Indonesian Journal of Electrical Engineering and Computer Science
Vol. 12, No. 2, November 2018, pp. 455~460
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v12.i2.pp455-460

Deep Learning for Roman Handwritten Character Recognition

Muhaafidz Md Saufi, Mohd Afiq Zamanhuri, Norasiah Mohammad and Zaidah Ibrahim
Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Shah Alam, Selangor, Malaysia

Article Info

Article history:

Received May 9, 2018
Revised Jul 12, 2018
Accepted Jul 30, 2018

Keywords:

AlexNet
CNN
Deep Learning
GoogLeNet
Handwritten Digit Recognition

ABSTRACT

The advantage of deep learning is that the analysis and learning of massive amounts of unsupervised data make it a beneficial tool for Big Data analysis. Convolution Neural Network (CNN) is a deep learning method that can be used to classify image, cluster them by similarity, and perform image recognition in the scene. This paper conducts a comparative study between three deep learning models, which are simple-CNN, AlexNet and GoogLeNet for Roman handwritten character recognition using Chars74K dataset. The produced results indicate that GooleNet achieves the best accuracy but it requires a longer time to achieve such result while AlexNet produces less accurate result but at a faster rate.

Corresponding Author:

Muhaafidz Md Saufi,
Faculty of Computer and Mathematical Sciences,
Universiti Teknologi MARA,
Shah Alam, Selangor, Malaysia.
Email: muhaafidz4ever@gmail.com

1. INTRODUCTION

One of the most widely used applications of Artificial Neural Network is Character Recognition [1-2]. Deep learning is one the most powerful machine learning methodology for solving critical problems of image processing, computer vision, natural language processing and signal processing. A key advantage of deep learning is that the analysis and learning of massive amounts of unsupervised data make it a beneficial tool for Big Data analysis [3]. Thus, deep learning often produces good results. Nevertheless, we must say that deep learning approaches require high computing resources compared to more traditional machine learning approaches. Indeed it necessitates a powerful GPU and a big database [4].

Over the past few years, deep learning on remote sensing imagery shows exceptionally good results on both optical (hyperspectral and multispectral imagery) and radar images and in mapping land areas. Scene classification of high resolution remote sensing image plays a major role for a various types of application, for instance land-use determination, natural hazard detection, network intrusion detection [5], geospatial object detection, and urban planning [6]. Recently, deep learning method especially convolutional neural network have shown their much stronger feature representation power in computer vision.

Convolution Neural Network (CNN) is a deep artificial neural network that is used mainly to classify image, cluster them by similarity, and perform image recognition in the scene. CNN has shown impressive performances in artificial intelligence tasks such as object recognition and natural language processing [4]. The general structure of a CNN consists of layers composed of neurons. A neuron takes input values, does computations and passes the result to the next layer. Besides image recognition, CNN also can be used in video analysis and also natural language processing [7].

For cases where the available data is not much, pre-trained CNN models such as AlexNet and GoogLeNet can be used. Both attain a low error when trained over the million of images contained in ImageNet [8]. GoogLeNet and AlexNet are often used in photo classification, as a large fraction of examples in ImageNet that are composed of photos. The use of pre-trained CNN model is to transfer the learning parameters. Rather than constructing and training a new network, we can take a pretrained network and uses it as a starting point to learn a new task and do fine tuning. Pre-trained CNN models have already learned a large set of features. Thus, the classification and recognition process can be achieved faster.

In this paper, a comparison between CNN and two of the popular pre-trained CNN models are conducted in terms of accuracy performance. These two models are GoogleNet and AlexNet. The objective of this comparison is to know which models are best suited to recognize the data.

2. THE PERFORMANCE OVERVIEW OF CNN MODELS USING HANDWRITTEN CAPITAL LETTER DATASET

The dataset needs to adhere with the constraints given by the CNN models. One of the constraint is regarding the size of the individual image. AlexNet needs to use a precise image size for training. Besides the constraints from the models, the hardware used also plays an important part on the performance. So, to test these 3 CNN models, a high performance gaming laptop is used to conduct the test. The laptop needs to have a dedicated NVIDIA Graphics Processing Unit (GPU) with a minimum of 8 gigabyte of Random Access Memory (RAM). The complete specification of the laptop used is listed in Table 1.

