International Journal of Application or Innovation in Engineering & Management (IJAIEM)
Web Site: www.ijaiem.org Email: editor@ijaiem.org
Volume 10, Issue 4, April 2021 **ISSN 2319 - 4847**

Analysis of Non-Linearity Accuracy for a Deep Learning model usingGPU on TensorFlow

T. Tritva Jyothi Kiran

PhD Scholar of APJ ABDUL KALAM UNIVERSITY, INDORE, India.

ABSTRACT

The ability to process large number of features makes deep learning very powerful when dealing with unstructured data. Still, Deep Learning algorithms can be excess for more complex problems because they require access to a vast amount of data to be effective. So many researchers analysed performance of Deep Learning models implemented using Python andin previous researchers work Linearity resulted less accuracy. To overcome these issues of accuracy, loss rate and linearity,in this paper I am showing the performance analysis ofDeep Learning modelon TensorFlow with Keras. TensorFlow emerged on top of the Pythonlibraries implemented by the Google. My model trained to classify the image on few thousand images and used Non-Linearity in network for images, and tested on TensorFlow weather the model is able to predicted accurately and successfullycompared the human brain to artificial neural networks in terms of image recognition and compared the performance analysis for accuracy of the model prediction on TensorFlow, and also compared the result with Python.Finally, TensorFlow with Non-Linearity showed high accuracy 88% on Intel® Core™ i3-7100U CPU.

Keywords:TensorFlow, CNN (Convolution Neural Network), Deep Learning, Python, RELU (Rectified linear unit), Max Pooling, Flattening.

1. INTRODUCTION

Deep learning is receiving a share of attention these days because of its reachingunparalleled levels of accuracy to the point where deep learning algorithms can outperform humans at classifying images. Depending on the features, our brain is classifying the things when we see things. The process of building a Convolutional Neural Network always involves four major steps [5] as shown in Figure (1).

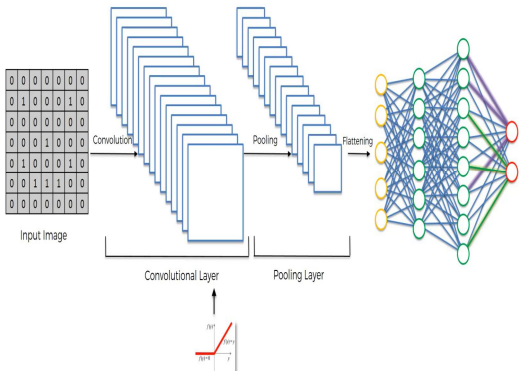

Figure (1) – Full Connection.

Step - 1: Convolution
Step - 2: Pooling

Step - 3: Flattening
Step - 4: Full connection

In Step-1 [2][5] the Multilayer Neural Networks trained with the backpropagation algorithm with a successful Gradient-Based Learning technique with appropriate network architecture, Gradient-Based Learning algorithms can be used to produce a complex decision surface that can classify high-dimensional patterns with minimal pre-processing.
The convolution is able to successfully capture the spatial and temporal dependencies in an image through the application of relevant filters. The evolution operation about feature detectors, Filters, future maps are described and implemented [2][5]. and the below shown Figure (2) describing how the transformations are carried clearly.

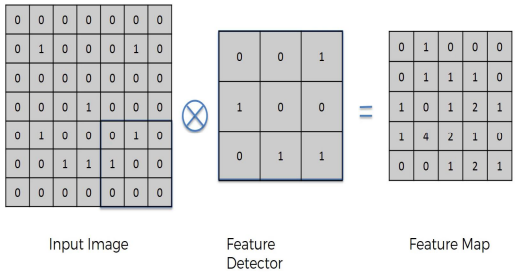

Figure (2)

[1][5]Because linearity is not good and we want more nonlinearity in our network for image recognition So used mathematical model shown below.

$$(f * g)(t) \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f(\tau)\, g(t-\tau)\, d\tau$$

