UNIVERSITI TUNKU ABDUL RAHMAN

# UCCD3074 Deep Learning for Data Science

# Reimplementation of ResNeXt for binary image classification with limited amount of data.

Research-based [x] Application-based [ ]



| Name | Samuel Tan Joo Woon (Leader) | Chew Yong Khang | Ang Jian Wei |
|---|---|---|---|
| Programme | CS | CS | CS |
| Student ID | 1907010 | 1902461 | 1900952 |
| Contribution | 1/3 | 1/3 | 1/3 |

# 1. INTRODUCTION

## 1.1 Background and Problem Statement

Deep convolutional neural networks have actually led to a series of breakthroughs in image classification. In order to obtain a complete understanding of image classification, our project is trying to reimplement an image classification model that was proposed in 2017, ResNext. ResNext is a model based on ResNet with the introduction of the concept of Aggregated Transformations.

This project aims to reimplement the model and verifying the performance of the model to two datasets, CIFAR-10 and pikachu custom dataset. Pikachu dataset is a custom dataset that is prepared by our own to perform pikachu image binary classification. In this dataset, the images are classified into pikachu and not pikachu. The challenge of this project is to reimplement and evaluate the performance of ResNext to the dataset.

## 1.2 Project Scope

The scopes within the research are as below:

1. The dataset used for evaluation of the model is CIFAR-10 and pikachu dataset.
2. The performance of the reimplemented ResNext model will be compared against pytorch official implementation of ResNext-50 and Resnet-18 in pikachu dataset and Resnet-18 only in CIFAR-10.
3. The experiment process will be conducted on Google Colab environment.

## 1.3 Project Objective

The main objective of this project is to reimplement the ResNext model and perform analysis to the performance of the model. To achieve the main objective, the sub-objectives are listed below:

1. To study and evaluate the performance of the model with CIFAR-10 and pikachu dataset.
2. To compare the performance of the reimplemented model against the pytorch official implementation of RexNext-50 and Resnet-18.

## 2. RELATED WORK

In Deep Neural Network, that is an issue that happened before ResNet was designed, which is a vanishing and exploding gradient issue. At that time, researchers likewise thought that the deeper of the neural network has better performance but after they compare the results between 56-layers and 20-layers, training error and test error for 56-layers is higher than 20-layers.

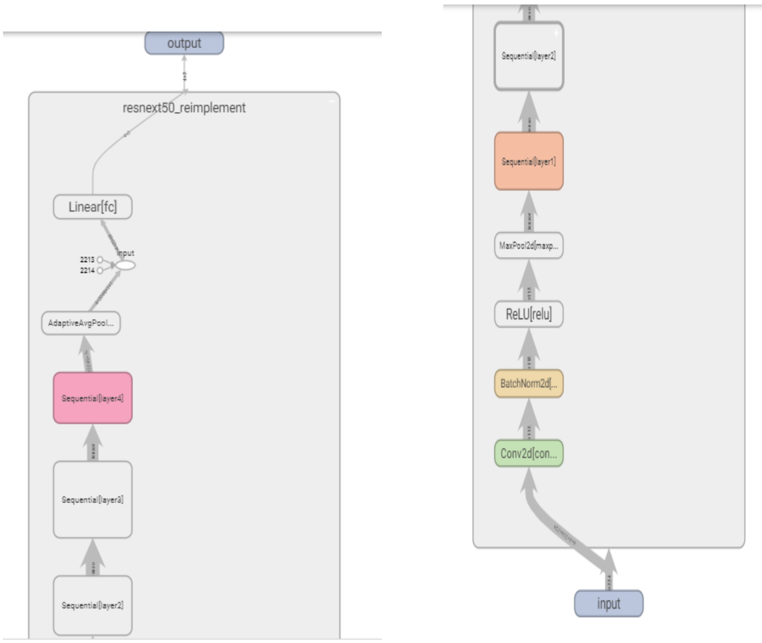
Plain Network, such as VGG nets is the convolutional layers mainly have 3x3 filters and two simple design which is same output feature map size and the layers have the same number of filters, and if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity layers that have a stride of 2. And in the training result on ImageNet for the plain net in 18 layers compared with 34 layers the training errors between are almost the same and the training error for 18 layers is lower than 34 layers. So in this training errors results, knows that the plain network for 34 layers was caused by the vanishing gradients.

Residual Network has the same design with the plain network, but it inserts the shortcut connection (Resblock) into the network which turns the network into counterpart residual version. Skip connections is an architecture that to batch normalize and remove the FC layers at the end of the network so that using this approach is used to replace layers, learn the underlying mapping and allow the network to fit the residual mapping. And using this approach is able to encounter the vanishing gradient problem due to the output never being zero, because the final output must add the input when applying skip connection. From the Image Net Classification using ResNet- 18 and ResNet-34, by comparing the result the training errors for Resnet-34 is lower than Resnet-18 which means that the performance for depth networks is better, and it resolves the vanishing gradient problem.

So for the Resnet architecture it can have countless layers and still be able to train easily without increasing the training error percentage and help to tackle the vanishing and exploding gradient problem using skip connection. Moreover, ResNet architecture has no necessities to send all neurons in every epoch and can improve the precision of the model.

