Image Splicing Detection Using Neural Network

1st Diellza Malazogu
Department of Computer Science
Michigan Technological University
Houghton, MI, USA
dmalazog@mtu.edu

2nd Jaffarus Sodiq
D*epartment of Computer Science*
Michigan Technological University
Houghton, MI, USA
jsodiq@mtu.edu

3rd Rishi Babu
Department of Physics
Michigan Technological University
Houghton, MI, USA
rbabu@mtu.edu

Abstract— With a digitalized world where a lot of information is being spread using the internet the risk of forging images and videos is a hazardous one regarding producing fake information for the general public. Through this project we tried to solve this problem by training three different Convolutional neural networks (CNN) architectures to detect image forgery, more specifically the Image Splicing category of image forgery. The dataset used in this project is a public Columbia Splicing Detection Evaluation Data Set with a fixed size of 128 pixels x 128 pixels including about the same number of authentic and spliced image blocks. In the project we gave a brief description about the architectures that we used; VGG-16, VGG-19 and ResNet50. The models were trained and tested based on the model's accuracy. In this paper we further presented the experimental results for each model, the visualized accuracy for each model across different epochs as well as a comparison between the different models and their results. Our results showed that overall the three models were pretty similar in their performance but VGG-16 overall performed better with a higher accuracy at 71% and loss value of 0.6. Moreover, in this paper it is also presented a summary of related work tackling this image forgery problem.

Key words : CNN, VGG-16, VGG-19, ResNet50, Image forgery, Image Splicing, Detection, Accuracy.

I. Background and Motivation

Digital age has revolutionized the world with the internet becoming one of the main sources of information and means for socializing. Moreover, this digital revolutionization also enabled people to share images from their daily lives to the world. However, these developments also enabled people to fake images and videos therefore creating fake information that can be distributed in internet leading to misinforming the general public.

Throughout the years tools such as photoshop and illustrator have given an ease of access to image editing and tampering. Through the use of elaborate background, feature insertion and geometry modifications of images the forged image can be of unprecedented realism and impossible to be detected by human eyes thus the internet is being flooded with tampered images with increasing frequency. These forged images have been used to create fake news, political agendas and financial conspiracies among others. Therefore to be clear about the truth of these images and stop the general public misinformation, designing a forgery detection tool is of the utmost importance in the current era.

Image Forgery can be largely divided into two subcategories: Copy Move Forgery and Image Splicing. The category that was tackled during this project is the Image Splicing category of Image Forgery.

Image splicing is a technique of cropping an image or part of it and pasting it to another image with the view of deceiving the viewer to fake news. This small image tampering can be traced through the object geometries and statistical methods.

The geometry based approach is looking at the aberrations in the lighting, motion blur, object geometry and background of the image through certain filters and tools. One specific example of this is to look at the image and determine the boundaries of the background and the targets.

The statistics approach to this is to look at the aberrations in the signal or the pixel of the image. Like detecting a sensor pattern noise, a compression artifact which needs some prior knowledge of the images. For example multiple photos taken from a single camera, is able to give a parameter allied the photo-response non-uniformity noise called the PRNU which is unique for a specific camera lens. PRNU can be used to detect any compression artifacts or other aberrations in the images. Even the presence of a colour filter array (CFA) demosaicing artifacts which is a mosaic of tiny colour filters over the pixels of an image gives evidence of image forgery. Also more information can be withdrawn from the history of the image formats and its manipulation. For an image saved in a compressed JPEG format, the manipulation of a certain portion of the image causes the discrete cosine transform (DCT) coefficients to change indicating image forgery.

Due to the above mentioned reasons, Image Splicing is harder to detect than Copy Move Forgery. Copy-Move Forgery is easy to detect due to the difference in outlines of the objects in the image, their textures, sizes among other things.

Over the past couple of years Deep neural networks such as Convolutional Neural Network(CNN), Recurrent neural network(RNN) and Generative Neural network (GAN) have been developed and widely used to tackle such problems. These networks are shown to predict high characteristic statistics from higher dimensional inputs and train the model for learning their representations at a higher rate. This allows the networks to be used in various fields including computer vision, artificial intelligence, predicting weather models and high performance simulations of real world problems. It is worth noting that CNN was also widely used during the COVID-19 pandemic which allowed scientists to predict the rate of spread of infections, the future trends in the infection rate, the efficiency of the vaccine and herd immunization cycles. To conclude among others this also inspired us and paved the way for using deep learning based networks for detecting image tampering which works quite well on the Columbia Image splicing detection dataset.