[1][3][5] Rectified activation units (rectifiers) are essential for state-of-the-art neural networks. The importance of rectifier neural networks is for image classification from two aspects. First, the Parametric Rectified Linear Unit (PReLU) that generalizes the traditional rectified unit. PReLU advances model fitting with closely zero additional computational cost and little overfitting risk. Second derives a robust initialization method that particularly considers the rectifier nonlinearities. This method empowers us to train enormously deep rectified models directly from scratch and to investigate deeper or wider network architectures. Based on PReLU networks previous work achieved 26% relative improvement on the Image classification. [5] To improve the accuracy, the structure of the convolutional neural networks (CNN) removed linearity and implemented a nonlinear activation function. [5] A mathematical model called the Rectified-Correlations on a Sphere" (RECOS) is projected and encountered weights defined a set of anchor vectors in theRECOS model. Anchor vectors signify the frequently taking place patterns (or the spectral components). At that point, the behaviour of a two-layer RECOS structure is analysed and compared with its one-layer counterpart and finally model is generalized to a multilayer system.

[1][3][5]RELU (Rectified linear unit) operations are described to understand the transformations used in the below shown Figure (3).

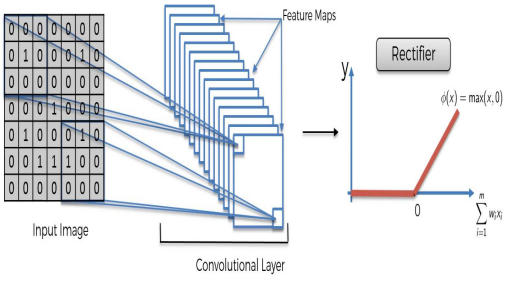

Figure (3) – RELU

convolution layer outputs are well depicted for clear understanding purpose by the below shown Figure (5).[4][5] a common practice to gain invariant features in object recognition models are to cumulative multiple low-level features over a small neighbourhood. Still, the discrepancies between those models makes a evaluation of the properties of deferential aggregation functions hard to gain insight into deferent functions by directly comparing them on architecture for several common object recognition tasks results show that a maximum pooling operation significantly outperforms subsampling operations. Regardless of their shift-invariant properties, covering pooling windows are no significant improvement over non-overlapping pooling windows. By applying this, old work achieved the state-of-the-art error rates of 4.57%.

In Step-2 Max pooling is applied [4][5]. [1] The pooling layer is in control for dropping the spatial size of the convolved feature to decrease the computational power required to process the data through dimensionality reduction and the fine work of Max pooling is described and showed clearly in the below Figure (4).

In Step-3 we have to proceed from pooled layer to flatten layer[5] to convert our input image into a suitable form for Multi-Level Perceptron, so flatten the image into a column vector.We can easily understand the transformations and changes occurs within the

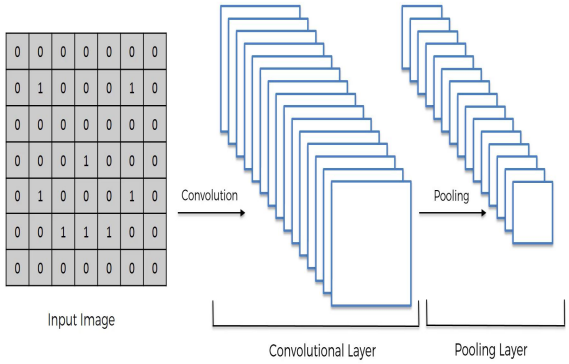

Figure (4) – Max Pooling.

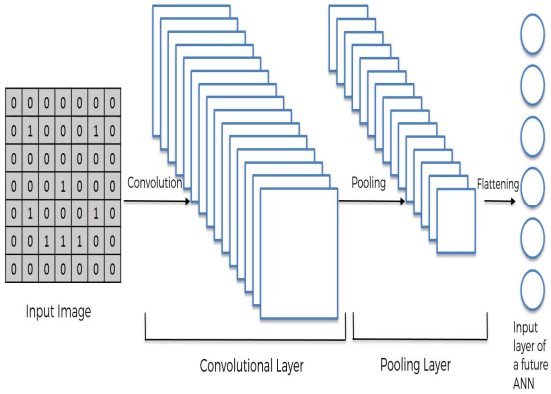

Figure (5) – Flattening.

