Relay Backpropagation for Effective Learning of Deep Convolutional Neural Networks

Li Shen[1,2], Zhouchen Lin[3,4,⋆], Qingming Huang[1]

[1] University of Chinese Academy of Sciences [2] University of Oxford
[3] Key Lab. of Machine Perception (MOE), School of EECS, Peking University
[4] Cooperative Medianet Innovation Center, Shanghai Jiaotong University
lishen@robots.ox.ac.uk zlin@pku.edu.cn qmhuang@ucas.ac.cn

Abstract. Learning deeper convolutional neural networks has become a tendency in recent years. However, many empirical evidences suggest that performance improvement cannot be attained by simply stacking more layers. In this paper, we consider the issue from an information theoretical perspective, and propose a novel method *Relay Backpropagation*, which encourages the propagation of effective information through the network in training stage. By virtue of the method, *we achieved the first place in ILSVRC 2015 Scene Classification Challenge.* Extensive experiments on two challenging large scale datasets demonstrate the effectiveness of our method is not restricted to a specific dataset or network architecture.

Keywords: Relay Backpropagation, Convolutional Neural Networks, Large scale image classification.

1 Introduction

Convolutional neural networks (CNNs) are capable of inducing rich features from data, and have been successfully applied in a variety of computer vision tasks. Many breakthroughs obtained in recent years benefit from the advances of convolutional neural networks [1,2,3,4], spurring the research of pursuing a high performing network. The importance of network depth is revealed in these successes. For example, compared with AlexNet [1], the utilisation of VGGNet [5] brings about substantial gains of accuracy on 1000-class ImageNet 2012 dataset by virtue of deeper architectures.

Increasing the depth of network has become a promising way to enhance performance. On the downside, such a solution is accompanied by the growth of parameter size and model complexity, thus poses great challenges for optimisation. The training of deeper networks typically encounters the risk of divergence or slower convergence, and prone to overfitting. Besides, many empirical evidences [5,6,7] (e.g., the results reported by [5] on ImageNet dataset shown in Table 1 (Left)) have suggested the improvement on accuracy cannot be trivially

[⋆] Corresponding author.

Model	ImageNet 2012	
	top-1 err.	top-5 err.
VGGNet-13	28.2	9.6
VGGNet-16	26.6	8.6
VGGNet-19	26.9	8.7

Model	Places2 challenge	
	top-1 err.	top-5 err.
VGGNet-19	48.5	17.1
VGGNet-22	48.7	17.2
VGGNet-25	48.9	17.4

Table 1. Error rates (%) on ImageNet 2012 classification and Places2 challenge validation set. VGGNet-22 and VGGNet-25 are obtained by simply adding 3 and 6 layers on VGGNet-19, respectively.

gained by simply adding more layers. It is in accordance with the results in our preliminary experiments on Places2 challenge dataset [8], that deeper networks even suffer from a decline on performance (in Table 1 (Right)).

To interpret the phenomenon, we should be concerned with the possibility of vanishing and exploding gradient, i.e., the crucial reasons that hamper the training of very deep networks with backpropagation [9] (BP) algorithm, as gradients might be prone to become either very small or very large through many layers. To investigate whether vanishing and exploding problems appear, we analyse the scale of the gradients at different convolutional layers during training, where takes the 22-layer CNN model (in Table 1) as an example, shown in Fig. 1. The average magnitude of gradients and their relative values with respect to weights are displayed, respectively. It can be observed that the gradient magnitude of lower layers does not tend to vanish or explode, but remain approximately stable in the progression. In practice, the issues of vanishing and exploding gradient have been largely coped with by aid of some techniques, e.g., rectifier neuron [10,11], refined initialisation scheme [12,7], and Batch Normalization [13].

However, from an information theoretical perspective [14,15,16], the amount of information derived from target outputs diminishes during propagation, although the gradient magnitude does not vanish. Such degradation would be amplified as network goes deeper. In order to effectively update network parameters, the error information should not go back too many layers. However, the problem is inevitable when optimising a very deep network with standard backpropagation algorithm.

address the problem, in this paper we propose a novel method, *Relay Backpropagation* (Relay BP) for training, which encourages *effective information* to pass through the network, i.e., declining the information information flow derived from supervision signals To accomplish the aim, the whole network is first divided into several segments. We introduce one or multiple interim output modules (including loss layer) after intermediate segments, and aim to minimise the ensemble of losses. More importantly, the gradients from different losses are propagated to the lowest layers of respective segments, namely, the gradient with respect to certain loss will propagate at most N consecutive layers, where N is smaller than realistic network depth. An example framework is depicted in Fig. 2 with two auxiliary output modules. In a word, we provide an elegant way to effectively preserve relevant information by shortening the path from out-