## 3. Implementation ResNeXt-50

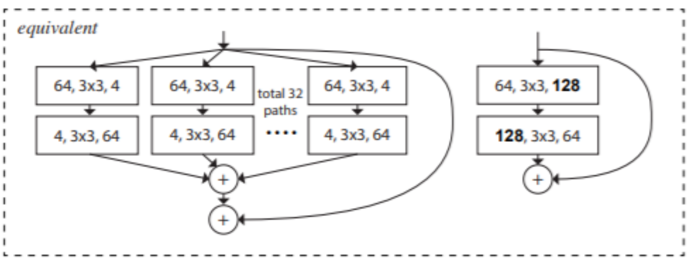
Figure 3.1: Use a top-down design diagram to show your overall system.




| ResNeXt-50 (32×4d) | |
|---|---|
| 7×7, 64, stride 2 | |
| 3×3 max pool, stride 2 | |
| 1×1, 128<br>3×3, 128, $C$=32<br>1×1, 256 | ×3 |
| 1×1, 256<br>3×3, 256, $C$=32<br>1×1, 512 | ×4 |
| 1×1, 512<br>3×3, 512, $C$=32<br>1×1, 1024 | ×6 |
| 1×1, 1024<br>3×3, 1024, $C$=32<br>1×1, 2048 | ×3 |
| global average pool<br>1000-d fc, softmax | |
| **25.0**$\times 10^6$ | |
| **4.2**$\times 10^9$ | |

ResNext is an image classification model that is based on Resnet with the concept of Aggregated Transformations. While it keeps the application of residual network from Resnet, inside each block of the residual block, the aggregated transformation is applied by distributing the input channel to different paths referred to as Cardinality, where it means the size of the set of transformations to be aggregated to perform convolution operation. It also can be viewed as having multiple convolution layers operate side by side to the input channel and the result is concatenated after the process.

Figure 3.2: The building block of ResNext.




### 3.1 Implementation Details

Our implementation of ResNeXt follows the figure 3.1 above. The structure of the model can split into 3 parts. The first part is the input block with one 7x7 convolution layer and one 3x3 max pool layer, the model then followed by residual blocks, where it consists of 3 convolution layers with the middle layer having cardinality of 32. The last part is followed by a global average pooling layer and fully connected layer for classification. The model structure for CIFAR-10 is ResNext-29 where it starts with a single 3x3 convolution layer then followed by 3 stages of residual block, each block has 3 convolution layers, and ends with an average pooling layer and fully connected layer (29 layer in total). Every convolution layer is followed by batch norm and relu layer. The optimizer for training the model is SGD with batch size of 8, initial learning rate of 0.1 and momentum of 0.9. The learning rate is divided by 10 for every 7 steps.

## 4. EXPERIMENT & EVALUATION

The evaluation metric used for our experiment is the cross entropy loss, accuracy and confusion matrix. Our expectation is to get the result that is close to the official ResNext-50 model to ensure the model is implemented correctly. Pikachu dataset consists of 774 images in the training set, 258 images in validation set and testing set. The result for all the models is close to each other, with the best result from Resnet-18.

Table 4.1: Performance in Pikachu dataset.

| **Performance metric** | **Reimplement ResNext-50 (our model)** | **ResNext-50** | **Resnet-18** |
|---|---|---|---|
| **Validation loss** | 0.5021 | 0.5047 | 0.3917 |
| **Test accuracy(%)** | 79.46 | 77.52 | 83.33 |
| **Confusion matrix** | TN: 95 FP: 34<br>FN: 19 TP: 102 | TN: 90 FP: 39<br>FN: 19 TP: 102 | TN: 102 FP: 29<br>FN: 14 TP: 102 |

CIFAR-10 dataset consists of 5000 images in training set and 1000 images in testing set. The model is able to train with CIFAR-10. However, due to time and resource constraints, we are only able to get the result below as we have run out of Google Colab and Google Drive usage. The result is showing as below:

Table 4.2: Performance in CIFAR-10.

| **Performance metric** | **Reimplement ResNext-50 (our model)** | **Resnet-18** |
|---|---|---|
| **Test accuracy(%)** | 45.06 | 66.38 |

## 5. CONCLUSION

The reimplementation of the model is a success as the result is close to the official implementation in the pikachu dataset. However, the result of CIFAR-10 is not satisfying as in the paper due to time and resource constraint. We have learned how to reimplement a model from a research paper and experience a deep learning model development pipeline in this project. The future extension of this project can be evaluating the model with other dataset such as ImageNet and COCO object detection.

## 6. CONTRIBUTIONS

The task has distributed evenly, like table below :

| **Team Members** | **Task Separation** |
|---|---|
| Chew Yong Khang | Prepare training data and preprocessing the data. |
| Samuel Tan Joo Woon | Design model architecture . |
| Ang Jian Wei | Train and test model. |

**GitHub Link**

https://github.com/samueltan3972/ResNext-Reimplementation (Our Repository)

**Referred Repository**

https://github.com/thkhoon/UCCD3074_Lab6

https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py

## REFERENCES


1) He, K. et al., 2016. Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
2) Xie, S. et al., 2017. Aggregated Residual Transformations for Deep Neural Networks. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
3) Krizhevsky, Alex. ,2012. Learning Multiple Layers of Features from Tiny Images. University of Toronto.
4) Ioffe, S., & Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.
5) Krizhevsky, A, 2009. Learning Multiple Layers of Features from Tiny Images.