Residual Attention Network for Image Classification

E4040.2020Fall.LZLZ.report
Lichirui Zhang lz2627, Weiwei Zhan wz2483
Columbia University

Abstract

In this project, we reproduced the paper, titled "Residual Attention Network for Image Classification" and re-implemented the Residual Attention Network which is a convolutional neural network applying the attention mechanism for image classification. We constructed networks with different trunk layer depth and evaluated their performance on the benchmark datasets CIFAR-10 and CIFAR-100. We encountered multiple challenges such as limited computational resources and insufficient training details in the original paper, and to best reproduce the results, we extensively explored different model architectures, hyperparameter configurations, and training strategies to achieve higher accuracy.

Similar to the original paper, our Residual Attention Networks show good robustness to different levels of noises and the residual attention learning is shown to effectively avoid the performance dropping for very deep networks. However, our test accuracies are consistently 10% lower than that reported in the original paper, although we extensively tweaked hyperparameters and network architecture. Interestingly, we found that more Attention Modules in each stage does not necessarily result in a better network performance, implying that stacking more modules in later stages rather than distributing them equally across all stages might help improve the test accuracy to the reported level.

1. Introduction

The attention mechanism is one of the most influential ideas in the deep learning field. Attention, roughly speaking, is a learnable map that describes the interdependence of a given input feature to the output or to itself. A more important element will be given more attention, i.e., a larger weight. Many models adopting the attention mechanism are the state-of-the-art in a wide range of fields, including but not limited to natural language processing [2, 3], image captioning [4], and even protein structure prediction [5]. The goal of our project is to understand how the attention mechanism can be applied to a convolution neural network and facilitate solving the image classification problem. We comprehensively studied the paper that, for the first time, introduced the attention mechanism into the image classification problem and achieved the state-of-the-art accuracy when it was published [6]. Our objective is to recreate the network model and reproduce the experimental results in the paper. However, by no means did we intend to achieve or surpass their accuracy, as our time and computational resources were limited, and the technical details are not clearly described in the original paper.

In this project, we recreated the Residual Attention Networks with different trunk layer depth and evaluated their performance on the benchmark datasets CIFAR-10 and CIFAR-100. Similar to the original paper, we also 1) tested the performance of multiple attention types on image classification, 2) evaluated the effectiveness of attention residual learning when training very deep networks, and 3) assessed the noise resistant property of Residual Attention Network under different noise levels. During our study, we encountered several technical challenges. First of all, as stated before, the original paper did not provide a thorough explanation for network architecture and training strategies. To best reproduce the results, we extensively explored different model architectures, hyperparameter configurations, and training techniques such as data augmentation and parameter initialization methods. Second, our computational resources are limited, causing difficulties in training very deep networks on the large ImageNet dataset. As a workaround, we trained our network on a resized ImageNet 32x32.

2. Summary of the Original Paper

2.1 Methodology of the Original Paper

The original paper proposed the Residual Attention Network as shown in Fig. 1. The network mainly adopted four strategies to achieve robust network performance. Firstly, the network is constructed by stacking multiple Attention Modules that can capture different attention-aware features. This can refine features from complex images as the layers go deeper. Secondly, the bottom-up top-down feedforward structure is used in each Attention Module, and therefore we can achieve an end-to-end trainable network for feature learning. Thirdly, attention residual learning is used to train this deep network in order to avoid the performance drop caused by naive stacking Attention Modules. Finally, the architecture is able to adopt many basic units including Residual Unit [7], ResNeXt [8], and Inception [9] to build the Attention Module, which gives the architecture a great flexibility.

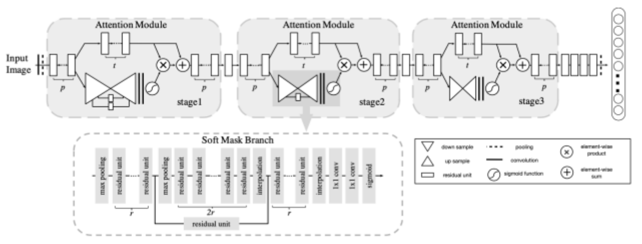


Figure 1. The architecture of the Residual Attention Network.

2.2 Key Results of the Original Paper

The original paper used CIFAR-10 and CIFAR-100 datasets to evaluate the performance of Residual Attention Network with different trunk layer depth. Table 1 shows that both attention-56 and attention-92 perform well on CIFAR-10 with test error of 5.52% and 4.99% respectively. The network with larger trunk layer depth performs better on both datasets, but it requires larger parameters as well.