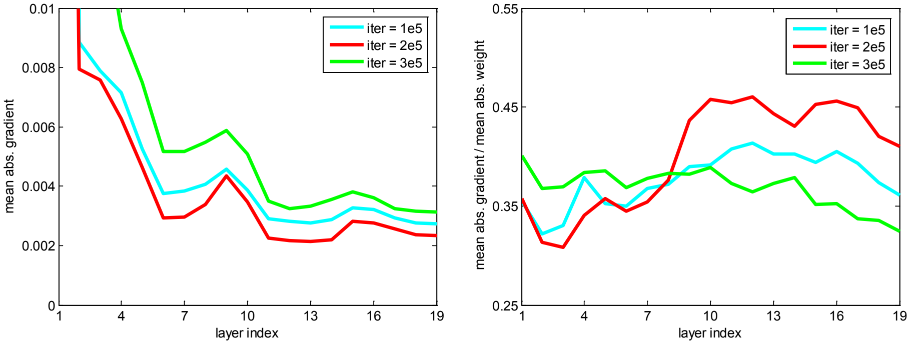

Fig. 1. Magnitude of the gradient at each convolutional layer of the 22-layer CNN model (i.e., 19 convolutional layers and 3 fully connected layers) in Table 1. A colour line plot the gradient magnitude of different layers at a certain number of iterations. (Left) Average magnitude of gradients. (Right) Relative magnitude of gradients, i.e., average magnitude of gradients divided by average magnitude of weights.

puts to lower layers, and meanwhile restrain the adverse effect of less relevant information which propagated through too many layers.

By virtue of Relay BP, we achieve the first place in ILSVRC 2015 Scene Classification Challenge, which provides a new large scale dataset involving 401 classes and more than 8 million training images. The benefits of the method are also verified on ImageNet 2012 classification dataset with another two famous network architectures, which demonstrates our method is not constrained to a specific architecture or dataset. We will make our models available to the research community.

2 Related Work

Convolutional neural networks have attracted much attention over the last few years. For image classification tasks with large scale data [17,18,19], there is a tendency of increasing the network complexity (e.g., the depth [5] and the width [20]), which brings about the difficulties of training the network. A range of techniques are exploited to address the problem by taking various angles. For example, Simonyan and Zisserman [5] propose to reduce the risk of divergence by initializing a deeper network with the aid of pre-training shallower ones. Refined initialization schemes are adopted to train very deep networks directly by drawing the weights from properly scaled distributions [12,7]. Moreover, the benefits of new activation functions [10,7,21] for training deep networks have been shown in extensive experiments. Besides, some studies are developed in the direction of finding better optimizers, such as stochastic gradient descent with momentum [22] and RMSProp [23], which is widely used and works well in practice.

