SVHN single digit classification using Deep CNN and VGG16

Samya Bose
18180523
MSc in Data Analytics
School of Computing
National College of Ireland

Venkata Devaraju Nandimandalam
18181422
MSc in Data Analytics
School of Computing
National College of Ireland

Shreyashi Mazumder
18181384
MSc in Data Analytics
School of Computing
National College of Ireland

Muhammad Imran Shaikh
17119308
MSc in Data Analytics
School of Computing
National College of Ireland

Abstract: Image detection and classification is one of the most sought after area of study while applying neural networks. One of the lucrative topic in this area is the detection and classification of Street View House Numbers. With the help of Google street view, it has been made possible to take pictures of doors of the houses. This has resulted in the formation of one of the biggest datasets called the SVHN dataset which has thousands of images of house numbers, which can be used to train machine learning models. In this study, we have made a comparison between two machine learning models which uses the SVHN dataset. The first one is traditional Deep CNN (Convolutional Neural Network) and the second one is VGG16. We have tried to compare the performances of the two models, and their accuracy. We have selected these deep learning models because the CNN is a good when it comes to image classification, and, on the other hand, VGG16 (Visual Group Geometry) architecture developed by Oxford is an extension of the traditional CNN, has much better performance and accuracy in theory. This study sets out to test this theory, and in the process classify the house numbers. Due to the limitation of time and processing power, this study is limited to single digit classification. The proposed models can be used for multiple digit classification as well, but is not in the scope of this study.

I. INTRODUCTION

The process in computer vision by using classification algorithm to classify an image based on visual contents. By using deep learning neural networks, it is possible to implement different techniques and recognize images with better accuracies. Deep learning is an artificial intelligence function and a subset of machine learning that investigates huge proportion of samples with the help of statistical learning made by means of many-layered artificial neural networks. ANNs (Artificial Neural Networks) are graphs having their basic computational units (nodes or neurons) grouped in sets usually called layers.In order to get a functional neural network, you got to have at least an input layer and an output layer, and one or more intermediate hidden layers. If the ANN does have more than one of such hidden layers, you got a deep neural network. Sometimes it is very difficult for computer to understand visual content. Humans make sense of what we see based on our experiences and memories. The gap of computer understanding of image classification is bridges with the help of parallel computing, deep learning and neural networks. Computer vision is main technique used for image processing, classification and detection. It is an offshoot of artificial intelligence that aims to teach computers to look at images and videos and understand them. It is basically an attempt at mimicking this capability of human's vision process by passing them on to machine learning algorithms and automating them. Regarding dataset, The Street View House Numbers (SVHN) has been sourced from Stanford.edu website which is real-world labeled image dataset of multiple house/door numbers obtained from Google Street View images. The dataset contains over 600,000-digit images divided into 10 classes and can be seen into two formats bounding boxes on character and centered around a single character image. The selection of file format is .mat file which is MATLAB file of 32 x32 inputs that follows same setup as MNIST dataset. The dataset is publicly available for non-commercial use only. [1] The main objective and motivation for the project is to classify images based on the given dataset of Street View house numbers by implementing two different deep learning classification algorithms i.e. CNN (Convolution Neural Network) and VGG16(Visual Geometry Group) neural network to achieve best accuracy of image classification. Moreover, SVHN dataset is one of the good Kaggle competition challenge, so it would be great to showcase our learned skills by implementing proposed algorithms to juice out the best improvement in the result accuracy. The research question is "Can **VGG16 neural network be used to improve classification accuracy of Multiple digit House/Door Numbers?** ". The programming language used for this project is Python programming language and an IDE is Jupyter Notebook. Python is a general-purpose programming language used mostly for any kind of platform. For our project the image processing in python is based on multiple libraries. The main libraries imported in our code are NumPy for multidimensional arrays, pandas for data munging, matplotlib and seaborn for visualization, TensorFlow for creation of deep learning models, Keras an opensource neural network API based python library

