Residual Attention Network for Image Classification

Rahul Lokesh
UNI: rl3164
rl3164@columbia.edu

Shivam Ojha
UNI: so2639
so2639@columbia.edu

Sushant Tiwari
UNI: st3425
st3425@columbia.edu

Abstract

Attention mechanisms are used in diverse fields such as time series analysis and prediction, sequential modeling, feature extraction and other complex memory based systems. Attention techniques are extensively employed in the deep neural networks over various network architectures for feature extraction and region specific learning. One such area of application is in image classification where by learning attention related features, masking the unwanted regions, relevant patterns can be identified based on the attention mechanism. Since this type of information gathering requires "intelligently" reading and learning complex spatial patterns and identifying relevant regions in the image, minimizing the performance drop, as the depth of the attention network is increased, becomes the most crucial part. Modularizing feature learning units by using residual network architecture helps in the performance retention. This allows us to train highly deep neural networks keeping base level of performance with the help of identity mapping and skip connections. We, through this paper, compare the effects of various learning mechanisms, attention types and optimization techniques on CIFAR-10 Attention-56 and Attention-92 as well as CIFAR-100 Attention-56 data sets over ResNet architecture to come up with an optimum neural network for image classification.

1. Introduction

Using Residual Attention Network for image classification task is a naive and unexplored technique. It is a very challenging task to come up with the model which takes in advantages of convolutional Neural Nets, is able to extract and learn complex patterns in an image besides being robust to the noise and parameter changes. Attention mechanisms for feature extraction combined with the residual network results in high performance retention and helps in generalization over vast variety of data sets.[3]

Authors of the original paper titled "Residual Attention Network for Image Classification"[4], proposed an attention learning mechanism and a network based on Residual Learning. Employment of Attention Residual Learning mechanism gives significant improvement over blindly stacking the Attention Modules as it reduces the drop in performance when multiple attention modules are used for extracting complex features in an image. They employed these residual units at various parts of the Attention Module like in trunk units and soft mask units besides adding the trunk unit output to the product of mask and trunk units outputs for identity mapping.

Although, authors of the original paper managed to achieve significant accuracy on image classification by employing ResNet architecture on CIFAR-10, CIFAR-100 and ImageNet datasets[4], there model is computationally complex, takes more time to converge and uses large number of parameters which slows down the convergence. We, through this project, have used methods which were employed in the paper like different learning mechanisms such as Attention Residual Learning and Naive Learning, Activation Attention Types such as Spatial Learning and Mixed Learning to compare the performance of our model for computational complexity as well as validation accuracy. We seek to reduce the variance and time for the convergence of our model, besides structuring the code in more efficient manner as compared to the paper. We present and compare the effect of these combinations on Attention 56 and Attention 92 for CIFAR-10 and Attention 56 for CIFAR-100 to come up with a valid conclusion and a robust model which can be used in any application of Residual Attention Learning.

2. Summary of the original paper

2.1. Methodology of the original paper

In the original paper[4], Attention Modules were used for 3 stages along with the Residual Network. It used standard ResNet structure, trunk branch in the Attention Modules extracts and processes relevant features and patterns from the image. It uses 't' number of residual units per module. This trunk branch is compatible with

any network architecture for residual attention learning. 'p' number of residual units are used in the attention module before residual and attention units. The output of these residual units is given to both the Trunk and the Soft Mask branches. Soft Mask Branch is used for eliminating the unwanted features and aids in feature extraction. It uses down sampling in the form of max pooling to extract main features followed by 'r' number of residual units between two max pooling units. The combinations of down sampling and residual units are followed by interpolation or up sampling units to make the spatial dimension in accordance with the output of the next stage. This is followed by the convolution layers and the activation attention function which in the default case is taken as sigmoid/mixed attention. For catching specific patterns, mask is used to control input representations like an advanced feedforward control system. The Attention Mask which is used in the attention module serves 2 purposes: to update the filter coefficients during back propagation and for attention feature learning in the forward path.

Layer	Output Size	Attention-56	Attention-92
Conv1	112×112	7×7, 64, stride 2	
Max pooling	56×56	3×3 stride 2	
Residual Unit	56×56	$\begin{pmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{pmatrix} \times 1$	
Attention Module	56×56	Attention ×1	Attention ×1
Residual Unit	28×28	$\begin{pmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{pmatrix} \times 1$	
Attention Module	28×28	Attention ×1	Attention ×2
Residual Unit	14×14	$\begin{pmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{pmatrix} \times 1$	
Attention Module	14×14	Attention ×1	Attention ×3
Residual Unit	7×7	$\begin{pmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{pmatrix} \times 3$	
Average pooling	1×1	7×7 stride 1	
FC,Softmax	1000		
params×10^6		31.9	51.3
FLOPs×10^9		6.2	10.4
Trunk depth		56	92

Figure 1. Residual Attention Network architecture-ImageNet [4]

There are several drawbacks to using Attention Modules in a stacked manner or using a single Attention module architecture without the residual units:

1. If only one attention mask branch is used, it would not be able to extract complex patterns and features in the noisy image or in the images with lot of pixel variance. Also, in this case, obtaining maximum information from the image, would require many channels.

2. Changes will be implemented only once for a single Attention Module, if the weights for the soft mask are wrongly varied resulting in the failure of feature learning, successive modules would not be able to "relearn".

Authors of the paper[4] presented Residual Attention Network architecture for solving these problems: Mask unit in the trunk branch helps in region-specific unique learning. This allows the mask units in different trunk branches to learn multiple complex patterns and features. Also, residual units and skip connections for each attention module help to retain performance over many layers.