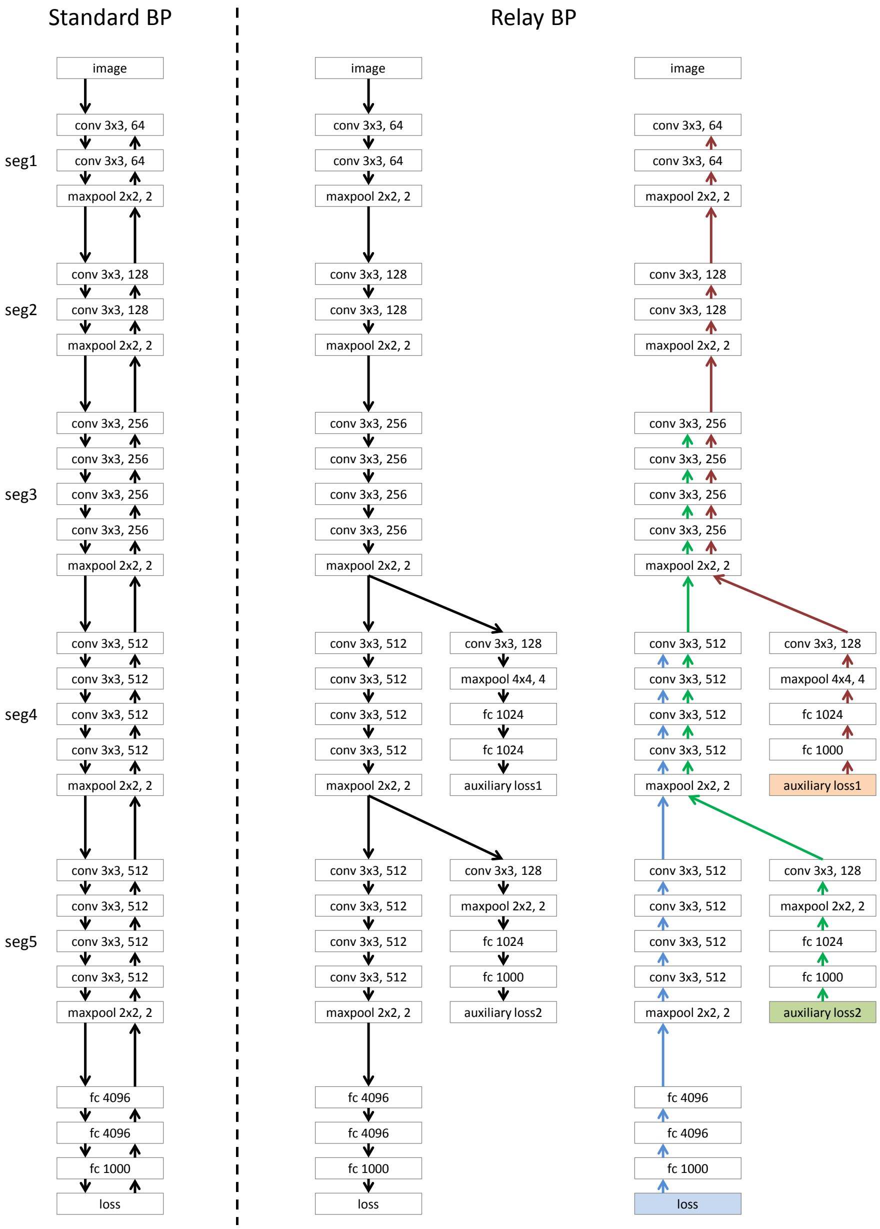

Fig. 2. (Left) VGGNet-19 network [5] with standard backpropagation algorithm. (Middle & Right) VGGNet-19 extended network with Relay backpropagation algorithm. This is an example with two auxiliary output modules, adding two branches on traditional VGGNet-19 architecture. The black arrows denote the forward propagation of information through the network, and the color arrows indicate the information (gradient) flows at backward propagation. This figure is best viewed on the screen.

It is particularly worthy of comparing our method with the work in [24,25], where temporary branches including classifiers are attached to intermediate layers, and helps to propagate the supervision information to lower layers with shortcuts. However, such multi-loss mechanism neglects information reduction due to long-term propagation, and the adverse effect of less relevant information for lower layers. Different from it, our method can effectively preserve relevant information and meanwhile restrain the adverse effect of less relevant information, thus obtain a model with better performance. In [26,27], powerful networks are obtained by adopting new structures, i.e., Inception module and Residual block, which are concurrent with our work, and also attend the ILSVRC 2015 Challenge. The two structures implement shortcut connections in different ways, however long-term propagation still exists when training a deeper network. Therefore, our contribution is orthogonal to these work, and network performance can be further improved benefitting from our method.

3 Standard BP and Information Reduction

Considering a feedforward neural network, which is comprised of L parameterised layers with weights W, each layer $l \in \{1, \cdots, L\}$ is followed by a non-linear transformation on its input variables h_{l-1} to yield the output h_l, i.e., $h_l = f_l(h_{l-1}; W_l)$, which is the input of consecutive layer $l+1$. The network receives sample $x \in \mathcal{X}$ as the starting input, i.e., h_0, and is learned to minimise the loss between the final output h_L and desired target $y \in \mathcal{Y}$, $\ell(y, h_L)$, over data $(\mathcal{X}, \mathcal{Y})$. Different transformations could be applied in the network, for clarity we omit the subscript of f.

When training with standard BP algorithm, optimisation at each iteration is comprised of forward propagation and backward propagation (as shown in Fig. 2 (Left)). The process of forward propagation is to feed x into and forward propagate through the network. Error is then generated, and the gradients with respect to neurons h and weights W are propagated with backward recurrence,

$$g^h_{l-1} = \frac{\partial \ell}{\partial h_{l-1}} = \frac{\partial f(h_{l-1}; W_l)}{\partial h_{l-1}} \frac{\partial \ell}{\partial h_l} = \delta^h g^h_l, \tag{1}$$

$$g^w_l = \frac{\partial \ell}{\partial W_l} = \frac{\partial f(h_{l-1}; W_l)}{\partial W_l} \frac{\partial \ell}{\partial h_l} = \delta^w g^h_l, \tag{2}$$

until the lowest layer (i.e., the first convolutional layer), which is backward propagation. Weights are updated according to the respective gradients on samples (i.e., batches of data). In other words, the information of error received by lower layers has flowed through many intermediate layers, whose number arises along with growth of network depth.

From an information theoretical point, the flow of information through network forms a Markov chain. The gradient g^h_L from topmost layer preciously represents the supervision signal (loss), which is transmitted to g^h_{L-1} and g^h_{L-2} in turn. According to Data Processing Inequality [15], $I(g^h_L; g^h_{L-1}) \geq I(g^h_L; g^h_{L-2})$, the amount of information about the signal is unable to increase during processing, and without attenuation only when the transformation is invertible. In practice, when information flow go across a series of layers, the amount of information is prone to reduction due to complicated transformations (e.g., ReLU and pooling), which implies less relevant information derived from loss is received at lower layers, making it difficult to leverage meaningful gradient for weight update. Such effect will amplify when information propagates deeper, ultimately hamper the performance of the whole network. In order to effectively update network parameters, information flow should not go back too many layers.