In order to detect Image Splicing of Image Forgery we trained three deep-learning algorithms in Columbia Splicing Detection Evaluation Data Set and reported their performances. More concretely we trained VGG-16, VGG-19 and ResNet50 architectures on the data set and obtained the accuracy of each in order to see which architectures can perform better. The three models were trained using Adam optimizer and binary cross-entropy to give a penalty to the model for each iteration.

After training the three models we compared the experimental results about which we will talk more in depth later-on and then came up with the model which we thought performed best to detect Image Splicing.

II. RELATED WORK

A. *Robust forgery detection for compressed images using CNN supervision (Baubacar Diallo etc, 2020)*

In this journal the author relied on a camera identification model based on CNN. Lossy compression such as JPEG is the most common type of intentional concealment of image forgery. The author trained CNN using different qualities of compressed and uncompressed images. For better interpretation of the CNN, the author proposed an in-depth supervision by first a visualization of the layer and an experimental analysis of the influence of the learned features. This method resulting more robust and accurate framework. [2]

B. *Image Level Forgery Identification and Pixel Level Forgery Localization via a Convolutional Neural Network (Haitao & Qiu, 2018)*

The author reports on a light-weight (5k parameters) using VGG architecture that allows for image forgery detection. VGG can perform well with an accuracy of 91% and AUC of 85% on test data. The author also modified network with a local anomaly detection network. This method achieved accuracy of 92% and AUC of 80% on test data. The author's result demonstrates a relatively light-weight detection network that is easy to train and adopt for image level forgery identification and pixel level localized area detection.[3].

C. *Image Forgery Detection Based on convolutional neural network (Guorui and Jian. 2020)*

This paper describes the use of Style Based Recalibration Module (SRM) filters for neural network and a high-pass filter to achieve image pre-processing for the Columbia dataset. The author reports on using this pre-processed data to train a five layer convolutional neural network and the robustness of the neural network on image forgery detection and classification. The reported accuracy of the model with 5 convolutional layers os 91.8% with average pooling and 89.57% accuracy with maximum pooling on the modified test data with less detection time. The model reported is robust in all cases and the one main drawback is its post-processing operations.[9]

III. DATASETS AND METODELOGY

In this project we be used the Columbia Splicing Detection Evaluation Data Set which are open for general public, provided from Columbia University. This data consists of two types of images; original image and image with photomontage (Spliced Image). The data set consists of two main categories Authentic category (Au) 933 Block and Spliced category (Sp) 912 image blocks. For both Au and Sp, there are five subcategories (with another three subcategories for TS, TT, and SS). The images are with a fixed size of 128 pixels x 128 pixels and the image blocks are extracted from images in the CalPhotos collection. Table 1 show the detail of data set.

For the authentic category the image blocks are cropped directly from the images where the images captured by the camera without any manipulations. For the spliced category, the images are spliced by object-based image splicing, like cropping an object and pasting it to another image without any post-processing. Then the cropped image blocks which contain part of the splicing boundary from the composite image as image blocks. The cut and paste of image objects are using Adobe Photoshop 6.0 [1].

This project split the data into two parts, the Training data set and the Testing data set (75% and 25%). We compared three different Neural Network frameworks to see which provided the best accuracy. We are using an Adam optimizer with a learning rate of 0.01 to optimize the neural network and the binary cross-entropy function as a loss function.

TABLE I. DATASET CATEGORY

Category	One Textured Background (T)	One Smooth Background (S)	Textured-Smooth Interface (TS)	Textured-texture Interface (TT)	Smooths-smooth Interface (SS)	TOTAL
Authentic (Au)	126	54	409	179	165	933
Spliced (Sp)	126	54	298	287	147	912

IV. OVERVIEW ARCHITECTURE

A. VGG-16

VGG-16 is a deep learning convolutional neural network model developed by K. Simonyan and A. Zisserman in their paper [10]. This model trained on the ImageNet dataset (which is a dataset of more than 14 million images with more than 1000 different labels), for the ILSVRC-2014 competition, achieved an accuracy over 92 percent. It is an upgrade over AlexNet which implements multiple 3x3 kernel-size filters over the large scale kernel-size filters of AlexNet

For the input of VGG-16, it maintains a strict (224, 224, 3) input image dimension. This input is passed through several convolutional layers each of which has a filter of 3x3 to capture almost all the directional configurations in each image and some with 1x1 filter which acts as a linear transformation of the input. The first and second layers consist of 64 channels of 3x3 filters with the same padding followed by a max pool layer with stride 2 which reduces the dimensions of the image to 112x112x64. This is followed by two convolutional layers of 128 size with another max pooling layer which reduces the image to 56x56x128. More

stacks of convolutional layers with 256 and 512 sizes with more pooling layers are used and a final max pooling is done over a 2x2 pixel size with stride 2.

