Residual Attention Networks

*ECBM4040 2021Fall DFTH report

Yixuan Zou yz4004
Columbia University

***Abstract*—Attention mechanisms is one of the most valuable breakthroughs in deep learning in 2010s and is widely applied in various field like computer vision (CV) and natural language processing (NLP). It is often used as an encoder-decoder form between layers of neural network like machine translation to mimic the process of attention concentrating. In this report, we will explore a specific model, "Residual Attention Network" proposed by paper [2], which is a convolutional neural network combining attention mechanisms with residual learning in image classification. The basic brick module used is a pre-active residual learning unit introduced by [3]. It is not simply adding attention layer between residual network, but also implant residual structure in the attention module. So, attention modules cooperating with residual units can also be stacked to a very deep model like ResNet. I implement the attention module proposed by [2] and test its performance on cifar10. By trying different hyperparameter and layer structure, the test accuracy is roughly about 85% by no more than 20 epochs.**

I. Introduction

The initial idea of attention mechanism is from human cognitive process of selectively concentrating on a particular part of field of view. Since neural network is designed to mimic human brain action, it is natural to add attention layer of module as a supplement. One early application of attention mechanism in deep learning is used in LSTM based RNN for sentence processing. When reading a sentence, people tend to focus on key words, which often appear in middle of sentence. So, in sequence to sequence, adding an encode/decode as attention module can imitate this action. In computer vision, people's attention on object can be modeled by applying a weight to digital image. For example, people's eyes tend to focus on something colorful and bright. So, we could consider neural network put a higher weight on the pixels of those objects.

Attention is generally divided into two types: one is top-down conscious attention, called focus attention. Focused attention refers to the attention that has a predetermined purpose, depends on the task, and actively and consciously focuses on an object; the other is bottom-up unconscious attention, called saliency-based (Saliency-based) attention. Saliency-based attention is attention driven by external stimuli, does not require active intervention, and has nothing to do with the task. If the stimulus information of an object is different from the surrounding information, an unconscious "winner-take-all" (like maxpooling take the largest value in the pool) or gating mechanism can turn attention to the object. Regardless of whether the attention is intentional or unintentional, most human brain activities rely on attention, such as memorizing information, reading or thinking: An example related to attention is the cocktail party effect. When a person is chatting with a friend at a noisy cocktail party, he can still hear the conversation of the friend despite the disturbance of the surrounding noise, while ignoring the voice of other people (focused attention). At the same time, if there are important words in the unnoticed background sound (such as his name), he will immediately notice (salience attention).

The attention in deep learning is derived from the attention mechanism of the human brain. When the human brain receives external information, such as visual information and auditory information, it often does not process and understand all the information, but only pays attention to it. Focusing on some significant or interesting information will help filter out unimportant information and improve the efficiency of information processing. The earliest starting point for using Attention in image processing is to use a mechanism similar to human brain attention, using only a small receptive field to process the Attention part of the image, reducing the dimension of calculation. Later, some people discovered that the convolutional neural network has the function of Attention. For example, in the classification task, the pixels activated by the high-level feature map are also concentrated in the area related to the classification task, that is, the salience map. It is often used in image detection and segmentation. So how to use Attention to improve the performance of the model on classification tasks? The article [2] provides a new way of thinking.

1. Propose a stackable network structure. Similar to the Residual Block in ResNet, the network structure proposed in this article is also stacked through a Residual Attention Module structure, which enables the network model to easily reach a deep level.

2. Propose a residual learning method based on Attention. Same as ResNet, the model proposed in this article also uses a residual method, so that very deep models can be easily optimized and learned, and have very good performance.

3. Bottom-up Top-down forward Attention mechanism. Other networks that use Attention often need to add a branch to the original network to extract Attention and perform separate training. The model proposed in this article can extract the Attention of the model in a forward process, so that the model can be trained It's even simpler.

Identify applicable funding agency here. If none, delete this.