Table 1. Laptop Specification

Model	MSI GL62-6QF
Operating System	Windows 10
Processor	Intel Core i7-6700HQ
RAM	8 gigabytes
Graphics Card	Nvidia GTX 960M

The dataset used for training on these three methods is Chars74K dataset [9]. The dataset consists of the images of handwritten capital letters, small letters, numbers and symbols. Each of the letter consists of 55 variations of identical images. Since the overall training takes a long time and the hardware used for the training is still not powerful enough, the training only covers a handwritten capital letters only, which are letters A to Z, thus, this makes the total number of images used in the experiment is 1430. The size of the images in the dataset is 1200 x 900 pixels. Reducing the resolution may decrease the validation accuracy.

2.1. Simple-CNN

A CNN consists of one or more convolutional layers, usually along with a subsampling layer, which are followed by one or more fully connected layers similar to a standard neural network. These complex architectures are built for classification problems by stacking multiple and different layers. Generally, there are four types of layers, which are convolution layers, pooling/subsampling layers, non-linear layers, and fully connected layers [10]. The first layer, convolutional layers are able to extract the local features because they restrict the receptive fields of the hidden layers to be local. Then, the pooling/subsampling layer reduces the resolution of the features. This process makes the features robust against noise and distortion. Neural networks generally and CNN specifically rely on a third layer, non-linear "trigger" function to signal distinct identification of identically features on each hidden layer. CNN may use a variety of specific functions such as rectified linear units (ReLUs) and continuous trigger (non-linear) functions to efficiently implement this non-linear triggering. The final layer, Fully connected layers, total up a weighting of the previous layer of features, indicating the precise mix of "ingredients" to determine a specific target output result. Figure 1 illustrates the simple-CNN architecture.

For this CNN model, the images are resized to a smaller size, 93x70 pixels to be precise. This is because by using the original resolution is impossible due to the limitation of Graphics Processing Unit (GPU) memory of the device used for this training. The training is set to 10 epoch with 10 iterations for every epoch. Based on the result, the training has reach the final iteration at Epoch 10 with 0.8077 of validation accuracy. The accuracy already near the maximum during the Epoch 9. Figure 2 shows the detail results of the training and validation process. Figure 3 listed some sample images that have been correctly classified.

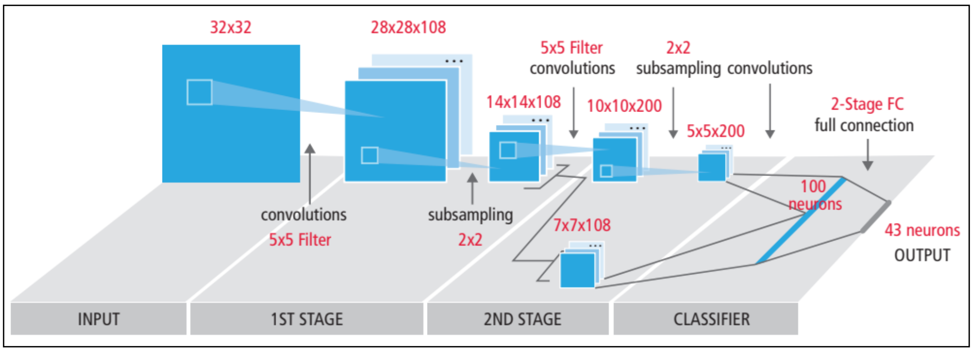
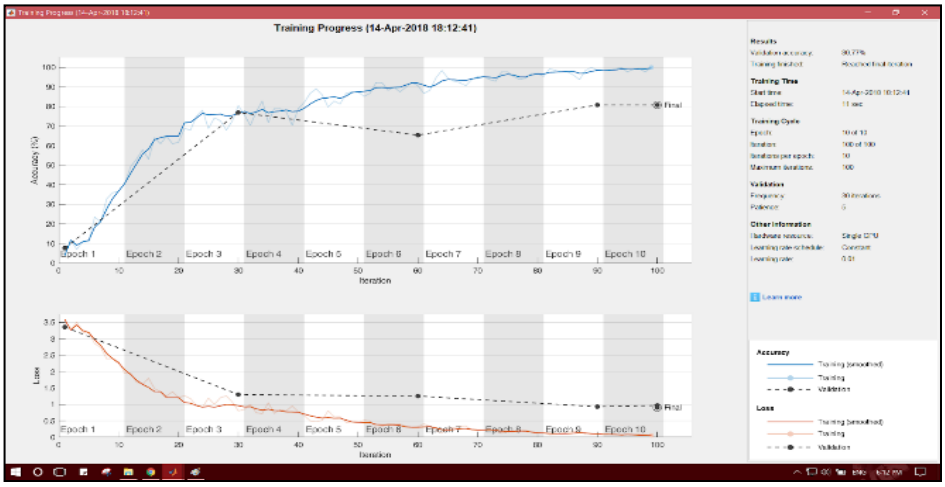


