A Comparative Analysis of ResNet Architectures

Conference Paper · December 2022

DOI: 10.1109/SMARTGENCON56628.2022.10083966

3 authors, including:

Piyush Nagpal

3 PUBLICATIONS **3** CITATIONS

SEE PROFILE

Ajitkumar Shitole

International Institute of Information Technology

29 PUBLICATIONS **190** CITATIONS

SEE PROFILE

2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)
Karnataka, India. Dec 23-25, 2022

A Comparative Analysis of ResNet Architectures

Piyush Nagpal
Department of Electronic and Telecommunication
Hope Foundation's International Institute of Information Technology,
Pune, India
piyushnagpal777@gmail.com

Shivani Atul Bhinge
Department of Computer Engineering
Hope Foundation's International Institute of Information Technology,
Pune, India
shivanibhinge@gmail.com

Prof.(Dr.) Ajitkumar Shitole
Department of Computer Engineering
Hope Foundation's International Institute of Information Technology,
Pune, India
ajitkumarsh1@gmail.com

***Abstract*— Neural networks today are becoming increasingly complex, from a few layers to more than 100 layers. The principal advantage of a totally deep neural network is that it may represent very complicated functions. Functions can be learned at different levels of abstraction, such as low-level boundary functions and high-level complex functions. However, the use of deep networks is not always efficient. A huge barrier to training them is vanishing gradients: very deep networks often have a gradient signal that goes to zero quickly, thus making gradient descent prohibitively slow. Deep residual networks are nearly like networks in which convolution, pooling, activation, and fully connected layers are superimposed. The only construct of a simple network that can be created as a residual network is the identifying link between the layers. Different types of ResNet can be developed depending on the depth of the network, such as ResNet-50 or ResNet-152. The number at the end of ResNet suggests the variety of layers in the community or the depth of the network. ResNet can be designed to any depth using ResNet's basic building blocks. In this article, we demonstrated a residual network with depths between 34 and 152 and tried to differentiate the architectures by training them on the same dataset.**

Keywords—ResNet, Residual Networks, Deep Residual Network, Cat-Dog Prediction.

I. INTRODUCTION

Researchers have found that it makes sense to say "deeper is better" when it comes to convolutional neural networks[2]. Therefore, with the demand for image processing and recognition, deep neural networks are becoming more complex and deepening.

Adding layers to a neural network can reduce accuracy. The more layers the network has, the more likely it is to extract more features from a complex image.

Different layers help detect different features. For example, the first layer can detect the edges of the image and the other layer can detect the shape. So adding more layers to this network can cause performance degradation issues because it can become less accurate or saturated. This is not due to overfitting, but due to vanishing gradient issues[6] , optimization features, and network initialization. This is due to the low rate of error in both training and testing as opposed to model overfitting. This is where the residual neural network comes into play. ResNet is called Residual Networks. A Residual Neural Network (ResNet) is an Artificial Neural Network (ANN) used for tasks such as image recognition. ResNet was first mentioned in the 2015 article Deep Residual Learning for Image Recognition by creators Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun[1] . ResNet solves this performance degradation or gradient loss problem[6] . ResNet residual neural networks are a special type of neural network, ANNs. ResNet is an extended version of neural networks because both contain activation layers of the BatchNormalization function of convolutional blocks[1] , but ResNet also contains residual blocks in addition to this.

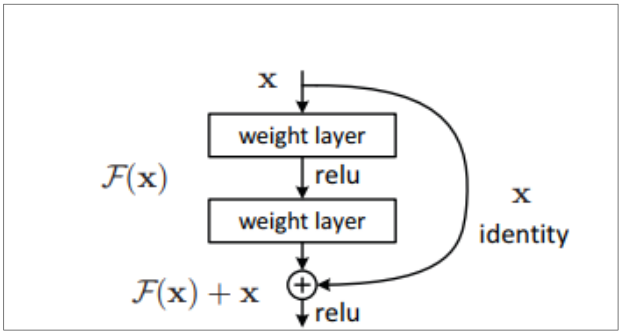

Fig. 1. Residual Block [18]

Figure 1 shows a direct connection with some intermediate layers omitted. This connection is called the skip connection [14, 15] and is the heart of the residual block[3]. Here we are adding the input layer to the output layer, so the output layer will not match. Since the output layer is different from the input layer, we add a convolution block to the shortcut path or skip connection [14, 15] to make the input and output look similar.

There are two ways to do the same thing. (1) Skip extra zero-padded joins to increase dimensionality. (2) Add a 1x1 convolutional layer to the input in the skip connection[14, 15]. Thus, the links in the residual block help the model learn the identification function, which allows the upper layer to perform as well as the lower layer.

Therefore, to add an extra layer to a deep network without compromising performance, the input and output layers match and limit as the network has to learn the identification function, which degrades performance.

