Multi-Class Image Classification

DEEP LEARNING ASSESMENT

FABIO PALLIPARAMBIL

Submitted on
15/01/2021

Table of Contents

1.0 Introduction

In this project we are implementing a deep learning algorithm knows as Convolutional Neural Network (CNN). The Convolutional Neural network is used for image classification and recognition because of its high accuracy, **it was created by Yann LeCunin 1998 and was widely used for handwritten digit recognition** (1). For this project we have been provided with a dataset known as CIFAR-10, It consists of 60,000 32x32 colour images in 10 objects, with 6,000 images per object. There are 50,000 training images and 10,000 test images.

2.0 Exploratory Analysis of CNN and CNN architectures

Deep Learning is a subset of Machine Learning, but Deep learning is highly demanded due to its performance. The main objective of machine learning is to predict the results according to the input data. But Deep learning can predict even though if it is different from the input data, Deep learning is a part of machine learning algorithms, but it refers to multi-layer filter architecture where it extracts useful features. Each layer of the input receives the output data of the previous layer. The advantages of using deep learning algorithms over Machine learning algorithms are Ability to Develop New Functions, Advanced analysis Capabilities, adaptability, transferable, feature engineering not required up to an extent, scales data effectively, Best performance and accurate results.

Deep Learning VS Machine Learning	
Deep Learning	Machine learning
Best Performance	Works Better on small data
Scale data effectively	Financially and computationally cheap
Adaptable and transferable	Easier to interrupt
Feature engineering not required in most scenario	
Ability to develop new functions	

A convolutional neural network is one of the best algorithms used for task such as computer vision, image recognition, image classification and handwritten digit recognition etc.
There are different types of architectures in CNN architectures, but for this project, we are focusing **on AlexNet, ZF-Net, VGG-Net, Network in Network (NiN), Google Net(inception-v1) and ResNeXt-50**. The architecture of a CNN is similar to that of the connectivity pattern of the Neurons in the Human Brain and was inspired by **David H.Hubel** and **Tosten Wiesal** while performing a series of experiments in cats and monkeys in the period between 1958 to 1959(1,PG:446). And this gave insights into the structure of the Visual Cortex.

- AlexNet (2012)
  - It is the advanced version of LeNet-5, the only difference is AlexNet have stacked a few more layer on top LeNet-5. AlexNet is the first stacked Convolutional neural network.
  - AlexNet is made up of 8 to 5 convolutional and 3 fully connected layers. It uses Max Pooling and Rectified Linear Units (ReLu) activation function. It uses 3 max-pooling layers and the architecture had 60 million parameters.
  - To reduce overfitting the authors have used 2 Regularisation techniques, they are **drop out** and **data argumentation**.

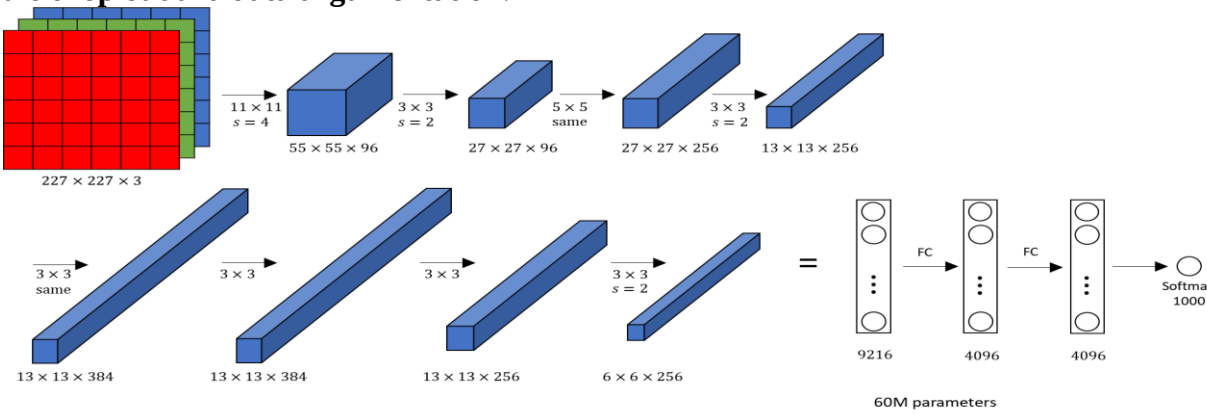


Figure 1 AlexNet (http://datahacker.rs/deep-learning-alexnet-architecture/)

- ZF-Net (2013)
  - ZF-Net was the winner of ILSVRC in 2013 but they displayed as Clarifai which is the company where ZF-Net were developed (7,9). It was invented by Dr Rob Fergus and his PhD Student at that moment, Dr Matthew D.Zeiler in YNU.
  - ZF-Net is a significantly improved model compared to AlexNet (8).
  - It had 96 convolutional layers,3 max-pooling layers, ReLu activation function and local normalization.

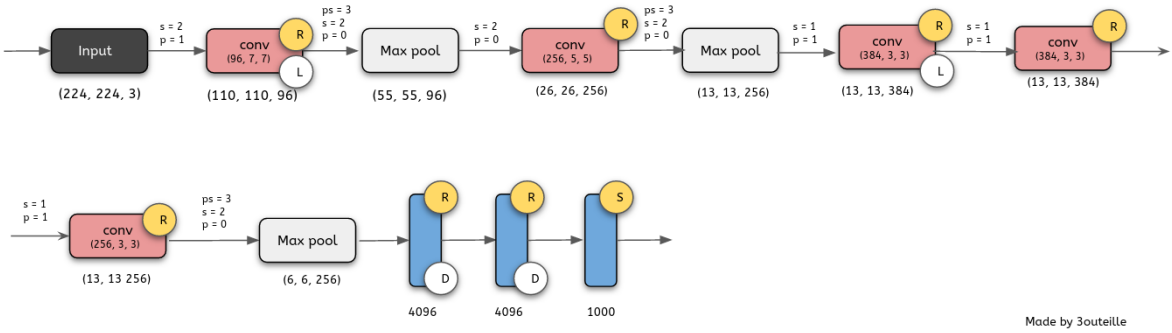


Figure 2 ZF-Net (https://hackmd.io/@bouteille/ByaTE80BI)

  - It uses Deconvnet techniques for visualisation. Deconvnet is a technique used to reverse the deep learning framework(Conv->Activation(Activation Function)>Pooling)such a way that we can visualize the features in pixel(7)