Network	params (10^6)	CIFAR-10 (err.%)	CIFAR-100 (err.%)
Attention-56	/	5.52	/
Attention-92	1.9	4.99	21.71
Attention-236	5.1	4.14	21.16

Table 1. Classification error (%) on CIFAR-10 and CIFAR-100.

Attention Residual Learning. The original paper also evaluated the performance of Attention Residual Learning (ARL) by comparing it against Naive Attention Learning (NAL) which naively stacks the Attention Modules. Table 2 shows that ARL consistently outperforms NAL results, and the performance increases with the depth of trunk layers.

Network	ARL (Top-1 err.%)	NAL (Top-1 err.%)
Attention-56	5.52	5.89
Attention-92	4.99	5.35
Attention-128	4.44	5.57
Attention-164	4.31	7.18

Table 2. Classification error (%) on CIFAR-10 for ARL and NAL.

Spatial Attention and Channel Attention. The original paper evaluated three kinds of attention: Mixed Attention, Channel Attention, and Spatial Attention. Table 3 shows that the mixed attention achieves the best performance, implying that adding no constraint and letting attention change adaptively with features can generate better results.

Activation Function	Attention Type	Top-1 err. %
$f_1(x)$	Mixed Attention	**5.52**
$f_2(x)$	Channel Attention	6.24
$f_3(x)$	Spatial Attention	6.33

Table 3. Classification error (%) of Attention-56 with different attention types on CIFAR-10 Dataset.

Noisy Label Robustness. Additionally, the original paper tested the network robustness by injecting different levels of noises to the training labels. Table 4 shows that Attention-92 network consistently outperforms the ResNet-164 across different levels of noises.

Noise Level	ResNet-164 err. (%)	Attention-92 err. (%)
10%	5.93	5.15
30%	6.61	5.79
50%	8.35	7.27
70%	17.21	15.75

Table 4. Test error (%) on CIFAR-10 with different levels of label noises.

3. Methodology of the Students' Project

3.1. Objectives and Technical Challenges

The overall objective of this project is to reproduce the experiments and results of the original paper to the greatest possible extent. To achieve this, we first built the Attention Module using the pre-activation residual unit and assembled the Residual Attention Network as shown in Fig. 1. Then we used different networks and attention mechanisms to conduct various experiments in the original paper. As the paper did not provide a clear description for their training details, we need to extensively explore hyperparameters and training strategies for different networks and experiments.

3.2. Problem Formulation and Design Description

In addition to constructing a single Attention Module, there are still other problems to solve in order to implement the whole Residual Attention Network with robust performance, e.g., how should we stack Attention Modules to achieve a very deep network? How should we design the structure of the mask branch to effectively capture features from complex images? Should we add additional constraints to the attention provided by the mask branch? To explore these questions, we design the following experiments with the reference to original paper.

Attention Residual Learning. We used the strategy of attention residual learning (ARL) to stack multiple Attention Modules. Inside each Attention Module, the mask branch can learn a same size mask $M(x)$that weighs the trunk branch output $T(x)$. Adapting the idea of residual learning, the output of Attention Module H is :

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

And for NAL, the output of Attention Module $H$is the direct dot product between soft mask with the trunk branch output:

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

where i and c are spatial positions and channel indices, respectively.

The first benefit of ARL is that the mask $M(x)$ can act as a feature filter that effectively strength the useful features extracted by $T(x)$. Also, by using ARL, the stacked Attention Modules can clarify the selected features as we move to deeper layers, and we expect that the deeper Residual Attention Network should achieve better performance.

Structure of Mask Branch. The mask branch uses a bottom-up top-down feedforward structure, which includes both feed-forward sweep and top-down feedback steps. As Fig. 2 shows, the soft mask branch firstly down samples through max pooling for multiple times to greatly increase the receptive field, then a linear interpolation up samples for several times and expands the global information. The number of upsampling equals that of downsampling to keep the image size unchanged. The output then goes through two convolution layers to be normalized by a sigmoid layer.

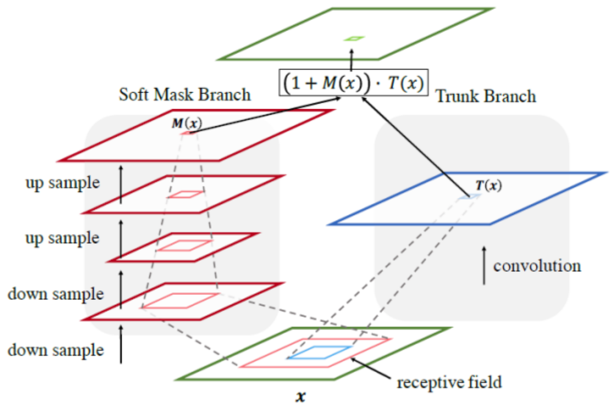


Figure 2. The structure of the mask branch and trunk branch.