Given a simple network, the output is H(x) = f(x). However, the output of the residual network is H(x) = f(x)+x. where x is the input to the function.

So H(x) = f(x) + x, so f(x) = 0 means H(x) = x. We get x as an output which is also an input. So the residual block makes it easy for the layer to read the identification function.

So we can add these ResNet blocks and form a deep network.

978-1-6654-5499-5/22/$31.00 ©2022 IEEE

1) Identification block - It is a standard block with the same input activation size and output activation size. (Refer to Figure 2 (a) for Identity Block.

2) Convolutional Block - If the size of the input block and the output block do not match, a convolutional layer is added to the skip join. The only difference from the identification block is that there is an additional CONV2D layer in the shortcut path. (See Figure 2 (b) for the convolution block).

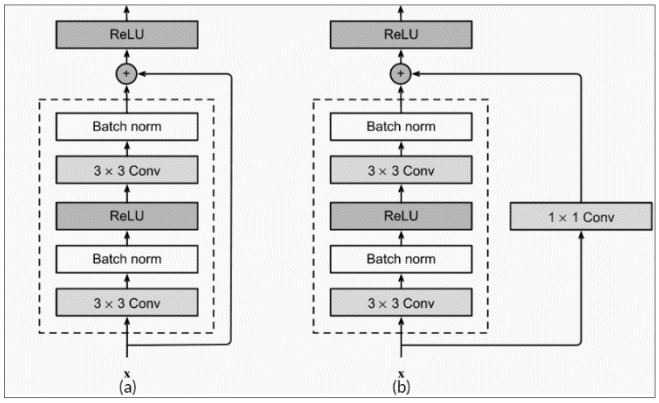

Fig. 2. (a) Identity Block (b) Convolutional Block[19]

Calculating output size of image

$$Output\ size = \left(\frac{n + 2p - f}{s} + 1\right) x \left(\frac{n + 2p - f}{s} + 1\right)$$

Fig. 3. Formula for Output Size of an image[2]

where “n” is the input volume, “p” is the padding, “ f ” is the size of the filter or kernel, and “s” is the number of steps. Step typically defines overlap between application operations. Conv2d, specifies the distance between successive convolutional filter applications. In a particular dimension, a value of 1 applies the operator to all rows/columns, a value of 2 means every second, and so on. In this article, we used the steps to reconstruct the input image. Therefore, if the step is 1, the shape is maintained as it is, and if the step is 2, the shape is changed, and there is a formula for this (refer to the formula in Fig. 3).

I. RELATED WORKS

Convolutional networks are powerful visual models that yield hierarchies of[8] capabilities. The paper by Jonathan Long, Evan Shelhamer, and Trevor Darrell indicates that convolutional networks by using themselves, trained to give up to cease, pixels to pixels, exceed the state of the art in semantic segmentation [8].

Their paper indicates the key insight to build “fully convolutional” networks that take input of the arbitrary size and produce correspondingly sized output with efficient inference and getting to know. They outline and element the distance of completely convolutional networks, explained their application to spatially dense prediction duties, and draw connections to prior [8] fashions. They adapted cutting-edge category networks (AlexNet[9], the VGG net[10], and GoogLeNet[11, 20] into completely convolutional networks and switch their found-out representations through fine-tuning [12] to the segmentation task. They then outline a singular architecture that mixes semantic facts from a deep, coarse layer with look data from a shallow, fine layer to produce correct and exact segmentations. Their completely convolutional network achieves the latest segmentation of PASCAL VOC (20% relative development to[8] sixty-two. 2% mean IU on 2012), NYUDv2, and SIFT flow, whilst inference takes much less than one 5th of a 2nd for a typical photograph.[8].

Gradient component centering is a new methodology for decomposing neural networks into biased and centered subnets which are then skilled in parallel. The decomposition can be implemented to any sample established thing in the network’s gradient and is designed such that the subnets are more amenable to optimization via gradient descent than the authentic network: biased subnets because of their simplified architecture, centered subnets because of a modified gradient that improves conditioning. The architectural and algorithmic changes mandated with the aid of this approach encompass both familiar and novel elements, often in prescribed mixtures. The framework suggests as an example that shortcut connections a well-known architectural characteristic have to work nicely along with slope centering, a brand new technique defined herein [13]. Their benchmark experiments undergo this prediction and demonstrated that issue-centering decomposition can speed up gaining knowledge appreciably without adversely affecting the skilled community’s generalization potential[13].