Two types of Learning Mechanisms were used in this paper:

Naive Attention Learning (NAL) : This involves direct multiplication of the trunk and mask branch outputs. As the range of the mask layers range from 0 to 1, this would lead to diminishing inputs for successive layers as the depth of the layer increases.

Attention Residual Learning (ARL) : This involves adding the trunk output to the dot product of trunk and mask branch output for identity mapping. So, even if the value of the dot product diminishes because of the mask branch, then also the trunk branch output with the similar input value be used for the successive stages. This identity mapping in this case will help in performance retention.

Also, this paper compared the results of three types of Activation Attention Functions in the soft mask branch:

Mixed Activation Attention: It uses Sigmoid activation without performing normalization for any channel.

Channel Activation Attention: It performs L2 normalisation for channels to remove variance in the pixels due to high ranges.

Spatial Activation Attention: It performs z-value normalisation by subtracting the mean from the data points in the channel and dividing by their standard deviation. The normalised value is passed through the sigmoid activation function for obtaining the soft mask output.

Authors of the paper [4] conducted experiments in 2 steps. In first step, they analysed each unit in the module and tried different attention mechanism types on CIFAR-10 and CIFAR-100 data sets. In the second step, they used ResNeXt and Inception for comparing the performance with the Residual Network architecture for attention based feature learning.

2.2. Key results of the reference paper

This section briefly covers the results of the paper we have referenced. The results of Residual Attention Network were evaluated on CIFAR-10, CIFAR-100 and ImageNet data sets[4].

For CIFAR-10, 4 types of Attention Module structures with varying trunk layer depth were employed by the authors of this paper. These are Attention-56, Attention-92, Attention-128 and Attention-164. Their model managed to achieve error rates of 4.99% with Attention-92 and 5.52% with Attention-56 using Attention Residual Learning (as described above). Error rates for CIFAR-10 as obtained in the original paper are described in the table below:

Network (Varying Trunk Layer Depth)	Attention Residual Learning (ARL)	Naïve Attention Learning (NAL)
Attention-56	5.52	5.89
Attention-92	4.99	5.35
Attention-128	4.44	5.57
Attention-164	4.31	7.18

Figure 2. Classification Error Rate(%) on CIFAR-10 [4]

This table summarises that as the number of Attention Modules employed per stage increase, there is reduction in % error rates for the proposed Attention Residual Learning. For Naive Attention Learning, reverse trend is observed i.e. with increase in the trunk layer depth from Attention-56 to Attention-164, classification error increases.

Attention Type	Details	Top-1 error (%)
Mixed Attention	Sigmoid Activation	5.52
Channel Attention	L2 Normalization	6.24
Spatial Attention	Z-Value Normalization followed by Sigmoid Activation	6.33

Figure 3. CIFAR-10 Attention-56 Test error% for different activation functions [4]

The paper achieved 5.52% top-1 error% with mixed attention activation function which is smaller than the error achieved by the spatial attention activation function (6.33% top-1 error). Hence, according to the results of the paper, using sigmoid activation without normalization for the soft mask branches will give lower error rates.

Using data augmentation, the authors of the paper[4], compared validation error for different architectures on ImageNet. By using Attention-92 structure, Top-1 % error of 19.5% was obtained by them. They obtained lowest Top-5 % error of 4.8% for ResNet-200 and Attention-92.

To summarise, the original paper discusses the importance of using Residual Attention Network for stacking

Network	params$\times 10^6$	FLOPs$\times 10^9$	Test Size	Top-1 err. (%)	Top-5 err. (%)
ResNet-152 [10]	60.2	11.3	224 × 224	22.16	6.16
Attention-56	31.9	6.3	224 × 224	**21.76**	**5.9**
ResNeXt-101 [36]	44.5	7.8	224 × 224	21.2	5.6
AttentionNeXt-56	31.9	6.3	224 × 224	**21.2**	**5.6**
Inception-ResNet-v1 [32]	-	-	299 × 299	21.3	5.5
AttentionInception-56	31.9	6.3	299 × 299	**20.36**	**5.29**
ResNet-200 [11]	64.7	15.0	320 × 320	20.1	4.8
Inception-ResNet-v2	-	-	299 × 299	19.9	4.9
Attention-92	51.3	10.4	320 × 320	**19.5**	**4.8**

Figure 4. Single crop validation error for ResNet [4]

Attention Modules. It recommends the use of Attention Residual Learning (ARL) over Naive Attention Learing (NAL). It achieves 3.90% error on CIFAR-10 with ResNet and 20.67% error on CIFAR-100. It achieves 0.6% accuracy improvement for the ImageNet in addition to reducing the trunk depth by 49%

3. Our Methodology for this project

We used different architectures of Residual Attention Network in combination with the activation attention types for CIFAR-10 and CIFAR-100 data sets. Through our work, we propose an architecture which would achieve decent test accuracy with low computational complexity. We study the effect of learning mechanisms, activation attention type, attention module stages and optimization techniques on the performance output of Residual Attention Networks in terms of testing accuracy and execution time to come up with the robust network. Though we use ResNet as the Network Architecture for attention based feature learning, our proposed model can be extended to any network architecture because of well organised and structured branches at each stage.