that runs on TensorFlow and other neural networks platforms. The outcome and results of evaluation of image classification would be based on the two machine learning algorithms i.e. CNN and VGG16. CNN is an ensemble of processing nodes arranged in a layer-by-layer manner normally trained end-to-end in a supervised manner using gradient descent-based algorithms such as stochastic gradient descent (SGD) algorithm. The process involved several different filters/kernels consisting of (randomly initialized) trainable parameters depending on the depth and filters at each layer of a network, which can convolve on a given input volume (the first input being the image itself) spatially to create some feature/activation maps at each layer. During this process, (through backpropagation) they learn by adjusting those initial values to capture the correct magnitude of a spatial feature on which they are convolving. These high number of filters essentially learn to capture spatial features from the input volumes based on the learned magnitude. CNN works better for image classification as comparison to RNN and ANN [2]. On the other hand, Vgg-16 based on convolutional neural network architecture. It is pre-trained model for **Keras.** In general, the number after the network name indicates the number of layers the architecture comprises of. The idea was to stack up layers to form a very deep convolutional neural network that would perform extremely well on tasks. Moreover, Vgg-16 bagged one of the awards in ImageNet 2014 challenge [3]. With the support of these two deep learning image classification models, it is possible to attain our desired outcome results.

II. Literature Review

Working Methodology: U. Fayyad et al. [4] has explained the detailed working methodology of KDD from getting data retrieved to getting output data as meaning information. retrieval of datasets from the targeted repositories. The next step is to pre-process and cleansing the data, that may include many steps like removal of noise, outliers and handling of missing values if applicable. The processed data is sent for data transformation or reducing dimension if required, go for mining the data with application of statistical methods and machine learning algorithms to derive meaningful patterns of data.

Different Architectures of Convolutional Neural Network (CNN): P. Sermanat et al. [5] has presented a framework of LSVRC2013 winner (Y. Lecun et al. [6] who proposed the AlexNet) a new technique called over feat on ImageNet dataset which aids in image classification, localizing and detecting defects. They have proposed an approach of learning and then finally determining the boundaries of the object and has applied it on three datasets. They have compared their approach with other approaches and finally got top 5 error rate of 13.6%. They also performed a classification on the validation datasets, where it is observed over feat to perform top 5 14.2% error rate using the average of seven fast models. Now let us discuss some famous frameworks introduced so far. Girshick et al. [7] has proposed an approach to perform object detection using R-CNN. It is a combination of CNN with Region proposals. The net consists of homogenous structure having 13 layers 3x3 and they fine-tuned the network, VGG16 architecture is named as O-net and the baseline as T-net. Their architecture has outperformed and increased mAP percentage. C. Szegedy et al. [8] has proposed a framework called Inception-v2. They participated in ILSVRC 2012 competition and used ImageNet dataset to work on Image classification, detection and recognition. Inception-v2 has many conv layers, inception modules and reduction technique variations in their architecture. This model has taken low resolution input with a good accuracy of recognitive performance. C. Szegedy et al. [9] explored the deep neural techniques and further worked with ImageNet dataset appeared in ILSVRC14 competition, in the first paper they introduced an Inception v1 (GoogLeNet) architecture, created on the concepts of Hebbian and multi-scale processing. The architecture has a total of 22 conv layers and the filters applied in inception model was learned and the layers were repeated multiple times. A. Krizhevsky et al. [10] has come up with a network named AlexNet that was applied on the subsets of ImageNet datasets previously used in ILSVRC-2010 and ILSVRC-2012 competition. The model has obtained good error rate of 15.3%, by implementing their network of 8 layers with weights consisting of 5 conv and 3 fully connected layers. They introduced some new features to their network and enhanced their GPU 2D implemented conv layer in the training CNN. The images trained on the net was RGB and took almost 5 days to finish it. K. He et al. [11] introduced residual nets that aid in image recognition as there are many drawbacks in deep neural nets implementation. In deep neural nets, training data is very difficult and time consuming. The network architecture is inspired from VGG net that has combined ensemble of different 101-layer residuals and achieved good accuracy rate. Reason behind getting better results than other traditional CNN models is not having complexity and presence of few filters. K. Simonyan and A. Zisserman [12] introduced the application of Deep CNN for Large scale Image recognition. The architecture named as ConvNet was applied on ImageNet datasets. The stack of Conv layers consists of having 3 fully connected layers. The configuration of fully connected layers is same in other networks. Here it has many layers for each class and final layer is the soft-max layer. C. Szegedy et al [13] studied that combining the residual connections with Inception architecture can accelerate the model performance. They ensembled inception and residual to form two different architectures called Inception-Resnet-v1 and Inception-Resnet-v2. Combining one inception v4 and 3 residuals has achieved a good error rate on the test data of ImageNet classification challenge. Inception Reset-v2 proven to show a very good recognition performance. R. Girshick [14] has presented Fast R-CNN, the input image is fed and passed through many conv layers and max-pooling will form a feature map. ROI pooling layer uses max pooling to convert feature maps into ROI. Each feature layers are fed into arrangements of full-connected layers that branched finally into 2 output layers, one uses softmax layers for classifying images and other through bounding boxes to perform regression. great thing about this model is, training model in a single stage and can update all layers, achieving higher detection rate (mAP) as