The paper presented by T. Raiko, H. Valpola, and Y. LeCun reworks the outputs of each hidden neuron in a multilayer perceptron network to have 0 output and 0 slope on common and uses separate shortcuts connections to model the linear dependencies rather. This modification goals at separating the issues of studying the linear and nonlinear elements of the whole enter output mapping, which has many blessings. They examine the theoretical houses of the transformation by way of noting that they make the Fisher statistics matrix in the direction of a diagonal matrix, and for this reason, a general gradient is closer to the natural gradient. They experimentally affirm the usefulness of the differences with the aid of noting that they make basic stochastic gradient mastering competitive with modern-day learning algorithms in pace, and they seem also to help discover answers that generalize better. The experiments consist of both types of small photographs and mastering a low dimensional representation for pix by way of using a deep unsupervised vehicle encoder network[14] The changes had been beneficial in all instances, with and without regularization and with networks from two to 5 hidden layers[14].

The influential Residual Networks designed by He Et al. Continue to be the gold preferred architecture in several medical publications[16]. They commonly function as the default structure in studies, or as baselines when new architectures are proposed. Yet there has been tremendous development in fine practices for training neural networks since the inception of the ResNet structure in 2015[16]. Novel optimization & data augmentation has expanded the effectiveness of education recipes. The paper offered by Ross Wightman, Hugo Touvron, and Herve J'egou, re-evaluated the performance of the vanilla ResNet-50 whilst trained with a procedure that integrates such advances[16]. They percentage aggressive training settings and pre-educated models inside the timm open supply library, with the desire that they'll serve as higher baselines for future work[16]. For

instance, with their education setting, a vanilla ResNet-50 reaches 84% pinnacle-1 accuracy at resolution 224×224 on ImageNet-Val without greater statistics or distillation[16].

Extremely profound convolutional networks with many layers have prompted critical decreases in blunders on serious benchmarks. Albeit the unrivaled expressiveness of the many layers can be profoundly alluring at test time, preparing exceptionally profound organizations accompanies its arrangement of[17] difficulties. The slopes can evaporate, the forward stream frequently diminishes, and the preparation time can be agonizing. To resolve these issues, we propose stochastic profundity, a preparation methodology that empowers the disconnected arrangement to prepare short organizations and utilizes profound organizations at test time[17]. We start with exceptionally profound organizations however during preparing, for every smaller-than-usual group, haphazardly drop a subset of layers and sidestep them with the personality capability. This straightforward methodology supplements the new progress of leftover organizations. It decreases preparing time considerably and further develops the test blunder fundamentally on practically all informational indexes that we utilized for assessment. With stochastic profundity, we can expand the profundity of lingering networks even past 1200 layers despite everything yielding significant upgrades in test mistakes (4.91% on CIFAR-10)[17].

In the paper written by Baoqi Li and Yuyao, they first deduced the backpropagation of networks and established an interpretation of ResNet that is similar to the ResNet. They then explained the gradient fading in a convolution neural network through the deduced backpropagation. lastly, the proposed an improved ResNet via adjustable shortcut connections and design and convex k strategy for improved ResNet according to the different region parameters changing rules[21].

In the paper written by using G. Montufar, R. Pascanu, K. Cho, and Y. Bengio, that suggests the take a look at the complexity of functions computable by using deep feedforward neural networks with piecewise linear activations in terms of the symmetries and the wide variety of linear areas that[22] they've. Deep networks are capable of sequentially mapping quantities of every layer's input area to equal output. In this way, deep fashions compute features that react equally to complex patterns of different inputs. The compositional shape of those functions allows them to reuse pieces of computation exponentially frequently in phrases of the network's depth. This paper investigates the complexity of such compositional maps and contributes new theoretical outcomes regarding the gain of intensity for neural networks with piecewise linear activation[22] features. Especially, the evaluation changed into no longer precise to an unmarried circle of relatives of fashions, and as an example, they rent it for rectifier and maxout networks. This stepped forward complexity bounds from pre-existing work and look into the behavior of devices in better layers[22].

II. GENERALIZING RESNET ARCHITECTURE

The architecture of ResNet has 4 stages as shown in Figure 4. The network can take input images that have height and width as multiples of 32 and the channel width can be 3 because we can have a max of 3 channels (R, G, B)[4] . For this paper, we have used the input image shape as (224 x 224 x 3)[2].

In ResNet 34 architecture we have 2 blocks first convolutional block and an identity block as discussed above[7] we use a convolutional block we have to change the shape of the input image for adding at the end of the block. In ResNet34 each block contains 2 layers of Conv2d, BatchNormalization, and an Activation function each and at the end, we add the initial input images to the output of these layers. Every ResNet architecture performs a few layers before going into stage 1, those layers are first, the initial convolution layer(conv2d) with a kernel size of 7 x 7 with strides as 2 and output filters as 64, and second, a max-pooling layer of kernel size as 3 x 3 and strides as 2.

