OPTIMIZERS COMPARISION

OBJECTIVE:

There have been developments of various types of optimizers depending on the approach they adopt to converge to the global minimum. In my paper I have done a comparative study on SGD, Adamax, Adagrad, Nadam, Adam, RMSprop and Adadelta. I have taken the default values of hyperparameters provided on the official Keras website to avoid any absurd results. In this finding the main goal was to find what difference in accuracy and loss are when we use different optimizers on a deep CNN network (here, ResNet-50).

TEST CASES:

I trained the model on a CIFAR-10 dataset. The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

The dataset is divided into five training batches and one test batch, each with 10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another. Between them, the training batches contain exactly 5000 images from each class.

The classes range from Airplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship and Truck.

The classes are completely mutually exclusive. There is no overlap between automobiles and trucks. "Automobile" includes sedans, SUVs, things of that sort. "Truck" includes only big trucks. Neither includes pickup trucks.

For training of this model and dataset the number of epochs was chosen to be 10 while if when not getting sufficient result the number of epochs was increases to 25. And also the batch size of each epoch was 32.

PROBLEM FACED:

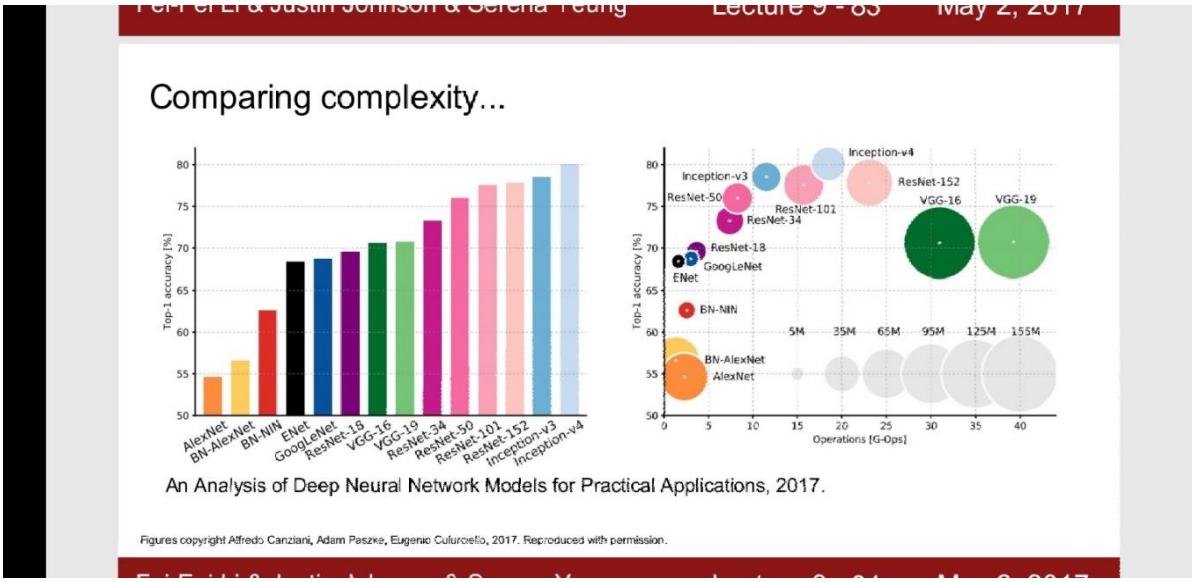
1. The dataset to choose to train: The dataset on which the model is to be trained shouldn't be very specific or short that it doesn't gives much variations to the model. And also it shouldn't be very much complex that it makes algorithm very much bulky.
2. To choose which type of Resnet network to use: I took two parameters while deciding the type of network or the depth of Resnet network to choose. The network should not be too small that it becomes inefficient to capture the difference in the performance and it should not be too bulky that it takes huge time to execute.
3. To decide the number of trainable layers: While training a deep network like Resne-50 through transfer learning. It becomes very important to decide the number of trainable layers in the model and correspondingly the number of layers to be frozen.