Additional Constraints to Attention. As mentioned before, the feature map inside a soft mask branch is normalized by a sigmoid activation function before we get the final soft mask output, which is corresponding to the mixed attention type f_1. In addition, we tested two other attention types - channel attention f_2 and spatial attention f_3 - by adding additional constraints to attention before the sigmoid activation function. The spatial attention normalizes the feature map from each channel before the sigmoid layer, while the channel attention adds L_2 normalization within all channels for every spatial location. The following equations show the calculation process of these three attention types:

$$f_1(x_{i,c}) = 1/(1 + exp(-x_{i,c}))$$
$$f_2(x_{i,c}) = x_{i,c}/||x_{i,c}||$$
$$f_3(x_{i,c}) = 1/(1 + exp(-(x_{i,c} - mean_c)/std_c))$$

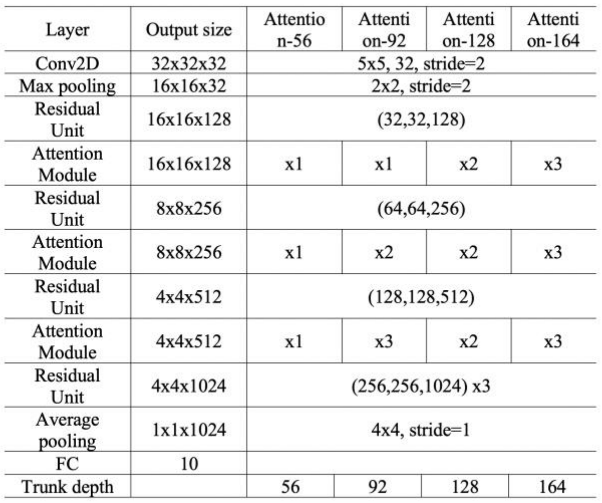
where i and c indicate the spatial positions and channel indices respectively, and x_i is the feature vector across three channels at the i-th spatial location. $mean_c$ and std_c represent the average value and standard deviation of the c-th channel image.

4. Implementation

In Sec 4.1, we describe our high-level implementation of the Residual Attention Network architecture, then illustrate our training algorithm data augmentation methods in detail. In Sec 4.2, we show the detailed architecture of our model, and our low-level implementation of the basic building blocks of the network. Our network and training scripts are available in this GitHub repository [1].

4.1. Residual Attention Network

Network Architecture. We constructed the Residual Attention Networks very similar to the architecture shown in Fig. 1, albeit small tweaks. One can stack m Attention Modules in each stage, resulting in an Attention-(36m+20) network with 36m+20 trunk depth. A special case is Attention-92. We experimented with both the {2, 2, 2} Attention Module configuration, and the {1, 2, 3} configuration. The {1, 2, 3} configuration performed better. Therefore, we decided to use the {1, 2, 3} configuration in our subsequent experiments.



Layer	Output size	Attentio n-56	Attenti on-92	Attenti on-128	Attenti on-164
Conv2D	32x32x32	5x5, 32, stride=2			
Max pooling	16x16x32	2x2, stride=2			
Residual Unit	16x16x128	(32,32,128)			
Attention Module	16x16x128	x1	x1	x2	x3
Residual Unit	8x8x256	(64,64,256)			
Attention Module	8x8x256	x1	x2	x2	x3
Residual Unit	4x4x512	(128,128,512)			
Attention Module	4x4x512	x1	x3	x2	x3
Residual Unit	4x4x1024	(256,256,1024) x3			
Average pooling	1x1x1024	4x4, stride=1			
FC	10				
Trunk depth		56	92	128	164

Table 5. Architecture details for Attention Residual Networks with different trunk depth.

Training Algorithm. Because the original paper did not provide a comprehensive training algorithm and hyperparameter description, to best reproduce the results of the original paper, we explored different hyperparameter configurations and training strategies for different datasets and Residual Attention Network architectures. We trained our models using Adam optimizer with a mini-batch size of 32. We set the initial learning rate to be 0.0001. Learning rate was divided by 5 when validation accuracy plateaued. The minimum learning rate was set to be 10^{-6}. The training process was terminated after 150 epochs or when no improvement was gained in validation accuracy in a consecutive 15 epochs. If the training process is terminated by early-stopping, the best weights would be restored. For some cases, e.g., very deep networks such as Attention-236 trained on the CIFAR-100 dataset, to prevent the model from overfitting, we applied a dropout layer before the last dense layer with a dropout rate 0.5, and a L_2 regularization with a factor 0.001. We trained our networks on a single NVIDIA Tesla T4 GPU.

