Densely Connected Inception Networks
CS2IS2 Final Project (2018/2019)

Supratik Banerjee

sbanerje@tcd.ie

Abstract. This paper embraces the observations made in some of the state-of-the-art Convolutional Neural Networks and brings together the best of their qualities. Bringing the best of Inception and DenseNet together, this paper introduces a new Network Architecture, DenseInception-Net. We solve the image classification problem with experimentation on the Fashion MNIST data-set. Further on, this paper questions the importance of deeper Convolutional Networks and compares the implementation of two modified versions of the Inception-v4 architecture comparing against the original Inception-v4, Dense-Net and ResNet architecture to answer the question. The answers derived from this experiment helps us assert and construct the DenseInception architecture. This paper also demonstrates a modification to the DenseInception architecture demonstrating a very shallow variant of it, which is in line with the experiments conducted to answer the above question. The results have been very positive, with Both the DenseInception-Net variants outperforming each and every network tested.

1 Introduction

Image Classification is a very important problem in the branch of Artificial Intelligence particularly under Computer Vision. Its importance arises from its applications in various critical fields, including Robotics, Autonomous Systems, etc. In the last few years the use of Convolutional Neural Networks has accelerated, ever since the record-breaking achievement demonstrated by Alex Net [1] in the 2012 ImageNet Competition [2]. Since then there has been a trend of using deeper architectures with smaller kernel sizes. A lot of research has investigated various architectures to better generalize features from image. This paper demonstrates experiments of Image classification using original implementations of two modified architecture of the Inception-v4 architecture from GoogLeNet [4] to test few hypotheses and subsequently introducing a new architecture.

The motivation of this project comes from my study from the module CS7GV1, for which I was studying the Inception-v4 architecture. The work presented in this paper is an extension of my previous work's future scope. My previous work demonstrated only the Inception-v4 architecture on Fashion-MNIST [5], and a reduced version of inception-v4 (Inception-v4-A) which I had developed just out of curiosity. My

implementation of the reduced version of Inception gave rise to few questions, which I thought I would explore later and with this project it was the best opportunity to do so. Seeing the performance of the stripped-down version of Inception, I questioned, whether it is important to have very deep network to achieve higher accuracy. My reduced architecture was outperforming the original Inception-v4 architecture for Fashion MNIST data-set. Motivated from my previous project, I picked up the Dense-Net architecture for this project. While studying Dense-Net, I decided to use some bit of the Dense-Net architecture and apply it to the Inception-v4 Blocks to reduce its parameters and compare its performance with the Inception-v4 Net. Thus, I removed the Reduction-A and Reduction-B blocks from the Inception-v4-A and replaced them with Transition Blocks used in Dense-Net (Inception-v4-B), which uses Batch Normalization, 2D Convolution with a kernel size of 1 and Average Pooling with a kernel size of 2. Its input size is that of the Reduction Block which it replaced, and the output size is the next Block's Input size.

2 Related Work

The search for a network architecture, which is fast, efficient and accurate has been accelerated since the advances promised by Alex-Net. Google Inc has been at the forefront of their research to discover such an architecture with GoogLeNet, and so have Microsoft and Facebook, with their contribution being ResNet [6] and Dense-Net [11] respectively. In this paper, we study the implementation of Dense-Net, ResNet and GoogLeNet and how Inception-v4-A and Inception-v4-B address a few important questions and how the proposed DenseInception Net perform against them.

3 Problem Definition and Algorithm

To address the Image Classification problem and for the experiments we will be using the Fashion-MNIST [5] dataset. It is a dataset comprising of 70,000 uniquely labeled images of size 28x28 belonging to 10 classes. This dataset was introduced to be a more challenging replacement to the MNIST dataset. When a model is excessively complex, having too many parameters compared to the number of training samples, over-fitting might weaken its generalization ability. To improve the network, many data augmentation and regularization approaches have been used, such as random cropping [1], flipping [3], dropout [7], batch normalization [9] and Random Erasing [8].