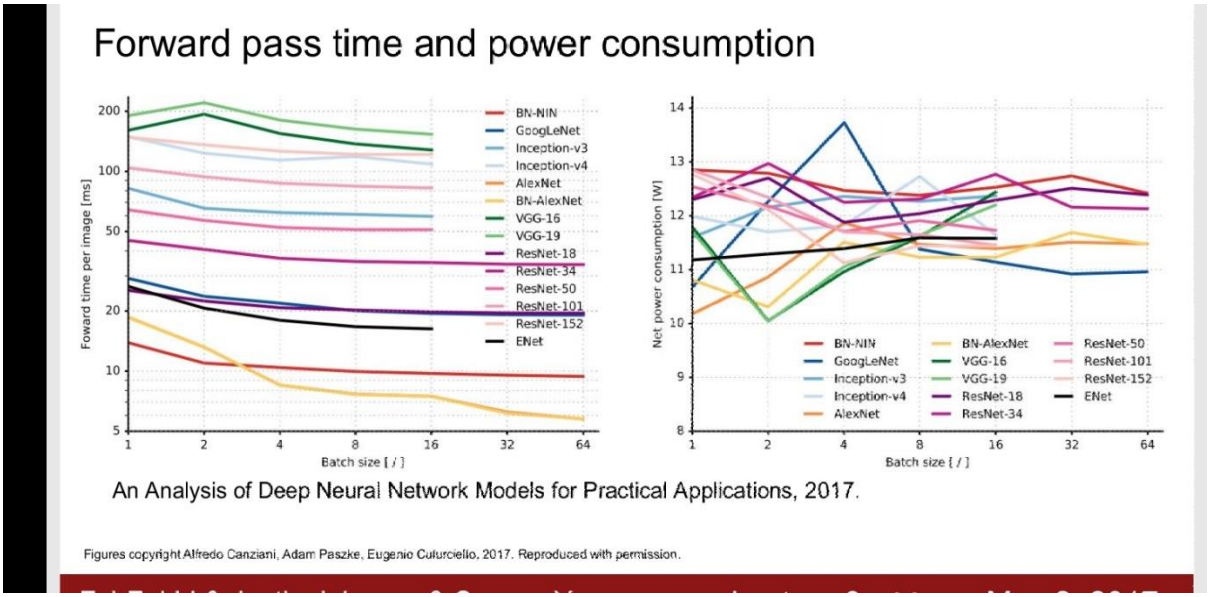
HOW TO OVERCOME THEM:

1. As mentioned above about the requirements of type of dataset required for this comparison process. Preference was **CIFAR-10** dataset because they not very huge as they have only ten classes to classify and also they are not very small as they have in total 60000 images where 6000 belongs to each class.

2. There are various types of ResNets available on the basis of number of layers like Resnet 34 B, Resnet 34 C, ResNet 50, Resnet 101, ResNet 152 and more. The requirement was to use a network which neither too short nor too bulky so the **ResNet 50** architecture was found appropriate. Its architecture is shown in diagram.

   Here, are some statistics to show the comparative study of various models.





This was shown in one of the lectures by **STANFORD UNIVERSITY** on **Computer Vision.**

In their Lecture they draw some conclusions for the models which are as follow.

VGG: Highest memory, most operations.