CIFAR-10: We analyze the proposed Residual Attention Network architecture and various constituent units at every attention module stage. For CIFAR-10 we used 2 different combinations of trunk layer depths which are Attention-56 and Attention-92. We used Mixed Activation attention type with Sigmoid activation for soft mask unit as well as Spatial Activation attention type with Sigmoid in combination with z-value normalisation for Attention-56 and Attention-92 network architectures. We also tested CIFAR-10 data set for the accuracy and execution time in case of Naive Attention Learning and Attention Residual Learning for this network combination. After analyzing the initial drop in performance by using SGD (with Momentum) as an optimizer for training Attention-56 and Attention-92 in CIFAR-10 data set, we propose and document the results using Adam as an optimizer. In addition we employ several combinations of pixel augmentation, epochs, kernel size

Layer	Output Size	Attention-56	Attention-92
Conv1	32 × 32	5 × 5, 64, stride 1	
Max pooling	16 × 16	2 × 2, stride 2	
Residual Unit	16 × 16	$\begin{pmatrix} 1\times1, 16 \\ 3\times3, 16 \\ 1\times1, 64 \end{pmatrix} \times 1$	
Attention Stage - 1	16 × 16	Attention × 1	Attention × 1
Residual Unit Down Sampling	8 × 8	$\begin{pmatrix} 1\times1, 32 \\ 3\times3, 32 \\ 1\times1, 128 \end{pmatrix} \times 1$	
Attention Stage - 2	8 × 8	Attention × 1	Attention × 2
Residual Unit Down Sampling	4 × 4	$\begin{pmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{pmatrix} \times 1$	
Attention Stage - 3	4 × 4	Attention × 1	Attention × 3
Residual Unit Down Sampling	4 × 4	$\begin{pmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{pmatrix} \times 1$	
Residual Unit	4 × 4	$\begin{pmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{pmatrix} \times 3$	
Average Pooling	1 × 1	4 × 4 stride 1	
FC, Softmax	10		
Trunk depth		56	92
Cifar10-attn-56-arl-spatial params $\times 10^6$		2.89	9.39
Cifar10-attn-56-arl-mixed params $\times 10^6$		2.89	-
Cifar10-attn-56-nal-mixed params $\times 10^6$		2.89	-
Cifar100-attn-56-arl-spatial params $\times 10^6$		2.94	-
Cifar100-attn-56-nal-mixed params $\times 10^6$		2.93	-

Figure 5. Our Architecture for Residual Attention Network

and strides of filters to optimize our model.

CIFAR-100: We analyze two model configurations for CIFAR-100: a) Attention-56 with Naive Attention Learning (NAL) and Mixed activation attention in soft-mask unit b) Attention-56 with Attention Residual Learning (ARL) and Spatial activation attention in soft-mask unit. Both these models use Adam as an optimizer.

We derive our conclusions on experiments based on the above mentioned model configurations and network architecture. We did not use ImageNet since it is a complex data set and training this data set with GPU resource available to us would have taken a lot of time for training to complete. Since the training time itself would take too long with ImageNet data set, we decided not to go ahead with it for our implementation and further experimentation. Our analysis will be applicable to any image data set requiring attention based feature learning.

4. Datasets and Data Preprocessing

CIFAR-10 and CIFAR-100 datasets consist of 50,000 training images and 10,000 validation images of size 32x32 with each image having 3 Red, Green, Blue (RGB) channels.

For CIFAR-10:

Training data shape: (50,000, 32, 32, 3)

Training labels shape: (50,000, 10)

Validation data shape: (10,000, 32, 32, 3)

Validation labels shape: (10,000, 10)

Shape of the training labels for CIFAR-10 is 50,000 X 10 whereas it is 50,000 X 100 for CIFAR-100. We calculated the mean and the standard deviation for every values per channel for CIFAR 10 and CIFAR 100 data sets. This normalization eventually speeds up the process. In addition, to improve the model performance, we used data augmentation techniques like horizontal flip, width and height shifting keeping the validation split as 0.2. We used image data generator to fit our training dataset.

For CIFAR-100:

Training data shape: (50,000, 32, 32, 3)

Training labels shape: (50,000, 100)

Validation data shape: (10,000, 32, 32, 3)

Validation labels shape: (10,000, 100)

5. Code Structure and System Requirements

The code[1] is structured to contain the residual attention network class in model_utils.py, which contains configurable attention modules, activation attention types, activation module stages and learning mechanisms. A residual unit function is created in res_units.py for adding residual unit before using activation in the residual attention network architecture. The learning_mech.py has attention Residual Learning function to implement attention residual learning module and Naive Attention Learning function to combine (dot product) outputs from trunk and soft mask branches without using the identity mapping/residual replication.

In the code, attention module is used to return model after addition of attention networks and residual units. The Trunk branch is used for feature processing and has 't' number of residual units in total. This can be adapted to any architecture like "ResNet". Soft mask branch which is used to mask the redundant environment in the image. After 'p' numner of pre-processing residual units, the output is given to trunk branch for feature processing and mask branch for masking the attention environment. Skip connection are used, which perform identity mapping. Their results are added to the output of the stacked layers.[5] The soft mask unit returns output for the soft mask unit which along with the trunk output can be used for Naive Learning or Residual Attention Learning. Attention module stage which defines attention module stage and returns output to the residual for the next step.

The Jupyter notebooks are then run on Google Cloud Platform (GCP) using Tesla T4 GPU. These notebook had code for training the model, adjusting the hyperparameters, data augmentation, summary and results as well as plots.