4 Relay BackPropagation

The motivation of our method is to propagate effective information through the network in backward propagation. We accomplish the target by using auxiliary output modules appropriately. Take VGGNet-19 network architecture for example, as shown in Fig. 2 (right). The whole network is first divided into several segments separated with max-pooling layers. For instance, from the first convolutional layer to the first max-pooling layer is considered as a segment, and the next segment starts from the third convolutional layer and ends to the second max-pooling layer. Thus, there are totally five segments, numbered 1 to 5 from lower to higher layers.

We attach one or multiple auxiliary output modules to intermediate segments. Fig. 2 is an example with two output modules (i.e., auxiliary loss 1 and loss 2), which are added after segment 3 and 4, respectively. In order to preserve the relevant information about loss, the gradient flows in the whole network are blocked, and one derived from each loss is required to go across at most N consecutive layers, where N is the upper limit of the numbers of layers that we deem that can carry enough relevant information. Namely, different losses are responsible for different parts of weight layers in the network. The information flows from different losses are represented with different colours in Fig. 2. Auxiliary loss 1 (coloured with red) would be propagated until the lowest one in segment 1, and auxiliary loss 2 (coloured with green) would be propagated until the lowest one in segment 3, and the primary loss (coloured with blue) would be propagated until the lowest layer in segment 4, respectively. On the other hand, it is equivalent to apply different step sizes for the flows in such a framework, and the size for the flow derived from primary loss can be regarded as zero at the lower segments, which is not based on gradient magnitude as adaptive optimisation methods (e.g., ADAM and RMSprop) modulated.

More importantly, there is overlapping between information flows at intermediate segments, such as segment 4 receives the information from primary loss and auxiliary loss 2. As our optimisation objective is to minimise the sum of the three losses, the updating on segment 4 would fuse the information passed through from the two losses. Consequently, segment 4 plays the role of the transition between the two information flows of primary loss and auxiliary loss 2, not only the transition between lower and higher layers trivially, that is why we call the method as *Relay Backpropagation*. The back-forwad step can be interpreted as update with multiple gradient flows respectivley across shorten networks, whereas jointly passing through the entire one. The lower layers seems to be isolated from the top-most loss, however, other information flow with identical target would affect them.

In summary, our method is characterised with two points: (1) Each loss (including the main and auxiliary ones) is responsible for the update of different layers in the network, i.e., a shorten sub-network, rather than all the layers below. Such mechanism is helpful to reduce the degradation of relevant information about loss and restrain the adverse effect of less relevant information due to long-term propagation. It is distinctively different from traditional multi-loss with standard BP algorithm [24,25], where the lower layers would be affected by diffuse ones. (2) Information flows from different losses exist overlapping at intermediate segments, which guarantees to coordinate information propagation in a very deep network.

In forward propagation step, information transmission follows the manner from input to output layers, where the activations generated at one layer are fed into its adjacent layer in turn. The black arrows in Fig. 2 (middle) indicate the directions of information flows through the network. It is consistent with standard BP for the network with auxiliary branches.

When testing an image, a prediction is made without considering auxiliary branches, as auxiliary supervision is introduced only to enhance the training of network. Consequently, there is no extra cost (parameter size and time expense) brought in testing stage, ensuring the test efficiency of model.

One might be concerned with: Where to add auxiliary output module? And which segments (or convolutional layers) should belong to the scope of certain loss? We apply the heuristic scheme based on empirical evidences in this work. Nevertheless, some intuitive rules can be considered for guidance. One insight is that it is inadvisable to add auxiliary output modules at too lower layers, since the patterns captured at these layers lack of sufficient discrimination for recognising a high-level concept (e.g., object or scene). Moreover, the depth of a network is an important factor to be considered. Adding an auxiliary branch might be enough if the network is not too deep. In general, the design can be adjusted flexibly according to specific requirements and practical experience.

5 Experiments

In this section, we evaluate Relay BP on Places2 challenge [8] and ImageNet 2012 classification dataset [19], and also investigate it on four different network architectures. We show Relay BP outperforms baselines significantly. The baseline methods are briefly introduced below:

- **Standard BP:** Given the network, information forward and backward propagation follow the rule of traditional backpropagation algorithm (e.g., in Fig. 2(Left)).

- **Multi-loss + standard BP:** One auxiliary output module (branch) is attached to parts of intermediate layers.

For a fair comparison, the network architecture in training stage (i.e., the architecture with temporary branches) is identical for our method and the baseline of multi-loss with standard BP. The difference lies in the scheme of information backward propagation. In the experiments, we only add one auxiliary branch for all architectures, as they are not too deep to tackle. Moreover, the increment of branch also brings about training computation cost. Therefore, the principle is adding the branches as few as possible. We intend to train extremely deeper networks by aid of multiple branches in future work.