compared as SPPnet and R-CNN. K. He et al. [15] has introduced SPPnet, where method is spatial pyramid pooling that eliminates the requirement of input image to have fixed size. The model was applied on ImageNet classification dataset which shows good accuracy rate than any other CNN models without requirement of any tuning. Feature maps were done only one time for the entire image then pooling the features taking arbitrary regions. The spatial pyramid pooling applied on each window of feature maps to make it a fix length of depiction of window. Like S. Ren et al [16] introduced Faster R-NN which is a combination of Fast R-CNN and RPN. Localizing object was done by proposing region proposal network (RPN). Both RPN and Fast R-CNN are trained all together to share conv features for object detection. Thus, this method instead of learning separate networks share and jointly optimize with bac propagation. The region proposals are computationally less expensive. So, comparing detection accuracy between RPN+VGG and Faster R-CNN, the later has showed mAP of 70.4 % with trained model.

Application of Deep neural nets in different field: Now there are several research studies available where faster R-CNN approach has been taken like, detection of objects like glomeruli in digital pathological images (human renal tissue images) [17] or classification of electrocardiogram images with the help of this deep neural nets, this technique can help in classification of different ECG signals and thus detect the possibility of occurrence of heart attacks. Implementation of this CNN model will bring a solution in the field of Medical Imaging and Healthcare [18]. Exploring popular open datasets like MNIST, solving the text detection and recognition problems by leveraging the Faster R-CNN models. J. Yang et al. [19] addressed the problem recognizing handwritten text through an algorithm based on deep learning, such as Faster R-CNN. It was observed that the method is more precise than the traditional OCR method.

III. Problem Statement

The question that we are trying to analyze in this study is if the modern architecture VGG16 which is an extension of the traditional CNN model better than the actual Deep CNN on the basis of accuracy, time taken and losses.

IV. Proposed Approach

In this study, we have considered KDD (Knowledge Discovery in Databases) methodology. According to this process, we are trying to gain some knowledge from the data, and are applying data mining methods. We divided our group into 2, and proceeded with one method each. At the end, we ran our codes in both the group's systems, so that there is uniformity in the test results. For our dataset, Figure 1 shows the KDD process that we have followed:

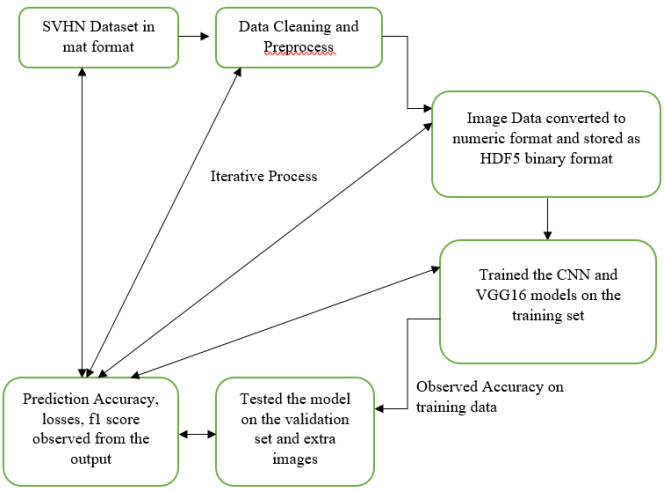

Fig 1: KDD process flow for this study.