In Step-4 [5]The flattened output is fed to feed-forward neural network and backpropagation applied to every iteration of training in full connection and is well described the implementation process is clearly shown in Figure (1) and after testing the full connection shows the CNN output matching within the layers with accuracy values as shown in Figure (6) clearly for the understanding purpose. The model is talented to differentiate between controlling and certain low-level features in images and classify them using the SoftMax Classification technique [5].

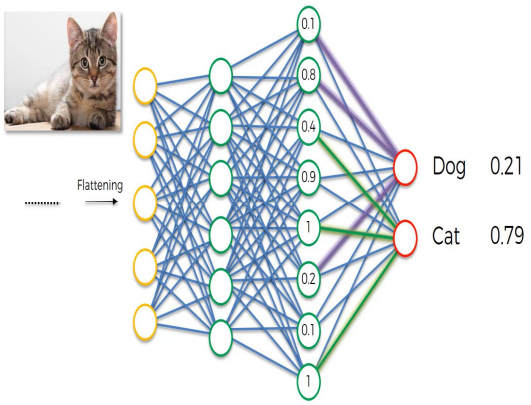

Figure (6) – Full Connection Results.

2. IMPLEMENTATION

Resolving an image classification problem, everywhere our goal will be to state which class the input image fits to. The way we are going to achieve it is by training an artificial neural network on few thousand images and make the NN (Neural Network) learn to predict which class the image belongs to.So, coming to the implementation part, I used Keras deep learning library in TensorFlow to build CNN (Convolutional Neural Network). First, I imported all the required Keras packages using which to build CNN, and every package is installed properly in my machine Intel® Core™ i3-7100U CPU, and I tested the code using TensorFlow.

I used all the required training and test dataset, there are 10,000 images in both folders, which is the training data as well as the test dataset.

The**TrainingSet**comprises two sub folders **cats** and **dogs**, each holding 8000 images of the respective category. And the folder **TestSet** contains two sub folders **cats** and **dogs**, each holding 2000 images of respective category. Then we added a convolution layer by using the "Conv2D" function. The input image our CNN is going to be taking is of a 64x64 resolution with RGB, which is a colour image,then applied the activation function and rectifier function.

To perform pooling operation on the resultant feature maps we get after the convolution operation is done on an image. The primary aim of a pooling operation is to reduce the size of the images as much as possible. But the key thing to understand here is that we are trying to reduce the total number of nodes for the upcoming layers then by taking our classifier object and added the pooling layer. We just reduced the complexity of the model without reducing its performance.Here and nowchange all the pooled images into a continuous vector through Flattening. Flattening is a very important step to understand. Fundamentally doing at this point is taking the 2-D array, that is the pooled image pixels and converting them to a one-dimensional single vector.

To create a fully connected layer, to this layer we are going to connect the set of nodes we got after the flattening step, these nodes will act as an input layer to these fully-connected layers. In place of this layer will be existing between the input layer and output layer, we can refer to it a hidden layer.

Dense is the function to add a fully connected layer,
The output layer should contain only one node, as it is binary classification. This single node will give us a binary output of either a Cat or Dogand we will be using a sigmoid activation function for the final layer. And set Optimizer parameter to choose the stochastic gradient descent algorithm. and set Loss parameter to choose the loss function.To finish, the system of measurement parameter is to choose the performance metric.

To fit CNN to the image dataset we pre-process the images to prevent over-fitting. Overfitting is when you get a great training accuracy and very poor test accuracy due to overfitting of nodes from one layer to another.**S**ubsequently,formerly we fit our images to the neural network, we need to perform some image augmentations on them, which is basically synthesising the training data. We are going to do this using Keras library for doing the

synthesising part as well as to prepare the training set as well as the test set of images. And creating synthetic data out of the same images by performing different type of operations on these images like flipping, rotating, blurring, etc.Once a neural network is accomplished on every training samples only in one pass, we say that one epoch is complete. So, training process should consist more than one epoch.In this case I have defined 25 epochs.