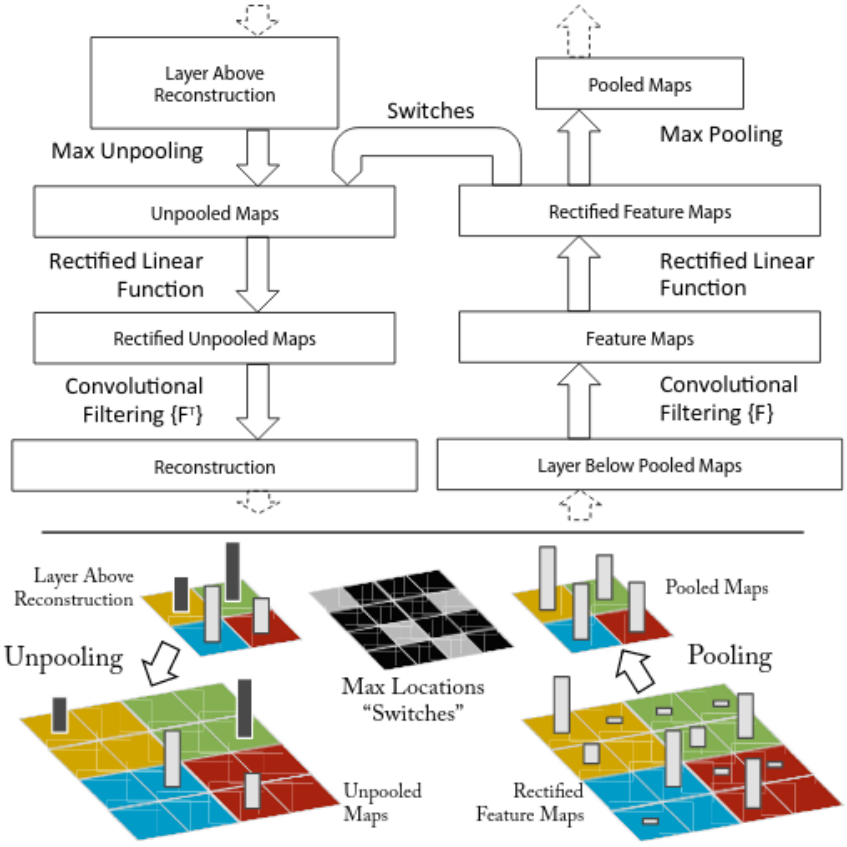


Figure 3 Deconvnet (https://hackmd.io/@bouteille/ByaTE80BI)

- It uses Unpooling because the max pooling operation is non-invertible. But we can obtain an approximate inverse by recording the location of the maxima within each pooling region.

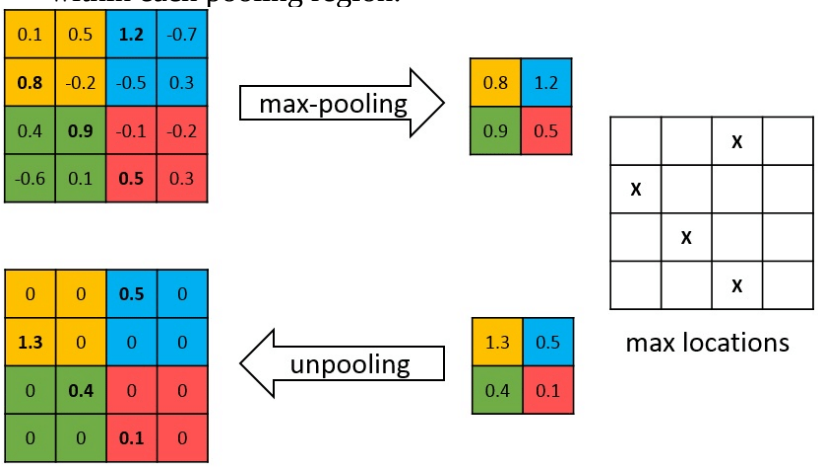


Figure 4 Unpooling (https://www.researchgate.net/figure/Pooling-and-unpooling-layers-For-each-pooling-layer-the-max-locations-are-stored-These_fig2_306081538)

- VGG-net (VGG-16) (2014)
  - It is a much deeper network compared to AlexNet. It uses smaller filters but with more depth to the network. By using more depth, you can achieve the same effective receptive fields as one 7x7 convolutional layer.
  - It has similar training procedures as AlexNet but does not use Local Response Normalisation (LRN). The fully connected layer (FC7) is a good feature

representation that can be used to extract feature from other data and generalize tasks

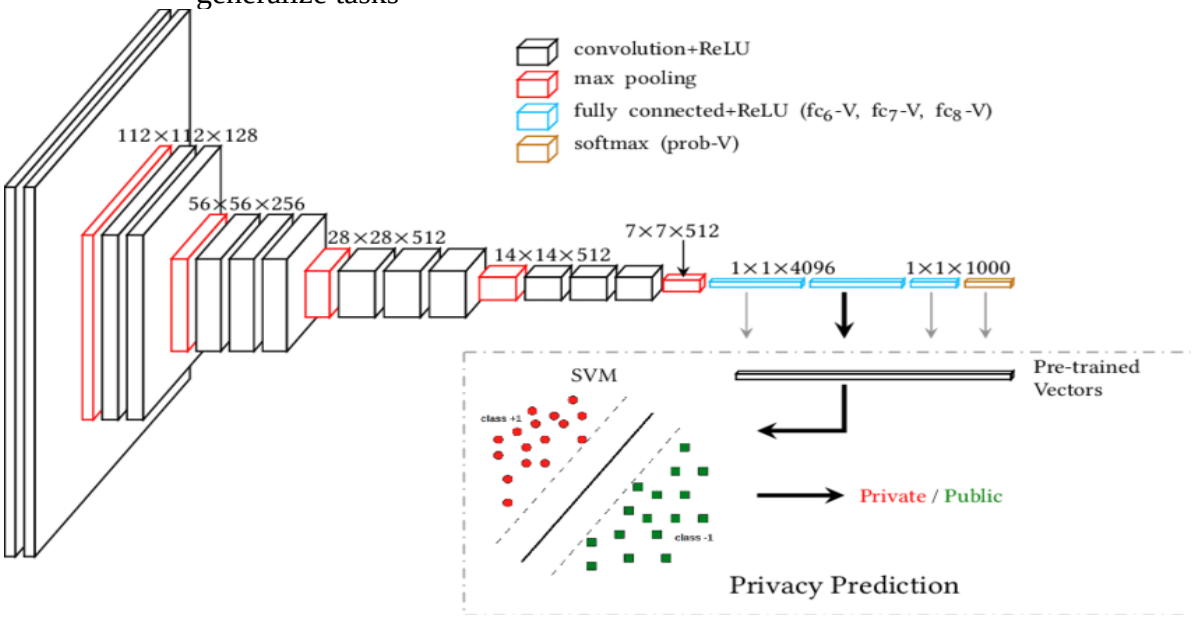
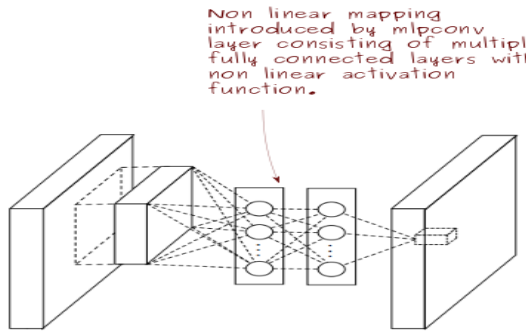


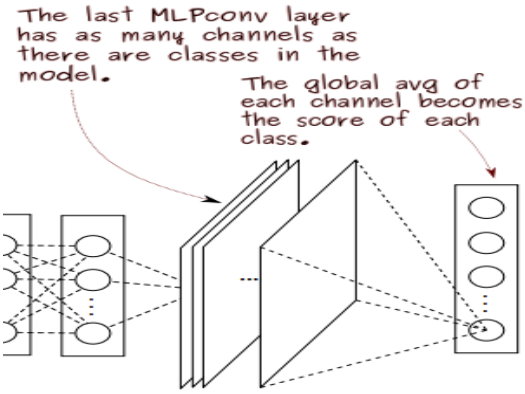
Figure 5 VGG16 (https://www.researchgate.net/figure/Image-encoding-using-pre-trained-CNN-1-We-employ-a-CNN-eg-VGG-16-pre-trained-on_fig2_331670582)

- It uses ensembles for best results. VGG16 has 13 convolutional and 3 fully connected layers. And it has 2 Max pooling layers.
- There are 2 types of VGG- net, they are VGG16 and VGG19. The only difference is VGG19 is very slightly better and it needs more memory. Therefore, VGG16 is mainly used and well known compared to VGG19.