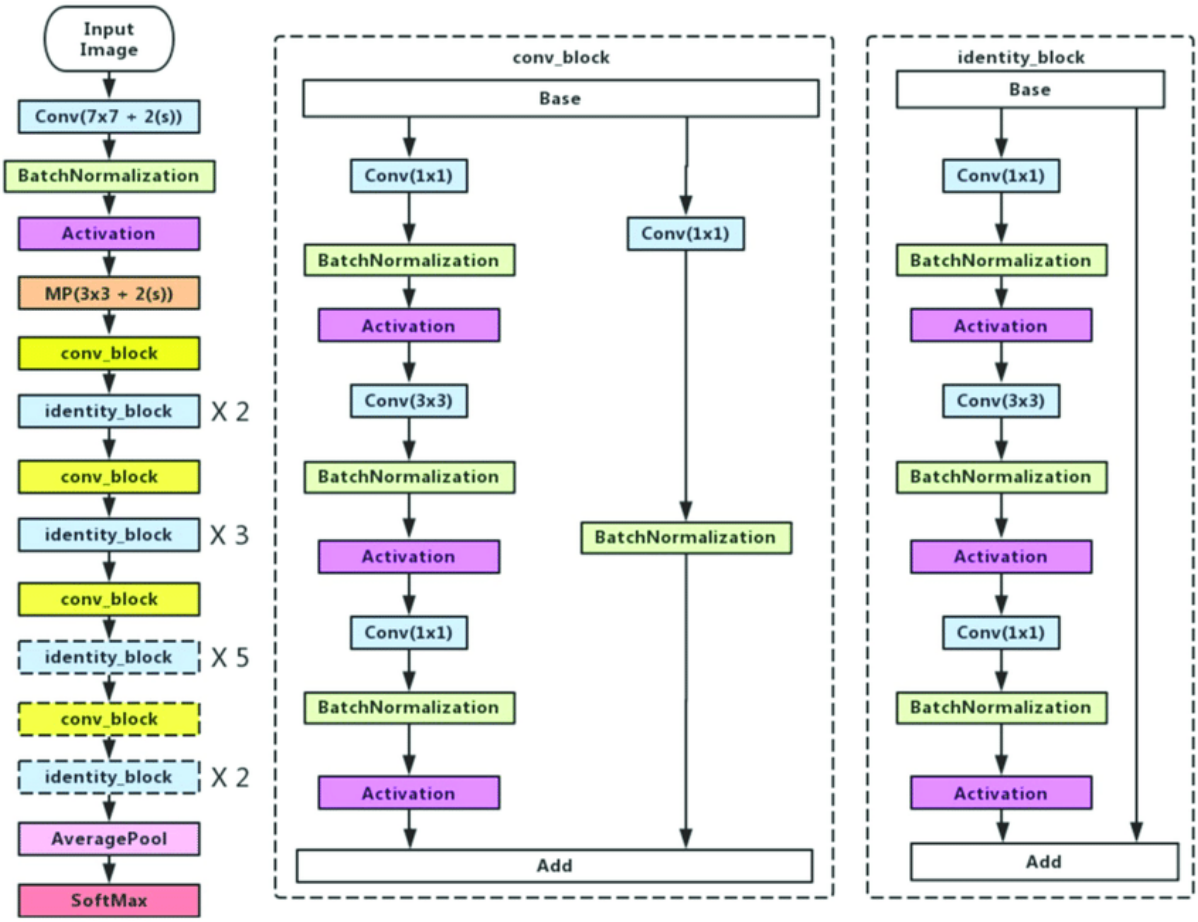
GoogleLeNet: Most efficient.

AlexNet: Smaller compute, still memory heavy, lower accuracy.

ResNet: Moderate efficiency depending on model, highest accuracy.

So, on basis of statistics shown, ResNet-50 architecture was chosen.

3. The first solution problem was found in Deep Learning Course offered on Coursera. In that course the instructor Mr Andrew NG tells that there is an inverse relationship in the amount of dataset on which the model is trained and the number of trainable layers. In this model we model we **freeze all layers except the last layer.**



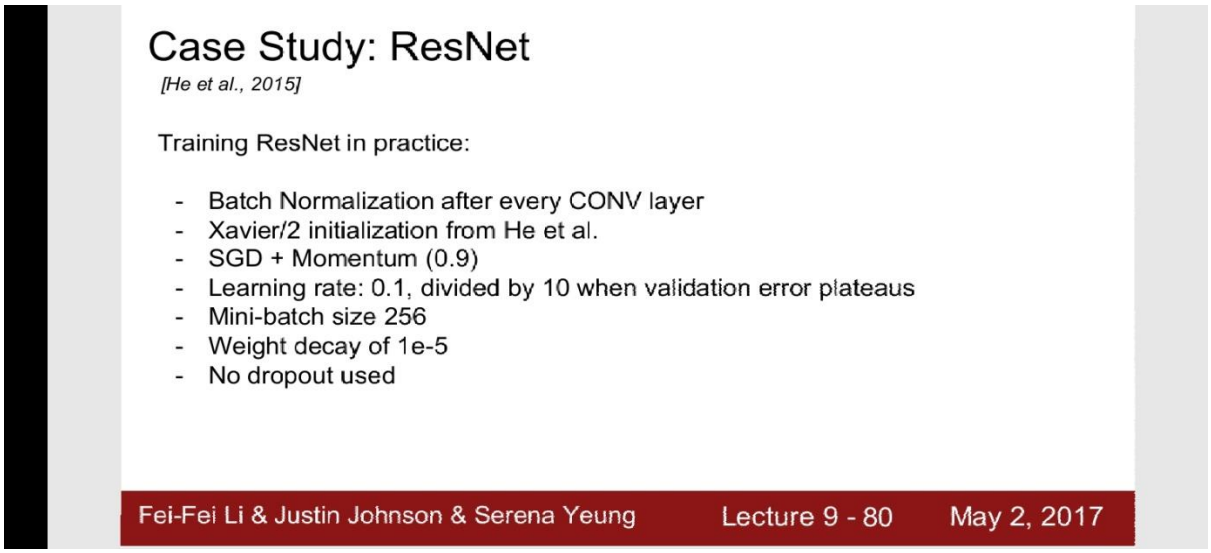
COMPARISION:

Now, the main part of this experiment is the comparison of the optimisers on the chosen model that is ResNet-50 and chosen dataset that is the CIFAR-10 dataset.