Datasets. The CIFAR-10 and CIFAR-100 datasets consist of 60,000 32x32 images of 10 and 100 classes respectively, with 50,000 training images and 10,000 test images. We took 10,000 images from the training dataset as a validation dataset. All images were rescaled per-pixel by 255.0. For training images, we randomly shifted width and height within 10%, then cropped a 32x32 part from the shape shifted images. The training images were randomly zoomed in or out by 10%, sheared by 10%, randomly rotated by an angle less than 20°, and randomly flipped horizontally. For very deep networks such as Attention-236, we increased the augmentation intensity by 50% to further suppress overfitting.

The original ImageNet dataset consists of 14, 197, 122 64x64 images of 1000 classes, including 1.2 million training images, 50, 000 validation images and 100, 000 test images. We used a resized ImageNet 32x32 as a workaround as we failed to load the data to memory. Different from the data preprocessing and augmentation methods we used for the CIFAR datasets, we applied the same methods described in the original paper. Specifically, we first padded 4 pixels on each size of the image, then randomly sampled a 32x32 crop from the original image or its horizontal flip. All images were rescaled per-pixel by 255.0.

4.2. Network and Units Design

Network Design. As shown in Fig. 1, we followed the network design illustrated in the original work, albeit small tweaks. In our implementation, the numbers of skip connections are 2, 1, 0 for each stage, which is in agreement with the setting in the original paper. The hyper-parameters *p*, *t*, and *r* denote the number of pre-processing Residual Units before splitting into trunk branch and mask branch, the number of Residual Units in trunk branch and the number of Residual Unit between adjacent pooling layers in the mask branch. We set *p* = 1, *t* = 2, *r* = 1 in our work, same as the setting for ImageNet in the original work.

Soft Mask Branch Design. Our implementation of the soft mask branch is illustrated by the pseudocode below. Note that we set *Skip* as a hyperparameter in our work, which denotes the number of skip connections in each attention unit. The *ResidualUnit* denotes the pre-activation residual unit, whose structure will be described later. For the NAL mechanism, we simply change the line

$$output = (1 + OutMask) * OutTrunk$$

to

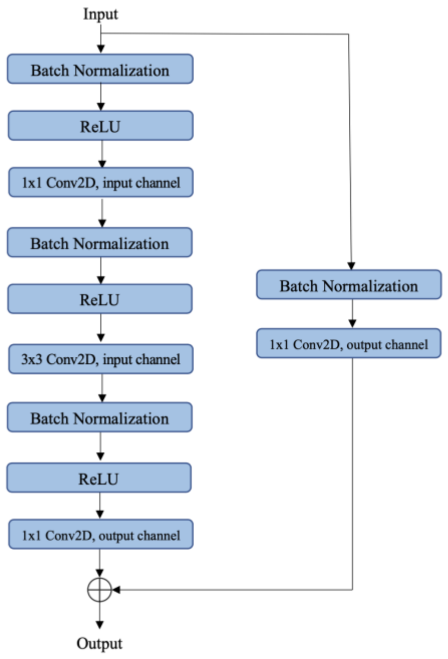
$$output = OutMask * OutTrunk$$

All the parameters that are not explicitly stated are the default settings in the TensorFlow version 2.2.

Algorithm 1: Mask Branch Implementation

```
Require: Input data $\mathbf{x}$, number of skip connections $n$
Set: $p = 1, r = 2, t = 1$
/* Pass the input through the pre-processing Residual Units
   before branch splitting                                  */
for $i \leftarrow 1$ to $p$ do
  | $x$ = ResidualUnit($x$)
end
/* Pass the data through the trunk branch                   */
Set $trunk\_output = x$
for $i \leftarrow 1$ to $t$ do
  | $TrunkOutput$ = ResidualUnit($TrunkOutput$)
end
/* Pass the data through the soft mask branch               */
// The first down sampling
Set OutMask = MaxPooling2D()(x)
for $i \leftarrow 1$ to $r$ do
  | $OutMask$ = ResidualUnit($OutMask$)
end
// Skip connections
Set $sc$ = []
for $i \leftarrow 1$ to $skip - 1$ do
  | $out\_sc$ = ResidualUnit($OutMask$)
  | $sc$.append($out\_sc$)
  | $OutMask$ = MaxPooling2D()($OutMask$)
  | for $j \leftarrow 1$ to $r$ do
  |   | $OutMask$ = ResidualUnit($OutMask$)
  | end
end
// reverse the sequence of $sc$
$sc$ = reversed($sc$)
for $i \leftarrow 1$ to $skip - 1$ do
  | // The first upsampling
  | for $i \leftarrow 1$ to $r$ do
  |   | $OutMask$ = ResidualUnit($OutMask$)
  | end
  | $OutMask$ = UpSampling2D()($OutMask$) // Skip connections
  | $OutMask$ = $OutMask$ + sc[i]
end
// The second upsamplping
for $i \leftarrow 1$ to $r$ do
  | $OutMask$ = ResidualUnit($OutMask$)
end
$OutMask$ = UpSampling2D()($OutMask$) // Output of soft mask
   branch
$OutMask$ = Conv2D(input_channel, (1, 1))($OutMask$)
$OutMask$ = Conv2D(input_channel, (1, 1))($OutMask$)
$OutMask$ = Activation('sigmoid')($OutMask$)
// Combine the trunk branch with soft mask branch: (1 +
   $OutMask$) * $OutMask$
$output$ = (1 + $OutMask$) * $OutTrunk$
// Output of attention unit: add the last residual unit
for $i \leftarrow 1$ to $p$ do
  | $output$ = ResidualUnit($output$)
end
return $output$
```