We decided to add a padding of 3 in front of these initial layers so that the shape of the input image is 230 x 230 x 3. So when the image receives the first convolutional layer, the output shape will be 112 x 112 x 64. Then do a max merge. This is the layer that gives the output as 55 x 55 x 3 before we get into step 1. So the image input for step 1 will be 55 x 55 x 64. For ResNet50, the first stage network is a mix of convolution blocks and identity blocks. As discussed above, each block consists of three layers of conv2d, batch normalization, and activation functions. For identification blocks, each level consists of steps equal to 1 and the filter and kernel size specified in the architecture (see Table 1). For the convolution block, the first level, conv2d, the batch normalization, and activation functions are two steps, the rest are identical to the same block, but after the third level, short concatenations are added. Responsible for adding the original image to the output. this block. Referring to Figure 2.1, step 1 consists of a convolution block with stride 1 and filters 64, 64, and 256, and two identification blocks with filters 64, 64, and 256. The first identification block has a core size of 3 and the rest have a core size of 1. This is the basic architecture of all steps in ResNet50, only the filters change. The flat line of the figure. 4 is an identification block, meaning the input and output are the same size and can be added directly. the dotted line in figure 2.1 denotes a convolution block that can be added when the input and output are of different sizes (i.e. the input is greater than the residual output), and the default solution is to use a 1x1 convolution with a stride of 2. The input is halved in height and width, but the channel width is doubled[4]. As you move from one step to the next, the channel doubles in width, and the input halves.

Table 1 summarizes the size of the output at each level and the dimensions of the convolution kernel at each point in the structure. In ResNet34, 2 layers are superimposed on each other for each residual function F. The two layers are 3x3, and 3x3 convolutions. Finally, the network has an intermediate pool layer and a fully connected layer containing 1000 neurons (the output of the ImageNet class). Similarly, for deep networks such as ResNet50, ResNet101, ResNet152, etc., bottleneck design is used. For each residual function F, 3 layers are superimposed on each other [5] . The three layers are 1×1, 3×3, and 1×1 convolutions . A 1x1 convolutional layer is responsible for reducing and restoring dimensions.

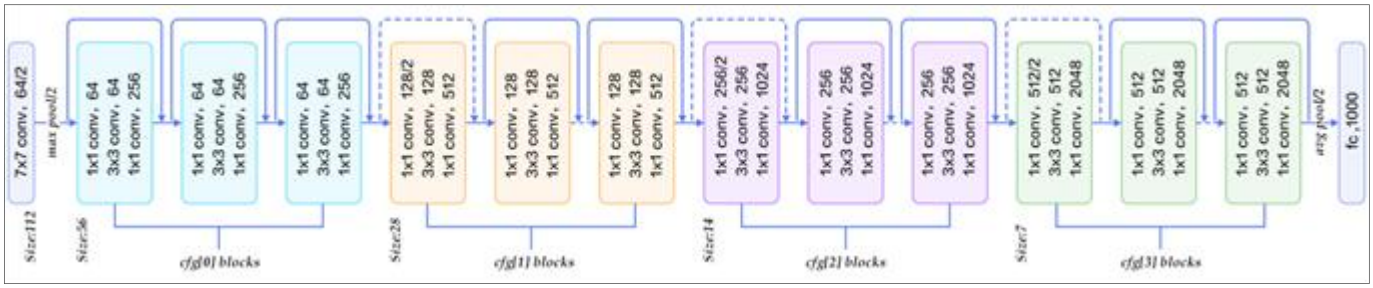

Fig. 4. Stages of ResNet[21]

TABLE I. SIZES OF OUTPUTS AND CONVOLUTIONAL KERNELS FOR DIFFERENT RESNET DEPTHS[2]

layer name	output size	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2			
conv2_x	56×56	3×3 max pool, stride 2			
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax			
FLOPs		3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

III. EXPERIMENTS

The dataset is 25000 which is split into train and test in 60 - 40 ratio respectively. Where equal labels of dog and cat are taken to avoid biased training data. After the split, the training data is 15000 and the test data is 10000. Moving on to the individual ResNet architecture experiments:

ResNet 34:

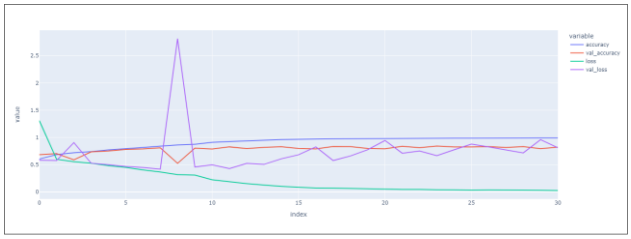

Fig. 5. Training and Testing accuracy and loss for ResNet34

Figure 5 shows the training accuracy which is around 99% and the test accuracy which is 82% highest, the training loss goes to 0.02 while the validation loss is up to 0.81.

And the confusion matrix (refer to figure 6) derives that 304 images were predicted dogs but were cats (refer to image 8). And 4252 were predicted cats but were dogs (refer to figure 9).