3.1 Inception Architecture

The Inception architecture is inspired by the "Network In Network" (NIN) [10] architecture. It uses the "MLP Convolution Layers" introduced in NIN which replaces linear convolutional layer with a multilayer perceptron convolutional layer. The

intuition behind this is to have a universal function approximator for feature extraction of local patches, as it can approximate more abstract representations. These perceptrons are also a mathematical equivalent to a 1x1 convolution kernel. The second concept that the inception architecture derives from NIN is the "Global Averaging Pooling". Even though the purpose behind this was to eliminate the use of a fully connected layer at the end, the inception module adds a dropout layer after applying Average Pooling and followed by a SoftMax layer. The first inception architecture [4] also introduced dimensionality reduction by applying convolution of filter size 1 before applying more expensive convolutions of filter sizes 3 or 5.

In [11] the Inception-v2 architecture was introduced with some new concepts like the "Spatial Factorization into Asymmetric Convolutions", this helps to further optimize the network by reducing the number of parameters. (see **Figure.1).** It demonstrates how any convolution of NxN, can be replaced by a 1xN convolution followed by a Nx1 convolution.). Example, A 3x3 convolution would take 9 parameters, whereas breaking it into 1x3 + 3x1 takes only 6 but gives similar results. This yields good results on medium grid size feature maps between 12x12 and 20x20.

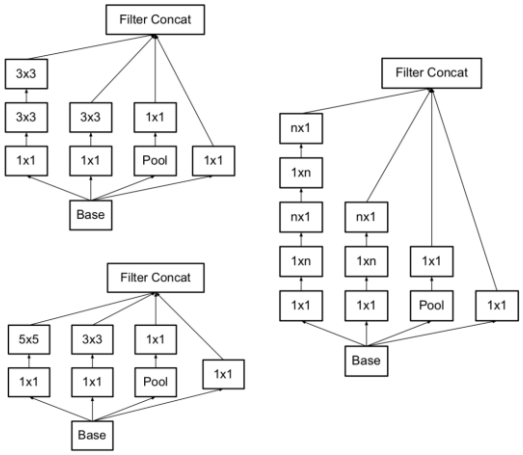

Figure.1. Spatial Factorization into Asymmetric Convolutions

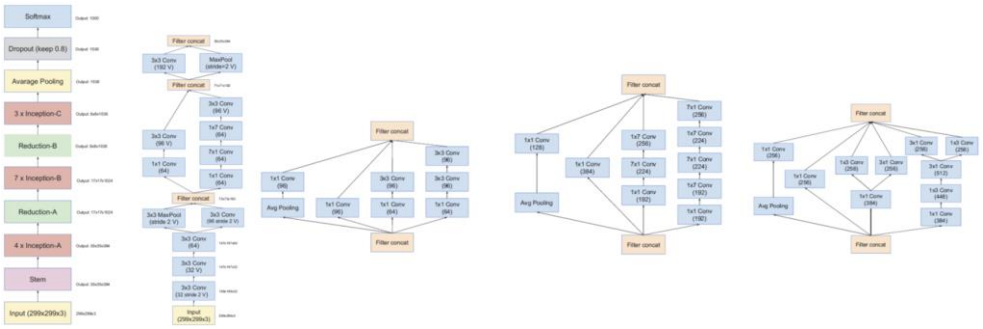

Figure.2. Inception-v4 Architecture

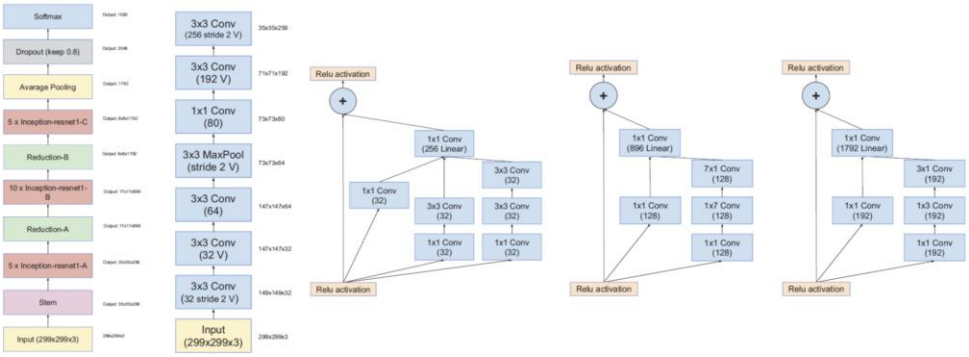

Figure.3. Inception-Resnet-v2 Architecture

The next iteration showcased the Inception-v4[13] architecture. (see **Figure.2.**) It was introduced with dedicated Inception-Reduction blocks to down sample and change the feature map size. It proposes another architecture alongside, that is the Inception-ResNet architecture (see **Figure.3.**), which introduces Residual connection in cheaper inception blocks.