Pre-activation Residual Unit. The pre-activation Residual Unit was implemented as illustrated in Fig. 1. We added a batch normalization layer before subsequent activation and convolution layers.



Input
Batch Normalization
ReLU
1x1 Conv2D, input channel
Batch Normalization
ReLU
3x3 Conv2D, input channel
Batch Normalization
ReLU
1x1 Conv2D, output channel
Batch Normalization
1x1 Conv2D, output channel
Output

Figure 3. Structure of the pre-activation Residual Unit.

5. Results

In this section, we report the reproduction results and compare our results with the original work. Also, we extensively discuss the difference between our work and the original paper, and what insights we can learn from these discrepancies.

5.1. Project Results

Mixed Attention, Spatial Attention and Channel Attention. In this experiment, we evaluated three different attention types, mixed attention f_1, channel attention f_2, and spatial attention f_3.

We trained Attention-56 networks using these three attention types, and summarized the results in Table 6. The mixed attention has the best performance. Therefore, the subsequent experiments were conducted using the mixed attention mechanism.

Activation Function	Attention Type	Top-1 err. % (this work)
$f_1(x)$	Mixed Attention	**14.43**
$f_2(x)$	Channel Attention	17.40
$f_3(x)$	Spatial Attention	16.45

Table 6. Classification error (%) of Attention-56 with different attention types on CIFAR-10 Dataset.

Attention Residual Learning. In this experiment, we reproduced the results on evaluating the effectiveness of attention residual learning mechanism by comparing attention residual learning (ARL) and naive attention learning (NAL).

We constructed Attention-56, Attention-128 and Attention-164 by stacking 1, 3 and 4 Attention Modules respectively in each stage, as described in the original paper. The Attention-92 is a special case. We illustrated our reasonings in Sec. 4.1. The networks were trained with ARL and NAL attention mechanisms and the results are demonstrated in Table 7. For ARL, as we stack more Attention Modules in each stage, the performance increases, except for a special case Attention-92, whereas for NAL, the accuracy decreases with the number of Attention Modules in each stage.

Network	**ARL (Top-1 err. %)**	NAL (Top-1 err. %)
Attention-56	**14.43**	16.15
Attention-92	**12.63**	13.18
Attention-128	**13.29**	14.33
Attention-164	**12.02**	14.15

Table 7. Classification error (%) on CIFAR-10 for ARL and NAL.

Noisy Label Robustness. We reproduced the experiment to test the noise resistant property of the Residual Attention Network on CIFAR-10 dataset. We injected multiple levels of noise to the training label, and the ratio of noisy labels was set to be 10%, 30%, 50%, and 70% respectively. ResNet-56 was selected as a baseline method for comparison.

The experimental results were summarized in Table 8. It is shown that the decrease of Attention-56 test error is significantly slower than that of ResNet-56 when we increase the noise level, although the test error of Attention-56 is slightly higher than ResNet-56 at the 10% noise level. The overall better performance of Attention-56 suggests that our experiment is able to reproduce the robustness of Residual Attention Network under different noise levels.

Noise Level	**ResNet-56 err. (%)**	**Attention-56 err. (%)**
10%	15.65	18.02
30%	24.95	21.06
50%	40.46	29.18
70%	61.58	40.68

Table 8. Test error (%) on CIFAR-10 under different levels of label noises.

Performance on the CIFAR-100 Datasets. We also tested the performance of Attention-56, Attention-92, Attention-128, and Attention-236 networks on CIFAR-100 dataset. We were unable to run Attention-452 due to insufficient GPU memory. The results were summarized in Table 9. The CIFAR-10 results were provided for comparison. All networks show significantly larger test error on CIFAR-100 than CIFAR-10 dataset. As stacking more Attention Modules, the accuracy first increases, and drops significantly for Attention-236.