6. Experiments and Training Procedure

We modeled the Residual Attention Network by taking different combinations of strides, filter size, optimizers, learning mechanisms for soft mask units, attention stages, activation attention types and epochs. We analyzed each model configuration for test data accuracy and execution time or convergence. In this work we present 6 sets of model configurations, with each model saved in google drive[2], which were most optimal in terms of above performance metrics:

6.1. CIFAR-10 Attention-56 Attention Residual Learning and Mixed Attention

In this model configuration, we used Attention-56 as the trunk layer depth. Attention-56 uses 1 Attention Module per stage in the three stages. We kept the learning mechanism as Attention Residual Learning (ARL) which adds the trunk branch output to the dot product of the outputs from trunk and soft mask branches thereby using the identity mapping/residual replication. We used Sigmoid (Mixed Activation) as the activation function for soft mask unit without adding extra normalization after the sigmoid funtion.

```
average_pooling2d (AveragePooli (None, 1, 1, 512)    0          activation_99[0][0]
flatten (Flatten)               (None, 512)          0          average_pooling2d[0][0]
dense (Dense)                   (None, 10)           5130       flatten[0][0]
==================================================================================
Total params: 2,892,970
Trainable params: 2,877,290
Non-trainable params: 15,680
```

Figure 6. CIFAR-10 Attention 56 ARL Mixed Attention: Parameters and Output layer dimensions

We used Adam optimiser instead of Nestrov SGD with momentum of 0.9 which was used in the original paper as SGD does not have adaptable learning rate and using learning rate scheduler according to the number of epochs, in this case, would have increased the number of parameters besides being time consuming. We used Early Stopping with the patience level of 15. We used batch size of 64 and trained the model for 250 number of epochs.

```
# Evaluate model
model.evaluate(X_test, y_test, verbose=1)
```

```
313/313 [==============================] - 4s 14ms/step - loss: 0.4449 - accuracy: 0.8626

[0.44491156935691833, 0.8626000285148621]
```

Figure 7. CIFAR-10 Attention 56 ARL Mixed Attention: Test Accuracy

Total number of parameters in this configuration were 2,892,970. Average time to run one epoch = 48.6 seconds (excluding first epoch). We obtained test accuracy of 86.26%

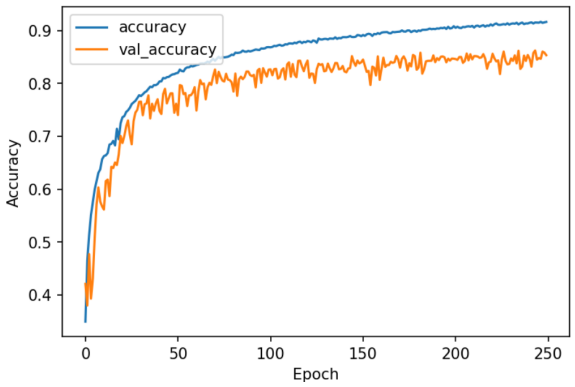

Figure 8. CIFAR-10 Attention 56 ARL Mixed Attention: Training and Validation Accuracy vs Epoch

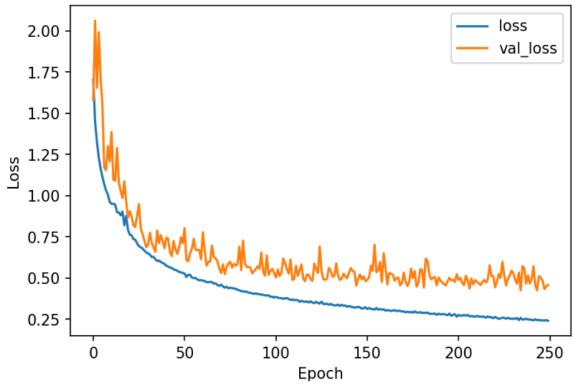

Figure 9. CIFAR-10 Attention 56 ARL Mixed Attention: Training and Validation Loss vs Epoch

6.2. CIFAR-10 Attention-56 Attention Residual Learning and Spatial Attention

In this model configuration, we used Attention-56 as the trunk layer depth. We used Attention Residual Learning (ARL) as the learning mechanism which solves the problem of performance loss due to the values in the soft mask layer output falling in between 0 and 1. We used Sigmoid followed by z-value normalization as the activation function for the soft mask unit.

We again used Adam optimiser instead of Nestrov SGD with momentum of 0.9 which was used in the original paper as SGD in this case also did not give good results and increased the converging time. We used Early Stopping

```
average_pooling2d (AveragePooli (None, 1, 1, 512)    0          activation_99[0][0]
flatten (Flatten)               (None, 512)          0          average_pooling2d[0][0]
dense (Dense)                   (None, 10)           5130       flatten[0][0]
==================================================================================================
Total params: 2,893,866
Trainable params: 2,878,186
Non-trainable params: 15,680
```

Figure 10. CIFAR-10 Attention 56 ARL Spatial Attention: Parameters and Output layer dimensions

with the patience level of 15. We used batch size of 64 and trained the model for 250 number of epochs.

```
# Evaluate model
model.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 4s 14ms/step - loss: 0.4671 - accuracy: 0.8666

[0.46714237332344055, 0.866599977016449]
```

Figure 11. CIFAR-10 Attention 56 ARL Spatial Attention: Test Accuracy

Total number of parameters in this configuration were 2,893,866. Average time to run one epoch = 47.7 seconds (excluding first epoch). We obtained validation accuracy of 84.70% and test accuracy of 86.65%

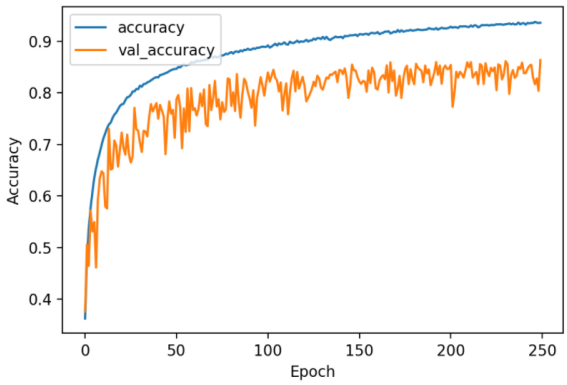

Figure 12. CIFAR-10 Attention 56 ARL Spatial Attention: Training and Validation Accuracy vs Epoch