3.2 ResNet Architecture

Given the drive to develop very deep models, this significance gives rise to a very important question according to the author: "is learning better networks as easy as stacking more layers?" [6]. An important issue that arises with very deep network is vanishing gradients, which hamper the convergence of a model. To address this issue the ResNet architecture (see **Figure.4.**) was developed, which show empirical results on how using residual networks can help increase the depth of a network up to 1000 layers. The primary idea behind having a residual layer is to add shortcut connection, which is simply adding information from the previous layer to the current layer and applying ReLu.

In the backward pass of Backpropagation, the bypass enables the error gradient information from the output layer of the network, to propagate without modifications by intervening layers to the top of the network. Thus, all layers in the network can access the unaltered information from the output layer. This property enables ResNets to go Ultra-Deep without sacrificing performance.

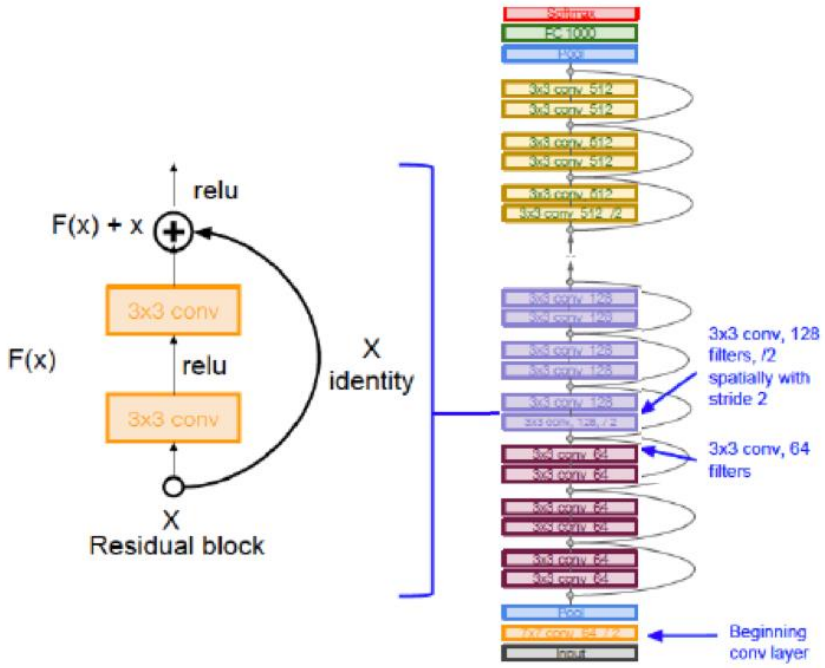

Figure.4 ResNet Architecture

3.2 DenseNet Architecture

DenseNet is a logical extension to ResNet. It shows us that CNN can be a lot deeper, efficient to train and accurate if they have shorter connections. In the quest to further mitigate the vanishing gradient and develop more efficient models, DenseNet architecture (see **Figure.5.**) implements something like that of ResNet. Instead of summing the residual, it concatenates the residual to pass it on to the next layer. The primary idea behind this is to share the same input with the close layers. Along with this, it adds a transition layer to reduce the feature map size along the four blocks. DenseNet provides very good results with lesser parameters.