Network	params x 10^7	CIFAR-10 (Top-1 err. %)	CIFAR-100 (Top-1 err. %)
Attention-56	3.52	14.43	40.00
Attention-92	4.47	**12.63**	39.83
Attention-128	4.13	13.29	**38.34**
Attention-236	6.09	13.74	52.39

Table 9. Comparisons of Residual Attention Networks with different trunk layer depth on CIFAR-10/100.

Performance on the ImageNet Dataset. Next, we tested Attention-52 on a resized ImageNet 32x32 dataset. The AttentionNeXt-56, AttentionInception-56 and Attention-92 were omitted because of the time limit. The training process was painfully slow, causing difficulties for us to evaluate the results and tune hyperparameters. We stopped the training process after 40 epochs. Our model has 27.26% top-1 accuracy.

5.2. Comparison of the Results Between the Original Paper and Students' Project

Spatial Attention and Channel Attention. We reproduced the overall trend that the mixed attention mechanism outperformed the other two attention mechanisms, but were unable to achieve the accuracies that the original paper reported. Our test accuracies are consistently about 10% lower than the test accuracies in the original paper. Moreover, in our experiments, the channel attention mechanism achieved a lower accuracy than the spatial attention whereas the original paper reported an opposite trend. The comparison is shown in Fig. 4.

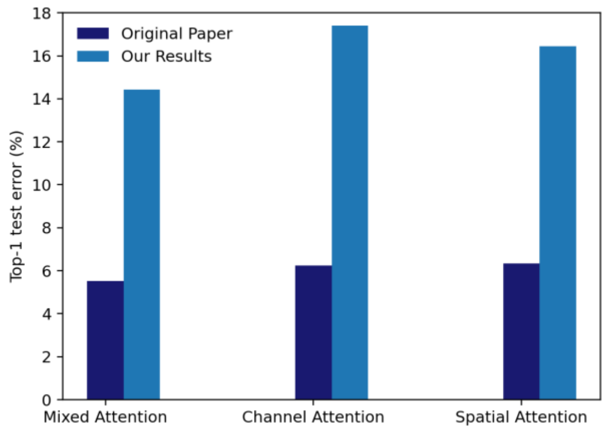


Figure 4. Comparisons of test error generated by different attention types against the original paper results.

Attention Residual Learning. Our results show that ARL outperformed NAL, but again, fail to reproduce the high accuracies demonstrated in the original paper. Same as the trend in the original paper, as we stacked more Attention Modules in each stage, the network performed better except for Attention-128, and NAL suffered degradation with increased number of Attention Modules. For the ARL mechanism, Attention-128 performed worse than Attention-92, different from the results in the original paper, in which the network's performance increased with the number of Attention Modules. Interestingly, for the NAL mechanism, we reproduced that Attention-92 outperformed the other networks. The differences between our results and the original results are shown in Fig. 5.

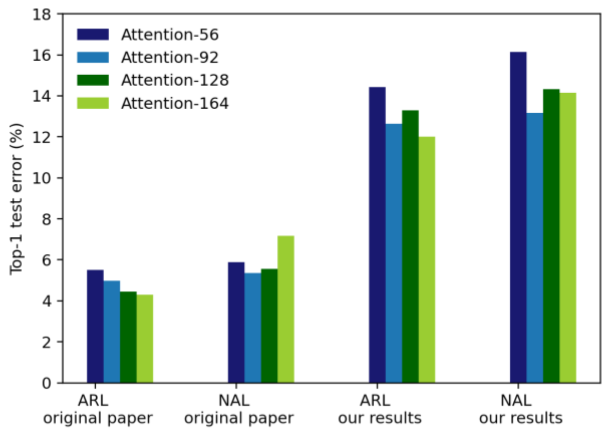


Figure 5. Comparisons between our results and original paper for ARL and NAL training.

Noisy Label Robustness. We were able to generally reproduce the superiority of Residual Attention Network over ResNet in terms of the noise resistant property, although ResNet performed slightly better at the 10% noise level. However, as we mentioned before, the test error of our attention network is significantly larger than that in the original paper. In our experiment, we selected the simplest model Attention-56 to save training time, and its robustness to noises might be slightly weaker than the Attention-92 in the original paper. However, even if we take this factor into consideration, the huge gap between our experiment and original paper results is still out of our expectation.