6.3. CIFAR-10 Attention-56 Naive Attention Learning and Mixed Attention

In this model configuration, we used Attention-56 as the trunk layer depth. We changed the learning mechanism to Naive Attention Learning (NAL) which computes the dot product of the outputs from trunk and soft mask branches

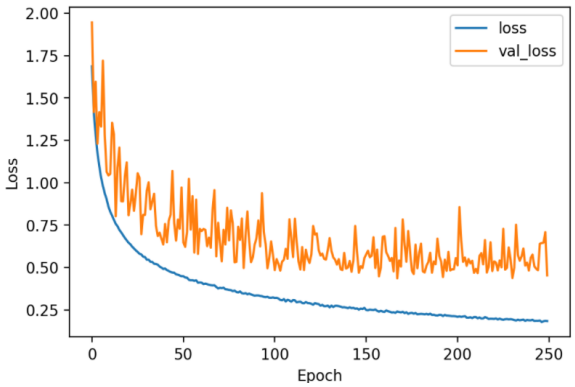

Figure 13. CIFAR-10 Attention 56 ARL Spatial Attention: Training and Validation Loss vs Epoch

without using the identity mapping/residual replication. We used Sigmoid activation as the mixed attention.

```
average_pooling2d (AveragePooli (None, 1, 1, 512)    0          activation_99[0][0]
flatten (Flatten)               (None, 512)          0          average_pooling2d[0][0]
dense (Dense)                   (None, 10)           5130       flatten[0][0]
==================================================================================================
Total params: 2,892,970
Trainable params: 2,877,290
Non-trainable params: 15,680
```

Figure 14. CIFAR-10 Attention 56 NAL Mixed Attention: Parameters and Output layer dimensions

We again used Adam optimiser instead of Nestrov SGD with momentum of 0.9 which was used in the original paper as SGD in this case also did not give good results and increased the time for convergence. We used Early Stopping with the patience level of 15 and a batch size of 64.

```
# Evaluate model
model.evaluate(X_test, y_test, verbose=1)
```

```
313/313 [==============================] - 6s 13ms/step - loss: 0.4867 - accuracy: 0.8566

[0.4866824150085449, 0.8565999865531921]
```

Figure 15. CIFAR-10 Attention 56 NAL Mixed Attention: Test Accuracy

Total number of parameters in this configuration were 2,892,970. Average time to run one epoch = 50.3 seconds (excluding first epoch). We obtained test accuracy of 85.66%

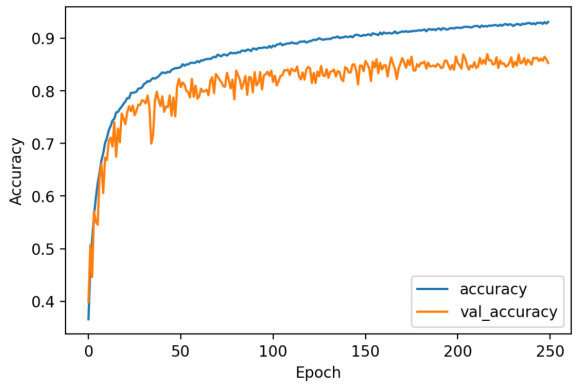

Figure 16. CIFAR-10 Attention 56 NAL Mixed Attention: Training and Validation Accuracy vs Epoch

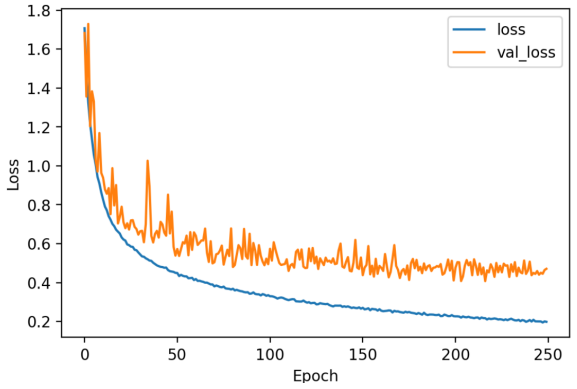

Figure 17. CIFAR-10 Attention 56 NAL Mixed Attention: Training and Validation Loss vs Epoch

6.4. CIFAR-10 Attention-92 Attention Residual Learning and Spatial Attention

In this model configuration, we used Attention-92 as the trunk layer depth. We used 1 Attention Module in Stage-1 of the Residual Attention Network, 2 Attention Modules in Stage-2 and 3 Attention Modules in Stage-3. We used the default Attention Residual Learning (ARL) and Sigmoid activation followed by z-value normalization as the spatial attention.

