Residual Dense Neural Network (ResDen)

Changyou Chen
Department of Computer Science
University at Buffalo
Buffalo, NY 14260
changyou@buffalo.edu

Utkarsh Shukla
Department of Robotics
University at Buffalo
Buffalo, NY 14260
ushukla@buffalo.edu

Swaminathan Pudukottai Jayaraman Murali
Department of Computer Science
University at Buffalo
Buffalo, NY 14260
swaminaa@buffalo.edu

Gulfam Hussain
Department of Computer Science
University at Buffalo
Buffalo, NY 14260
gulfamhu@buffalo.edu

Abstract

Deeper neural networks are difficult to train and pose vanishing gradients problems while training the network. To overcome these challenges, various neural network architectures have been proposed in recent times and these newly proposed architectures have helped researchers in the deep learning area in the image classification category by improving the accuracy rate. Resnet and Densenet are few such neural net architectures which have helped the training of networks that are substantially deeper than those used previously. Neural networks with multiple layers in the range of 100-1000 dominates image recognition tasks, but building a network by simply stacking residual blocks limits its optimization problem as pointed in the ResNet paper and the authors have shown that architectures with thousands layers do not add much and performs similar to architecture with hundred layers or even worse than that. This paper proposes a network architecture Residual Dense Neural Network (ResDen), to dig the optimization ability of neural networks. With enhanced modeling of Resnet and Densenet, this architecture is easier to interpret and less prone to overfitting than traditional fully connected layers or even architectures such as Resnet with higher levels of layers in the network. Our experiments demonstrate the effectiveness in comparison to ResNet models and achieve better results on CIFAR-10 dataset.

1 Introduction

We have seen that due to recent advancement and popularity, the research activity of different neural network architecture has been increasing a lot in the deep learning area. The main reason for researchers focusing more in the design of these neural network architecture is because of an increase in the layers in each of the proposed neural networks in the recent time and it motivates to explore the different connectivity patterns and similarity between these architectures to understand how effectively they are performing.

Based on various research done on the plain simple plain deeper network, It has been observed that plain deeper network pose below constraints:

- Vanishing/ Exploding Gradients: The neurons in the network die during the training process and are not able to reach the first layer during the backpropagation process.

- Harder Optimization: Due to simply stacking layers in the plain deeper network, it becomes very difficult to train such a plain deep network and optimization problem arises.

To overcome these issues of training simple plain deep neural networks [Kaiming He et al]. came up with a neural network called Residual Neural Network (ResNet) which included the idea of skipping the layers in between.

A residual neural network (ResNet) is an artificial neural network that consists of deep layers of neural networks but with skip connections to jump over some of the layers. Plain networks are difficult to train because when they start converging, its accuracy gets saturated and then degrades rapidly. Plain networks exhibit a higher training loss with an increase in number of layers and thus, there is no point in having a large number of layers. A solution to this deep plain network is proposed through Residual Neural network (ResNet) which utilizes the idea of shortcuts where the added layers are identity mapping, and includes double or triple layer skips that contain regularization (ReLU) and batch normalization learned from the shallower model.

ResNet helped the multiple layers to learn features at various levels of abstraction. This approach helped to design a deep neural network layer with much lesser training/test loss thereby achieving better accuracy, and it has even worked well on different datasets like COCO dataset.

In ResNet, layers are reformulated as learning residual functions with respect to the layer inputs, instead of learning unreferenced functions.

In the similar manner, considering the "**dense topology**" of Resnet where skip connection is used through Identity mapping to next forward layers after skipping 2-3 layers in between, a new neural architecture was proposed by [Huang et al.] called Densenet in which each layer was connected to every other layer in a feed-forward fashion.

In this paper, we present a simple architecture called **ResDen** which utilises the idea of both Resnet and Densenet as these two architectures follow the similar dense topology and includes the skip connections which makes use of addition and concatenation operations belonging to Resnet and Densenet respectively and combines the features and pass it to next layer.

To show the efficiency and effectiveness of the proposed ResDen model, we conduct extensive experiments on CIFAR-10. The proposed ResDen model requires fewer parameters than the existing state-of-the-art architectures while achieving better or at least comparable results. If you look at the CIFAR-10 dataset, ResDen-52 surpasses ResNet-56 and DenseNet-40 with 47% less parameters. The main contributions of this paper are as follows:

• ResNet and DenseNet have the same kind of architecture and path topology – "**dense topology**".