The stack of convolutional layers is followed by three Fully Connected layers, where the first and second layer have 4096 channels and the third layer integrates a 1000-way ILSVRC classification with 1000 channels [ref]. The final activation layer used in the network is a soft-max layer.

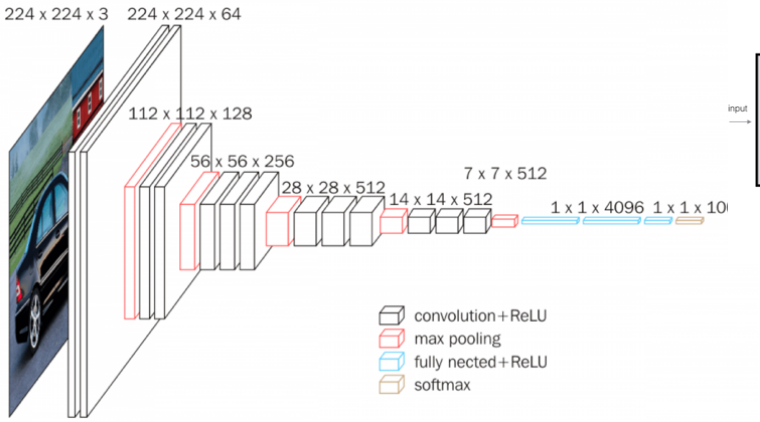

Figure 1. The Architecture of VGG-16

B. VGG-19

We implement this model as a convolutional neural network and evaluate it on the forgery detection dataset. The initial convolutional layers of the network extract feature from the image while the fully connected layers predict the output probabilities. One of the network architectures that we use is VGG-19. VGG-19 is a convolutional network with 19 layers. VGG stands for the visual geometry group invited to oxford university for the ImageNet large-scale visual competition in 2014. VGG using 224x224 pixel as an input. VGG-19 use two layers 3x3 filters instant a large filter, Besides the 3x3 filter, VGG also provide a 1x1 convolutional filter which acts like linear transformation follow by ReLu unit [5]. We add a final function using SoftMax activation to get the probability of original and spliced image.

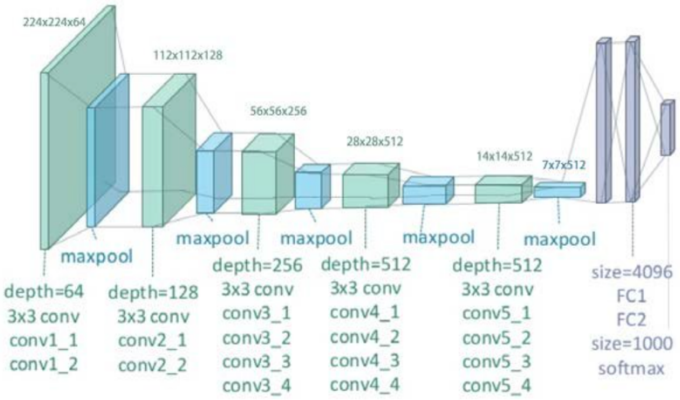

Figure 2. The Architecture of VGG-19

C. ResNet50

A residual neural network(hereinafter ResNet) is an artificial neural network. This network builds on constructs known from pyramidal cells in the cerebral cortex. It is worth mentioning that this model was the winner of ImageNet challenge in 2015. What distinguishes ResNet is that it allows us to train extremely deep neural networks with 150+layers.

The ResNet-50 model consists of 5 stages each with a convolution and Identity block. Each convolution block has 3 convolution layers, and each identity block also has 3 convolution layers. The ResNet-50 has over 23 million trainable parameters. The architecture that we used for our project is as shown below.