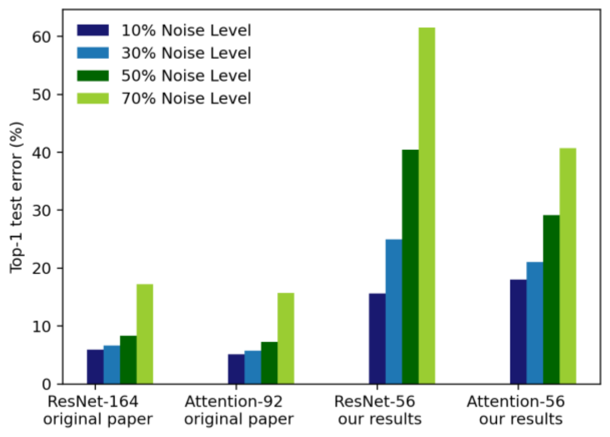


Figure 6. Comparisons between our results and the original paper under different levels of noises.

Performance on the CIFAR-100 Dataset. In the original paper, the test error is around 20% on the CIFAR-100 dataset. Our test error, however, is around 40%. We did not observe that as we stack more Attention Modules, the accuracy increases. For example, Attention-92 outperformed the other networks in the CIFAR-10 experiment, and Attention-128 has the best performance in the CIFAR-100 experiment. In addition, the number of parameters for our networks and theirs are different. Since the original paper did not report the CIFAR-100 results for Attention-56 and Attention-128, we omitted showing their result comparison in the figure.

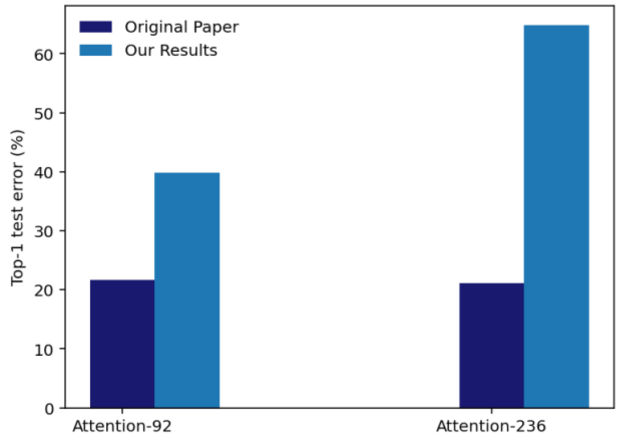


Figure 7. Comparisons between our results and original paper for Attention-92 and Attention-236 on the CIFAR-100 dataset.

Performance on the ImageNet Dataset. Due to limited time, we were not able to fully reproduce the experiments on the ImageNet dataset. The original paper tested numerous different basic units on the ImageNet dataset. We only tested Attention-56 on a resized ImageNet dataset. Our model could achieve a 27.26% top-1 accuracy, far lower compared with their 78.24% top-1 accuracy.

5.3. Discussion of Insights Gained

Our test accuracies of the networks trained on the CIFAR-10 and CIFAR-100 datasets are worse than the results from the original paper, by a large margin of 10%. As a starting point, we used the same data augmentation methods, hyperparameter settings, parameter initialization techniques [10], and training algorithms as described in the paper. However, our optimization diverged due to the large initial learning rate 0.1. We had to reduce the initial learning rate to 0.0001. We also changed the batch size from 64 to 32, built a more complex model, and used the Adam optimizer instead of SGD with the Nesterov momentum and weight decay because such changes could lead to higher accuracies in our experiments. Our networks suffered from a severe overfitting problem as shown in Fig. 8, especially for very deep networks. Our training accuracies were similar or higher than the reported test accuracies in the original paper but our test accuracies were significantly lower than theirs. Although numerous methods, including data augmentation, L_2 regularization, early stopping and dropout, were applied to alleviate overfitting, the overfitting problem could not be eliminated, and our test accuracies failed to reach the reported level. In all previous reproduction efforts we could find for the original paper, only one reached or even surpassed the test accuracy reported in the original paper, using the same hyperparameter settings but a different distributed training strategy from the paper. All the others used their own settings, resulting in similar or worse results compared with our work. This suggests that albeit it is possible to achieve the original results with the original settings, it is tricky and difficult. We also doubt that early stopping negatively impacted our results, because the original paper terminated the training process after 160k iterations, but our stopping criterion is whenever the validation accuracy plateaus. Since in the ResNet experiments, their validation accuracy can still be improved after a plateau stage, we suspect if we have not early stopped the training process, we would be able to reproduce the original results. Our learning rate schedule is also different from theirs. The authors divided by 10 at 64k and 96k iterations. However, we schedule the learning rate to be divided by 10 whenever the validation accuracy plateaus. Our final learning rate was 10^{-6}, much smaller than theirs. Such a small learning rate is likely to make the training process slow.