.

average_pooling2d (AveragePooli	(None, 1, 1, 512)	0	activation_159[0][0]
flatten (Flatten)	(None, 512)	0	average_pooling2d[0][0]
dense (Dense)	(None, 10)	5130	flatten[0][0]

Total params: 9,396,778
Trainable params: 9,365,738
Non-trainable params: 31,040

Figure 18. CIFAR-10 Attention 92 ARL Spatial Attention: Parameters and Output layer dimensions

We used Nesterov SGD with momentum = 0.9, decay = 0.0001 and learning rate = 0.1. We defined the learning rate scheduler to reduce learning rates by 10 for the epochs 65, 120 and 180 while using the same learning rate = 0.1 otherwise. Adam optimiser was taking lot of time to converge with the accuracy values saturating with the epochs whereas Nesterov SGD with momentum gave better performance for Attention-92.

```
# Evaluate model
model.evaluate(X_test, y_test)
```

```
313/313 [==============================] - 10s 22ms/step - loss: 0.6249 - accuracy: 0.7874
[0.6248746514320374, 0.7874000072479248]
```

Figure 19. CIFAR-10 Attention 92 ARL Spatial Attention: Test Accuracy

Total number of parameters in this configuration were 9,396,778. Average time to run one epoch = 65 seconds (excluding first epoch). We obtained test accuracy of 78.74%

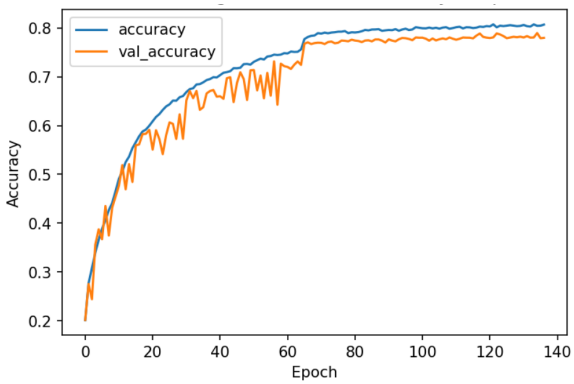

Figure 20. CIFAR-10 Attention 92 ARL Spatial Attention: Training and Validation Accuracy vs Epoch

6.5. CIFAR-100 Attention-56 Naive Attention Learning and Mixed Attention

In this model configuration, we used Attention-56 as the trunk layer depth. We used the learning mechanism of Naive Attention Learning (NAL) and Sigmoid activation as the mixed attention activation function.

We used Adam optimiser instead of Nestrov SGD with momentum of 0.9 which was used in the original paper as SGD in this case did not give good results and increased the time for convergence. We used Early Stopping with the patience level of 15 and a batch size of 64.

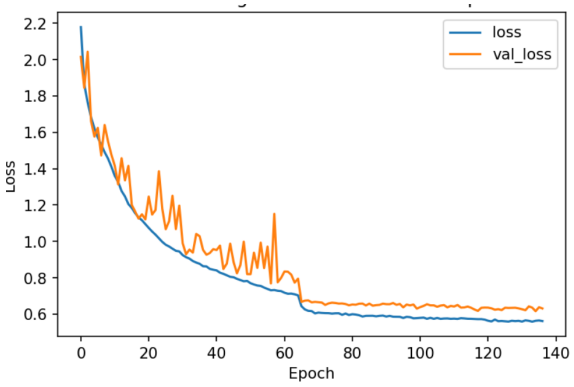

Figure 21. CIFAR-10 Attention 92 ARL Spatial Attention: Training and Validation Loss vs Epoch

average_pooling2d (AveragePooli	(None, 1, 1, 512)	0	activation_99[0][0]
flatten (Flatten)	(None, 512)	0	average_pooling2d[0][0]
dense (Dense)	(None, 100)	51300	flatten[0][0]

Total params: 2,939,140
Trainable params: 2,923,460
Non-trainable params: 15,680

Figure 22. CIFAR-100 Attention 56 NAL Mixed Attention: Parameters and Output layer dimensions

```
# Evaluate model
model.evaluate(X_test, y_test, verbose=1)
```

```
313/313 [==============================] - 7s 15ms/step - loss: 1.9709 - accuracy: 0.5346

[1.9709231853485107, 0.534600019454956]
```

Figure 23. CIFAR-100 Attention 56 NAL Mixed Attention: Test Accuracy

Total number of parameters in this configuration were 2,939,140. Average time to run one epoch = 47 seconds (excluding first epoch). We obtained test accuracy of 53.46%

6.6. CIFAR-100 Attention-56 Attention Residual Learning and Spatial Attention

In this model configuration, we used Attention-56 as the trunk layer depth. We used the default Attention Residual Learning (ARL) and Sigmoid activation followed by z-value normalization as the spatial attention.

.

We used Adam optimiser instead of Nestrov SGD with momentum and Early Stopping with the patience level of 15 and a batch size of 64.

Total number of parameters in this configuration were 2,940,036. Average time to run one epoch = 47 seconds (except first epoch). We obtained test accuracy of 54.76%

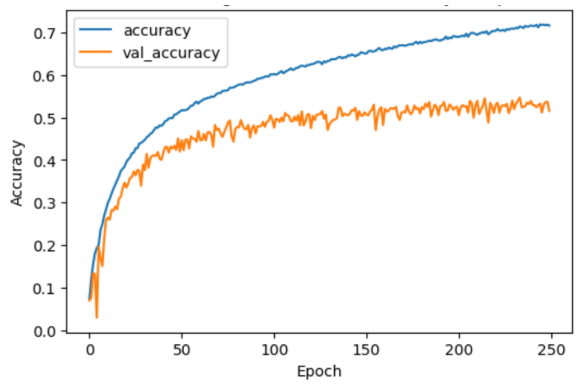

Figure 24. CIFAR-100 Attention 56 NAL Mixed Attention: Training and Validation Accuracy vs Epoch