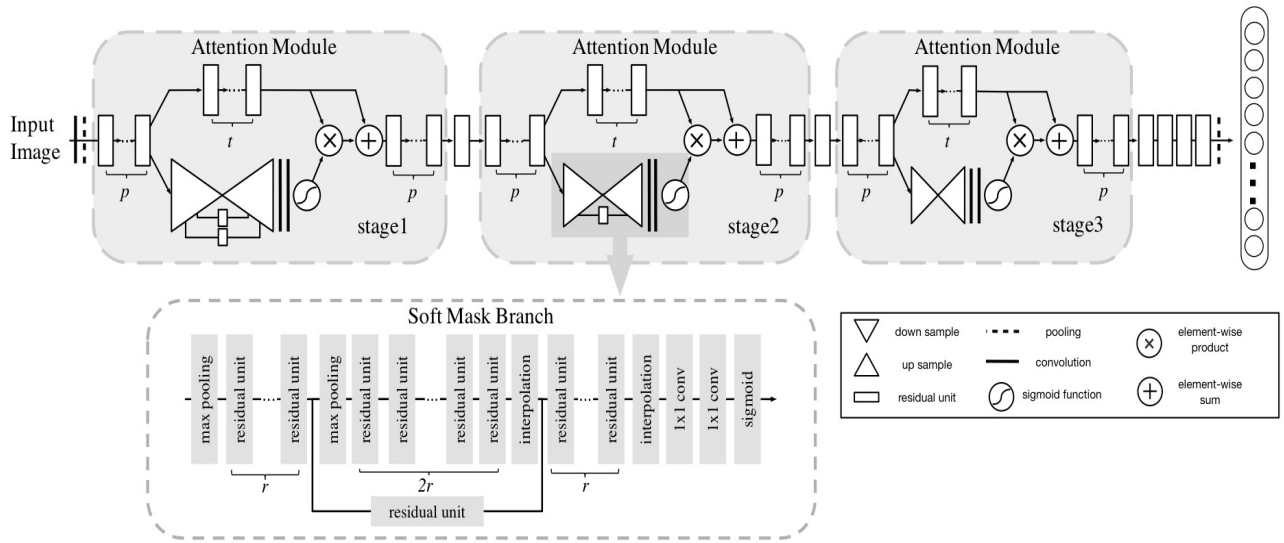

Fig. 1. Residual Attention Module Structure.

II. Residual Attention Netwroks

Overall structure is in figure1. The residual attention network is formed by starting at a preprocessing maxpooling layers, stacking multiple layers of attention modules connected by several residual units and ending by flatten and dense layer output for classification. Each attention module is divided into two parts: mask branch and trunk branch. The main branch performs feature processing. It can use any network model. The author uses the pre-activated residual unit, ResNet as the basic brick module of the residual attention network. If input x, then trunk brunch generates $T_{i,c}(x)$, where i is index of pixels and c is index of channels. The mask branch uses a combination of bottom-up and top-down attention to learn a mask $M_{i,c}(x)$ that has the same size as the main output. Then, attention module outputs

$$H_{i,c}(x) = M_{i,c}(x) * T_{i,c}(x)$$

which is equivalent to applying weight $M_{i,c}(x)$ on trunk output $T_{i,c}(x)$. In the attention module, the attention mask acts as a feature selector during forward and as a gradient update filter during back propagation, since when calculating derivative of ϕ, mask brunch staies same.

$$\frac{\partial H_{i,c}(x)}{\partial \phi} = \frac{\partial M_{i,c}(x;\theta) * T_{i,c}(x;\phi)}{\partial \phi} \tag{1}$$

$$= M_{i,c}(x;\theta) * \frac{\partial T_{i,c}(x;\phi)}{\partial \phi} \tag{2}$$

where ϕ is parameter of trunk feature selector and θ is parameter of mask brunch. This makes the attention module very robust to noise and can effectively reduce the influence of noise on gradient updates. The background occlusion, complex scenes, and appearance changes in the training images require a variety of attention. If the method of stacking attention modules is not used, more channels are needed to cover the combined attention of different factors. And an attention module can only modify the feature once, so the fault tolerance rate is very small.