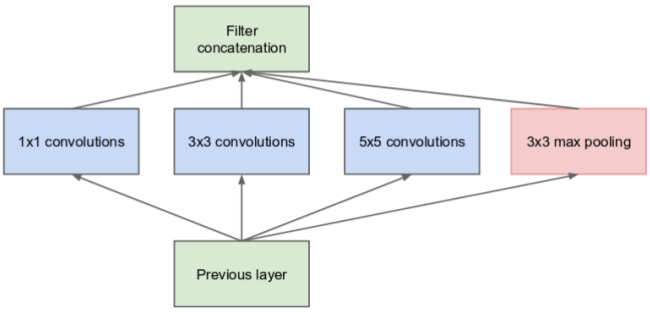
- Network in Network (3)
  - The NiN is inspired by the Inception line of deep architectures from Google. This is the precursor to the Google Net, and ResNet This formed the basis of the Inception architecture.
  - Two new concepts were introduced in this CNN architecture design and they are **MLPconv** and **Global Average pooling** (3).
  - **MLPconv**: Replaced linear filters with nonlinear Multi Linear Perceptron to extract better features within the receipt field. This helps in better abstraction and accuracy. (3)



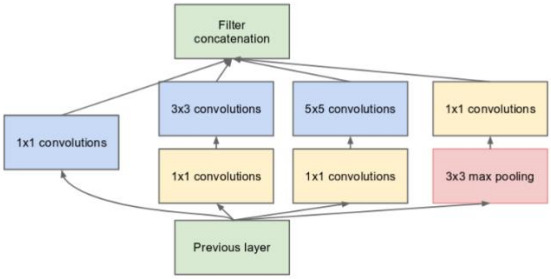
- **Global Average Pooling**: Got rid of the fully connected layers at the end thereby reducing parameters and complexity. This was replaced by the creation of as many activation maps in the last layer as there are classes. This was followed by averaging these maps to arrive at final scores, which is passed to SoftMax. This is performant and more intuitive. (3)



- Google Net (Inception-v1,2014)
  - It was proposed by research at google in 2014 in the research paper titled "Going Deeper with Convolutions" (1).
  - It has provided a significant decrease in error rate as compared to previous winner AlexNet and ZF-Net and significantly less error rate than VGG.
  - The 22-layer architecture with 5M parameters is called the Inception-**v1**. But there is a more improved version, and they are Inception-v3(2015), Inception-v4(2016) and also Inception-Resnet-V2(2016). (6)



(a) Inception module, naïve version



(b) Inception module with dimension reductions

- In inception-v1 has Designed a good local network topology ("network in network (NIN)") and then stacked these modules on top of each other. They have designed the architecture with efficient inception modules. (1)
- It Applies parallel tower of a convolutional network with a different filter. This idea is motivated by Arora et al in from the paper "*Provable bounds for learning some deep representations*". (5)
- it had a pooling operation of 3x3.

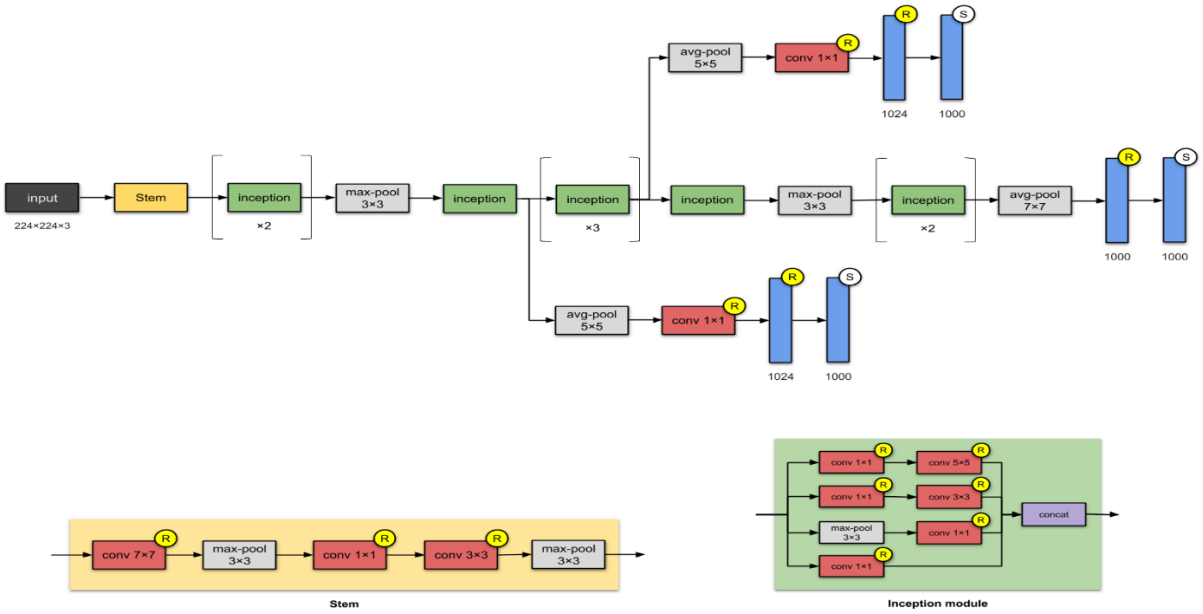


Figure 6 Inception-v1

- ResNeXt-50 (2017)
  - ResNet was developed by UC San Diego and Facebook AI Research. Its the next dimension on top of the ResNet, this next dimension is called a cardinality dimension.
  - It contrasts to in Network-in-Network it is Network -in- neuron, it expands along a new dimension instead of a linear function. A non-linear function is performed for each path.
  - The dimension of cardinality controls the number of more complex transformation.
  - It has 25 M parameters and compared to ResNet, it is made by adding parallel tower(cardinality), branches or path within a module. And the ResNext-50 has a total of 32 towers(12).

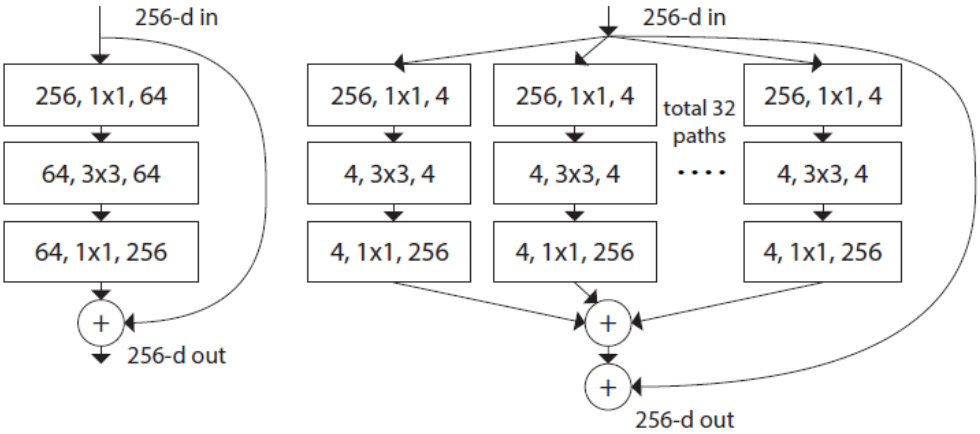


Figure 7 ResNeXt-50 (https://towardsdatascience.com/review-resnext-1st-runner-up-of-ilsvrc-2016-image-classification-15d7f17b42ac)