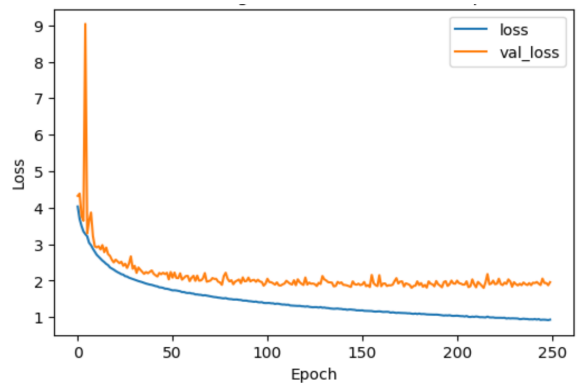

Figure 25. CIFAR-100 Attention 56 NAL Mixed Attention: Training and Validation Loss vs Epoch

average_pooling2d_1 (AveragePoo	(None, 1, 1, 512)	0	activation_199[0][0]
flatten_1 (Flatten)	(None, 512)	0	average_pooling2d_1[0][0]
dense_1 (Dense)	(None, 100)	51300	flatten_1[0][0]

Total params: 2,940,036
Trainable params: 2,924,356
Non-trainable params: 15,680

Figure 26. CIFAR-100 Attention 56 ARL Spatial Attention: Parameters and Output layer dimensions

```
# Evaluate model
model.evaluate(X_test, y_test, verbose=1)
```

```
313/313 [==============================] - 7s 15ms/step - loss: 1.7788 - accuracy: 0.5477

[1.7787835597991943, 0.5476999878883362]
```

Figure 27. CIFAR-100 Attention 56 ARL Spatial Attention: Test Accuracy

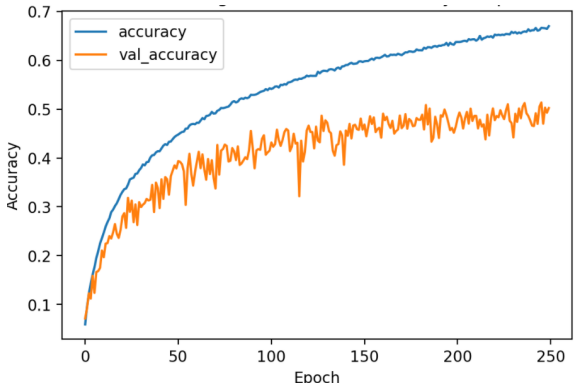

Figure 28. CIFAR-100 Attention 56 ARL Spatial Attention: Training and Validation Accuracy vs Epoch

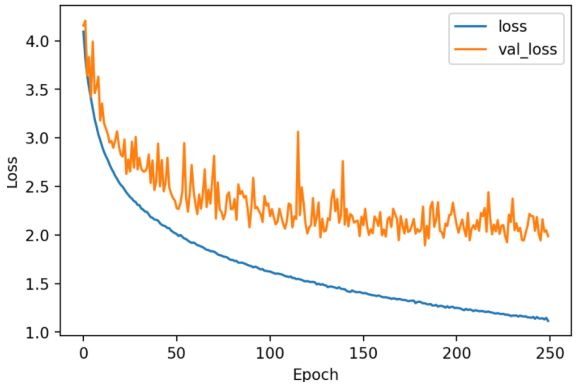

Figure 29. CIFAR-100 Attention 56 ARL Spatial Attention: Training and Validation Loss vs Epoch

7. Results

7.1. Accuracy

Dataset	Attention Network-Trunk Layer Depth	Optimization Type	Activation Attention Type	Learning Mechanism	Test Accuracy
CIFAR-10	Attention-56	Adam	Spatial	ARL	86.65%
CIFAR-10	Attention-56	Adam	Mixed	ARL	86.26%
CIFAR-10	Attention-56	Adam	Mixed	NAL	85.66%
CIFAR-10	Attention-92	Nesterov SGD with momentum	Spatial	ARL	78.74%
CIFAR-100	Attention-56	Adam	Mixed	NAL	53.46%
CIFAR-100	Attention-56	Adam	Mixed	ARL	54.76%

Figure 30. Accuracy for different model configurations

For CIFAR-10, We obtain highest test accuracy of 86.65% with Attention-56 network, Adam optimizer, Attention Residual Learning (ARL) and Spatial attention model configuration. For CIFAR-100, we obtain higher accuracy (54.76%) with Attention Residual Learning (ARL).

7.2. Training Time

Dataset	Attention Network-Trunk Layer Depth	Optimization Type	Activation Attention Type	Learning Mechanism	Approx. Training Time per Epoch
CIFAR-10	Attention-56	Adam	Spatial	ARL	47.7 sec
CIFAR-10	Attention-56	Adam	Mixed	ARL	48.6 sec
CIFAR-10	Attention-56	Adam	Mixed	NAL	50.3 sec
CIFAR-10	Attention-92	Nesterov SGD with momentum	Spatial	ARL	65 sec
CIFAR-100	Attention-56	Adam	Mixed	NAL	47 sec
CIFAR-100	Attention-56	Adam	Mixed	ARL	47 sec

Figure 31. Training Time per Epoch for different model configurations

For CIFAR-10, We obtain the lowest training time of 47.7 seconds with Attention-56 network, Adam optimizer, Attention Residual Learning (ARL) Spatial attention. For CIFAR-100, we obtain almost comparable training time of 47 seconds with both Naive Attention Learning (NAL) and Attention Residual Learning (ARL).

7.3. Efforts in Optimizing results

To reduce the computation time for each epoch, we changed the initial model configuration from having strided convolution to convolution followed by max pooling for the input image. This resulted in decrease in number of parameters to be learnt by the network and thereby reducing the computation time for each epoch.