CNN:

While using Convolutional Neural Network, we have kept in mind the KDD process, and likewise, we have proceeded with the model. The first step of the process was to get the data, which we have downloaded from the Stanford site. We selected the .mat file, which was present there, as it had the digits in the House Numbers cropped to make it a single digit and the images were already labelled. All the other details were still there in the picture like the colours and pixel density. The next step was pre-processing, which included the normalization of the data, creating a balanced test set, and then finally grey scaling of data. It was found that the dataset had a label 10 which is inaccurate. So, the labelling was changed, and the 10 labels were changed to 0, making the label range from 0-9. After the grey scaling was done, the total size of the training set and the validation set was found to be 2.28GB and 101.69MB respectively. These image sets were then one hot encoded and saved to HDF5 file. The variables were released, to save the memory. These files were then called for the training and testing of the model, which has been explained in detail in the Applied Methods section of this paper. Next, the results were evaluated, and conclusion was drawn.

VGG16:

Vgg16 neural network is another one of the finest proposed approaches when it to image classification. Vgg-16 is constructed over convolutional neural network architecture. It is pre-trained model for **Keras.** In general, the number after the network name indicates the number of layers the architecture comprises of. The idea was to stack up layers to form a very deep convolutional neural network that would perform extremely well on tasks. Moreover, Vgg-16 bagged one of the awards in ImageNet 2014 challenge. Regarding its architecture and working principal, VGG network is characterized by its simplicity, using only 3×3 convolutional layers stacked on top of each other in increasing depth. Reducing volume size is handled by max pooling. In this project we have adopted VGG-16 network in order to classify house digits. VGG-16 network has been successfully applied to image recognition task. The layers define in Vgg-16 model architecture are 16 layers in total

which are divided into 5 Max pool layers,13 layers are convolution layers and 3 of them are dense layers. Altogether when it sums up to 21 layers but only 16 of them are weighted layers. It doubles the number of features after each polling layer to boost the strength to noise by introducing maxpooling and create the CNN translation equivariant which results in doubling the number of channel layers. For computer vision applications transfer learning (which is process of reusing the weights of the models from pre trained-models) plays an important role in development of. Transfer learning basically allows flexibility to use directly pre-trained models for feature extraction. Furthermore, In ILSVRC (ImageNet large scale visual recognition competition) ImageNet dataset which consist over 14 million images having 1000 label classes is used to train and test VGG-16 network and it accomplish 92.7% among top five accuracy models in the competition. Moreover, VGG-16 outperforms against AlexNet network as VGG-16 model use 3x3 filter size whereas AlexNet use 11x11 filter size in first convolution layer and 5x5 in the second convolution layer. The original VGG-16 network takes images of size 224x224 as an input. In this project, we use transfer learning, to do this the images of size 32x32 are fed to the network as an input and change the final layer from 1000 to 10 as our dataset contains 10 classes. So, for this project, proposed model will be test against CNN to achieve best accuracy from these models which would help to classify images up to their best level. On other hand above defined KDD methodology has been utilized to create and design the process flow.

V. Applied Methods

We have applied Deep CNN and VGG16 which is an extension of traditional CNN in this study. The deep CNN uses the images after grey scaling from the dataset, whereas VGG16 uses the full RGB format of the images. This has been done, as the deep CNN takes more processing power and time as per theory, for image processing. The application of these methods are as follows.

Deep CNN:

We selected this method, as the assumption of the architecture is that the input data is an image. This makes it easier to encode some properties into the architecture and get the desired output. Talking about the architecture, there are mainly 4 layers in the CNN model. These are: Convolutional Layer, RELU (Rectified Linear Unit) layer, Pooling layer and finally Fully connected layer.