Hence we conclude that the model accuracy of ResNet34 architecture is 82.25%. (Refer to figure 7)

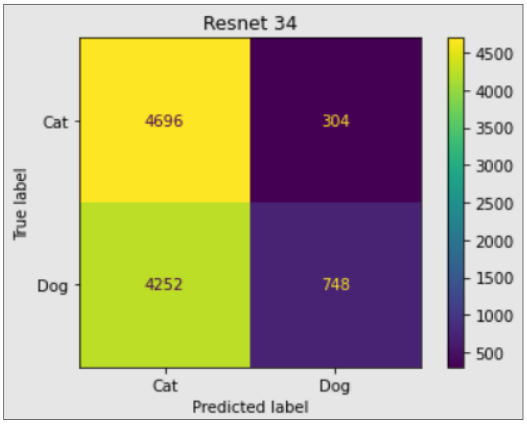

Fig. 6. Confusion Matrix of ResNet34.

```
[ ] old_model34 = load_model('/content/gdrive/MyDrive/Cloud Research/model/Training Models/resnet34/dog_cat_resnet34_after_spilt.h5')

[ ] loss, acc = old_model34.evaluate(images_test, Y_test, verbose=2)
    print("Trained model, accuracy: {:5.2f}%".format(100 * acc))

    313/313 - 120s - loss: 0.8111 - accuracy: 0.8225 - 120s/epoch - 382ms/step
    Trained model, accuracy: 82.25%
```

Fig. 7. Model Accuracy for ResNet34

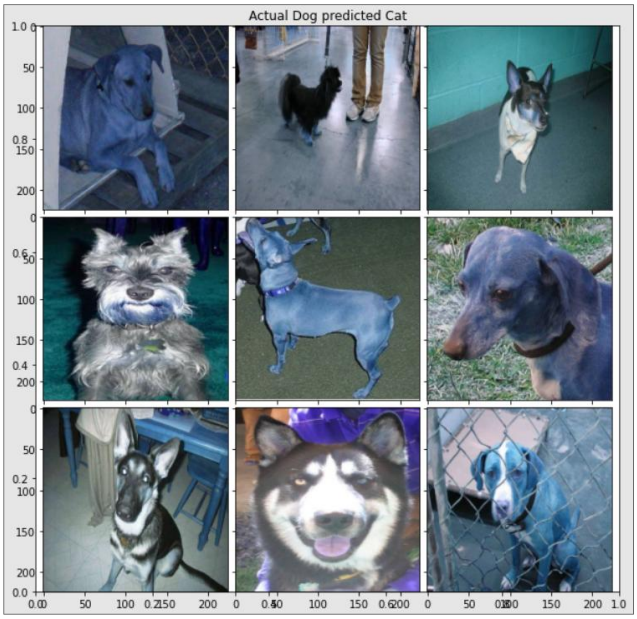

Fig. 8. Top 9 images for Actual Dog Predicted Cat for ResNet34.

Fig. 9. Top 9 images for Actual Cat predicted Dog for ResNet34

ResNet 50:

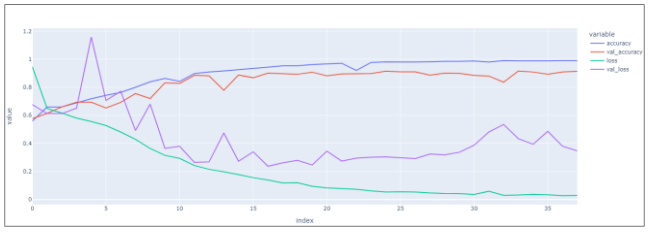

Fig. 10. Training and Testing accuracy and loss for ResNet50

Figure 10 shows the training accuracy which is around 99% and the test accuracy which is 91.39% highest, the training loss goes to 0.029% while the validation loss is up to 0.34%.

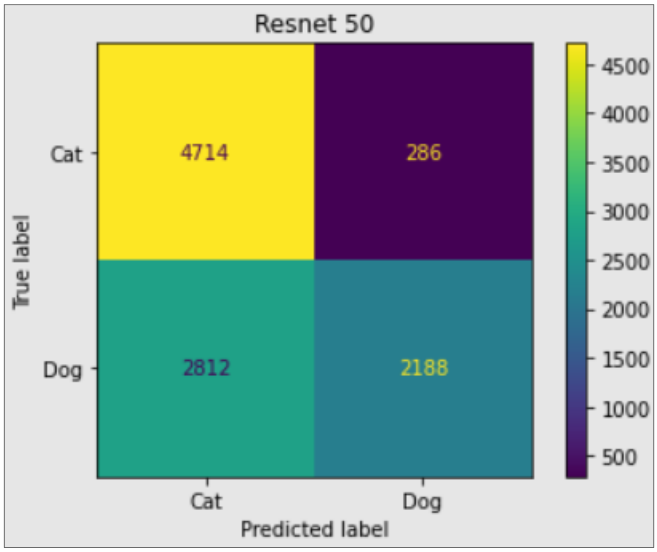

Fig. 11. Confusion Matrix of ResNet50.