Figure 1. Simple-CNN architecture [10]



Results
Validation accuracy: 90.77%
Training finished: Reached final iteration

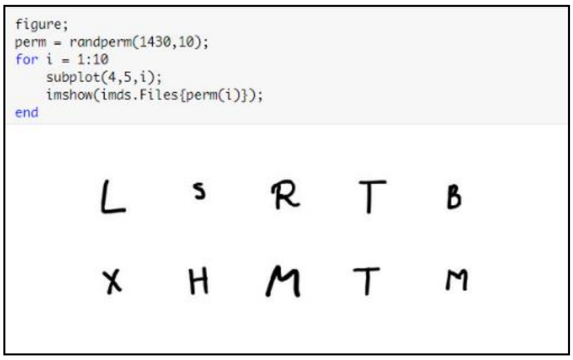
Training Time
Start time: 14-Apr-2018 18:12:41
Elapsed time: 11 sec

Training Cycle
Epoch: 10 of 10
Iteration: 100 of 100
Iterations per epoch: 10
Maximum iterations: 100

Validation
Frequency: 30 iterations
Patience: 5

Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.01

Figure 2. Simple-CNN training and validation details



```
figure;
perm = randperm(1430,10);
for i = 1:10
    subplot(4,5,i);
    imshow(imds.Files{perm(i)});
end
```

Figure 3. Simple-CNN test results

2.2. AlexNet

AlexNet is made up of 5 convolutional layers, max-pooling layers, dropout layers and 3 fully connected layers. Relu is applied after every convolutional and fully connected layer. Dropout is applied before the first and the second fully connected layer [11]. Figure 4 illustrates the architecture of AlexNet. For this training, the images size are back to its default size. The training is set to 5 epoch with 101 iterations per epoch. The results show that the training has reached the final iteration with 0.9423 validation accuracy. As the iteration process on AlexNet is tenfold than Simple-CNN, the training accuracy is already hit 0.90 in the middle of the second epoch. The results of the training can be seen in Figure 5 while Figure 6 listed some sample images that have been correctly classified by AlexNet.

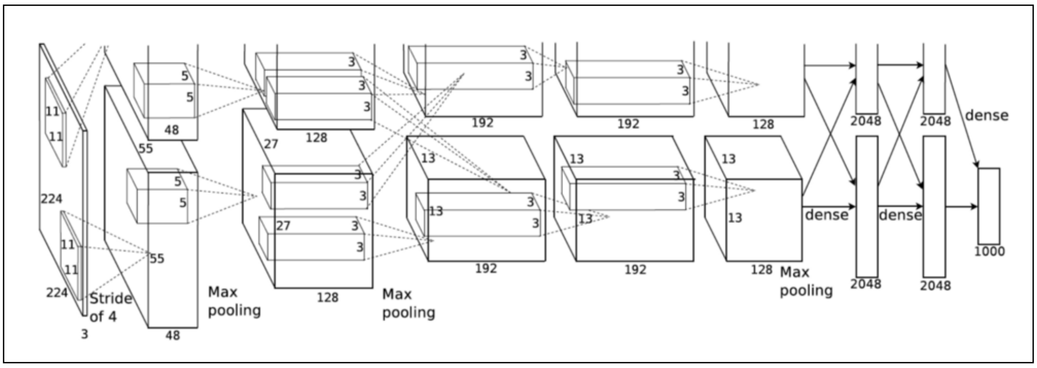
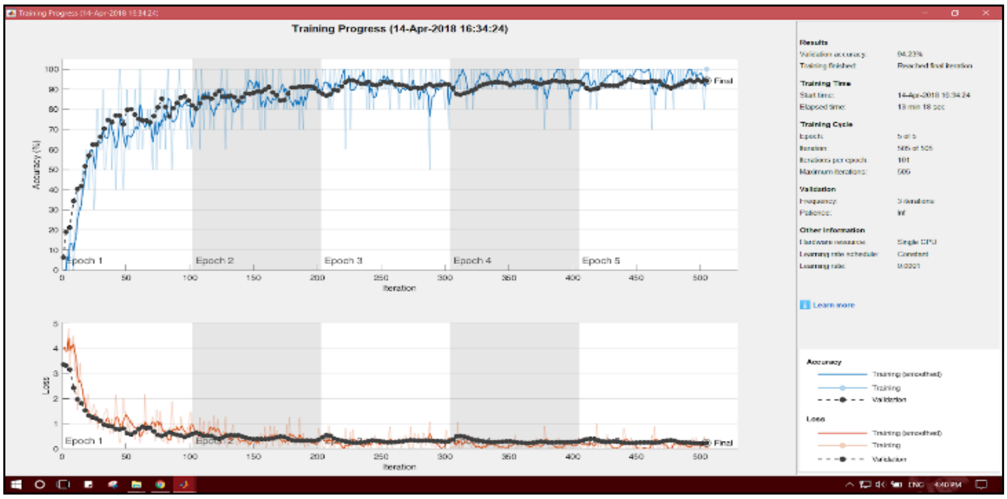