Starting with the Input, the data that we got, had coloured images of house numbers. The images had to be cropped to a uniform size so that there are no variations in the image sizes when it is fed into the model. So, the images were cropped to 32 x 32 format. Now, we know that RGB colouring will be having 3 channels, however, it becomes too tedious for the CNN to work on 3 channels, and takes a lot of processing power. So, the images were grey scaled so that the channels are reduced to 1, making the image format (32 x 32 x 1).

Next, the Convolutional layer, we have created two 5 x 5 filter layers, where the first layer slides over the input image and calculates a dot product of the filter entries and the input position. The second layer does the same but over the first convolutional layer. The two layers has been done for the better feature extraction from the image. Due to the grey scaling, the depth of the convolutional layer was set to 1. The RELU layer refers to the layer having the activation function which are applied to every neuron.

The pooling layer which comes next has been used to downsize the dimensions of the convolutional layer. The down sampling has been done to a 2 x 2 receptive field. This discards almost 75% of the data from the convolutional layer. This loss of data is of advantage to us as it reduces the memory use and frees up space for the next layers, and also, reduces the chances of overfitting the data.

The last layer is the fully connected layer, which has connections to all the output and activations of the previous layer. The fully connected layer has 128 nodes with 5 softmax activation layers.

So, the total architecture of the model is as per Figure 2:

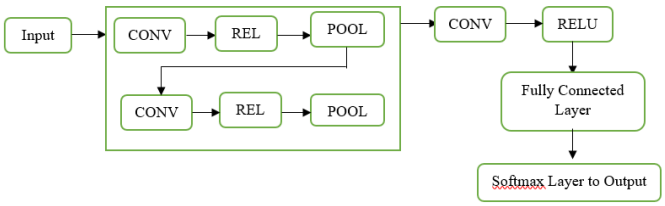

Fig 2: Architecture of the Deep CNN model

VGG16:

For classifying the numbers in Street view house number images dataset for solving the image classification task we are using the VGG16 model. It is a convolution neural network architecture consisting of 16 layers these consist of convolution layers, pooling layers, activation layers, and fully connected layers. VGG16 achieved 92 percent accuracy when trained on the ImageNet dataset. By replacing the filters in first and second convolution layers it exceeds the Alexnet network. Instead of large hyperparameters, VGG16 has convolution layers of 3x3 and it maintains consistency of convolution and pooling layers in the architecture. We are training the SVHN dataset with VGG16 in Keras. Keras is an open-source library that is used for implementing deep learning models it runs on top of TensorFlow it uses the TensorFlow framework. Keras consist of high-level APIs and architecture is simple and user-friendly.

Before training the model, we need to preprocess the data we are using 32x32 mat files consist of street view house number images with help of SciPy library loading the mat files and giving the corresponding labels to each pixel of images. Transforming the input data represents the pixel values of three channels and act as input to CNN first layer. Converting the training and testing data to float32 and normalizing the pixel value by using the largest pixel value 255 and dividing the pixels with it after normalizing the value ranges between 0 and 1. There are ten categories present in the data and converting

the labels to categories with the help of a to_categorical function.