3.0 Implemented CNN Analysis

The CNN implemented for this project is aspired by Alex-Net Architecture. I have used the method of stacking multiple Convolutional layers, but my main aim was to create a simple, efficient and less computationally intense Neural network for the prediction of the CIFAR-10 dataset. The implemented CNN consist of 6 Conv2D Layers,3Max Pooling Layer, 6 Batch normalisation layers, 3 Drop out layers and 2 fully connected layers. The activation function used is ReLu for all the Conv2D layer, but for the final fully connected layer SoftMax activation have been used as its performing multi-classification task. Padding has been applied to all the Convolutional layer. The weight and bias have been initialised as well, the parameters chosen for weight is he_uniform (kernel initializer = "he_uniform") and for bias is initialised as zero (bias initializer = "zero").

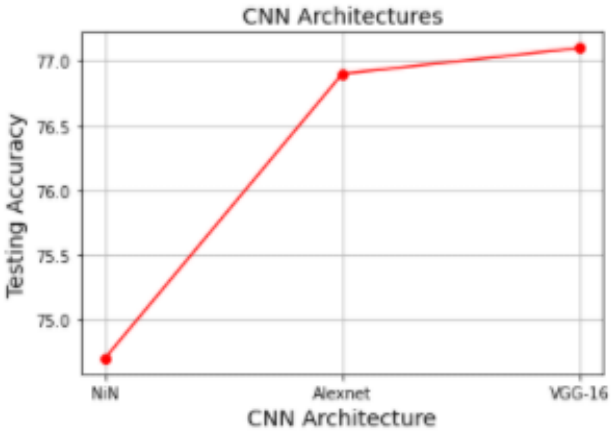
4.0 Empirical evaluation of the implemented CNN

There are multiple experiments done to explain the reason behinds the selected parameters for CNN.

4.1 CNN Architectures-Experiment 1

CNN architectures are different models created by different people and It has been explained in section 2.0. For this experiment, I choose AlexNet, NiN and VGG-16. According to the results achieved it shows VGG-16 and AlexNet performed nearly the same regarding accuracy. But for this project, I decided to go ahead with AlexNet. The main reason is AlexNet is simple and small model compared to VGG16, but AlexNet had nearly the same accuracy score and it can be less computationally expensive.

Different CNN Architecture		
	Training Accuracy(%)	Testing Accuracy(%)
AlexNet	76.9	76.9
VGGNet	86.7	77.1
Network in Network	95.7	74.7



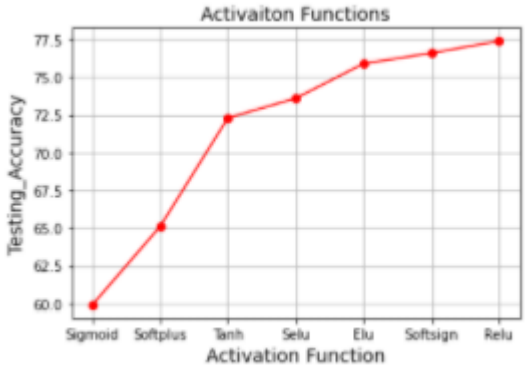
Therefore, my choice is **AlexNet** for the CIFAR-10 dataset.

4.2 Activation Function - Experiment 2

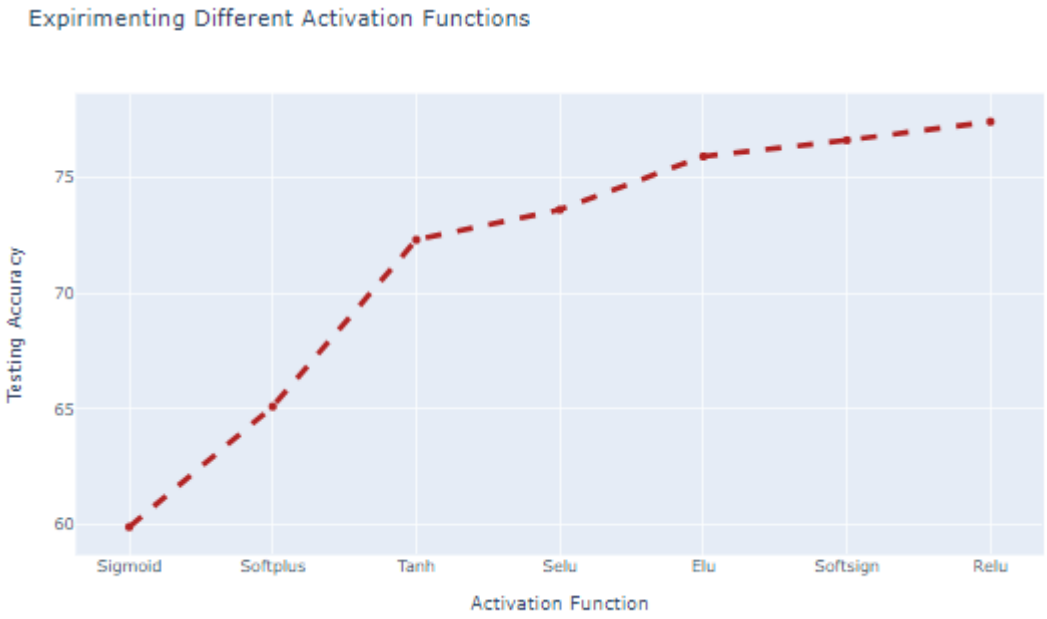
An activation function is a function that is added into an artificial neural network to help the network to learn complex patterns in the data(13). According to the results achieved during the experiment, it shows that **ReLu** function had the highest accuracy. ReLu function is one of the widely used and well-known activation functions.

.

Activation function		
	Training Accuracy(%)	Testing Accuracy(%)
ReLu	97.5	77.4
Selu	95.1	75.9
Elu	93.2	73.6
Softsign	94.0	76.6
Softplus	92.2	65.1
Tanh	93.3	72.3
Sigmoid	64.3	59.9

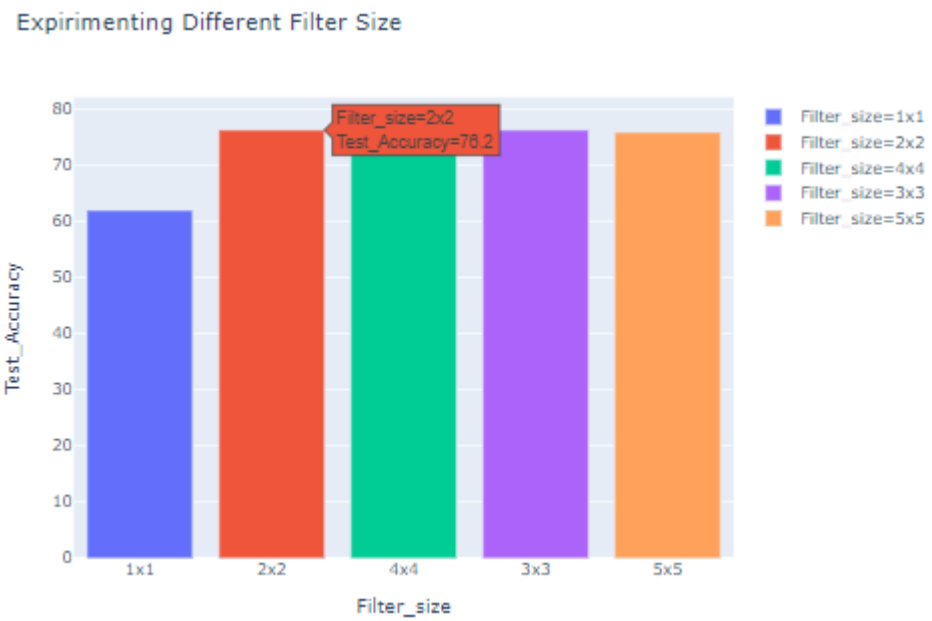


.