Figure 4. AlexNet architecture [11]



Results
Validation accuracy: 94.23%
Training finished: Reached final iteration

Training Time
Start time: 14-Apr-2018 16:34:24
Elapsed time: 13 min 18 sec

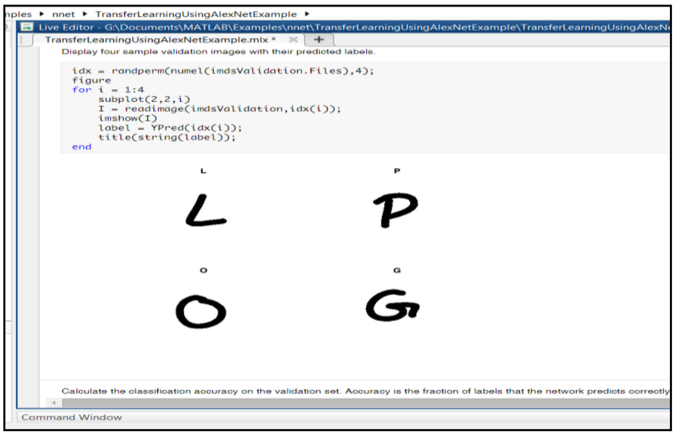
Training Cycle
Epoch: 5 of 5
Iteration: 505 of 505
Iterations per epoch: 101
Maximum iterations: 505

Validation
Frequency: 3 iterations
Patience: Inf

Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.0001

Learn more

Figure 5. AlexNet training and validation details


nples ▸ nnet ▸ TransferLearningUsingAlexNetExample ▸
Live Editor - G:\Documents\MATLAB\Examples\nnet\TransferLearningUsingAlexNetExample\TransferLearningUsingAlexN
TransferLearningUsingAlexNetExample.mlx *
+
L
P
O
G
Command Window

Display four sample validation images with their predicted labels.

```
idx = randperm(numel(imdsValidation.Files),4);
figure
for i = 1:4
    subplot(2,2,i)
    I = readimage(imdsValidation,idx(i));
    imshow(I)
    label = YPred(idx(i));
    title(string(label));
end
```

Calculate the classification accuracy on the validation set. Accuracy is the fraction of labels that the network predicts correctly

Figure 6. Sample AlexNet test results

2.3. GoogLeNet

GoogLeNet is a pre-trained CNN that has been trained on over a million images and can classify images into categories. The network takes an image as input and outputs a label for the object in the image together with the probabilities for each of the object categories. It has 22 layers. This architecture (Figure 7) is out of the norm, it veered off from the general approach of simply stacking conv and pooling layers on top of each other in sequence [12]. For GoogLeNet CNN model, the images size is also using their default size. GoogLeNet training and validation details shiwn in Figure 8. The results of GoogLeNet are shown in Figure 9 along with the training percentages of each letter. The training is set to 6 epochs with 101 iterations per epoch. The results show that the training has reached the final iteration with 0.9447 of validation accuracy.