And the confusion matrix (refer to figure 11) derives that 2188 images were predicted dogs but were cats (refer to image 13). And 2812 were predicted cats but were dogs (refer to figure 14).

Hence we conclude that the model accuracy of ResNet34 architecture is 91.39%. (Refer to figure 12)

```
[ ] old_model50 = load_model('/content/gdrive/MyDrive/Cloud Research/model/Training Models/resnet50/dog_cat_resnet50_after_spilt.h5')

[ ] loss, acc = old_model50.evaluate(images_test, Y_test, verbose=2)
    print("Trained model, accuracy: {:5.2f}%".format(100 * acc))

    313/313 - 31s - loss: 0.3473 - accuracy: 0.9139 - 31s/epoch - 100ms/step
    Trained model, accuracy: 91.39%
```

Fig. 12. Model Accuracy for ResNet50

Fig. 13. Top 9 images for Actual Dog Predicted Cat for ResNet50.

Fig. 14. Top 9 images for Actual Cat predicted Dog for ResNet50

ResNet 101:

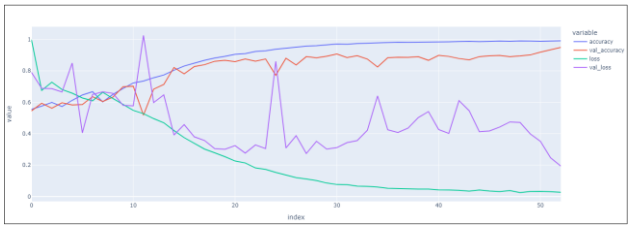

Fig. 15. Training and Testing accuracy and loss for ResNet101

Figure 15 shows the training accuracy which is around 99% and the test accuracy which is 94.89% highest, the training loss goes to 0.027% while the validation loss is up to 0.1939.

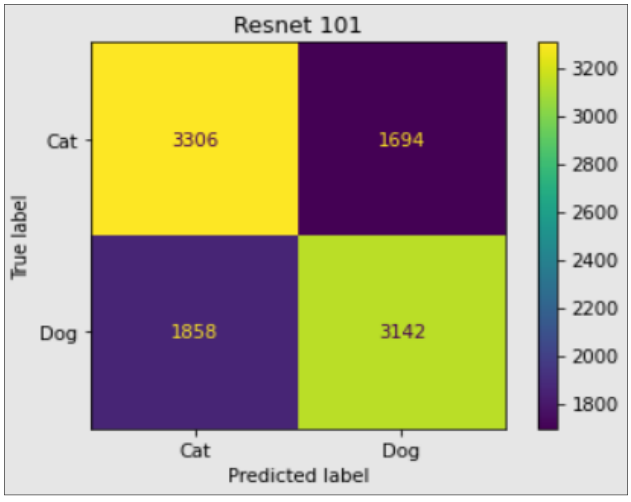

Fig. 16. Confusion Matrix of ResNet101.

And the confusion matrix (refer to figure 16) derives that 1694 images were predicted dogs but were cats (refer to image 18). And 1858 were predicted cats but were dogs (refer to figure 19).

Hence we conclude that the model accuracy of ResNet34 architecture is 94.89%. (Refer to figure 17)

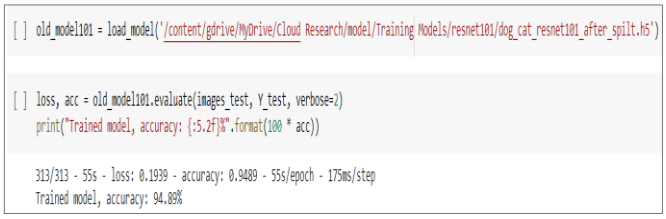

Fig. 17. Model Accuracy for ResNet101

Fig. 18. Top 9 images for Actual Dog Predicted Cat for ResNet101.

Fig. 19. Top 9 images for Actual Cat predicted Dog for ResNet101

ResNet152:

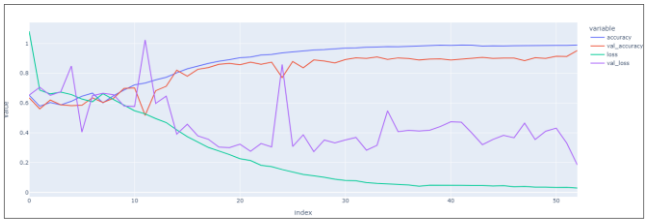

Fig. 20. Training and Testing accuracy and loss for ResNet152

Figure 20 shows the training accuracy which is around 99% and the test accuracy which is 95.29% highest, the training loss goes to 0.029% while the validation loss is up to 0.1850.