4.3 Filter Sizes - Experiment 3

In the Convolutional neural network, multiple filters are taken to slice through the image and map them one by one and learn different portions of an input image. Filters are located between the feature map and input image. According to the experiments conducted it shows that 5x5 filter provides slightly better accuracy regard to 3x3. But I have decided to 3x3 because it provides nearly the same accuracy and using a smaller filter size it will help to reduce computational cost and time(14).



Filter Sizes		
	Training Accuracy(%)	Testing Accuracy(%)
1x1	69.3	61.9
2x2	89.4	76.2
3x3	94.8	77.9
4x4	94.7	76.6
5x5	94.5	75.8

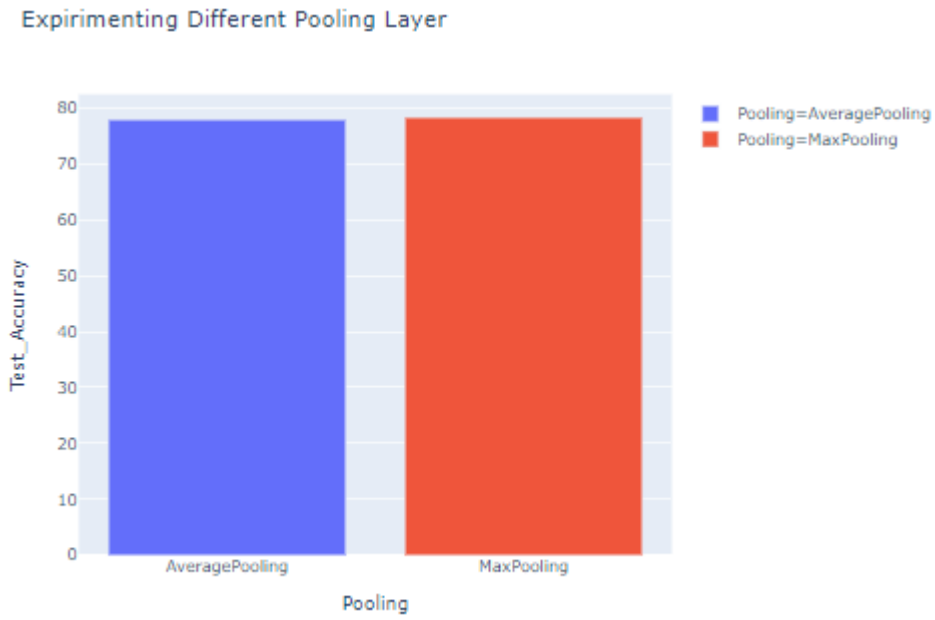
4.4 Filter Numbers - Experiment 4

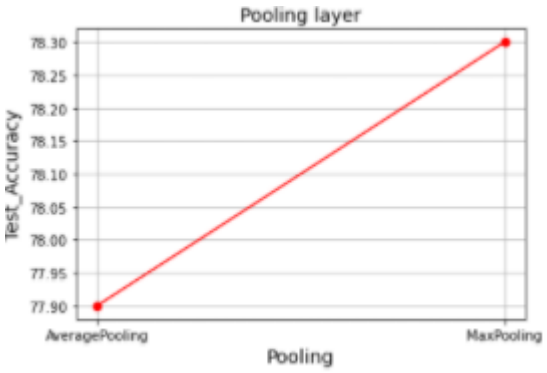
I choose to go with 32 as the number of the filter as the accuracy is nearly the same.

Number of Filter		
	Training Accuracy(%)	Testing Accuracy(%)
32	95.11	77.9
64	95.59	77.9
128	96.9	79.9

4.5 Pooling Layers - Experiment 5

Pooling layer is a layer added after the convolutional layer, specifically after a nonlinearity(eg: ReLu) has been applied to the feature maps output by a convolutional layer. **Pooling is required to downsample the detection of features in feature maps**. Pooling layer will always reduce the size of each feature map by a factor of 2. The 2 common pooling functions are **Average pooling** and **Maximum Pooling**.





Pooling Layers		
	Training Accuracy(%)	**Testing Accuracy(%)**
Average Pooling	**91.7**	**77.9**
Max Pooling	**94.54**	**78.3**

4.6 Loss Function - Experiment 6

Loss is nothing but a prediction error of Neural Net. And the method to calculate the **loss** is called **Loss Function**. Entropy is the measure of uncertainty in a certain distribution, and cross-entropy is the value representing the uncertainty between the target distribution and the predicted distribution. Sparse Categorical Cross Entropy is versions of Binary Cross-Entropy, adapted for several classes. Sparse categorical cross-entropy should be used when the classes are mutually exclusive(16).

Figure 8 Hinge

Figure 9 squared hinge

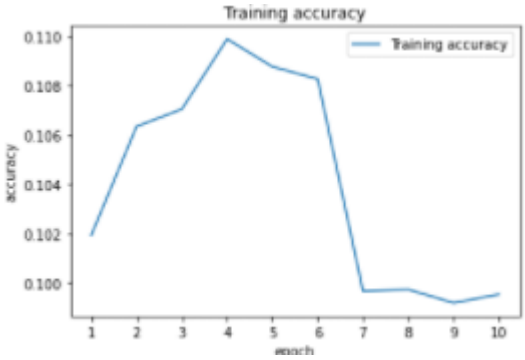


Figure 10 log cosh

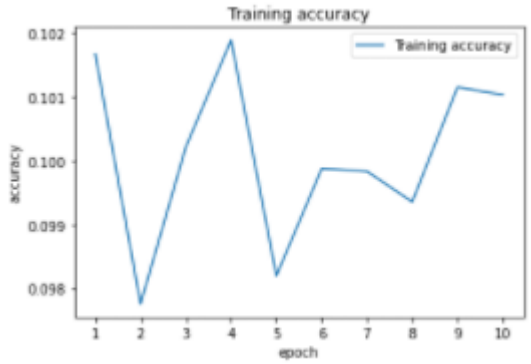


Figure 11mean_squared_logarithmic_error

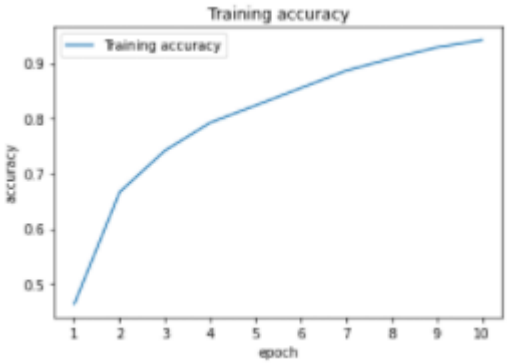


Figure 12 sparse_categorical_crossentropy

As you can see from the results of the experiments sparse_categorical_crossentropy is the best loss function to be used in the model.