• ResDen with higher modularity is being proposed, which has a more efficient connection – and mixing of flexible inner link modules and outer link modules.

• ResDen-52 demonstrates its superior efficiency, in parameters and accuracy, over the state-of-the-art architectures.

Figure 1 and Figure 2 shows the basic blocks of ResNet and DenseNet respectively as shown below:

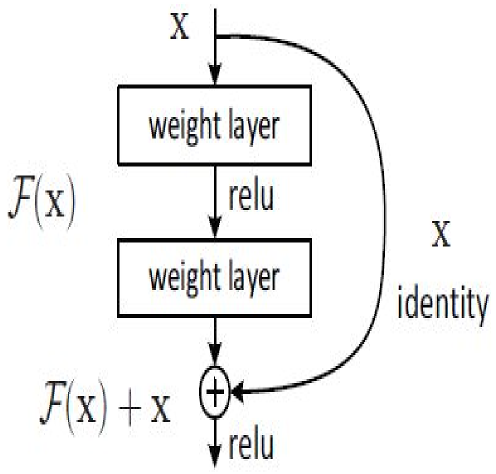


Figure 1: Residual learning: a building block

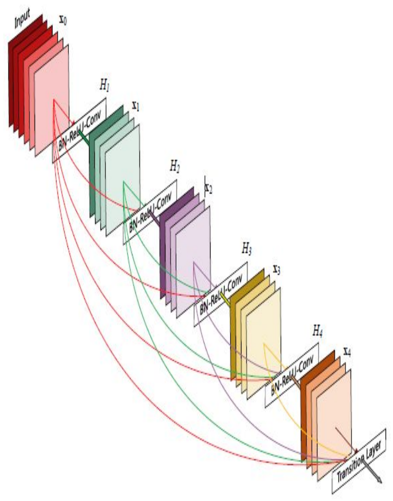


Figure 2: DenseNet building block

2 Related Work

After the introduction of AlexNet model in 2012 for the image classification with 8 layers, architectures with more number of layers became a talk in the deep learning area with researchers focusing on the impact of more layers in the neural networks. One such neural network is called Residual Neural Network (ResNet)[1] with layers starting from 18 upto 150+ were introduced and it performed well as compared to all previous networks. Since then, there have been variants of Residual networks implemented in the recent time like ResNext [8], Wide Residual Neural Network [11], ResNet Of ResNet (RoR), highway networks [12], Inception models, Inception-ResNets [10] which all utilized the idea of residual blocks and skip connections and they all promote learning capability by expanding the width or depth of the model. Furthermore, DenseNet achieved comparable accuracy with deep ResNet by proposing the densely connected topology, which connects each layer to its previous layers by concatenation. Instead of drawing representational power from extremely deep or wide architectures, DenseNets exploit the potential of the network through feature reuse, yielding condensed models that are easy to train and highly parameter efficient. Concatenating feature-maps learned by different layers increases variation in the input of subsequent layers and improves efficiency. This constitutes a major difference between DenseNets and ResNets. Compared to Inception networks, which also concatenate features from different layers, DenseNets are simpler and more efficient

3 Dataset

We selected the CIFAR-10 Dataset which consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.
Here are the classes in the dataset, as well as 10 random images from each of the class as shown in Figure 3 below:

Figure 3: CIFAR-10 dataset- pictorial class representation

4 Experiment

4.1 Dense Topology

Over here we talked about how ResNet and DenseNet have been derived from dense topology, and the difference between them is only of addition(+) and concatenation(||).

DenseNet is derived from “dense topology” . For DenseNet [Huang et al., 2017], the input of the nth layer is the concatenation of the outputs from all the previous layers. Hence, we can write DenseNet as: $\mathbf{P = H(X^0 \parallel X^1 \parallel \cdot\cdot\cdot \parallel X^{n-1})}$, where X denotes the layers and H is the composite function, which contains, batch normalization, convolution and activation function. The connection in DenseNet is only the concatenative connection (“||”) which increases the feature dimension gradually along the depths. It concatenates the raw features from previous layers to form the input of the new layer. Concatenation allows the new layer to receive the raw features directly from previous layers and it also improves the flow of information between layers.

