials for the SVHN dataset.

4 Conclusion

In this report, we have proposed a method that dynamically optimizes the selection of image processing filters for pre-processing the input images during the weight training of the CNNs. The experimental results show that the proposed method can select the appropriate image processing filters and improve the predictive performance. Additionally, the proposed method selected the different image processing filters for each channel. While we did not evaluate the computational cost for the training and prediction in the experiment, the additional computational cost of the proposed method is not so expensive because our method just adds the transformed images as the inputs. In future works, we will evaluate the performance of the proposed approach in terms of the accuracy and computational complexity. Also, we plan to develop the method that can dynamically construct more complicated image transformation (e.g., the sequence of the image processing filters).

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pp. 1106–1114, 2012.

[2] S. Shirakawa, Y. Iwata, and Y. Akimoto, "Dynamic Optimization of Neural Network Structures Using Probabilistic Modeling," in *32nd AAAI Conference on Artificial Intelligence (AAAI 2018)*, pp. 4074–4082, 2018.

[3] S. Saito, S. Shirakawa, and Y. Akimoto, "Embedded Feature Selection Using Probabilistic Model-Based Optimization," in *Student Workshop in Genetic and Evolutionary Computation Conference 2018 (GECCO 2018)*, pp. 1922–1925, 2018.

[4] K. Simonyan, and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *3rd International Conference on Learning Representations (ICLR 2015)*, 2015.

[5] T. DeVries, and G. W. Taylor, "Improved Regularization of Convolutional Neural Networks with Cutout," *preprint arXiv:1708.04552*, 2017.

[6] S. Amari, "Natural gradient works efficiently in learning," *Neural Computation* 10(2): 251–276, 1998.

[7] S. Baluja, "Population-based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning," *Technical Report Tech Rep CMU-CS-94-163*, Carnegie Mellon University, 1994.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, pp. 770–778, 2016.

[9] K. He, X. Zhang, S. Ren, and J. Sum, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," in *2015 IEEE International Conference on Computer Vision(ICCV)*, pp. 1026–1034, 2015.