4.7 Regularization Methods - Experiment 7

Regularization is a technique which makes slight modifications to the learning algorithm such that the model generalizes better. This in turn improves the model's performance on the unseen data as well. It helps to reduce overfitting of the model which will lead to more accurate predictor model.

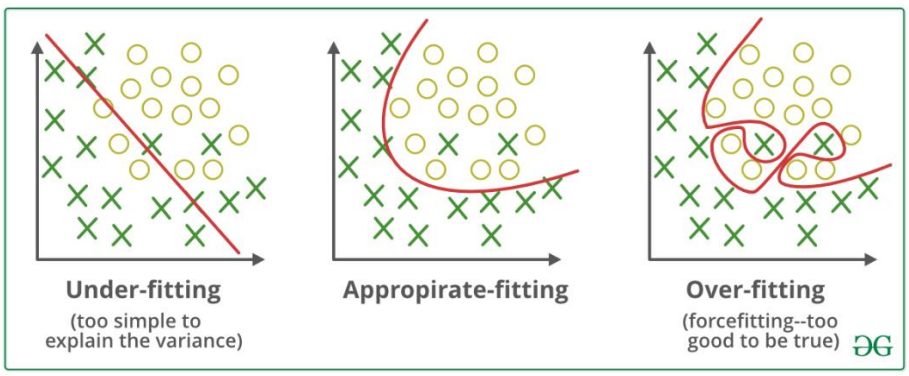


Figure 13 https://www.geeksforgeeks.org/regularization-in-machine-learning/

The 3 types of regularisation methods **are L2Regularisation, drop out and Data Argumentation.L2 regularization** is also known as weight decay as it forces the weights to decay towards zero.

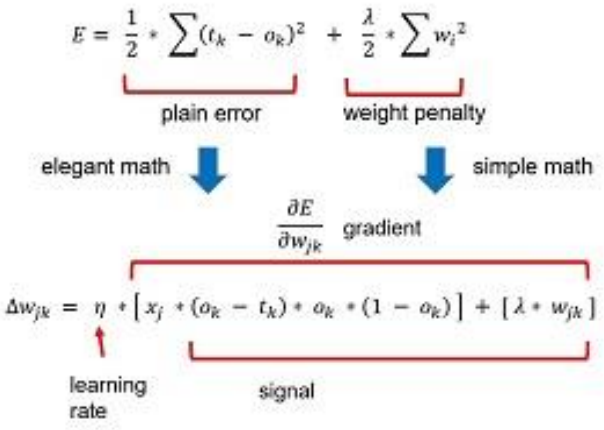


Figure 14 L2 Regularisation technique https://visualstudiomagazine.com/articles/2017/09/01/neural-network-l2.aspx

Drop out is a technique where randomly selected neurons are ignored during training.

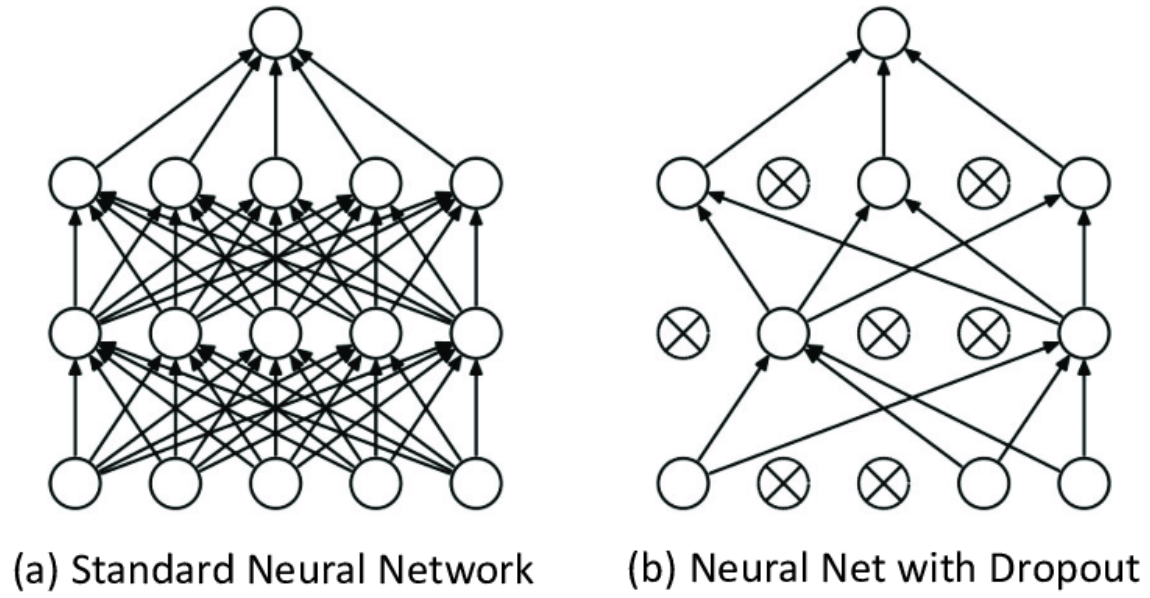


Figure 15 Drop Out https://www.researchgate.net/figure/4-An-illustration-of-the-dropout-mechanism-within-the-proposed-CNN-a-Shows-a_fig27_317277576

Data argumentation is a technique to artificially create new training data from existing training data.

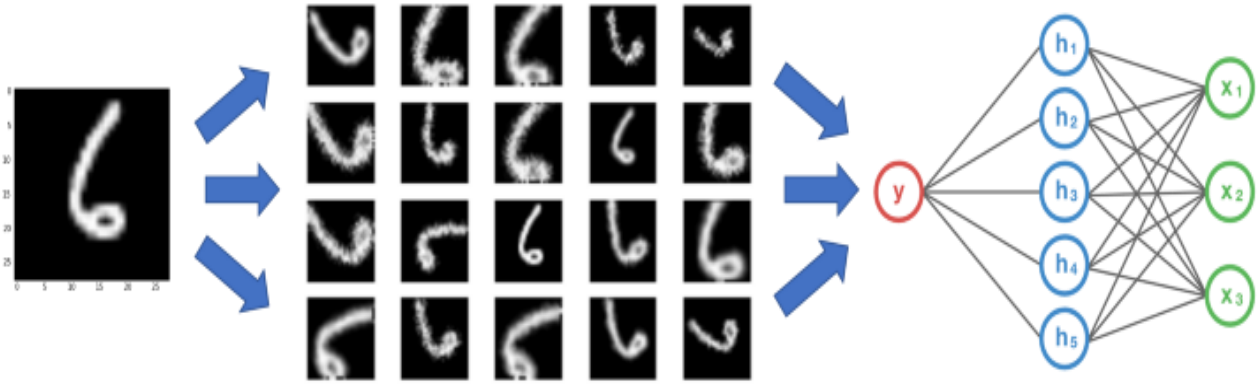
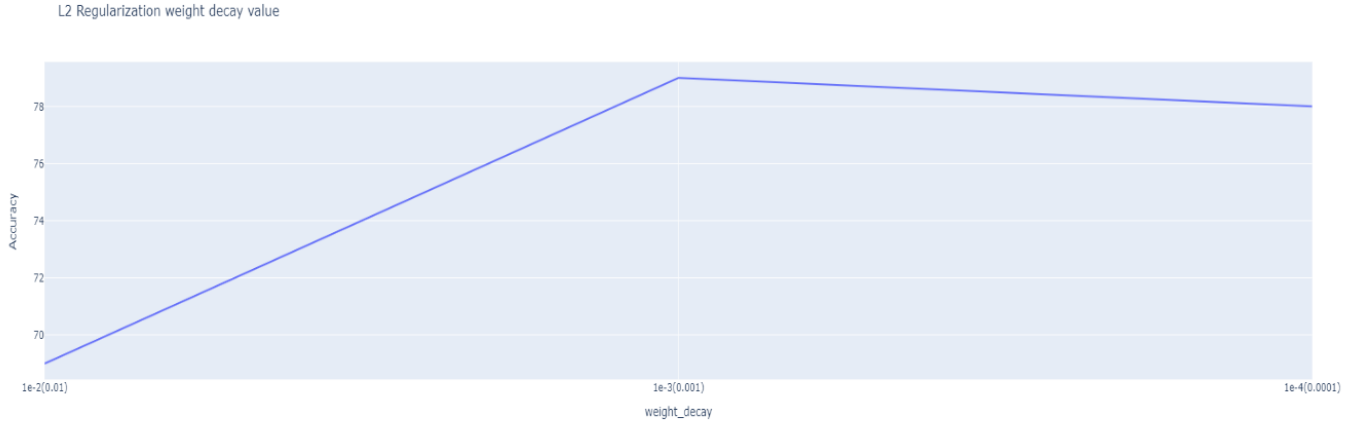