We experimented with Adam and Nesterov SGD with momentum optimizers for all combinations of our models to come up with suitable learning rate for each model so it had better convergence and high accuracy.

To reduce overfitting, we added dropout and batch normalization layer after the stage 3 attention network. This resulted in model performance degradation as it decreased the validation accuracy by a significant margin. We added patience level of 10 initially to prevent the model from overfitting but in most of our models we observed significant instability in the initial few epochs, thereby, we increased the patience to a higher value 15. Due to this our model was slightly overfitting at the end but yet we observed that the test accuracies were comparable to the validation accuracies and there was no significant drop in the model performance.

8. Discussion- Results and Architecture

Instead of training on ImageNet for 224X224 dimensions as was done in the paper[4], we used CIFAR-10 and CIFAR-100 datasets which have 32X32 image dimensions for generalising our results. We used almost similar

Residual Attention Network structure with max pooling layer followed by residual unit. 3 Attention Module stages were stacked with 3 and 6 Attention Modules in the three stages for Attention-56 and 92 respectively. Each Attention Modules have Residual Units and Down Sampling units for performance retention. Our architecture did not vary too much with respect to the original as we focused on studying the effect of different units on model performance through our experiments.

As shown in the table for accuracy and training time for different model configurations, for CIFAR-10, we managed to obtain a higher test accuracy of 86.65% with Attention-56, as compared to the accuracy obtained by Attention-92 (78.74%). This suggests that we can have higher accuracy with a relatively simple network having fewer trunk layer depth provided that Attention Residual Learning (ARL) and Spatial Attention is used in soft mask units. This model also had the lowest training time per epoch of 47.7 seconds. Also, for CIFAR-10 Attention-92, Adam optimiser had a high convergence time and had saturation in accuracy values so using Nesterov SGD with momentum optimizer gave better results. We obtain higher test accuracy in CIFAR-10 (1% difference) for Attention Residual Learning as compared to Naive Attention Learning. This reiterates that by employing skip connections identity mapping through residual units, the performance degradation can be reduced.

8.1. Comparisons with the original paper

Though the authors of the original paper[4] managed to achieve error rates of 4.99% with Attention-92 and 5.52% with Attention-56 using Attention Residual Learning for CIFAR-10, they did not explicitly try different combinations of constituent network structures, optimizers, learning mechanisms. We, on the other hand, tested different architectures and hyperparamters to achieve a test accuracy of more than 85% (86.65%) with relatively simpler Attention-56, ARL Adam optimiser. Our model for CIFAR-10 takes only 47.7 seconds per epoch to train and can generalise for any state-of-the-art network architecture.

We reproduce the Attention-92 structure with CIFAR-10 dataset obtaining test accuracy of 78.74% which is 17% lower as compared to the original paper.Moreover, we have built this model with 9.3 million parameters as compared to the 1.9 million parameters used by them. For CIFAR-100, we had an error rate of 45% as compared to the 20.45% error in the original paper. However, we did manage to reduce variance and execution time with considerable accuracy with few of our configuration trials for CIFAR-10 Attention-92 and CIFAR-100 Attention-56 models.

8.2. Problem Faced

We faced challenges such as reducing the number of total trainable parameters for each model configuration. We experimented with different combinations of batch normalization and dropout layers to see their effect on training times and accuracy. Hence, we used max pooling followed by convolutions and changed strided convolutions to decrease the number of parameters and the training time.

Another challenge that we faced was to reduce overfitting and high variance in the model. We had to come up with an optimum optimiser for each case, with right learning rate, learning mechanism and the patience level. To debug the errors at every stage in the Residual Attention Network, we modularized our code and by following a proper structure, we were able to not only remove the errors but also study the output at each stage by adding dummy layers in the code.

9. Conclusion and Insights gained

Through this project, we build a robust Residual Attention Network architecture for Image Classification through our experiments and analysis on different combinations of strides, optimizers, learning mechanisms for soft mask units, attention stages, activation attention types and epochs. We gain insights on effect of trunk branch, soft mask branch, residual units for attention-based feature learning. We reiterate the importance of Attention Residual Learning over Naive Attention Learning for deeper network performance.

10. Contributions

Full Name	Sushant Tiwari	Shivam Ojha	Rahul Lokesh
UNI	st3425	so2639	rl3164
Fraction of total contribution	1/3	1/3	1/3
Contributions	Building model structure, coding and debugging, Paper Report, Model Training and evaluation	Building model structure and coding, Git hub files, Paper Report, Model Training and evaluation	Building model structure and coding, Code debugging, Paper Report, Model Training and evaluation
Note: Each project member was equally involved in each of the above-mentioned tasks by taking up different sections and dividing each task (such as coding, model training etc.) among themselves			

Figure 32. Contributions of Team Members

References

[1] Github of our implementation: Residual attention networks. github.com/ecbme4040/e4040-2021fall-project-RANC-rl3164-so2639-st3425$. 4

[2] Our implemented models. drive.google.com/drive/folders/1R7stBXi9x7ga_2523Re12TWYuHHewDFU?usp=sharing. 5

[3] Residual attention network — attention-aware features (image classification). towardsdatascience.com/review-residual-attention-network-attention-aware-features-image-classification-7ae44c4f4b8. 1

[4] C. Qian S. Yang C. Li H. Zhang X. Wang X. Tang F. Wang, M. Jiang. Residual attention network for image classification. arxiv.org/pdf/1704.06904.pdf. 1, 2, 3, 9, 10

[5] S. Ren K. He, X. Zhang and J. Sun. Deep residual learning for image recoginition, 2015. arXiv: 1512.03385. 4