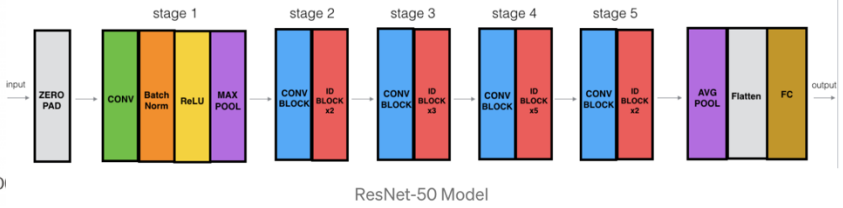

Figure 3. The Architecture of ResNet50

V. Preprocessing Procedure

As regard data pre-processing firstly we split the data into training and test set with total 1624 images as training and 221 images as testing respectively. Furthermore, taking into consideration that the three models that we used VGG16, VGG19 and ResNet50 as an input take the image size 224x224 and considering that our original data have an image size of 128x128 we resized the size of these images into 224x224 in order to be in line with the input size required by these three models.

Moreover, during training of ResNet50 model we noted an overfitting so in order to avoid overfitting we further pre-processed the data for ResNet50, more precisely the images were converted from RGB to BGR, then each color channel was zero-centered.

VI. Machine Learning and Algorithm

In this project we are using the Neural Network approach to classify image forgery. According to Keijsers, Neural networks are mathematical models that use learning algorithms inspire by the brain to store information [5].

Neural networks consist of three different parts: Input Layer, Hidden Layer, and Output Layer. Each layer correspondent to the other and build a network with an activation function.

Convolution Neural Network (CNN) has deep feed-forward architecture and has astonishing ability to generalize in a better way as compared to networks with fully connected layers [6]. An individual neuron in the next layer is connected to some neurons in the previous layer, this local correlation is termed as receptive field. The convolution layer is followed by pooling that will reduces number of trainable parameters and introduces translation invariance [7]. The last part that uses in CNN is the activation function. In this project, we mostly use the linear activation function ReLu

$$\phi(x) = \max(0, x) \qquad [1]$$

We optimize for binary cross entropy in the output of our model. We use binary cross entropy because it is easy to optimize and we using a binary classification as out final output. The loss function optimize using Adam optimizer. The Adam optimization algorithm is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications in computer vision and natural language processing.

Adam optimization is popular on deep learning due to the good result during many deep learning algorithm. In 2015 Diedreik and Jimmy conducting an experiment to see how well Adam optimization in two machine learning algorithm. The experiments that Adam to be robust and well-suited to a wide range of non-convex optimization problems in the field machine learning [8].

The algorithm that we use to conduct the neural network in TensorFlow. TensorFlow is an open-source library that provides a tool to build and train machine learning. Tensorflow itself has a large CNN architecture to implement in data set with a robust implementation.

VII. Experimental Result

After training the three different architectures and obtaining their accuracies as well, we obtained the following results presented below for each architecture separately.

A. VGG-16

VGG - 16 ran for 50 epochs and was trained on the Columbia dataset with 138,359,546 parameters. The model achieved a 70 percent accuracy on the testing dataset during the evaluation process.

The model was optimized with the Adam optimizer with a learning rate of 0.01. The model accuracy increased at a steady rate till the runtime was closed.

The accuracy per epoch rate shows a rapid increase in the initial stages with a steady but slow increase in the accuracy in the later stages. Given more resources, it can show a significant improvement in the accuracy.

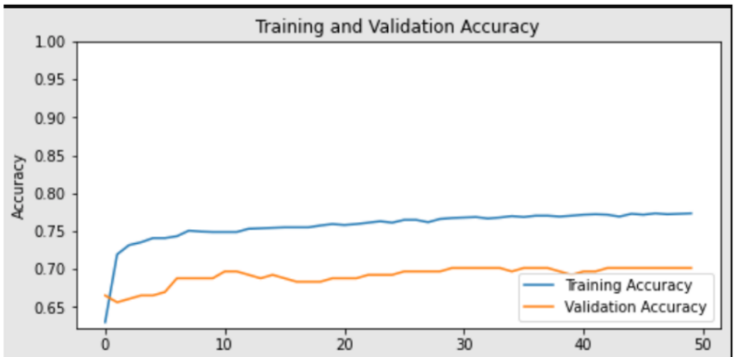

Figure 4. Accuracy plot per epoch of VGG-16

B. VGG-19

The second experiment that we did for the data set is using VGG-19 architecture. VGG-19 has more layers compared to the previous version VGG-16.

In this experiment we run the VGG-19 in 50 epochs to see how well the architecture predicts the authentic and spliced images.VGG-19 consists of 143,669,242 parameters with 19 layers. Figure 3 shown the accuracy plot for each iteration in VGG-19.