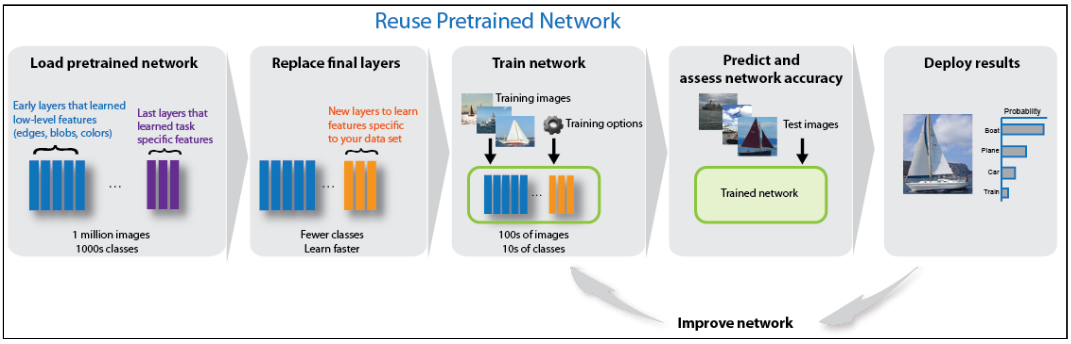
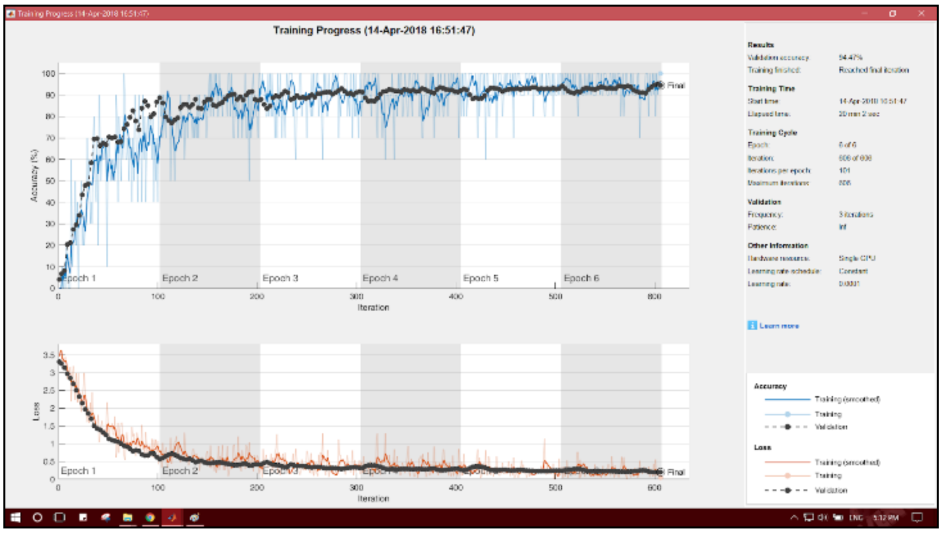


Figure 7. GoogLeNet architecture [12]



Results
Validation accuracy: 94.47%
Training finished: Reached final iteration

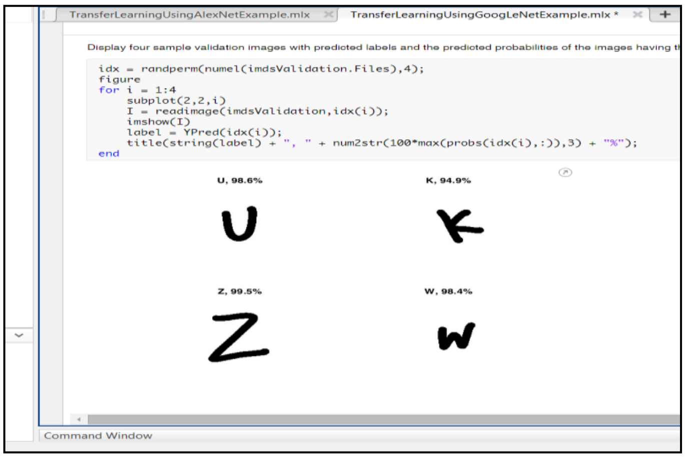
Training Time
Start time: 14-Apr-2018 16:51:47
Elapsed time: 20 min 2 sec

Training Cycle
Epoch: 6 of 6
Iteration: 606 of 606
Iterations per epoch: 101
Maximum iterations: 606

Validation
Frequency: 3 iterations
Patience: Inf

Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.0001

Figure 8. GoogLeNet training and validation details


TransferLearningUsingAlexNetExample.mlx
TransferLearningUsingGoogLeNetExample.mlx *
U, 98.6%
K, 94.9%
Z, 99.5%
W, 98.4%
Command Window

Display four sample validation images with predicted labels and the predicted probabilities of the images having t

```
idx = randperm(numel(imdsValidation.Files),4);
figure
for i = 1:4
    subplot(2,2,i)
    I = readimage(imdsValidation,idx(i));
    imshow(I)
    label = YPred(idx(i));
    title(string(label) + ", " + num2str(100*max(probs(idx(i),:)),3) + "%");
end
```

Figure 9. Sample GoogLeNet test results

3. CONCLUSION

Table 2 shows an overview of the accuracy performance of the three models based on Chars74K dataset. By looking at Table 2, we can see that GoogLeNet is better than simple-CNN and AlexNet but it took a longer time to achieve this result.