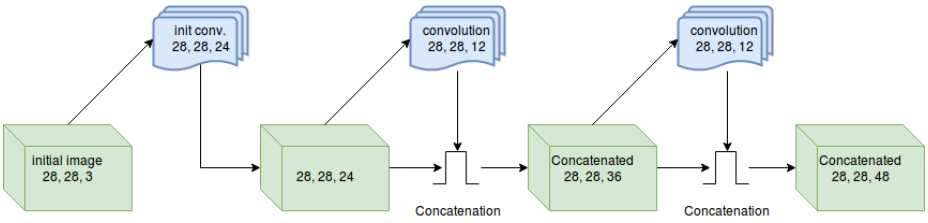

Figure.5. DenseNet Architecture

3.3 Inception Modified Architectures

To achieve the DenseInception-Net some preliminary modifications were made to the Inception Network to test the stability of the network and to confirm few hypotheses. As the trend towards the Image Recognition Network architecture suggests, "going deeper is better". I confirmed this by stripping down the entire Inception-v4 architecture. As described above, the architecture uses 3 inception blocks by the names Inception-A, Inception-B and Inception-C and two reduction blocks by the names Reduction-A and Reduction-B. the inception blocks are placed in the following manner in the network sequence, 4xA, 7xB and 3xC with Reduction-A between A-B and Reduction B between B-C. We formulate a hypothesis that even shallow network should be able to exploit a smaller Image size since Inception Network was designed for Images of size 224x224. To test this, all the Inception Blocks and Reduction Blocks are just used once. (see **Figure.6.**) This architecture is called Inception-v4-A for convenience of referencing.

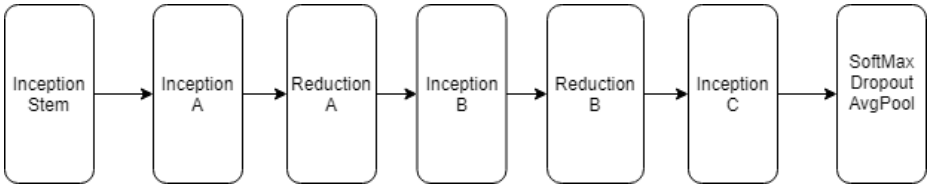

Figure.6. Inceptin-v4-A Architecture

Before working on the DenseInception-Net, one more Important hypothesis was to be tested. What is the performance difference between using just a DenseNet transition Block vs using pure Inception Reduction Blocks? The reduction blocks were replaced by transition blocks. This architecture is the Inception-v4-B. (see **Figure.7.**)

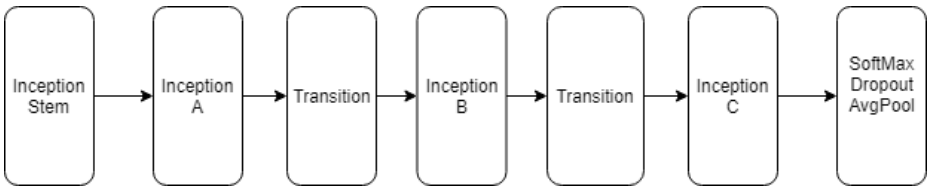

Figure.7. Inceptin-v4-B Architecture

3.4 DenseInception Network Architecture

The philosophy behind working on this was to take the best properties from the best networks and test their performance. Inspired by the Inception architecture we use the spatial factorization [11] and the dimensionality reduction [4] used in inception network. Since DenseNet is a logical extension to ResNet and shows promising results, the architecture implements the Dense like network connection by concatenating the residuals over adding them up. Instead of the reduction block, we stick with using the Transition Blocks used in DenseNet as they have a lesser memory footprint. For our Stem we choose a 2D convolution with filter size 3 and padding 1 to adjust the feature maps for the network. Lastly, we end with the same global average, dropout and linear SoftMax. (see **Figure.8.**)