A. Attention Module

Although the attention module has a greater effect on image classification, simply superimposing the attention module will not cause the performance of the model to increase for numerical stability. The authors point out that the mask branch needs to be followed by Sigmoid as the activation function, in order to output the feature map with normalized weights, but normalizing the output to 0 to 1 and then multiplying with the main branch will make the output response of the feature map weaker. Multi-layer superposition of this structure will make the value of each point of the final output feature map vanish to zero at end of training. Furthermore, the output from the mask branch may destroy the advantages of the main branch. For example, replacing the shortcut mechanism in the residual connection with the mask branch will cause the gradient of the deep network to not be returned well. In order to solve the above problems, the authors get inspiration from residual network. Instead of directly mulipling weight to trunk branch, they want mask branch learning the residual of weights and element-wised add the obtained attention feature map and the main feature map, so the output is expressed as:

$$H_{i,c}(x) = (1 + M_{i,c}(x)) * T_{i,c}(x)$$

$M_{i,c}(x)$'s value is between 0 and 1. By adding an identity map from trunk output, it has no issue on numerical stabiliy. Comparing to classic residual learning

$$H_{i,c}(x) = x + T_{i,c}(x)$$

, $T_{i,c}(x)$ is responsible for learning residual variance from input to output, but here $T_{i,c}(x)$ is a output from main network, $M_{i,c}(x)$is a filter imposed on main output. By this approach, keeping stack attentions will increase the ability of network.

B. Soft Mask Branch

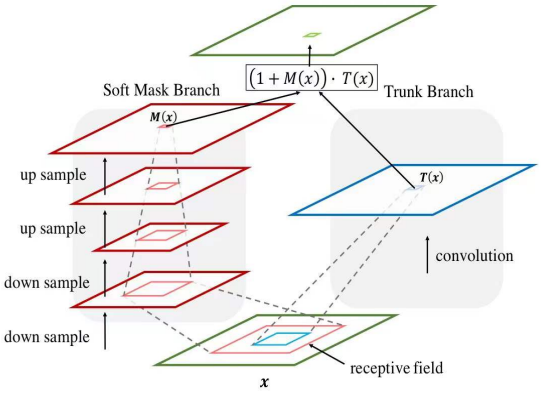

Fig. 2. Residual Attention Module Structure.

From the paper [2], the mask branch includes fast feed-forward sweep and top-down feedback steps. The former quickly collects the global information of the entire image, and the latter combines the global information with the original feature map. In a convolutional neural network, these two steps are expanded into a bottom-up top-down full convolution structure. The author mainly uses a simple FCN-like network to implement mask branching. Starting from the input, perform max pooling several times to achieve a rapid increase in the receptive field after a few Residual Units. After reaching the lowest resolution, the feature is magnified back through a symmetrical network structure, that is, linear interpolation is used after Residual Units. The number of linear interpolations used is consistent with the maximum pooling number to ensure that the input and output sizes are the same. Then, connect 2 consecutive 1×1 convolutional layers, and finally connect a Sigmoid layer to normalize the output to [0,1]. Added skip connections between bottom-up and top-down can capture different proportions of information.

C. Activation of Attention Module

There are three types of activation function applied on output of attention corresponding three kinds of attention. Mixed Attention:

$$f(x_{i,c}) = sigmoid(x_{i,c})$$

Channel Attention:

$$f(x_{i,c}) = \frac{x_{i,c}}{\|x_{i,c}\|}$$

Spatial Attention:

$$f(x_{i,c}) = sigmoid((x_{i,c} - \mu_c)/\sigma_c)$$

where μ_c and σ_c is mean and standard variance of a pixel on channels.

Figure Labels: In third column, 3 pairs in each cell represent 3 convolutions layers in residual unit with stride and channels. All padding condition is set as "same".

III. Experiments and Results

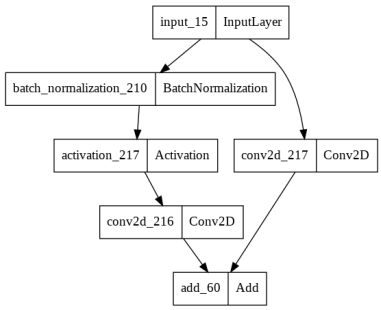

Fig. 3. Residual unit1.

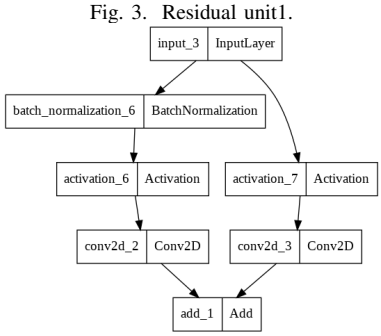

Fig. 4. Residual unit2.