Table 2. The Performance Overview on CNN Models Using Capital Letters from Chars74K Dataset.

CNN Model	Simple-CNN	GoogLeNet	AlexNet
Validation Accuracy	0.8077	0.9447	0.9423
Training Finished	Yes	Yes	Yes
Elapsed Time (s)	11	1202	798
Epoch	10 of 10	6 of 6	5 of 5
Iteration	100 of 100	606 of 606	505 of 505
Validation Frequency	30 iterations	3 iterations	3 iterations

Deep learning is one of the most powerful machine learning methods for solving critical problems in computer vision. In this paper, three models of Deep Learning have been used to test their accuracy performance. Based on the experimental results, the recommended model for learning alphabets is AlexNet as it takes a shorter time to learn the data where it only took 150 seconds per epoch while the nearest rival, which is GoogLeNet took 200 seconds per epoch. This is due to the architecure of these models where the layers in AlextNet are not as deep as in GoogLeNet. The Simple-CNN model does not produce a good result compared to GoogLeNet or AlexNet because it requires tremendous amount of data for training purposes. Future research is to enhance the simple-CNN architecture and increase the size of the dataset to obtain more accurate and faster results.

4. ACKNOWLEDGEMENT

The authors would like to thank Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Shah Alam, Selangor, for sponsoring this research.

REFERENCES

[1] Patil V.& Shimpi S. (2011). Handwritten English character recognition using neural network Elixir Comp. Sci. & Engg. 41 5587-5591

[2] S Mutalib, R Ramli, SA Rahman, M Yusoff, A Mohamed. (2008). Towards emotional control recognition through handwriting using fuzzy inference. ITSim 2008.

[3] Nijhawan, R.., Sharma, H., Sahni, H., & Batra, A. (2018, April). A deep learning hybrid CNN framework approach for vegetation cover mapping using deep features. In Signal-Image Technology & Internet-Based Systems (SITIS), 2017 13th International Conference on. IEEE

[4] Tuama, A., Comby, F., & Chaumont, M. (2016, December). Camera model identification with the use of deep convolutional neural networks. In Information Forensics and Security (WIFS), 2016 IEEE International Workshop on (pp. 1-6). IEEE.

[5] KA Jalil, MH Kamarudin, MN Masrek. (2010). Comparison of machine learning algorithms performance in detecting network intrusion. Networking and Information Technology (ICNIT)

[6] Cheng, G., Ma, C., Zhou, P., Yao, X., & Han, J. (2016, July). Scene classification of high resolution remote sensing images using convolutional neural networks. In Geoscience and Remote Sensing Symposium (IGARSS), 2016 IEEE International (pp. 767-770). IEEE.

[7] Z Ibrahim, D Isa, R Rajkumar, G Kendall. (2009). Document zone content classification for technical document images using artificial neural networks and support vector machines. Applications of Digital Information and Web Technologies, ICADIWT'09

[8] Ballester, P., & de Araújo, R. M. (2016, February). On the Performance of GoogLeNet and AlexNet Applied to Sketches. In AAAI (pp. 1124-1128).

[9] The Chars74K dataset. Retrieved from http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k/

[10] Hijazi, S., Kumar, R., & Rowen, C. (2015). Using convolutional neural networks for image recognition. Cadence Design Systems Inc.: San Jose, CA, USA.

[11] Gao, H. (2017, August 07). A Walk-through of AlexNet – Hao Gao – Medium. Retrieved from https://medium.com/@smallfishbigsea/a-walk-through-of-alexnet-6cbd137a5637

[12] Deshpande, A. The 9 Deep Learning Papers You Need To Know About (Understanding CNNs Part 3). Retrieved from https://adeshpande3.github.io/The-9-Deep-Learning-Papers-You-Need-To-Know-About.html