ResNet is also derived from “dense topology”. As we know that Resnet contains additional functions, where all the previous layers are added. Given the standard definition from [He et al., 2016b], ResNet comprises a skip connection that bypasses the non-linear transformations H’ with following functionality-: $\mathbf{R_l = H'(R_{l-1})}$, where H’ is the composite function with non-linear transform, R is the feature map with skip connection. ResNet poses the additive form (“+”) that operates on the entire feature map. Combination of features from previous layers by element-wise addition, which makes the features more expressive and eases the gradient flow for optimization simultaneously.

4.2 ResDen Networks

In this section, we have discussed the inner/outer link modules. The symbol “+” and “||” represent addition and concatenation, respectively as discussed above. So let’s see the implementation of Inner and Outer link modules-:

In inner link modules we have taken ResNet feature maps, which is based on the additive connections. Following the above preliminaries, we denote the output R in the following equations for L layers:

$F_l^{in} = \Sigma^l_{i=0} R_l = R_{l-1} + H_l^{in}(R_{l-1})$ (1)

Where H_l^{in} is the non-linear composite function for the inner link module.

In the outer link module we have taken concatenated connections, which is from DenseNet. So showing in equation-:

$F_l^{out} = \Sigma^l_{i=0} R_l = R_{l-1} \| H_l^{out}(R_{l-1})$ (2)

So from equation 1 and 2, we have-:

$\mathbf{F = (R_{l-1} + H_l^{in}(R_{l-1})) \| H_l^{out}(R_{l-1})}$ "ResDen equation"

Since **K** is the growth rate term mentioned in DenseNet paper[Huang et al.] for the feature maps. We also introduce **K1** and **K2** for inner and outer layers respectively. So, from the above equations $\mathbf{H_l^{in}}$ and $\mathbf{H_l^{out}}$ represent K1 and K2. That is, k1 is the inner link size for inner link modules, and k2 controls the outer link size for outer link modules. As for the positional control for inner link modules, we simplify it into two choices – "fixed" or "unfixed". The "fixed" term is easy to understand – all the features are merged together by addition over the same fixed space, as in ResNet. Here is the explanation for "unfixed": there are extremely exponential combinations to pose the inner link modules' positions along multiple layers, and learning the variable position is currently unavailable since their arrangement is not derivable directly. So for tackling this problem, we remove the fixed part from the growth rate and use only unfixed positions with the addition. Figure 4 shown below displays the basic building blocks of ResDen:

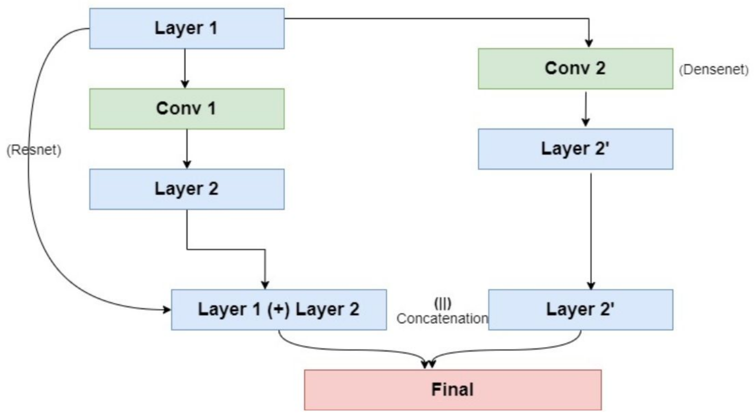


Figure 4: ResDen Architecture

4.3 Cutout Regularization

We performed the image classification on the CIFAR dataset using various approaches and the corresponding results were analyzed and recorded for each of the baseline models like Resnet and Densenet with the new model architecture ResDen.

We analyzed our model using Cutout regularization as against the general regularization techniques available for neural networks. Neural networks are more prone to overfitting due to dense architecture, and cutout regularization helps to overcome this issue. Cutout regularization focuses on regularizing the data space compared to other regularizations which focus in the function space of the model itself and thus it works better than other regularizations in the image classification category. Cutout distributes the most of the feature activations centrally towards the initial network layers which shows that the network is using more information from the data space thus forcing a CNN model to develop a more diverse set of features for classifying images. Cutout can be easily achieved by masking out the contagious squares of an image and can combined with any other existing data augmentation techniques to increase the model efficiency. We can see the cutout pictorial representation in the below Figure 5:

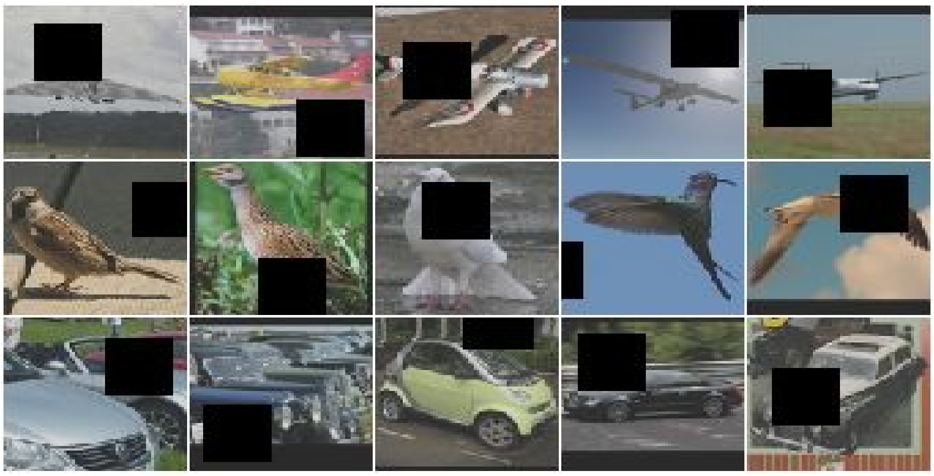


Figure 5: Cutout regularization on images

4.4 GELU(Gaussian Error Liner Units)

In this paper we have also discussed the use of the GELU activation function [5] compared to RELU function which is generally used in the high performing neural network. We perform our analysis using both the activation functions and see that the GELU performs well as compared to RELU and provides better accuracy results comparatively.

GELU is basically a new activation function and it is more robust to noisy inputs. It combines the properties from dropout, zoneout, and RELUs making it a viable alternative to previous nonlinearities. With GELU, the architecture outperformed significantly as compared to others by accommodating varying learning rates. The general form of the GELU can be given as,

$$\mathrm{GELU}(x) = 0.5x\left(1 + \tanh\left(\sqrt{2/\pi}(x + 0.044715x^3)\right)\right)$$

$$\mathrm{GELU}(x) = xP(X \le x) = x\Phi(x)$$
$$\approx 0.5x\left(1 + \tanh\left[\sqrt{2/\pi}\,(x + 0.044715x^3)\right]\right)$$

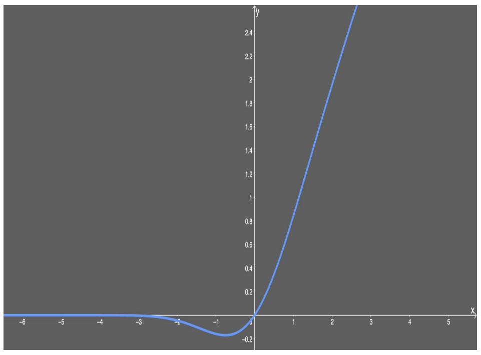


Figure 6: GELU activation function

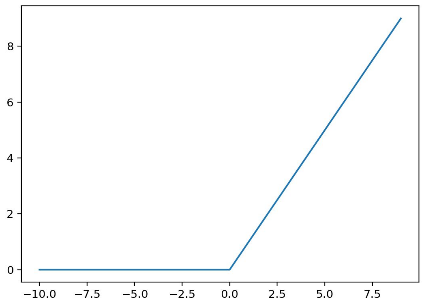


Figure 7: ReLU activation function

4.5 Implementation Details

The proposed model has multiple ResDen layers in each of the three blocks(similar to Dense Net). Each Resden layer has a structure that follows the figure. We make use of a composite function that comprises a bottleneck layer which reduces the number of parameters along with batch normalization and activation function. We are using CIFAR 10 as our dataset and convolution operation is done on the input(size of 32*32*3) with max(k1, 2*k2) output channels. We choose these parameters which balances both the accuracy and the complexity of the model. For the convolution layers with kernel size 3*3, inputs were zero padded by one pixel to keep the size of the feature map fixed. We also have a transition layer similar to Densenet which uses 1*1 convolution layer followed by 2*2 average pooling between two contiguous blocks. At the end of the last block, a global average pooling is performed and then a softmax classifier is attached. On the whole the input feature map size of three blocks are 32*32, 16*16, 8*8 respectively.