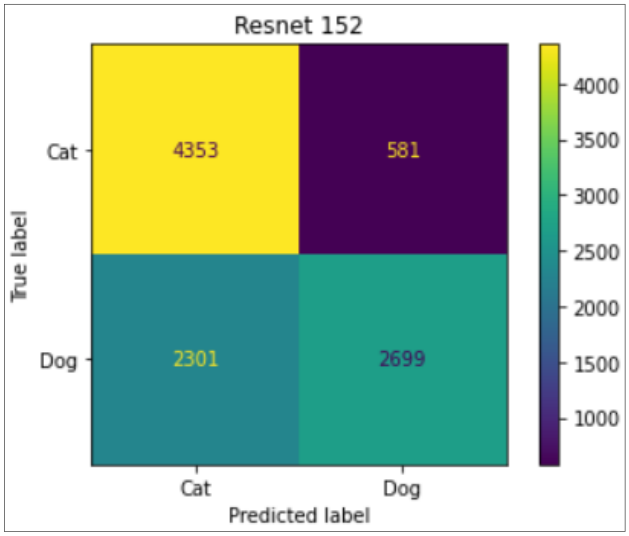

Fig. 21. Confusion Matrix of ResNet152.

And the confusion matrix (refer to figure 21) derives that 581 images were predicted dogs but were cats (refer to image 23). And 2301 were predicted cats but were dogs (refer to figure 24).

Hence we conclude that the model accuracy of ResNet152 architecture is 95.29%. (Refer to figure 22)

```
[ ] old_model152 = load_model('/content/gdrive/MyDrive/Cloud Research/model/Training Models/resnet152/dog_cat_resnet152_after_spilt.h5')

[ ] loss, acc = old_model152.evaluate(images_test, Y_test, verbose=2)
    print("Trained model, accuracy: {:5.2f}%".format(100 * acc))

    313/313 - 76s - loss: 0.1850 - accuracy: 0.9529 - 76s/epoch - 244ms/step
    Trained model, accuracy: 95.29%
```

Fig. 22. Model Accuracy for ResNet152

Fig. 23. Top 9 images for Actual Dog Predicted Cat for ResNet152.

Fig. 24. Top 9 images for Actual Cat predicted Dog for ResNet152

TABLE II. COMPARISON BETWEEN RESNET ARCHITECTURES

	Accuracy	Loss
ResNet34	82.25%	81.11%
ResNet50	91.39%	34.73%
ResNet101	94.89%	19.39%
ResNet152	95.29%	18.5%

Table 2 summarizes the experiments done in this paper. It shows that ResNet follows a trend that the higher the depth of the ResNet architecture the higher the accuracy.

The Validation Set:

For the validation set we randomly selected 5000 images from the dog's category and 5000 images from the cat's category so a total of 10,000 images were for the validating set. To maintain the similarity or to see the exact difference in all the models we have taken the same images for the training and validation of all the models.

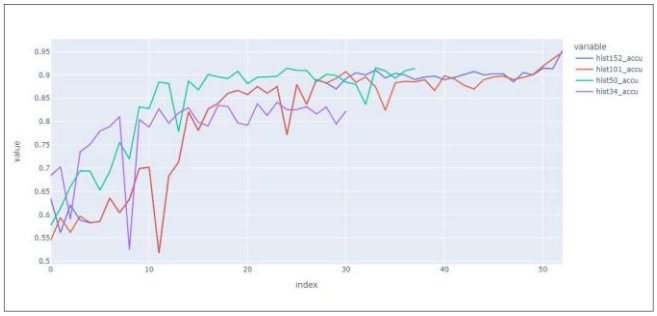

Fig. 25. Validation Accuracy comparison of Resnet 34, 50, 101, and 152

Figure 25 summarizes all the accuracy on the validation dataset for all 4 models: i) ResNet34: 82.25%, ii) ResNet50: 91.38%, iii) ResNet101: 94.89%, iv) ResNet152: 95.29%

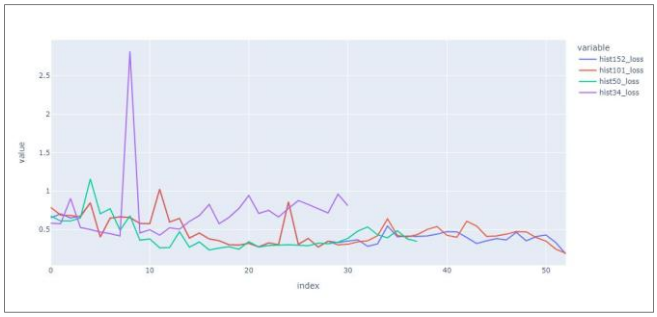

Fig. 26. 6 Validation Loss comparison of Resnet 34, 50, 101, 152.

