3 Layer Deep Neural Network

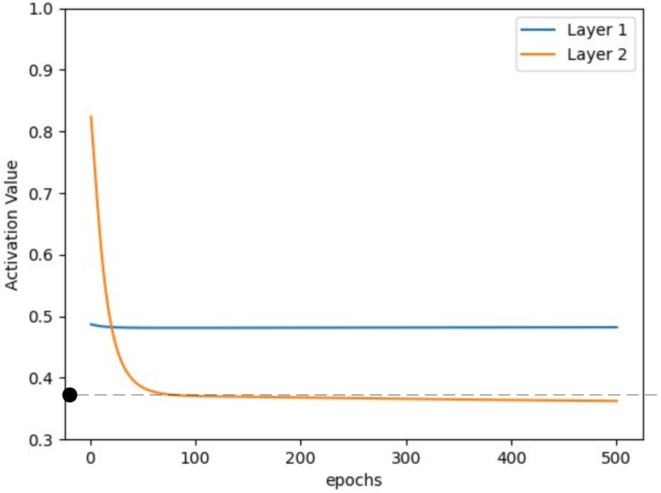
Activation Function	epochs	Xavier Initilization		He Initialization		Kumar Initialization	
		Training accuracy	Testing accuracy	Training accuracy	Testing accuracy	Training accuracy	Testing accuracy
Sigmoid	3000	55.13%	66.0%	54.33%	66.0%	59.89%	76.0%
relu	3000	82.25%	74.0%	86.29%	78.0%		

Deep Neural Network

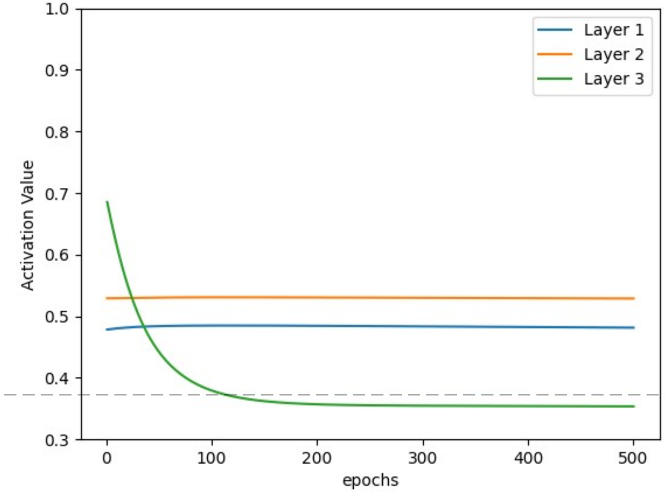
Activation Function	Layers	epochs	Learning rate		
				Training accuracy	Testing accuracy
Sigmoid	2	500	0.1	57.92%	64.0%
	3	500	0.1	55.64%	70.0%
	5	500	0.1	52.32%	66.0%