Figure.8. DenseInception Architecture

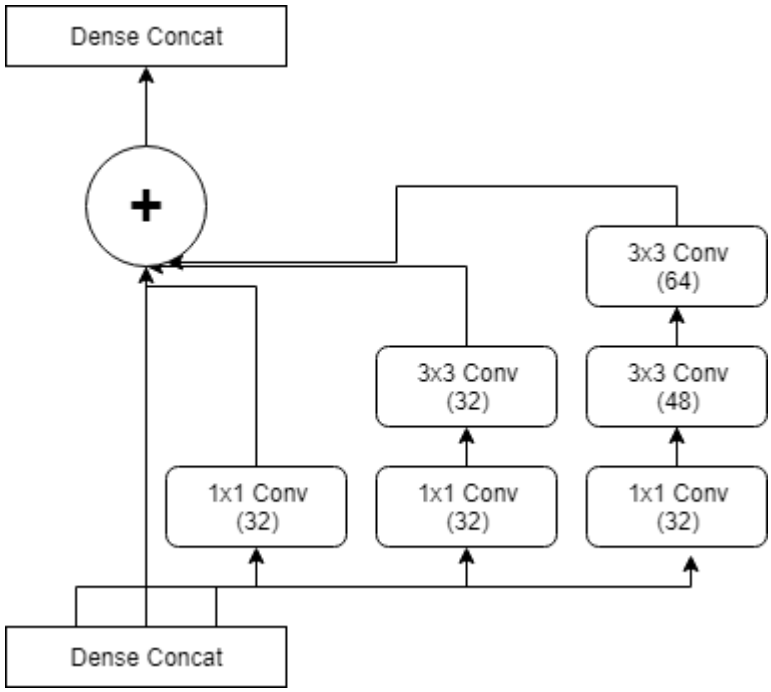

Figure.9. DenseInception A Schema

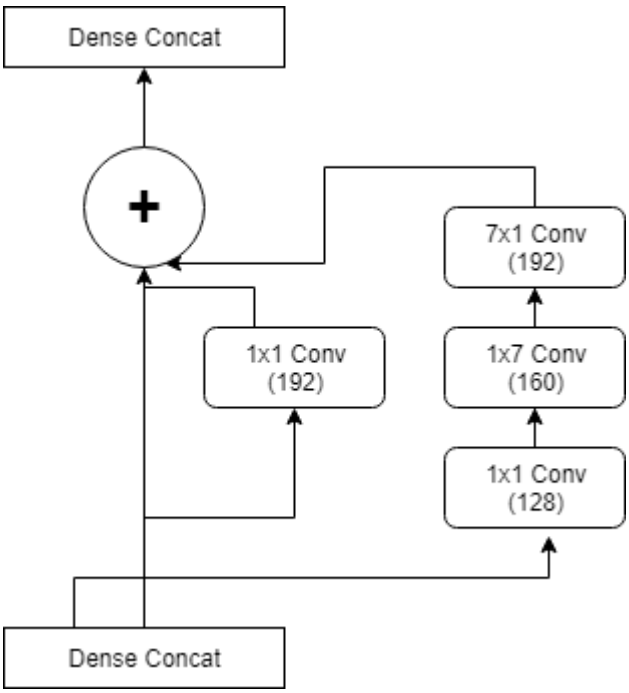

Figure.10. DenseInception B Schema

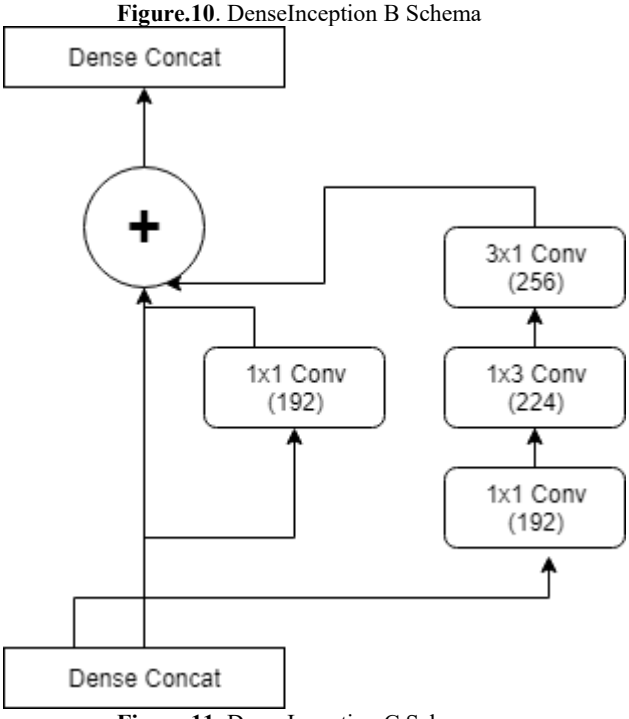

Figure.11. DenseInception C Schema

4 Experimental Results

The experiment was conducted on seven models. The first being the full Inception-v4 architecture, second is a reduced version of inception-v4 (see **Figure.6.**), third, Inception-v4 with Transition layer, instead of Reduction Block (see **Figure.7.**), fourth, a ResNet50, followed by DenseNet121. Lastly, we have the DenseInception Net and a reduced version of it just like Inception-v4-A (see **Figure.12.**). This experiment uses an original implementation of Inception-v4 as it is only available for 224x224 images, so the parameters had to be redone to work for Fashion MNIST which has images of size 28x28. The implementations of ResNet and DenseNet were used from GitHub.