4.6 Training

For our model we use a training batch size of 64 and train for 300 epochs. The learning rate is initially set to 0.1 and we use a learning rate scheduler to divide the learning rate by 10 percent at epochs 120 and 200 respectively. We used a weight decay of 10^{-4}, Nesterov momentum of 0.9 without dampening and used the weight initialization by [15].

- ➢ Used Bottleneck layers with composite function
- ➢ Pooling layers are used to reduce the size of layers.
- ➢ Used k1 and k2 to indicate the growth rate of resnet and densenet feature map respectively.
- ➢ In total we 52 additive(104) layers including the fully connected layer.

5 Results

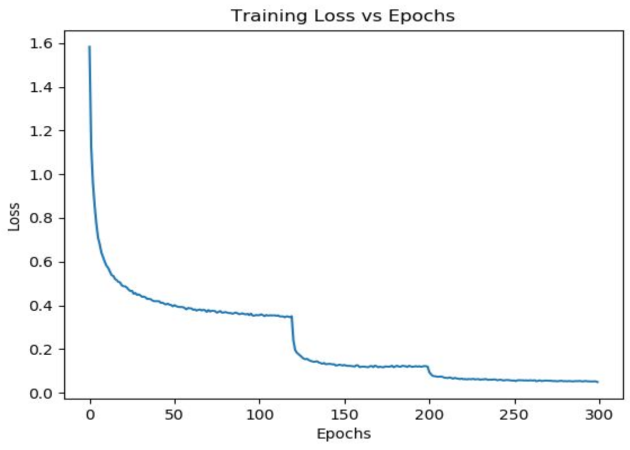
Following are the results that we got and compared with the baseline models that we had from the papers-:

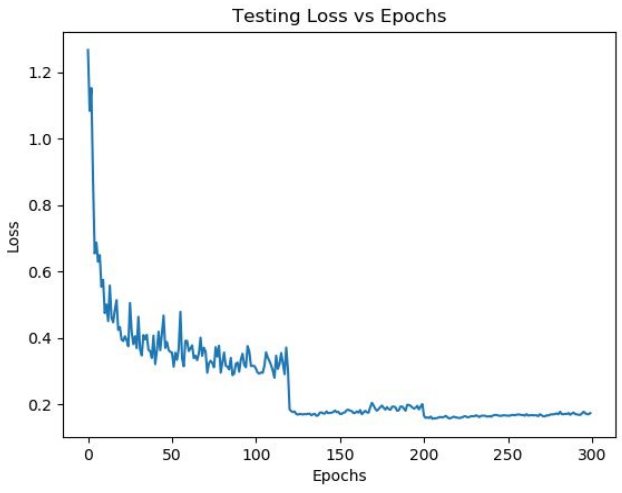
Model	Number of Parameters	Test Accuracy(%)
Baseline ResNet 56 (Normal Implementation)	0.85 M	93.03
Baseline ResNet 56 (including CutOut regularization)	0.85 M	92.90
DenseNet (L=40)	1.00 M	94.76
ResDen-52 (Normal Implementation)	0.51 M	94.66
ResDen-52 (including CutOut regularization)	0.51 M	95.21
ResDen (including CutOut regularization and GELU)	0.51 M	95.32

Table 1: Results

The calculation of number of parameters could be found along with the code in the text file **‘modelsummary.txt’**

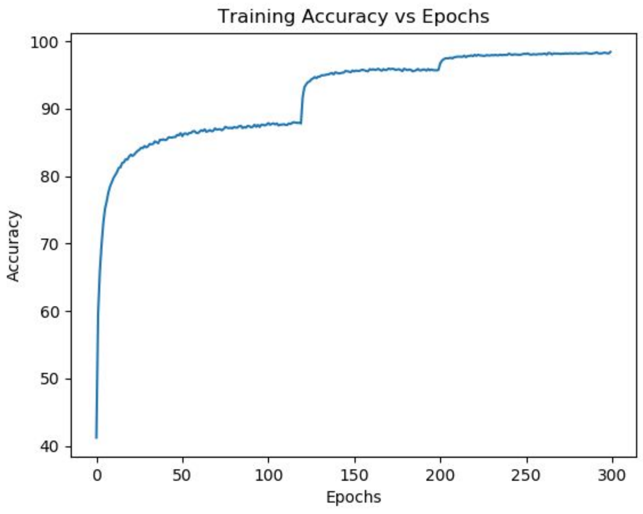
Training Loss vs Testing Loss for ResDen 52