5.1 Places2 Challenge

We evaluate our method on the Places2 challenge dataset [8], which is used in ILSVRC 2015 Scene Classification Challenge. This dataset includes images belonging to 401 scene categories, with 8.1M images for training, 20K images for validation and 381K images for testing. To mimic the real-world frequencies of scene occurrence, there is a non-uniform distribution of images per category for training, ranging from 4,000 to 30,000. The classification performance of the challenge is evaluated using the top-5 error, which allows an algorithm to identify multiple scene categories for an image, because a scene is likely to be described with different words.

Network Architectures. Relay BP is independent on the network architectures used. We investigate two types of deep convolutional neural network architectures on the Places2 challenge dataset, as shown in Table 2. The model A is based on VGGNet-19 [5], and simply adds 3 convolutional layers on the three smaller feature maps (56, 28, 14). The model B uses a 7×7 convolutional layers and a modified inception module as building block. We also incorporate spatial pyramid pooling (spp) [28] into the models, where the pyramid configuration is 7×7, 3×3, 2×2 and 1×1. Dropout regularization is applied to the first two fully-connected (fc) layers, with the dropout ratio 0.5. We use Rectified Linear Unit (ReLU) as nonlinearity and do not use Batch Normalization [13] in the two networks. The experiments involving Batch Normalization will be seen in Section 5.2. The auxiliary classifier ② is used in multi-loss standard BP and Relay BP, rather than standard BP. The loss weight of the auxiliary classifier is set to 0.3. The "gradient" in Table 2 shows the details of backward propagation in Relay BP.

Class-aware Sampling. The Places2 challenge dataset has more than 8M training images in total. The numbers of images in different classes are imbalanced, ranging from 4,000 to 30,000 per class. The large scale data and non-uniform class distribution pose great challenges for model learning.

input size	gradient	model A	model B
224×224	②	[conv 3×3, 64] × 2 maxpool 2×2, 2	[conv 7×7, 128, stride 2] × 1
112×112	②	[conv 3×3, 128] × 2 maxpool 2×2, 2	maxpool 2×2, 2
56×56	②	[conv 3×3, 256] × 5 maxpool 2×2, 2	[modified inception, k 64] × 4 maxpool 2×2, 2
28×28	①②	[conv 3×3, 512] × 5 maxpool 2×2, 2	[modified inception, k 128] × 4 maxpool 2×2, 2
-	-	auxiliary classifier ②	
14×14	①	[conv 3×3, 256] × 5 spp, {7, 3, 2, 1}	[modified inception, k 128] × 4 spp, {7, 3, 2, 1}
-	①	fc 4096	
-	①	fc 4096	
-	①	fc 401, classifier ①	

Table 2. Architectures of the networks used for ILSVRC 2015 Scene Classification. The convolutional layer is denoted as "conv <receptive field>, <filters>". The max-pooling layer is denoted as "maxpool <region size>, <stride>". Our modified inception module concatenates the outputs of a 1×1 convolution with k filters, a 3×3 convolution with k filters and two 3×3 convolution with 2k filters. ① and ② indicate which layers the gradients propagate to.

To address this issue, we apply a sampling strategy, named "class-aware sampling", during training. We aim to fill a mini-batch as uniform as possible with respect to classes, and prevent the same example and class from always appearing in a permanent order. In practice, we use two types of lists, one is class list, and the other is per-class image list, i.e., 401 per-class image lists in total. When getting a training mini-batch in an iteration, we first sample a class X in the class list, then sample an image in the per-class image list of class X. When reaching the end of the per-class image list of class X, a shuffle operation is performed to reorder the images of class X. When reaching the end of class list, a shuffle operation is performed to reorder the classes. We leverage such a class-aware sampling strategy to effectively tackle the non-uniform class distribution, and the gain of accuracy on the validation set is about 0.6%.

Training and Testing. Our implementation is based on the publicly available library Caffe [29], where function "BackFromTo" is called for each loss. Compared to standard BP, weight gradient blob are accumulated thus there is no extra memory cost, and data gradient blob can be reused by different flows by simply modifying "split_layer", which has no extra memory cost. Overall, like multi-loss method, a bit memory cost is needed for auxiliary branches compared to standard BP.

Method	model A		model B	
	top-1 err.	top-5 err.	top-1 err.	top-5 err.
standard BP	50.91	19.00	50.62	18.69
multi-loss + standard BP	$50.72_{(0.19)}$	$18.84_{(0.18)}$	$50.59_{(0.03)}$	$18.68_{(0.01)}$
Relay BP	$49.75_{(1.16)}$	$17.83_{(1.17)}$	$49.77_{(0.85)}$	$17.86_{(0.83)}$

Table 3. Single crop error rates (%) on Places2 challenge validation set. In the brackets are the improvements over "standard BP" baseline.