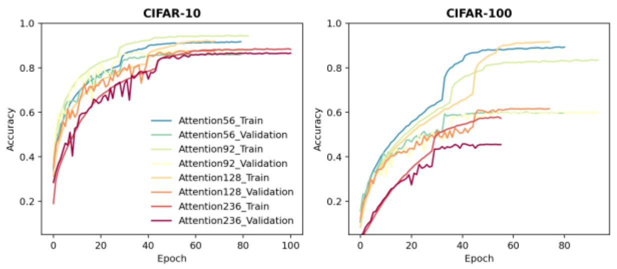


Figure 8. Training process of networks with different trunk depth on CIFAR-10 and CIFAR-100 datasets.

Interestingly, we found that more Attention Modules in each stage does not mean a better performance of the network, different from the conclusion in the original paper. On the CIFAR-10 dataset, our Attention-92 outperformed Attention-128. A possible reason is that we used a {1, 2, 3} Attention Module configuration in Attention-92 and the {2, 2, 2} configuration in Attention-128. As we stack more Attention Modules in later stages, rather than distributing them equally in each stage, the network might be more capable in capturing important low-level features distilled in the later stages, resulting in better performance. It is therefore worthwhile conducting more experiments to test whether it is possible to surpass the original accuracies using this Attention Module configuration. Another probable reason is that we used the same data augmentation methods and hyperparameter settings for all the networks in the CIFAR-10 experiment. Attention-128 and Attention-236 overfitted to the training set. In the original paper, the authors also seem to use the same training strategy for all the networks. Since no training history was provided, we wonder whether they also encountered the overfitting problem and how they solved it.

Apart from the performance difference between our networks and theirs, we found that our networks have different numbers of parameters from theirs. Their Attention-92 network, for example, has 1.9×10^6 parameters, nevertheless ours has 4.47×10^7 parameters, about 20 times larger. Namely, our model is much more complex. We initially built and tested the models that had the similar number of parameters compared with the original models. However, they did not perform well. The models could only achieve 60% test accuracy. We decided to double the number of input and output channels for every residual unit, resulting in a 16 times larger number of parameters, similar to the parameter number for the original ImageNet model. Our more complexed models were able to reach a higher accuracy, but prone to overfitting.

6. Conclusion

In this project, we re-implemented the Residual Attention Network which is a convolutional neural network applying the Attention Modules for image classification. We constructed networks with different trunk layer depth and evaluated their performance on the benchmark datasets CIFAR-10 and CIFAR-100. Our Attention-92 network outperforms the other networks on CIFAR-10 with 12.63% test error, while Attention-128 achieves the best performance on CIFAR-100 with 38.34% test error, suggesting that the accuracy does not necessarily increase as we stack more Attention Modules.

Similar to the original paper, our Residual Attention Networks show good robustness to different levels of noises, and we succeed to show that the residual attention learning can effectively avoid the performance dropping for the very deep networks. However, our test accuracies on CIFAR-10 are consistently 10% lower than that reported in the original paper. And for CIFAR-100 dataset, this accuracy gap can reach even larger than 20% and our networks suffer from a severe overfitting problem.

Although we extensively explored different model architectures, various hyperparameter configurations, and numerous training strategies including data augmentation, L_2 regularization, early stopping and dropout, it's still difficult for us to reach the reported accuracy level in the original paper and the overfitting issue cannot be thoroughly alleviated. Future exploration could be focused on further adjusting the parameter size to an appropriate level that can reach desirable performance and avoid overfitting simultaneously. Also, we found that stacking more modules in later stages rather than distributing them equally across all stages might help improve the test accuracy to the reported level.

7. References

[1] https://github.com/ecbme4040/e4040-2020fall-project-lzlz-lz2627-wz2483
[2] P. Zhou *et al.*, "Attention-based bidirectional long short-term memory networks for relation classification," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2016, pp. 207-212.
[3] W. Yin, H. Schütze, B. Xiang, and B. Zhou, "Abcnn: Attention-based convolutional neural network for modeling sentence pairs," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 259-272, 2016.
[4] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, "Image captioning with semantic attention," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4651-4659.
[5] E. Callaway, "'It will change everything': DeepMind's AI makes a gigantic leap in solving protein structures," *Nature*, 2020.
[6] F. Wang *et al.*, "Residual attention network for image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3156-3164.
[7] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European conference on computer vision*, 2016: Springer, pp. 630-645.
[8] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492-1500.
[9] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," *arXiv preprint arXiv:1602.07261*, 2016.
[10] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026-1034.

8. Appendix

8.1 Individual Student Contributions in Fractions

UNI	wz2483	lz2627
Last Name	Zhan	Zhang
Fraction of (useful) total contribution	1/2	1/2
What I did 1	Construct the network	Construct the network
What I did 2	Model training & Code comment	Model training & Parameter tuning
What I did 3	Github & Paper	Github & Paper