Self-Attentive Layer Aggregation

Liangchen Luo
Department of Earch and Space Science
Peking University
luolc@pku.edu.cn

Abstract

This paper is a report of the final assignment from the deep learning course given by Prof. Zhihua Zhang.

1 Introduction

In deep networks, using only the last layer makes the model "forget" distant layers (Wang et al., 2018). Therefore, there is no easy access to features extracted from lower-level layers if the model is very deep. To address this problem, layer aggregation (Yu et al., 2018) has been proposed to fuse information across layers in various popular tasks in the fields of computer vision and natural language processing. However, most of the previous methods combine layers in a static fashion in that their aggregation strategy is independent of specific hidden states.

Inspired by the success of attention mechanism in natural language understanding and generation, we propose a self-attentive layer aggregation method, which is able to fuse the layers with different attention weights based on the specific hidden state value. The proposed method can be broadly applied to different kinds of deep neural architectures.

2 Self-Attentive Layer Aggregation

An attention function can be described as mapping a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key.

To introduce attention mechanism in layer aggregation, we obtain the query, keys and values from all layers of a given model. Suppose a model contains L layers $\{x_i\}_{i=1}^{L}$, where x_i is the output of the i-th layer. We compute the query, keys, and values as:

$$Q = f_Q(x_L), \tag{1}$$

$$K_i = f_K(x_i, i), \tag{2}$$

$$V_i = f_V(x_i, i), \tag{3}$$

where the query Q and keys $\{K_i\}_{i=1}^{L}$ have dimension d_k and the values $\{V_i\}_{i=1}^{L}$ have dimension d_v. We compute the dot products of the query with all keys, divide each by $\sqrt{d_k}$ and apply a softmax function to obtain the weights on the values. In practice, we compute the attention function with the keys and values packed together into matrices K and V:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V. \tag{4}$$

In this way, the output of the attention function fuses the information from all layers dynamically with different weights for the layers based on their values.

Preprint. Under review.

3 Experiments

We apply the self-attentive layer aggregation architecture to ResNet (He et al., 2016), and call the new network as the SA-ResNet. We conduct experiments with ResNet and SA-ResNet on the CIFAR-10 image classification dataset (Krizhevsky & Hinton, 2009), which consists of 50K training images and 10K testing images in 10 classes. The experimental results show that SA-ResNet achieve considerable improvements over the original ResNet.

Implementation details. We mainly follow the settings in He et al. (2016) to train the models on the training set and evaluated on the test set. For both ResNet and SA-ResNet, the network inputs are 32×32 images, with the per-pixel mean subtracted. The first layer is 3×3 convolutions. Then we use a stack of $6n$ layers with 3×3 convolutions on the feature maps of sizes $\{32, 16, 8\}$ respectively, with $2n$ layers for each feature map size. The numbers of filters are $\{16, 32, 64\}$ respectively. The subsampling is performed by convolutions with a stride of 2. The network ends with a global average pooling, a 10-way fully-connected layer, and softmax. There are totally $6n + 2$ stacked weighted layers. For SA-ResNet, we apply average pooling to each layer output x_i to obtain the value vector $\hat{x}_i$ with the same size of the last layer. Then we apply a linear transformation on $\hat{x}_i$ to obtain the key of the i-th layer with $d_k = 32$, and the query Q shares the same value with the key K_L of the last layer.

We use a weight decay of 0.0001 and momentum of 0.9, and adopt the weight initialization and BN without dropout in He et al. (2016). The models are trained with a mini-batch size of 128 on one GPU. We start with a learning rate of 0.1, divide it by 10 at 32K and 48K iterations, and terminate training at 64K iterations. We follow the simple data augmentation in for training: 4 pixels are padded on each side, and a 32×32 crop is randomly sampled from the padded image or its horizontal flip. For testing, we only evaluate the single view of the original 32×32 image.

We test SA-ResNet with 20, 32, 44 and 56 layers, and compare them with the results reported in He et al. (2016). Table 1 shows the comparisons of SA-ResNet and ResNet. SA-ResNets outperforms all the counterparts with same number of layers.

Table 1: Comparisons of SA-ResNet with ResNet on CIFAR-10.

#layers	**ResNet**	**SA-ResNet**
20	8.75	7.73
32	7.51	6.68
44	7.17	6.45
56	6.97	**6.32**
110	6.43	-

It is also worth noticing that the testing error of the 56-layer SA-ResNet is comparable to that of 110-layer ResNet on CIFAR-10. These results indicate that the self-attentive architecture can greatly compress ResNet without losing the performance.

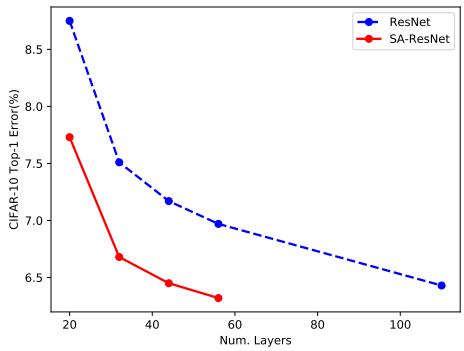

Figure 1: SA-ResNet is an efficient structure that can achieve same level of accuracy with only half of the layers of ResNet on CIFAR-10.

4 Conclusion

We propose a self-attentive layer aggregation method, which is able to fuse the layers with different attention weights based on the specific hidden state value. Experimental results show the proposed method can greatly improve the deep neural models.

References

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, 2009.

Qiang Wang, Fuxue Li, Tong Xiao, Yanyang Li, Yinqiao Li, and Jingbo Zhu. Multi-layer representation fusion for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics (CICLing)*, pp. 3015–3026, Santa Fe, New Mexico, USA, August 2018. Association for Computational Linguistics.

Fisher Yu, Dequan Wang, Evan Shelhamer, and Trevor Darrell. Deep layer aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2403–2412, 2018.