Once we have the test image, we will prepare the image to be sent into the model by converting its resolution to 64x64 as the model only accepts that resolution. Then we are using predict () method on classifier object to get the prediction. As per the prediction will be in a binary form, we will be getting either a 1 or 0, which will represent a dog or a cat respectively.

3. RESULTSANDCOMPARISIONS

This work is simulated and tested on Intel® Core™ i3-7100U CPU.

In the starting Epoch, Though the model showed 0.64 loss but surprisingly at the last epoch the model showed only 0.25 loss rate, it's really great achievement of accuracy with very low loss rate of 0.1. The result of performance analysis is clearly depicted and can be observed in the figure (7).

In the similar way, though the accuracy is only 60% at the starting epoch but surprisingly at the last epoch the model given greater improvement on accuracy as 88%. It's really a greater improvement percentage on previous works [1][2][3][4][5]. The performance analysis result is clearly depicted and can be observed in the Figure (8) shown below.

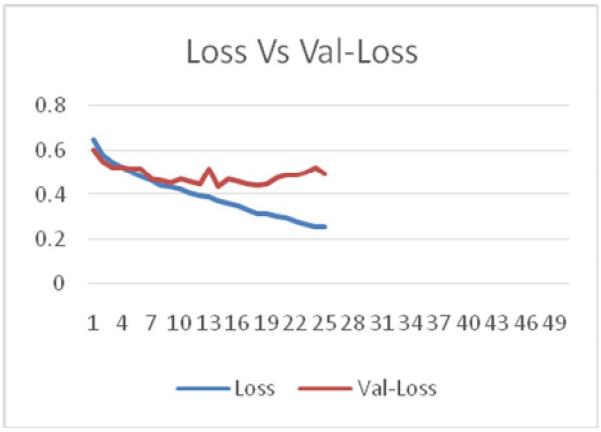

Figure (7) – Loss Vs Val-Loss

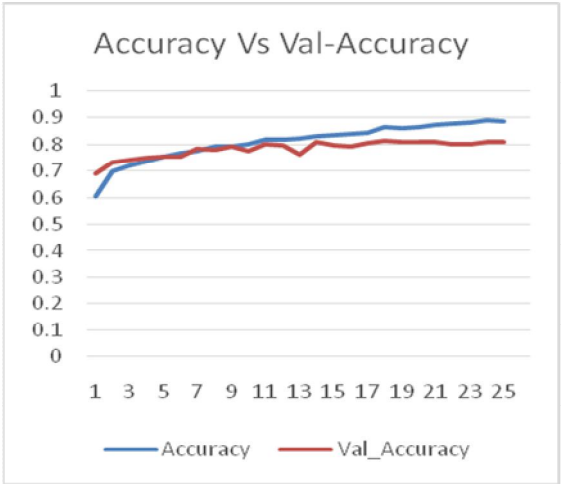

Figure (8) – Accuracy vs Val_Accuracy

The results of the SoftMax and Cross-Entropy methods are shown in Figure (9). The model given improved results as the result values observed in previous works [1][2][3][4][5] and the result observation saying that the Cross-Entropy is the preferred method for classification.

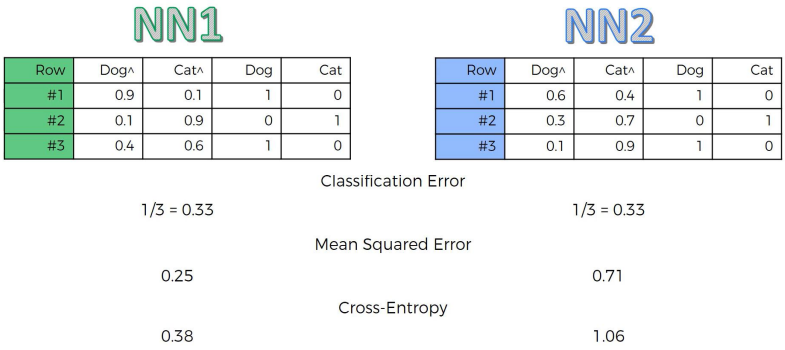

Figure (9).