Figure 16 Data argumentation https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/

According to the experiments conducted it shows that regularisation techniques help to significantly increase the performance of the model. But as you can see Data argumentation did not perform well. Therefore, I have selected L2 regularisation and drop out techniques in my CNN model.

Regularisation Methods		
	Training Accuracy(%)	Testing Accuracy(%)
L2 Regularisation	86.6	79.6
Drop out	81.3	80.1
Data Argumentation	69.7	73.8

Experiment with Drop Out	
	Accuracy (%)
Using same drop rate	78
Using drop rate incrementally	80



Methods to reduce Overfitting		
	Training Accuracy(%)	Testing Accuracy(%)
Earlier Stopping	80.5	79.5
Weight initialisation	80.8	78.9
Batch Normalisation	81.6	80.6

4.8 Earlier Stopping - Experiment 8

Early stopping is a method that allows you to specify an arbitrarily large number of training epochs and **stop** training once the model performance stops improving on a holdout validation dataset. According to our experiment conducted it shows that **Early stopping** helps the model to perform a little better(Refer figure above)

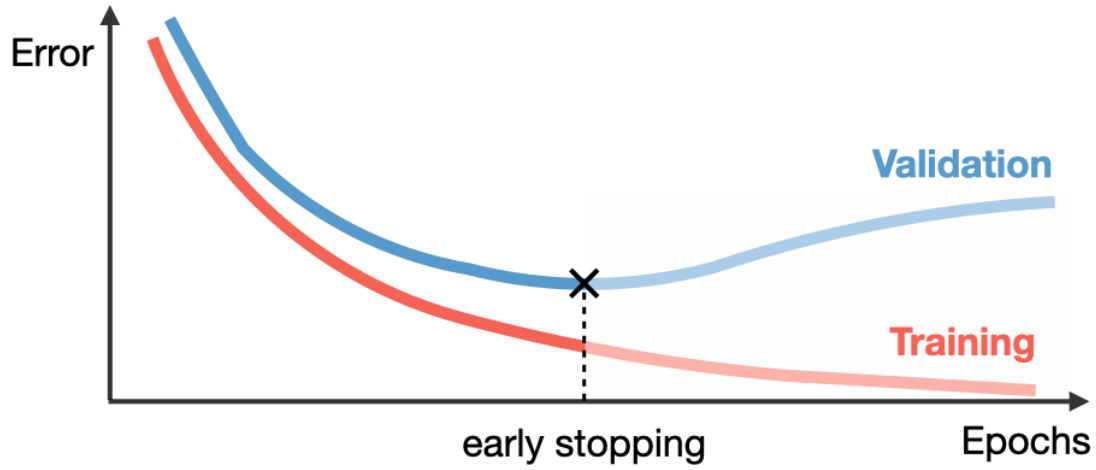


Figure 17 Early stopping https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-deep-learning-tips-and-tricks

4.9 Weight Initialisers - Experiment 9

Weight initialization aim is to prevent layer activation outputs from exploding or vanishing during a forward pass through a deep neural network. According to our experiment conducted it shows that weight initialisation helps the model to perform a little better(Refer figure above).

$$\pm \frac{\sqrt{6}}{\sqrt{n_i + n_{i+1}}}$$

Figure 18 https://towardsdatascience.com/weight-initialization-in-neural-networks-a-journey-from-the-basics-to-kaiming-954fb9b47c79

4.10 Batch Normalisation - Experiment 10

Batch normalization is a technique for training very deep neural networks that standardizes the inputs to a layer for each mini batch. This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks(15). According to our experiment conducted it shows that Batch normalization helps the model to perform a little better(Refer figure above)

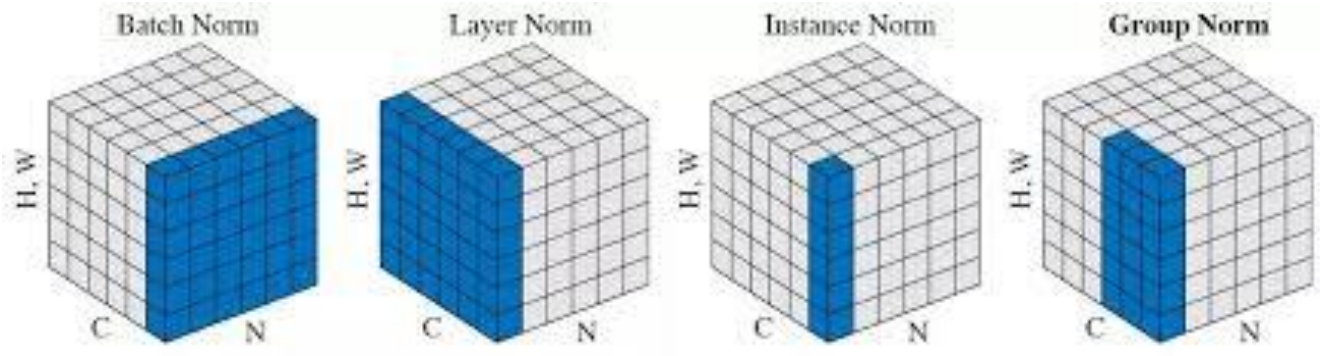


Figure 19 https://medium.com/syncedreview/facebook-ai-proposes-group-normalization-alternative-to-batch-normalization-fb0699bffae7

4.11 Gradient Descent - Experiment 11

As you can see the result mini-batch gradient performed better. And bigger batch size is better than smaller batch sizes, the sweet spot in our experiment is 390.6 per mini batch (batch size = 128).