During the coding, two types of residual unit are tried. Both have four convolution layers. In type2, it has four corresponding batch normalization layers while type1 has only three. Difference is whether the input and residual share a same batch normalization layer. It turns out residual unit 2 has much less computation cost and could converge much faster, since it has less parameters and we stack a lot of such units in the model. So, in later experiments, we adopt residual unit1. The fourth convolutional layer is just to make sure resiual and input have same number of channels and size.

Table 1

Layer	Output Size	Attention 92
Residual Unit	(32,32,32)	(1,32),(3,32),(1,128)
Attention Module	(32,32,128)	Attention step1*1
Residual Unit	(16,16,256)	(1,64),(3,64),(1,256)
Attention Module	(16,16,256)	Attention step2*2
Residual Unit	(8,8,512)	(1,128),(3,128),(1,512)
Attention Module	(8,8,512)	Attention step3*3

For hyperparameter t=2, trunk branch is just two connected residual unit, same in all three attention module. But since we use a middle residual unit between attention modules which reduces image size twice and double the channels. Thus, with going deeper, in third attention module we have input size 8*8. Small input shape represents a small region of image, like a specific area of objects. So it is not necessary having

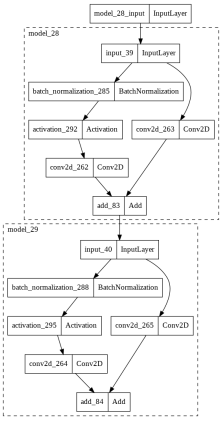

Fig. 5. Trunk Branch with t=2.

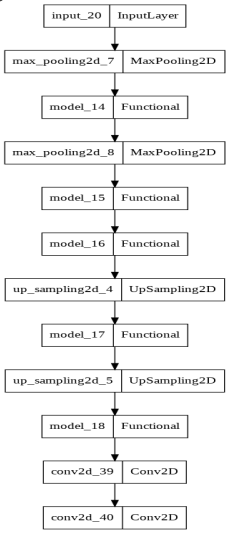

Fig. 6. Mask Branch in step1 with r=1.

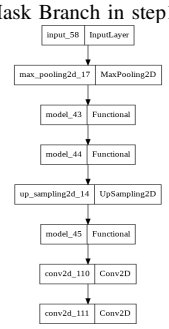

Fig. 7. Mask Branch in step2 with r=1.

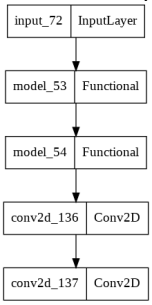

Fig. 8. Mask Branch in step3 with r=1.

too many maxpooling and upsampling layers as in shallow attention modules which receive a much larger input. Thus, in second and third module, we reduce the maxpooling and upsampling layers. In keras, there are two ways to magnifiy image, upsampling and conv2dTranspose and we adopt the fisrt one. Transpose convolution induces higher number of parameters and unnecessary linearity. By trying different combination of hyperparameters, the results have no much significant differences. But computation cost differs a lot. I tried to truncate training in 20 epochs to make sure results are comparable. The overall result is not as good as in paper whose test error is 5%, since my model encountered overfitting (train acc equals to 1.0 but test acc only reach to around 0.85). More work need to done on the optimizing process, since I found reducing the decay rate of adam after several epochs can improve the test accuracy. By far, I only conduct experiments on one set of hyperparameters. Trying different number of convolution channels in residual units may tune the model to get a better result. Also, other techniques should be adopted to improve generalization of model.

Table 2, Cifar10 Test Accuracy

Hyperparameters (p,t,r)	Attention 56	Attention 92
(1,2,1)	0.793	0.851
(2,2,1)	0.773	0.848
(1,3,1)	0.787	0.807
(1,2,2)	0.811	0.843

REFERENCES

[1] Itti L, Koch C. Computational modelling of visual attention. Nat Rev Neurosci. 2001 Mar;2(3):194-203. doi: 10.1038/35058500. PMID: 11256080.

[2] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang and Xiaoou Tang. Residual Attention Network for Image Classification, 2017; arXiv:1704.06904.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep Residual Learning for Image Recognition, 2015; arXiv:1512.03385.