Figure 26 summarizes all the losses on the validation dataset for all 4 models:

i) ResNet34: 81.11%,
ii) Resnet50: 34.73%,
iii) ResNet101: 19.39%,
iv) ResNet152: 18.5%

IV. CONCLUSION

In this study we focused on four ResNet architectures, starting from ResNet 34 to ResNet 152 (34, 50, 101, 152). We have compared all these ResNets and calculated their accuracy and loss and hence concluded as the ResNet layers increase we get better accuracy. The dataset used for this experiment had 2 classes cat and dog and the task was to predict provided the image whether it is a cat or a dog. It was a 25000-long dataset split in 60 – 40% as training and testing data respectively (15000 – train, 10000 – test). We started with a residual block which is identity and convolutional blocks followed by the respective architectures. Hence we conclude that ResNet34 gave an accuracy of 82.25% where 81.11% of images out of 10000 were mispredicted. In ResNet50 accuracy is 91.39% and almost 34.73% of images were detected incorrectly. Accuracy in ResNet101 is 94.89% and around 19.4% is predicted the opposite. In ResNet152 we observed an accuracy of 95.29% and 18.5% of images from the test dataset were mispredicted. (Refer to Table 2) Hence we conclude that from ResNet34 to ResNet152 the accuracy increases as the layers increase.

REFERENCES

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition”, In IEEE, 2015 https://arxiv.org/abs/1512.03385 .

[2] Pablo Ruiz, “Understanding and visualizing ResNets”, In towards data science ,2018 https://towardsdatascience.com/understanding-and-visualizing-ResNets-442284831be8

[3] Syed Abdul Gaffar Shakhadri, “Build ResNet from Scratch With Python !”, In Analytics Vidhya, 2021 https://www.analyticsvidhya.com/blog/2021/06/build-ResNet-from-scratch-with-python/

[4] ANKIT SACHAN, “Detailed Guide to Understand and Implement ResNets”, In cv-tricks website, 2019 https://cv-tricks.com/keras/understand-implement-ResNets/

[5] Jason Brownlee, “A Gentle Introduction to 1×1 Convolutions to Manage Model Complexity”, In cv-tricks website, 2019 https://machinelearningmastery.com/introduction-to-1x1-convolutions-to-reduce-the-complexity-of-convolutional-neural-networks/

[6] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In AISTATS, 2010. https://fleuret.org/dlc/materials/dlc-slides-5-5-initialization.pdf

[7] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In ICCV, 2015. https://arxiv.org/pdf/1502.01852.pdf

[8] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015. https://arxiv.org/pdf/1411.4038.pdf

[9] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012. https://dl.acm.org/doi/pdf/10.1145/3065386

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014. https://arxiv.org/pdf/1409.1556.pdf

[11] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. CoRR, abs/1409.4842, 2014. https://arxiv.org/pdf/1409.4842.pdf

[12] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In ICML, 2014. https://arxiv.org/pdf/1310.1531.pdf

[13] N. N. Schraudolph. Accelerated gradient descent by factor-centering decomposition. Technical report, 1998. https://nic.schraudolph.org/pubs/facede.pdf

[14] T. Raiko, H. Valpola, and Y. LeCun. Deep learning made easier by linear transformations in perceptrons. In AISTATS, 2012. http://proceedings.mlr.press/v22/raiko12/raiko12.pdf

[15] T. Vatanen, T. Raiko, H. Valpola, and Y. LeCun. Pushing stochastic gradient towards second-order methods–backpropagation learning with transformations in nonlinearities. In Neural Information Processing, 2013. https://arxiv.org/pdf/1301.3476v3.pdf

[16] Ross Wightman, Hugo Touvron, and Herve J'egou. ResNet strikes back: An improved training procedure in timm. In arXiv:2110.00476v1 2021 https://arxiv.org/pdf/2110.00476.pdf

[17] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep networks with stochastic depth. In European Conference on Computer Vision, 2016. https://arxiv.org/pdf/1603.09382.pdf

[18] Sabyasachi Sahoo. Residual blocks — Building blocks of ResNet. In Towards Data Science, 2018 https://towardsdatascience.com/residual-blocks-building-blocks-of-resnet-fd90ca15d6ec

[19] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, Dive into Deep Learning. At arXiv preprint arXiv:2106.11342, 2021 http://d2l.ai/chapter_convolutional-modern/resnet.html

[20] Aditya Chatterjee, Evolution of CNN Architectures: LeNet, AlexNet, ZFNet, GoogleNet, VGG and ResNet. In OpenGenus 2015 https://iq.opengenus.org/evolution-of-cnn-architectures/

[21] B. Li and Y. He, "An Improved ResNet Based on the Adjustable Shortcut Connections," in IEEE Access, vol. 6, pp. 18967-18974, 2018, doi: 10.1109/ACCESS.2018.2814605. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8315010

[22] G. Montufar, R. Pascanu, K. Cho, and Y. Bengio. On the number of linear regions of deep neural networks. In NIPS, 2014. https://arxiv.org/pdf/1402.1869.pdf