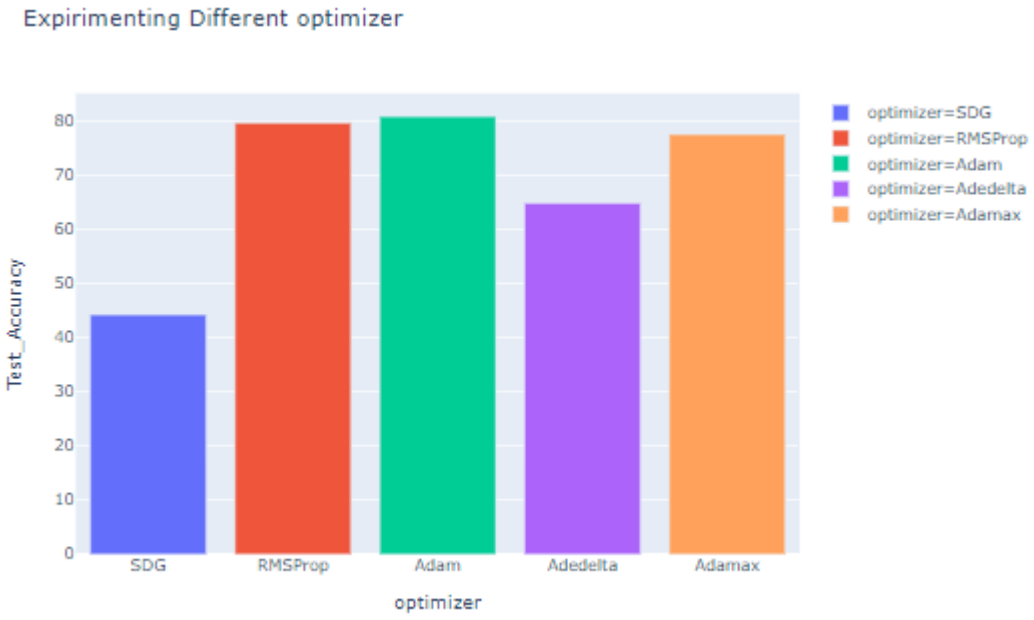
Gradient Descent		
	Training Accuracy(%)	**Testing Accuracy(%)**
Batch Gradient Descent	**78.8**	**79.6**
Mini – Batch Gradient Descent	**81.1**	**80.5**

mini batch gradient decent	
	Accuracy (%)
Batch Size=64	79
Batch Size=128	81
Batch Size=1024	71

4.12 Optimisation Algorithms - Experiment 12

Optimizers are algorithms used to change the attribute of the neural network such as weights and learning rate to reduce the loss. Reduction in losses will result in more accurate results possible. Some of the optimization algorithms are listed below.

Experimenting Different Optimizers		
Learning rate = 0.001	Training Accuracy(%)	Testing Accuracy(%)
SDG	43.5	44.2
RMSProp	80.1	79.6
Adam	80.8	80.8
Adadelta	65.3	64.8
Adamax	77.7	77.5



According to the results achieved during the experiment, it shows that Adam optimizer with learning rate= 0.001 and epsilon = 0.001 performed better compared to others. Therefore, I will be choosing Adam as the optimizer for the model(More in Appendix).

5.0 Conclusion

According to all the experiments conducted on the multiclass image classification model, it proves that the hyperparameters tuning in Convolutional neural network are playing a big role in its performance and accuracy of the models. The architecture in this experiment is aspired by AlexNet, due to its simple, fast and computationally efficiency. The Hyperparameters tuned are filter sizes, padding, activation function, drop out weight, bias, L2Regularisation and Batch normalisation. The model implemented has an overall 81% prediction accuracy for multi-image classification. In future, I would try more experiments on different types of architecture, more diversified learning rates and data argumentation by reducing the data set. This can push the limits to get an insight into how the model would react to these tests and will experiment with multiple hyperparameter tuning for higher accuracy of the model.

6.0 Reference

1. O'REILLY, **Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow**, Aurelian Geron, 2019,[Book]
2. Towards data science, Difference between local response normalization and batch normalisation, Available at[Online]https://towardsdatascience.com/difference-between-local-response-normalization-and-batch-normalization-272308c034ac
3. Teleported.in , available at[online] : http://teleported.in/posts/network-in-network/
4. Towards data science , Illustrate 10 CNN architectures availbleat['online']https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d
5. Towards data science, A simple guide to the version of inception network, available at[Online]: https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202
6. Medium ,Review: ZFNet — Winner of ILSVRC 2013 (Image Classification) available at [online] : https://medium.com/coinmonks/paper-review-of-zfnet-the-winner-of-ilsvlc-2013-image-classification-d1a5a0c45103
7. Towards data science, squeeze and excitation network, available at ['online']https://towardsdatascience.com/squeeze-and-excitation-networks-9ef5e71eacd7

8. Towards data science, everything you need to know about activation function in depth ,Available at[Online]: https://towardsdatascience.com/everything-you-need-to-know-about-activation-functions-in-deep-learning-models-84ba9f82c253
9. Medium ,how to choose the size of VNN filter, available at[Online]https://medium.com/analytics-vidhya/how-to-choose-the-size-of-the-convolution-filter-or-kernel-size-for-cnn-86a55a1e2d15#:~:text=Read%20More-,How%20to%20choose%20the%20size%20of%20the,or%20Kernel%20size%20for%20CNN%3F&text=Basically%2C%20We%20divide%20kernel%20sizes,till%205x5%20for%202D%20Convolution.

7.0 Appendix

4.12.1 SDG

Experimenting SDG optimizer by tuning different parameters	
	Accuracy (%)
Normal	61.1
Learning rate = 0.01	57.3
Learning rate = 0.001	44.2
Learning rate = 0.0001	27.6
Momentum = 0.2	65.1
Momentum = 0.6	71.9
Momentum = 0.9	79.1
The parameters to choose are learning rate = 0.01 and momentum = 0.9	

4.12.2 RMSProp

Experimenting RMSprop optimizer by tuning different parameters	
	Accuracy (%)
Normal	79.4
Learning rate = 0.01	60.2
Learning rate = 0.001	80.1
Learning rate = 0.0001	62.4
Momentum = 0.2	80.2
Momentum = 0.6	79.5
Momentum = 0.9	63.4
The parameters to choose are learning rate = 0.001 and momentum = 0.2	

4.12.3 Adam

Experimenting Adam optimizer by tuning different parameters	
	Accuracy (%)
Normal	81.2
Learning rate = 0.01	66.5
Learning rate = 0.001	80.4
Learning rate = 0.0001	59.8
epsilon = 0.01	77.0
epsilon = 0.001	81.5
epsilon = 0.0001	80.1
The parameters to choose are learning rate = 0.001 and epsilon = 0.001	

4.12.4 Adadelta

Experimenting Adadelta optimizer by tuning different parameters	
	Accuracy (%)
Normal	22.3
Learning rate = 0.01	37.5
Learning rate = 0.001	20.3
Learning rate = 0.0001	11.8
epsilon = 0.01	61.2
epsilon = 0.001	63.9
epsilon = 0.0001	63.3
The parameters to choose are learning rate = 0.01 and epsilon = 0.001	

4.12.5 Adamax

Experimenting Adamax optimizer by tuning different parameters	
	Accuracy (%)
Normal	77.6
Learning rate = 0.01	73.2
Learning rate = 0.001	78.1
Learning rate = 0.0001	50.4
epsilon = 0.01	74.2
epsilon = 0.001	79.2
epsilon = 0.0001	76.6
The parameters to choose are learning rate = 0.001 and epsilon = 0.001	