Deep Neural Network

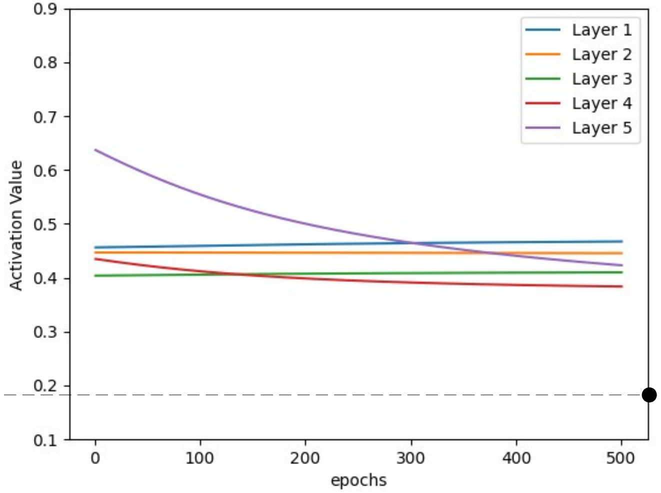
Sigmoid



Sigmoid



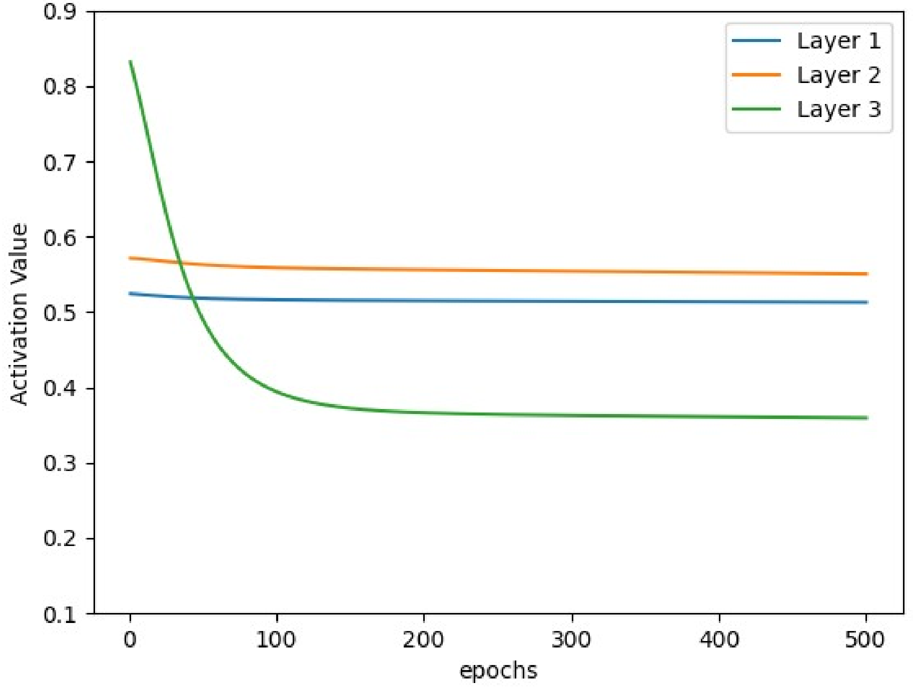
Sigmoid



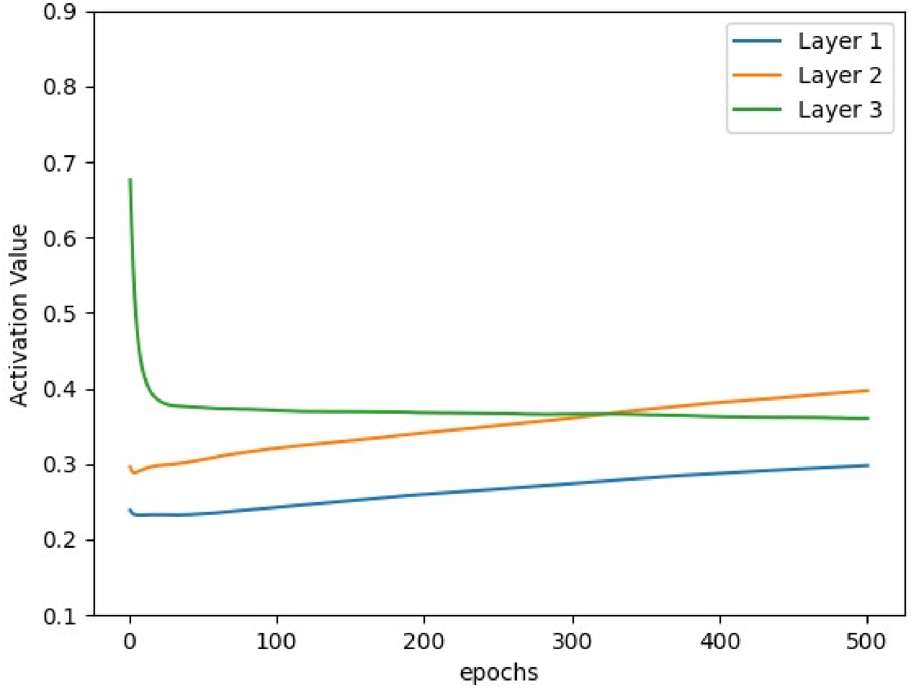
Deep Neural Network

Activation Function	Layers	epochs	Learning rate		
				Training accuracy	Testing accuracy
Sigmoid	3	500	0.1	56.32%	64%
relu	3	500	0.1	62.63%	74%