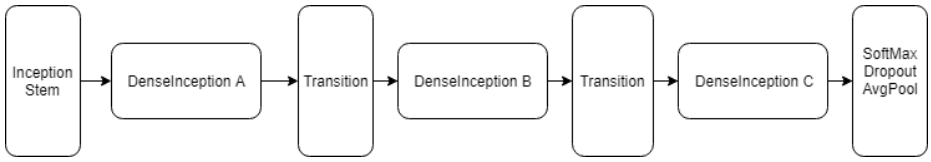

Figure.11. DenseInception Reduced Architecture

The models were trained on three different machines with GPUs in the following order, Nvidia GTX 980m, GTX 1080 Ti and 4x RTX 2080 Ti. The initial tests started with GTX 980m, but soon had to be changed, since it was taking over 48 hours to train the Inception Network. With an upgrade to the GTX 1080 Ti, the performance was better, but still took about 35 hours to train Inceptin-v4. Finally running out of time I had to rent a server with 4x RTX 2080 TI, where the inception model was training under a day and it was possible to further continue my experiments.

Out of all the models, Inception-v4 was the biggest model to train. The ResNet, DenseNet, Inception-v4-A and Inception-v4-B were trainable under 3-7 hours on 4 RTX. While The reduced variant of DenseInception-Net was trainable under 1 hour on 1 RTX, the DenseInception was only tested on 3 RTX as 1 GPU was busy running the reduced version, and there was no more time left to give another training run. The DenseInception architecture was trained in under 12 hours on 3 RTX.

- Methodology: As mentioned earlier, the Fashion-MNIST data-set has been used to test all the models. The reason behind this choice is that it is a very challenging and a new dataset and not all available papers use this for baseline. All models were run for 300 epochs using Stochastic Gradient Descent. SGD used a learning rate of 0.01 with 0.9 momentum and a decreasing learning rate at 4% every 8 epochs. The models were evaluated on few criteria. First and the most important, accuracy, followed by the size of model's parameter. For the result comparison, we will be evaluating the performance of DenseInception against Inception-v4, ResNet50 and DenseNet121.

- Results:

Model	Accuracy	Parameter
Inception-v4	93.5	41.16M
Inception-v4-A	94.3	13.48M
Inception-v4-B	94.45	10.70M
ResNet50	93.93	0.78M
DenseNet121	94.43	6.7M
DenseInception	95.25	14.78M
DenseInception Mod	95.0	1.01M