The VGG16 model was imported from the Keras library and importing models and layers from Keras library the VGG16 is a sequential model so we are using the sequential function from Keras and importing the con2d, flatten and dense layers which are useful in creating the last layers of the model. We can create the VGG16 model by keeping channels at last or first with height and width. During the creation of the model, the arguments play a major role. We are including the three fully connected layers to network with the help of include_top to true. The models available in Keras are trained on ImageNet data we can use those weights are else we can initialize the random weights we assigned the random weights. The channels can be at the end or front based on data format the height and width are not smaller than 32 and should contain three channels the input shape we are using was (32, 32, 3) and applying the max pooling. We can classify the classes in the image with the help of classes and need to use only when include_top is true, and no weights are specified. The compile method was used, and it consists of arguments like metrics, loss, and optimizers, etc. we can call the optimizer by its name or instantiate before passing to model clipnorm and clipvalue are two main parameters common in all optimizers. There are many optimizers present in the Keras in our model we are using the Adadelta optimizer, which is an extension of Adagrad, using moving window it adapts learning rates and we can use the default values for learning rate and decay factor for Adadelta optimizer. We trained the model by using the Adadelta and Adam optimizer, but accuracy was very less when we used Adam optimizer, so we used Adadelta. Loss and optimizers are two parameters to compile the model. The output is in categorical format, so we are using the categorical_crossentropy loss function for converting an integer to categorical we are using Keras utility. Metrics are used to find the performance of the model there are available metrics in Keras, and we can create the own functions and found the precision, recall and f1 scores for training and testing data. The fit method is used to train the model with epochs, batch_sizes and validation_split the value between 0 and 1 and it shows how much fraction of training data used for validation.

VI. RESULTS

Deep CNN: As mentioned earlier, the SVHN dataset consists of images of house numbers. This can be seen from Figure 3. The Deep CNN uses the grey scaled images for the model training and testing, whereas the VGG16 model takes the whole RGB coloured picture for training and testing.

Fig 3: Dataset images with full colour

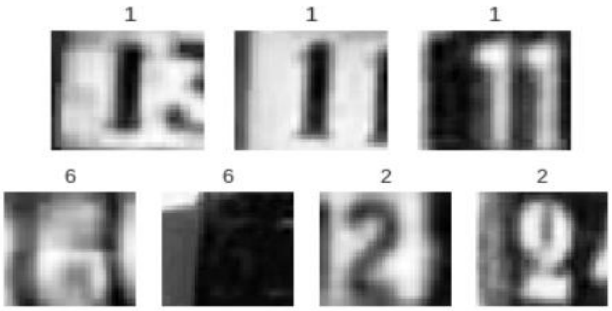

Fig 4: Grey Scaled images of the dataset used for Deep CNN

To avoid repetition, we have not included the test and validation set images which are somewhat similar. The Deep CNN model training yielded us with correctly classified digits, and some incorrectly classified ones. This can be seen from Figure 5 and 6.

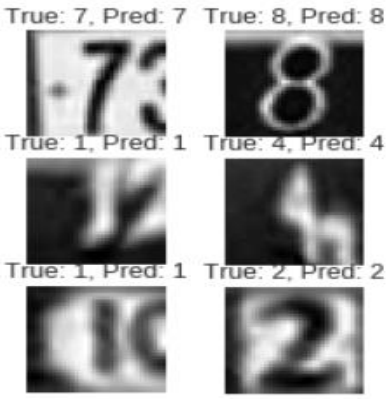

Fig 5: Correctly classified digits

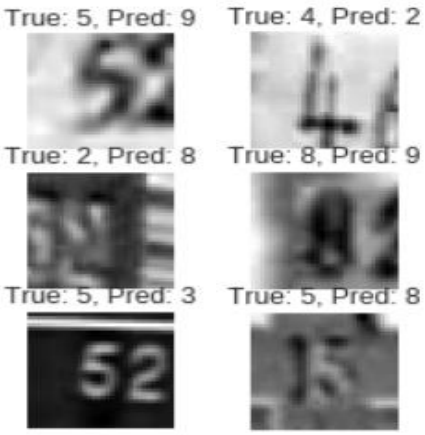

Fig 6: Incorrectly classified digits

The next figure i.e. Figure 7, shows the losses over the period of steps in the case of validation data and training data. In this figure, the green line indicates validation loss and the red line indicates training loss. We can see from the graph that there is a gradual decrease of losses in case of training data, although it is a steep decline. In the case of validation data, we see that

there is a stepwise decline in the losses at almost the same number of steps as that of the training data.