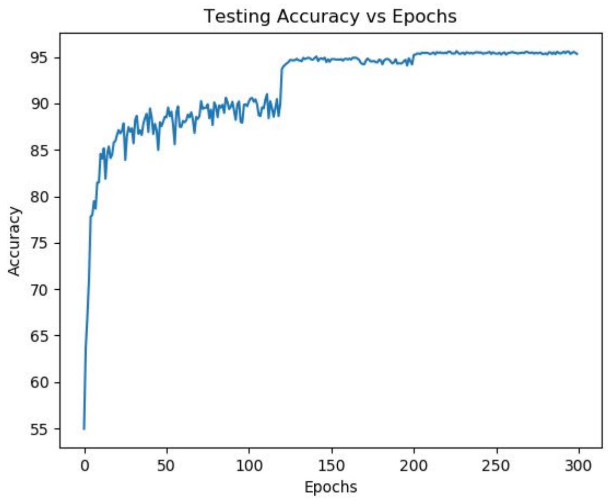




-

Training Accuracy vs Testing Accuracy for ResDen 52:





6 Future Works

There is recent neural architecture called NASNet proposed by Barret Zoph et al. [16] which basically presents the idea of learning transferable architectures for scalable Image recognition. Using NASNet, we can further enhance our study on this model to explore below activity in future:

- Possibility of finding unwanted layers using NASNet which do not contribute much in the middle.
- Using NASNet, performing hyperparameters tuning to achieve the best accuracy rate rather than manually tuning the parameters and analyzing the results. It will help in saving time.
- Could not analyze NASNet model on ResNet due to feasibility of high number of GPUs required.

7 Conclusion

In this paper we present a modification on the existing ResNet model for Image recognition with Dense architecture and perform our analysis on the CIFAR-10 datasets. Based on our analysis and experiment done, we found that baseline Resnet with DenseNet model works better and results in good accuracy. Along with general implementation, we also tried Cutout regularization against the normal regularization and found out that it provides better features for image classification and the overall accuracy of the model increases. There are different activation functions available in which Relu is one activation function which is being used very frequently in all the modern neural network architecture. We compared the RELU with a relatively new activation function called ‘GELU’ and found that the performance and time efficiency of our architecture improves as compared to RELU. Extensive experimental results represent that our proposed ResDen model is efficient in parameter.

8 Acknowledgements

We are grateful to Prof. Changyou Chen, co-instructors Le Fang and Zhenyi Wang and our teaching assistant Mohammad Abuzar Shaikh for constantly providing support and help throughout our project work and guiding us with their valuable feedback, which has been essential in this work. This project is completed under their guidance and satisfies the Final project requirement for Deep Learning course.

9 References

[1] He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing; Sun, Jian (2015-12-10). "Deep Residual Learning for Image Recognition".

[2] Residual neural network (https://en.wikipedia.org/wiki/Residual_neural_network)

[3] Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In CVPR, 2017.

[4] Günter Klambauer, Thomas Unterthiner, Andreas Mayr; "Self-Normalizing Neural Networks [2017]"

[5] Dan Hendrycks, Kevin Gimpel; "GAUSSIAN ERROR LINEAR UNITS (GELUs)"

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Identity Mappings in Deep Residual Networks. 2016.

[7] Karen Simonyan, Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014.

[8] S. Xie, R. Girshick, P. Dollar, Z. Tu and K. He. Aggregated Residual Transformations for Deep Neural Networks. arXiv preprint arXiv:1611.05431v1,2016.

[9] A. Veit, M. Wilber and S. Belongie. Residual Networks Behave Like Ensembles of Relatively Shallow Networks. arXiv:1605.06431v2,2016.

[10]Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke; "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning". 2016

[11] Sergey Zagoruyko, Nikos Komodakis; "Wide Residual Networks". 2016

[12] Rupesh Kumar Srivastava, Klaus Greff, Jürgen Schmidhuber; "Highway Networks". 2015

[13] https://www.henryailabs.com/3-2-19.html

[14] https://mlfromscratch.com/activation-functions-explained/#/

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassinghuman-level performance on imagenet classification.In ICCV, 2015.

[16] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, Quoc V. Le "Learning Transferable Architectures for Scalable Image Recognition", 2018.