Method	model A		model B	
	top-1 err.	top-5 err.	top-1 err.	top-5 err.
standard BP	48.67	17.19	48.29	16.89
multi-loss + standard BP	$48.55_{(0.12)}$	$17.05_{(0.14)}$	$48.27_{(0.02)}$	$16.89_{(0.00)}$
Relay BP	$47.86_{(0.81)}$	$16.33_{(0.86)}$	$47.72_{(0.57)}$	$16.36_{(0.53)}$

Table 4. Single model error rates (%) on Places2 challenge validation set. In the brackets are the improvements over "standard BP" baseline.

We train models on the provided Places2 challenge training set, do not use any additional training data. The image is resized isotropically so that its shorter side is 256. To augment the training set, a 224×224 crop is randomly sampled from a training image, with the per-pixel mean subtracted. The random horizontal flipping and standard colour shift in [1] are used. We initialise the weights using [7] and train all networks from scratch. We train the networks by applying stochastic gradient descent (SGD) with mini-batch size of 256 and a fixed momentum of 0.9. The learning rate is initialised to 0.01, and is annealed by a factor of 10 when the error plateaus. The training is regularised by weight decay (set to 0.0002). We train all models up to 80×10^4 iterations. In testing, we take the standard "single crop (centre crop)" protocol in [25]. Furthermore, we use the fully-convolutional testing [5] to report the performance of single model. The image is resized isotropically so that its shorter side is in {224, 256, 320, 384, 448}, and the scores are averaged at multiple scales.

Comparisons of Results. Table 3 lists the results of the three methods with "single crop" testing strategy. Compared with standard BP, the baseline "Multi-loss + standard BP" shows better performance by introducing auxiliary supervision on intermediate layers, however the superiority is marginal, even negligible with regard to model B. In contrast, our method achieves significant improvement over standard BP, as well as consistently outperforms "Multi-loss + standard BP" (approximately 1.0% on model A and 0.8% on model B based on top-5 measure). It is notable that the improvement on model B is less than the one on model A. The shortcut connections in modified Inception modules make it possible to propagate information with shortcuts, somewhat alleviates the information reduction. This is also the reason of ineffectiveness of "Multi-loss

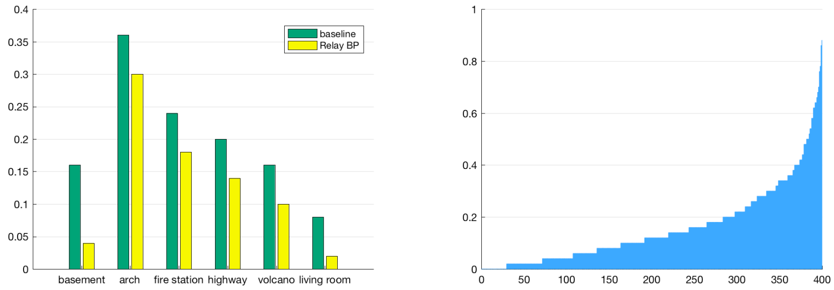

Fig. 3. (Left) Top-5 error rate(%) of example classes based on model B architecture on validation set. (Right) Pre-class top-5 error rate(%) based on model ensemble in ascending order.

Team name	top-5 err.
Ntu_rose	19.33
Trimps-Soushen	17.98
Qualcomm Research	17.59
SIAT_MMLAB	17.36
WM (model A)	17.35
WM (model B)	17.28
WM (model ensemble)	**16.87**

Table 5. The competition results of ILSVRC 2015 Scene Classification. The top-5 error rates (%) is on Places2 challenge test set and reported by the test server. Our submissions are denoted as “WM”.

+ standard BP” on model B. Nevertheless, our method is capable of improving the performance on model B. It confirms our insight that restraining the adverse effect of less relevant information is helpful for training deep neural networks.

For a comprehensive comparison, we also report the model performance with “single model” testing strategy in Table 4. Clear advantage can be observed in our method compared to the baselines. It is worthy of mentioning that the improvement of single model over centre crop is less, about 1.5% top-5 error diminished from 17.83% (single crop) to 16.33%, while empirical results on ImageNet 2012 classification dataset suggest the performance gain is approximately 3.0% [7,13]. To display further details about the result, we list top-5 error rates of example classes on validation set in Fig. 3(left), which are based on the architecture of model B. Distinguished superiority can be observed compared to standard BP, where the improvements on concept “basement”, ’living room” are 12% (relatively 300%), 6% (relatively 300%).