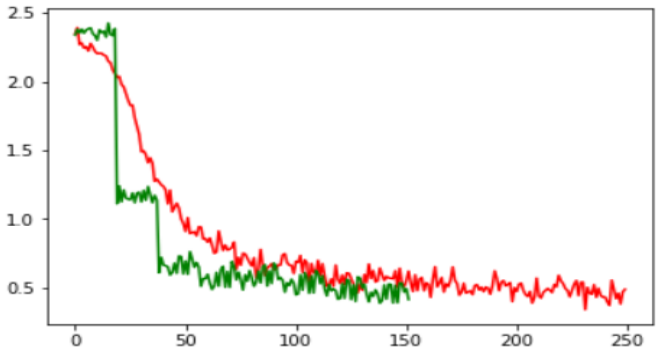

Fig 7: Training ad Validation Losses

Figure 8 is a straight forward confusion matrix for the accuracy of the Deep CNN model. Here the diagonal shows the correctly predicted labels. As we can see, the percentage in the diagonal is quite high which means that the model has correctly classified those labels. The final accuracy of the Deep CNN model on the training dataset was found to be 91%, and on the test dataset was found to be 83.6%.

Fig 8: Confusion Matrix

VGG16: The model performance can be calculated by using the evaluation methods like Accuracy, Precision, Recall, and F1 scores. The VGG16 model which we created predicted the number with good accuracy on both training and testing data. 73257 images present in the training data out of it 7362 samples are used for validation. We used batch_size of 200 and 10 epochs.

The following graph shows the accuracy of train and validation data at each and every epoch the accuracy of train and validation data was increased suddenly after the third epoch and it shows the train loss and validation loss is reduced suddenly after the second epoch.

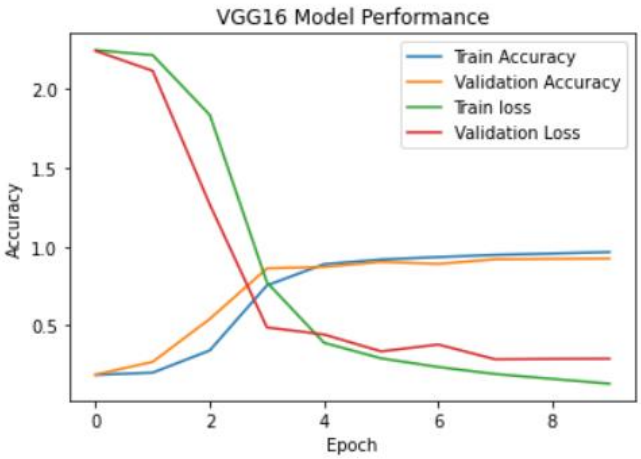

Fig 9: VGG16 model performance

The following figure explains the numbers that the model predicted accurately and with the trust percentage example the ground truth value was 5 and model predicted with 5.

true	guess	trust
5	5	0.993704
2	2	0.995610
1	1	0.999996
10	10	0.993434
6	6	0.956039
1	1	0.999817

Fig 10: Correctly classified

The following figure shows the wrong predictions done by the model actual ground truth value was 3 and the model guessed it as 10.

true	guess	trust
3	10	1.000000
8	1	1.000000
2	1	1.000000
2	1	1.000000
1	3	0.999998
1	5	0.999998

Fig 11: Incorrectly Classified

The created model was evaluated on the test data the test data consist of 26032 images the following are the test result values of loss, accuracy, F1 score, precision and recall. The accuracy

of test data is around 92% which indicates that the model performed well on test data.

```
Test loss: 0.2872140811511259
Test accuracy: 0.9277811646461487
Test F1 Score: 0.9291015267372131
Test Precision: 0.9397420287132263
Test Recall: 0.9190340638160706
[0.2872140811511259, 0.9277811646461487,
 0.9291015267372131, 0.9397420287132263,
 0.9190340638160706]
```

Fig 12: Model performance indicators

VII. Evaluation

From the results of both the methods, it is clearly visible that the accuracy of the VGG16 model is much higher than the Deep CNN model. Mainly when it comes to the classification of the test data, the VGG16 architecture outshines the traditional CNN. Also, it is true that the CNN used grey scaled images for its classification, as the use of RGB images is much more computationally expensive and time consuming. But the VGG16 model did the classification in the RGB images with ease, which reflects the superiority of the newer architecture. It should also be kept in mind, that when it comes to the classification of house numbers, we take the accuracy threshold at 99%, as inaccurate classification can lead to major problems in the real world scenario. In this regard, the VGG 16 is a better model for the above mentioned classification.