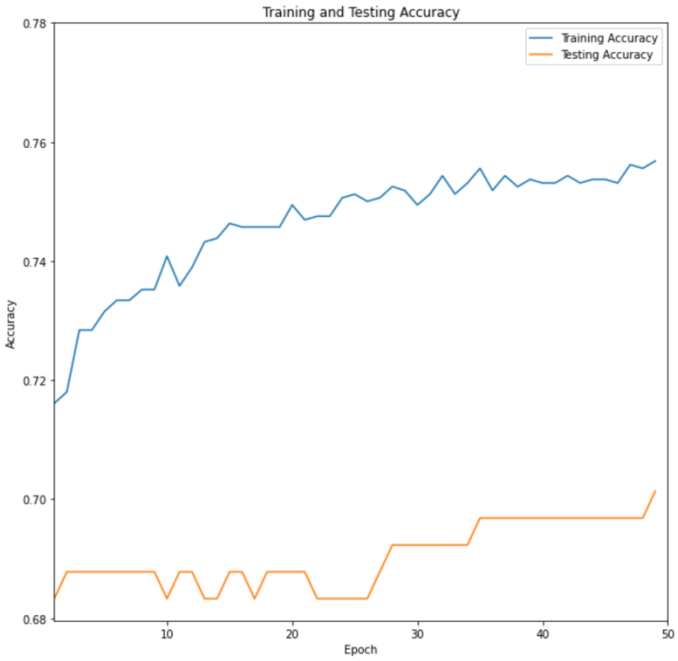

Figure 5. Accuracy plot per epoch of VGG-19

From Figure 5 we can see for each epoch that the accuracy in testing data increases slowly following the training data's accuracy. Approximately it takes 15 hours to run 50 epoch. T

The final model can achieve 75% accuracy on training data and 70% accuracy in testing data. It is worth mentioning that VGG-19 is slow learner which can also be shown by the fact that the decrease of loss value after each epoch is relatively small.

The testing model comes to be more stable on a certain period while at the same time the training accuracy fluctuates more in the entire epoch. The accuracy graph shows that the result is not optimum yet because the accuracy keeps increasing for each epoch. Due to the limitation of GPU, we keep the iteration into 50 epochs.

C. ResNet50

Finally, the third experiment that we conducted for the data set is using ResNet50 architecture. ResNet is way deeper than VGG16 and VGG19 but the model size is actually substantially smaller due to the usage of global average pooling rather than fully connected layers, thus this reduces the model size down to 102MB for ResNet50.

In this experiment we run the ResNet50 in 50 epochs to see how well the architecture predicts the authentic and spliced image. ResNet50 consists of 23,591,810 total parameters with 50 layers. Figure 4 showed the accuracy plot for each iteration in ResNet50.

Figure 6. Accuracy plot per epoch of ResNet50

D. Model Comparison

TABLE II. TESTING RESULT

Architecture	Accuracy	Loss
VGG-16	71%	0.6160
VGG-19	70%	0.5943
ResNet-50	65%	0.6640

Table 2 shows the comparison of accuracy and loss for each architecture. We can see both VGG-16 and VGG-19 show similar accuracy results in testing data and can perform very well compare with ResNet-50. However, VGG-19 takes longer to train compared to the other two architecture.

VGG-19 can take 15 hours to train the architecture in the Columbia data set; meanwhile, VGG-16 took 8 hours and ResNet took only about 50 minutes to train the model. This time consumption makes VGG-19 not suitable for this data set because the accuracy result is less than VGG-16 and the loss value is relatively similar (the difference is insignificant).

From Table 2 we can see that the best architecture for the Columbia data set is VGG-16 that has an accuracy of 71% and a slight loss value in the testing result. However, it took approximately 8 hours to rain the model that requires a high GPU in the process.

ResNet, on the other hand, has the smallest accuracy compared to the other two Architectures. The advantage of this architecture is that ResNet can perform faster to train the model. ResNet took only about 50 minutes to train the model for 50 epochs.

VIII. CONCLUSION

In this report we tested three different models with different architecture and proposed an architecture that will be more suitable for detecting image splicing based on the Columbia Image Splicing Detection Dataset.

The experimental results obtained from training different architectures show approximately similar results in accuracy and loss. Cross entropy loss is adopted to improve the generalization ability of each of the models. The significant difference is the time to train the model. VGG models take more time to train the model and are much slower when compared with ResNet50.