And I also tested the same method of implementation model on Python for the comparison with TensorFlow. As of my expectations the TensorFlow showed greater improvement in ETA and accuracy as compared with the Python, it can be observed by the output of Python given below.

PYTHON OUTPUT

Epoch 1/25

250/8000 [.....] - ETA: 1:20:29 - loss: 0.6933 - accuracy: 0.5015

4. CONCLUSION

The model is built and can be trained on any type of class you want, I used the image dataset of cats and dogs only as example for my work. This model for example, can take a brain scan as an input and predict if the scan contains a tumour or not accurately.

In this paper I showed the performance analysis of CNN (convolutional neural network) Deep Learning model on TensorFlow with Keras. My model will be trained on few thousand images of cats and dogs, and tested on TensorFlow weather be able to predicted accurately if the given image is of a cat or a dog and successfully compared the human brain to artificial neural networks in terms of image recognition and compared the performance analysis of the accuracy of the model prediction on TensorFlow, and also compared the result with Python and also with the previous works.

To my knowledge, my result is the first to surpass human-level performance 88% on this visual recognition challenge.

5. SCOPE FOR FUTURE WORK

Linearity is not good and how we want more nonlinearity in our network for image recognition to achieve more accuracy and how to improve SoftMax and cross entropy to reduce error rates needs further improvements are left over in this work. Also need to extend the tests for GPU and other hardware accelerations and need to analyse accuracy.

6. REFERENCES

[1]C. C. Jay Kuo, Understanding Convolutional Neural Networks with A Mathematical Model, University of Southern California, Los Angeles, CA 90089-2564, 2016.

[2] Yann LeCun, LeonBottou, YoshuaBengio, and Patrick Haffner, Gradient-Based Learning Applied to Document Recognition, PROC OF THE IEEE,NOVEMBER1998.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun,Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification, Microsoft Research, 2015.

[4] Dominik Scherer, Andreas Muller, and Sven Behnke, Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition, 20th International Conference on Artificial Neural Networks (ICANN), September 2010.

[5] Jianxin Wu, Introduction to Convolutional Neural Networks, LAMDA Group, National Key Lab for Novel Software Technology, Nanjing University, 2017.
[6] Y. lecu,B. boseter, J. SDenker,D Henderson, R.E. hovard, W. hubbred, AND L. Djackal, Backpropagation applied to handwritten zip code reconization, NeuralComputation,VOL1,NO.4PP.541-551,1989.
[7] B. H. Juang, Deep neural networks a developmental perspective, APSIPATransactions on Signal and Information Processing 5 (2016) e7.
[8] A. Ahmed, K. Yu, W Xu, Y. Gong, and E. Xing.Training hierarchical feed-forwardvisual recognition models using transfer learning from pseudo-tasks. ComputerVisionECCV 2008.
[9] F. Agostinelli, M. Hoffman, P. Sadowski, and P. Baldi, Learning activation functions to improve deep neural networks.arXiv:1412.6830, 2014.

ABOUT THE AUTHOR:

I am Mrs. T. Tritva Jyothi Kiran with 9+ years of work as Assistant Professor in Computer Science Department. Previously I have completed AICTE funding Project on IEEE802.11e in JNTUH and published in IEEE conference. I have been awarded Two times the National Award for Excellence "Adarsh Vidya Saraswathi Rastriya Puraskar" from Glacier Global Management in 2020. And Received "Women Researcher" Award from VDGOOD Professional Association 9th conference. Extant I am working in the research domain Deep Learning using TensorFlow and Swarm Intelligence. You can find my Lectures during COVID in my Blog is tritvajyothikiran.blogspot.com.