VIII. Conclusion

Finally, we can say that the study was successful in recognizing the better architecture for the classification of the SVHN dataset. Though it was computationally expensive to conduct the whole process, the presence of platforms like Colab helped us a lot to carry out this study. It was evident beyond reasonable doubt that the model VGG16 is a better extension of the traditional Deep CNN model, when it comes to the classification of the images.

As far as future work is concerned, the same study can be conducted with the help of RNN and other architectures such as Resnet and VGG19, and check for the best fit for the dataset, which includes better accuracy, training times, and the computational power required.

References

[1] MNIST. http://ufldl.stanford.edu/housenumbers

[2] H. Eghbali and N. Hajihosseini, "Deep Convolutional Neural Network (CNN) for Large-Scale Images Classification", SSRN Electronic Journal, 2019. Available: 10.2139/ssrn.3476258.

[3] X. Liu, M. Chi, Y. Zhang and Y. Qin, "Classifying High Resolution Remote Sensing Images by Fine-Tuned VGG Deep Networks", IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium, 2018. Available: 10.1109/igarss.2018.8518078 [Accessed 30 March 2020].

[4] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “The KDD process for extracting useful knowledge from volumes of data,” Commun. ACM, vol. 39, no. 11, pp. 27–34, Nov. 1996, doi: 10.1145/240455.240464.

[5] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, “OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks,” arXiv:1312.6229 [cs], Feb. 2014, Accessed: Apr. 18, 2020. [Online]. Available: http://arxiv.org/abs/1312.6229

[6] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998, doi: 10.1109/5.726791.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation,” in 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, Jun. 2014, pp. 580–587, doi: 10.1109/CVPR.2014.81

[8] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, Jun. 2016, pp. 2818–2826, doi: 10.1109/CVPR.2016.308.

[9] C. Szegedy et al., “Going deeper with convolutions,” in 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, Jun. 2015, pp. 1–9, doi: 10.1109/CVPR.2015.7298594.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386

[11] R. Girshick, “Fast R-CNN,” arXiv:1504.08083 [cs], Sep. 2015, Accessed: Apr. 19, 2020. [Online]. Available: http://arxiv.org/abs/1504.08083

[12] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition,” arXiv:1406.4729 [cs], vol. 8691, pp. 346–361, 2014, doi: 10.1007/978-3-319-10578-9_23

[13] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” arXiv:1602.07261 [cs], Aug. 2016, Accessed: Apr. 19, 2020. [Online]. Available: http://arxiv.org/abs/1602.07261

[14] Y. Kawazoe et al., “Faster R-CNN-Based Glomerular Detection in Multistained Human Whole Slide Images,” J. Imaging, vol. 4, no. 7, p. 91, Jul. 2018, doi: 10.3390/jimaging4070091
[15] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition,” arXiv:1406.4729 [cs], vol. 8691, pp. 346–361, 2014, doi: 10.1007/978-3-319-10578-9_23
[16] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031
[17] Y. Kawazoe et al., “Faster R-CNN-Based Glomerular Detection in Multistained Human Whole Slide Images,” J. Imaging, vol. 4, no. 7, p. 91, Jul. 2018, doi: 10.3390/jimaging4070091
[18] Y. Ji, S. Zhang, and W. Xiao, “Electrocardiogram Classification Based on Faster Regions with Convolutional Neural Network,” Sensors, vol. 19, no. 11, p. 2558, Jun. 2019, doi: 10.3390/s19112558
[19] J. Yang, P. Ren, and X. Kong, “Handwriting Text Recognition Based on Faster R-CNN,” in 2019 Chinese Automation Congress (CAC), Hangzhou, China, Nov. 2019, pp. 2450–2454, doi: 10.1109/CAC48633.2019.8997382.