The final results as seen from Table 1 show that VGG-16 can perform well when compared with other architectures. VGG-16 can classify the authentic and spliced image with 71% accuracy. VGG-16 train time takes 8 hours to complete 50 epochs. At the same time, ResNet50 is less accurate than other architectures but can perform faster in training the model for the same given iterations. Comparing the trade-off between accuracy, training time and resource usage, VGG-16 performs fairly well compared to VGG-19 and ResNet50.As a result, for this dataset we recommend using VGG-16 to predict the splicing image due to the better accuracy and training time that is still acceptable. Further investigation is required to check the validity of these models with more training dataset and better computing resources.

IX. LIMITATION

The models took a lot of time and computing resources to converge. We performed all the calculations on the Google Colab platform to take advantage of their GPU computing abilities. The main limitation we encountered was the runtime on the Colab notebook. Google Colab provides a maximum of 12 hours of runtime each with their graphical processing unit(GPU) and Tensor Processing Unit(TPU). The GPU allocated was a single 12GB NVIDIA Tesla K80 GPU that can be used up to 12 hours continuously. This GPU resource was used entirely by the models when testing for higher epochs (above 60). Google's policy to restrict the usage of backend GPU for computing temporarily disabled one of the members accounts for 2 days.

As a result, it disabled our investigation of running our models for a longer time and epochs. Since these models have been shown to have a higher accuracy in the original research papers, where they used multiple GPUs for weeks to train the model, we are certain our models will achieve higher accuracy provided we get better and faster computing resources.

X. TEAM CONTRIBUTION

Initially the three members together came up with the idea of choosing the topic of the project after doing research and deciding which topic is more in line with the course objectives. Afterwards each member took the responsibility of training and working in one of the models:

1. VGG-16: Rishi Babu

   Rishi wrote the VGG-16 code and trained the same on Google Colab. The model was tested for initial 30 epochs, 50 epochs and 100 epochs when the computing resources were restricted.

2. VGG-19: Jaffarus Sodiq

   Jaffarus wrote the VGG-19 code and trained the same on Google Colab. This model was tested for

initial 30 epochs and 50 epochs and with 2 different learning rates of 0.01 and 0.001

3. ResNet50: Diellza Malazogu

   Diellza wrote the ResNet50 code and trained the same on Google Colab. The model was tested for initial 30 epochs, 50 epochs and 100 epochs.

After the initial training for 100 epoch failed and the computing resources were restricted to one of the team members, the group decided to train the models for 50 epochs each keeping in mind the restriction of the available resources. After training the models each member of the group contributed into writing their experimental results, combining those results with the rest of the team's results and then overall contributing into writing the final project report and presentation. They divided the project outline among each other and wrote different parts of it as well as presented their parts in the project presentation.

REFERENCES

[1] Tsong,T. & Chang,S. (2004). A Data set of authentic and spliced image block. Columbia University. Retrieved from https://www.ee.columbia.edu/

[2] Diallo,B. etc. (2020). Robust forgery detection for compressed images using CNN supervision. Science Direct. Retrieved from https://www.sciencedirect.com/science

[3] Deng,H. & Qiu,Y. (2019). Image Level Forgery Identification and Pixel Level Forgery Localization via a Convolutional Neural Network. Standford University. Retrieved from http://cs230.stanford.edu/projects_fall_2019/reports/26260910.pdf

[4] Wei, J (2019). VGG Neural Netwirks : The Next Step of After AlexNet. Towards Data Science. Retrieved from https://towardsdatascience.com/vgg-neural-networks-the-next-step-after-alexnet-3f91fa9ffe2c

[5] Keijers,N.W.L (2010). Encyclopedia of the movement disorders. Science Direcet. Retrived from https://www.sciencedirect.com/topics/neuroscience/neural-networks/pdf

[6] Nebauer, C. (1998) Evaluation of convolutional neural networks for visual recognition. IEEE Transactions on Neural Networks 9 (4): 685- 696.

[7] Indolia,S etc. (2018). Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach. Science Direct. Retrieved from https://www.sciencedirect.com/science/article/pii/S1877050918308019

[8] Kingma, D & Ba, J. (2015). Adam : A Method for Stochastic Optimization. Cornell University. Retrieved from https://arxiv.org/abs/1412.6980

[9] F. Guorui, Wu Jian, Image Forgery Detection Based on the Convolutional Neural Network. Retrieved from https://dl.acm.org/doi/pdf/10.1145/3383972.3384023

[10] Karen Simonyan and Andrew Zisserman(2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556