ILSVRC 2015 Scene Classification Challenge. *By virtue of Relay BP, our “WM” team won the 1st place in ILSVRC 2015 Scene Classification task.* Table 5 shows the results of this challenge. We combine five models of different

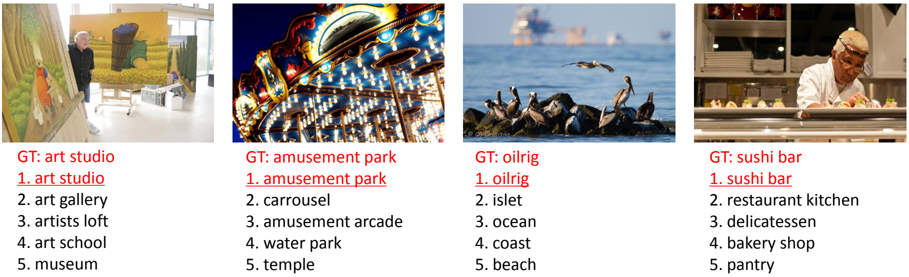

Fig. 4. Exemplars successfully classified by our method on Places2 challenge validation set. For each image, the ground-truth label and our top-5 predictions are listed.

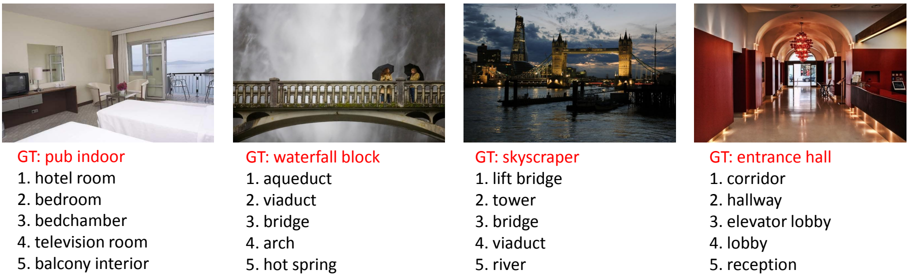

Fig. 5. Exemplars incorrectly classified by our method on Places2 challenge validation set. For each image, the ground-truth label and our top-5 predictions are listed.

architectures and input scales, and achieve 15.74% top-5 error on validation set. Fig. 3(right) display the per-class results of our method, which are reordered in ascend. We can observe that the model is capable of distinguishing most of the concepts, i.e., less than 20% error rates on more than 75% classes. Meanwhile, few concepts suffer from poor results, e.g., only 12% accuracy on " library/outdoor", which typically lack of distinctive characters apart from the others with similar appearance. For the final result, our top-5 error is 16.87% on the testing set, which is roughly 1.1% worse than the validation result. We conjecture that there might be a distribution gap between validation and testing data, similar degradation has also been observed by other teams [8]. Such phenomenon also implies the difficulty of task, as the scene concepts are typical associated with large intra-class divergence and sample amount would play a crucial role for the distribution of classes. Compared to the single model, the improvement of model ensemble over the one with architecture B is 0.4%. From single crop to single model, and further to model ensemble, the improvement is consistently lower than expected. We conjecture that training with large scale data enhances the capability of single view, leading to the difficulties of further improvement with model ensemble.

Fig. 4 shows some exemplars in Places2 challenge validation set, which are successfully classified by our method. The predicted labels are in descending order of confidence. Even though many high-level scene concepts exhibit large variance on intra-class appearance, our method could still recognise them easily. On the other hand, we also show some examples that incorrectly classified in Fig. 5. These predictions seem to be reasonable, although they fail in context of evaluation measure. A scene image might be typically described by multi-labels. Moreover, the composing of a scene is mostly complicated, such as a place is comprised of multiple objects and the same object might appear in different places. The loose connections between scene and object concepts increase the difficulties of scene recognition.

5.2 ImageNet 2012 Classification

We evaluate our method on the ImageNet 2012 classification dataset [19], which has become one of the benchmarks to assess the progress of image classification. This dataset includes images belonging to 1000 classes, with 1.2M images for training, 50K images for validation and 100K images for testing. The classification performance is measured by the top-1 and top-5 error rates. We use the provided data for training models, do not use any additional data.

Configurations. Recently, residual networks [27] introduce shortcut connections, and has achieved state-of-the-art performance on ImageNet 2012 classification dataset. Moreover, [26] utilizes the ”Inception-v3” architectures, and yielded comparable classification accuracy. We use the 50-layer residual network (ResNet-50) [27] and the Inception-v3 architectures [26] to evaluate Relay BP. For both architectures, we do not use scale jitter augmentation [5] during training. Standard SGD is applied to train the networks. Other configurations (including data augmentation, network architectures, and training/testing methodology) remain unchanged as [27,26]. More details about the configurations can be found in [27,26]. For Relay BP, we add one auxiliary branch with the loss weight set to 0.3. The gradient overlapping segments of primary and auxiliary loss range from “conv4_1” to “conv4_4” (ResNet-50), and “inception4a” to “inception4d” (Inception-v3), respectively. As the scheme of multi-loss has been included in Inception-v3, we omit the baseline “Multi-loss + standard BP” in Table 6.