Table.1. Model accuracy and parameter comparison

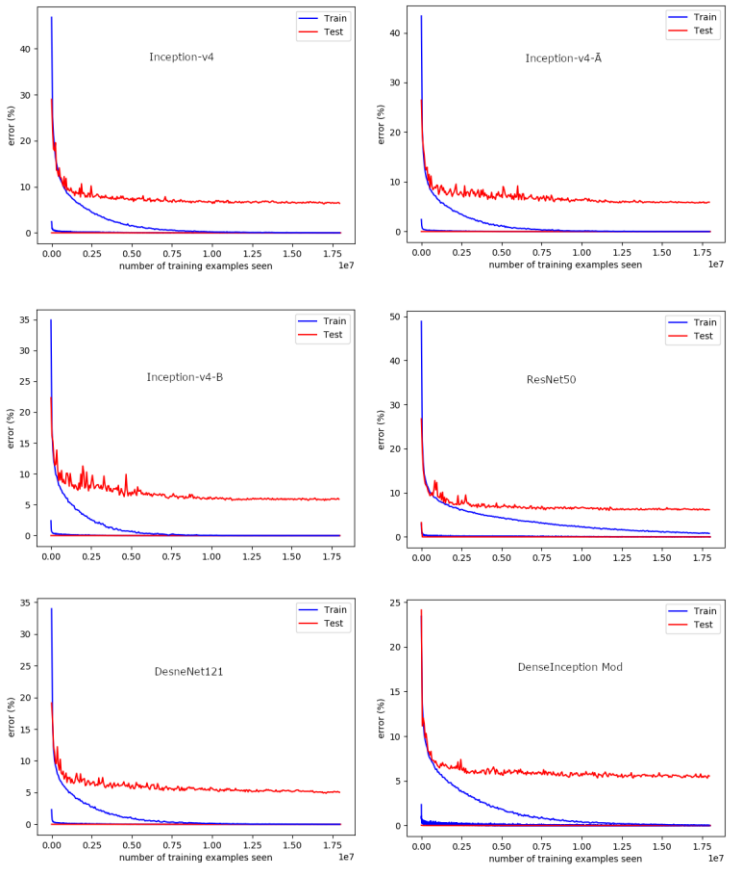

Chart.1. Model error plot

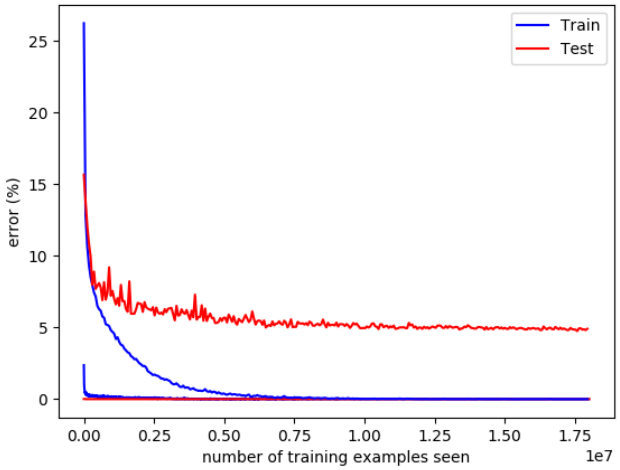

Chart.2. DenseInception Model error plot

- Discussion: The above charts give an overview to how the models fair against training examples. Also, with the spikes on the test line, we can see how long each model takes to converge. The trained Inception-v4 architecture has the greatest number of training parameters with a good accuracy. The Inception-v4-A surprisingly shows better accuracy with lesser parameters. Inception-v4-B has slightly lower parameter than that of Inceptoin-v4-A but also slightly higher accuracy. This might be an indication and an answer to the question proposed above, that maybe very deep network is not a very important requirement to achieve better networks. This leads us to the next experiment, that is, swapping Inception Reduction blocks for DenseNet Transition Blocks, as the Inception-v4-B model outperformed Inception-v4 and Inception-v4-A. Following the success of the experiments, the next step was to design the full architecture and test a shallower version of it. As it can been seen from the results in **Table 2.** Even the shallower network out performs every other network with only 1.01 million parameters. With this we can have a good overview of how the DenseInception Network fares against some of the state-of the art techniques:

5 Conclusions

The DenseInception Network introduced in this paper, has been derived more in a way as art than science, as it is derived with just understanding the important intuition being contributed by each of the above-mentioned networks and not looking at any scientific or mathematical proofs. This leaves scope for further work and development with the model, where the input and output features of the network layers can be

constructed with more care. While further improvising DenseInception for layer feature count, we can also test how we can shrink the layers a bit more in size to bring only optimal number of convolutions needed and expand them in depth in comparison to DenseNet.

As we observed that shallow network works really well, but this observation might differ with images of higher resolution. Also, the models should be tested against the Inception-Resnet architecture. Both these things couldn't be tested due to shortage of time, as these models can take way too much time to train.

References

1. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Communications of the ACM, vol. 60, no. 6, pp. 84–90, 2017.
2. O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge", 2014.
3. K. Simonyan, A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", ICLR 2015.
4. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015.
5. H. Xiao, K. Rasul, R. Vollgraf, "Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms", 2017
6. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016
7. N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting", JMLR, 2014.
8. Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang. "Random Erasing Data Augmentation". arXiv preprint arXiv:1708.04896, 2017.
9. S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift", ICML, 2015.
10. M. Lin, Q. Chen, and S. Yan, "Network in network", ICLR, 2014.
11. G. Huang, Z. Liu, L. van der Maaten, K. Q. Weinberger, "Densely Connected Convolutional Networks", 2016.
12. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
13. C. Szegedy, S. Ioffe, V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning", arXiv:1602.07261 (2016).