The following data are after the completion of ten epochs. While in Adadelta cases of 25 epochs is also taken due to not getting sufficient results.

One of the recommendation from STANFORD UNIVERSITY is shown here, and in the upcoming comparison table a detailed analysis is done on this..

Here, SGD with some conditions is recommended. So, the result after applying this on the given model and dataset in shown in the table.



Optimiser name	Hyper-Parameters values	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
Adam	Lr = 0.001 beta_1 = 0.9 beta_2 = 0.999 epsilon = 1e-07	0.7352	0.7732	0.8469	0.7259
Adagrad	Lr = 0.001 In_ac_va = 0.1 epsilon = 1e-07	1.6748	0.4274	1.3305	0.5856

Adamax	Lr = 0.001 beta_1 = 0.9 beta_2 = 0.999 epsilon = 1e-07	0.6369	**0.8172**	0.8748	0.7274
RMSprop	Lr = 0.001 rho = 0.9 mom = 0.0 epsilon = 1e-07	0.6341	**0.8193**	0.9116	0.7328
SGD	Lr = 0.01 mom = 0.0 (Default)	0.8099	0.7532	0.8880	0.7174
	Lr = 0.1 Mom = 0.9 (for 10 and 25 epochs respectively)	0.9375 0.6158	0.7158 **0.8179**	0.9766 1.0272	0.6984 0.7129
	Lr = 0.01 Mom = 0.9	0.7577	0.7711	0.8595	0.7245
	Lr = 0.001 Mom = 0.9	0.8452	0.7421	0.8797	0.7177
Nadam	Lr = 0.001 beta_1 = 0.9 beta_2 = 0.999 epsilon = 1e-07	0.7486	0.7672	0.8364	0.7281
Adadelta	Lr = 0.001 rho = 0.95 epsilon = 1e-07	Two observations are taken one with **10** and **25** numbers of epochs respectively.			
		Using 10 number of epochs			
		2.8353	0.1183	2.2782	0.1528
		2.8937	0.1119	2.3150	0.1425

		Using 25 number of epochs			
		2.6140	0.1511	2.0678	0.2724

Lr = learning rate; mom = momentum; In_ac_va = initial_accumulator_value

MY INSIGHT:

In the convolutional network, used for this optimiser comparison, we feed an image which an array of positive of positive integers ranging from 0 to 255 with dimension equal to the number of pixels in row and columns respectively. Now, this array is passes through a 50 layered Res-net. And last we get a final **softmax** output which gives the algorithm an average idea of pixels intensity distribution and also of different features of a particular image of a given class. This helps the model to learns to generalise the various variations of feature ranging from colour variations, edges, contrast, etc. and apply them to get better results.

Here, the role of the optimiser is to make the model learn or generalise faster and in efficient manner. In technical terms, it helps model to converge the global minimum or to move around the points nearer to the global minimum.

Here, in the above comparison, the optimiser Adadelta comes out to be very slow in this process of convergence. Even after increasing the number of epochs to 25 it was not showing any significant accuracy. So, on seeing its progress epoch number **150-200** would be able to give satisfactory result but it is quite slower as compared to optimiser like RMSprop, Adamax and Adam in this case. These three optimisers mentioned previously are proven to have fast update rule in this case.

CONCLUSION:

From above table we can conclude that **RMSprop, Adamax and Adam** optimised the above model very well, while **Nadam and SGD** were relatively lesser and finally **Adagrad and Adadelta** were not able to optimise the model efficiently and gave poor accuracy in training as well as validation set.

FUTURE WORK:

As for this model we saw RMSprop, Adamax and Adam performed really well. So, the obvious target should be to design such an optimiser which would have approach adopted by these optimisers and should in return amplify them. So, it should have a decaying learning rate along with decaying momentum so that **may goes slow when global optimum is near** and also the update rule **to learn more in beginning and goes relatively slower on maturity** or when it is nearer to the global maximum.

BY : AYUSH ISHAN..