Results Analysis and Discussion. Table 6 lists the classification errors achieved in single model. The results in the first row are the ones reported in [27] and [26], respectively. And the second row displays the results by our re-implementation. There is slight difference between the two rows, mainly because of the diversity of details in implementation, which has been described in the section of ”Configurations”.

The models trained with Relay BP achieve better classification performance compared to the ones trained with standard BP. The accuracy improvement is 0.44% on top-5 measure, and 0.91% on top-1 measure based on ResNet-50 network. Besides, there are 0.46% and 0.66% improvement on top-5 and top-1

Method	dataset	ResNet-50		Inception-v3	
		top-1 err.	top-5 err.	top-1 err.	top-5 err.
standard BP [27,26]	val	20.74	5.25	18.77	4.20
standard BP (re-implement)	val	21.17	5.37	19.18	4.43
Relay BP		20.26	4.93	18.52	3.97
Relay BP	test	-	4.95	-	**4.03**

Table 6. Single model error rates (%) on ImageNet 2012 classification dataset.

measure based on Inception-v3 architecture. The common characteristic of the two architectures is the utilisation of shortcut connections, although the implementations are different. As we have mentioned in above sections, shortcuts make the gradient of final outputs easily reach lower layers, thus are able to prevent the information reduction due to long-term propagation. This is also the evidence of only adding one auxiliary branch in Relay BP. Nevertheless, the network performance can be enhanced by aid of our method, which further demonstrates the promise of our insight that restraining the adverse effect of less relevant information is effective for improving network performance. Because of the high baselines, the improvement is so difficult, which highlights the effectiveness of our method. Moreover, we also report the results on test dataset (submitted to test server) to verify that the obtained results are not overfitting to the dataset. We only submitted the two results in the last half year, and the result 4.03% outperforms the best one of single model reported in ILSVRC 2015 ImageNet Classification task.

6 Conclusion

In this paper, we proposed the method *Relay Backpropagation*, which encourages the flows of informative gradient in backward propagation when training deep convolutional neural networks. Relevant information can be effectively preserved, and the adverse effect of less relevant information can be restrained. The experiments with four different network architectures on two challenging large scale datasets demonstrate the effectiveness of our method is not restricted to certain network architecture or specific dataset. As a future direction, we are interested in theoretical and mathematical support for the method.

Acknowledgments

Z. Lin is supported by China 973 Program (grant no. 2015CB352502), NSF China (grant nos. 61272341 and 61231002), and Microsoft Research Asia Collaborative Research Program.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS. (2012)
2. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014)
3. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: Closing the gap to human-level performance in face verification. In: CVPR. (2014)
4. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: CVPR. (2014)
5. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR. (2015)
6. Srivastava, R.K., Greff, K., Schmidhuber, J.: Highway networks. In: ICML Deep Learning Workshop. (2015)
7. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: ICCV. (2015)
8. Zhou, B., Khosla, A., Lapedriza, A., Torralba, A., Oliva, A.: Places2: A large-scale database for scene understanding. http://places2.csail.mit.edu/ (2015)
9. LeCun, Y., Bottou, L., Orr, G., Muller, K.: Efficient backprop. Neural Networks: Tricks of the trade (1998)
10. Nair, V., Hinton, G.: Rectified linear units improve restricted boltzmann machines. In: ICML. (2010)
11. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectified nonlinearities improve neural network acstic models. In: ICML. (2013)
12. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: ICAIS. (2010)
13. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167 (2015)
14. Kamimura, R.: Information Theoretic Neural Computation. World Scientific (2002)
15. Cover, T.M., Thomas, J.A.: Elements of Information Theory. Wiley-Interscience (2006)
16. Tishby, N., Zaslavsky, N.: Deep learning and the information bottleneck principle. In: IEEE Information Theory Workshop. (2015)
17. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. In: NIPS. (2014)
18. Xiao, J., Ehinger, K., Hays, J., Torralba, A., Oliva, A.: Sun database: Exploring a large collection of scene categories. IJCV (2014)
19. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: Imagenet large scale visual recognition challenge. IJCV (2015)
20. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional neural networks. In: ECCV. (2014)
21. Goodfellow, I.J., Warde-Farley, D., Mirza, M., Courville, A., Bengio, Y.: Maxout networks. arXiv:1302.4389 (2013)
22. Sutskever, I., Martens, J., Dahl, G.E., Hinton, G.E.: On the importance of initialization and momentum in deep learning. In: ICML. (2013)
23. Tieleman, T., Hinton, G.: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning (2012)

24. Lee, C.Y., Xie, S., Gallagher, P., Zhang, Z., Tu, Z.: Deeply-supervised nets. In: AISTATS. (2015)
25. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: CVPR. (2015)
26. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. arXiv:1512.00567 (2015)
27. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. (2016)
28. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: ECCV. (2014)
29. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv:1408.5093 (2014)