Deep Neural Network



Sigmoid

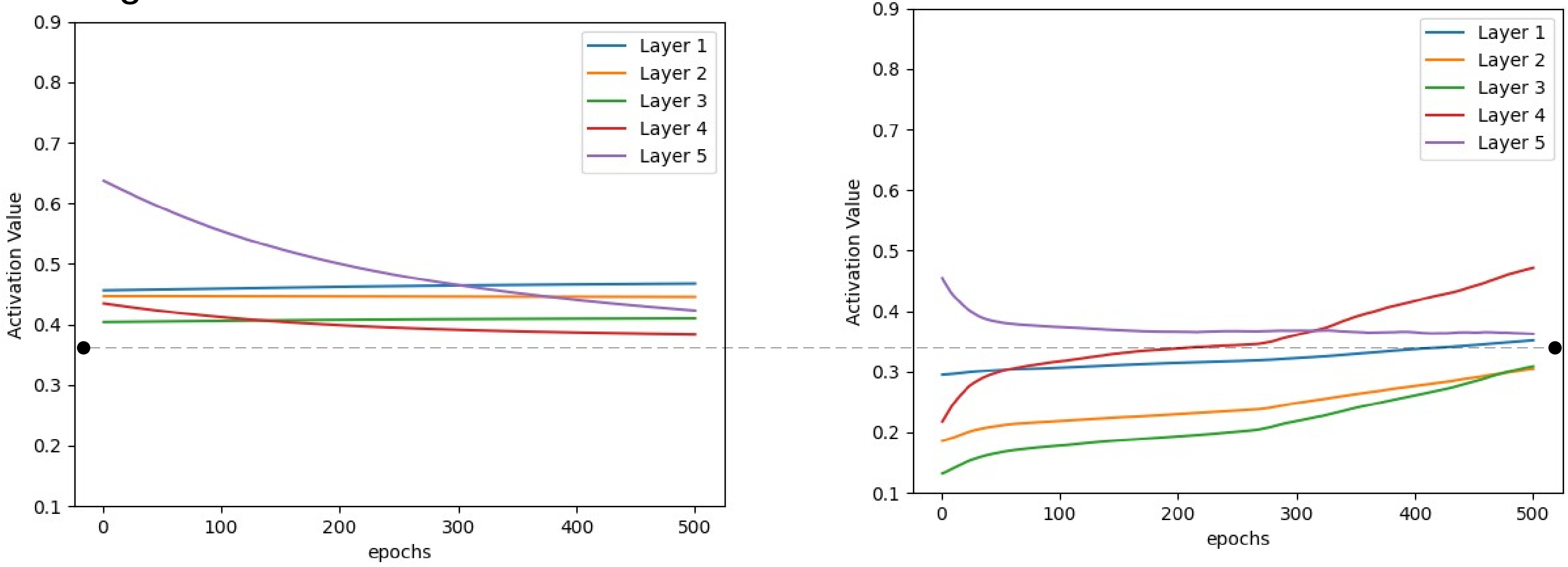


relu

Deep Neural Network

Activation Function	Layers	epochs	Learning rate		
				Training accuracy	Testing accuracy
Sigmoid	5	500	0.1	52.32%	66.0%
relu	5	500	0.1	59.94%	70.0%

Deep Neural Network

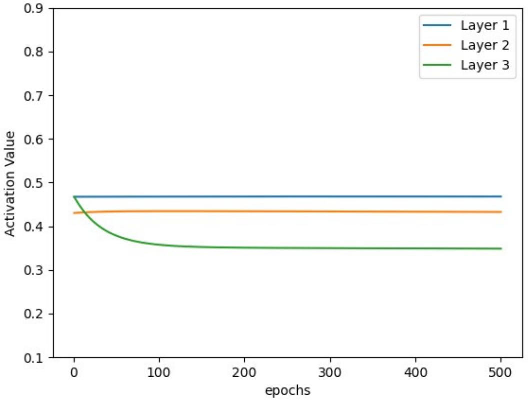


Sigmoid

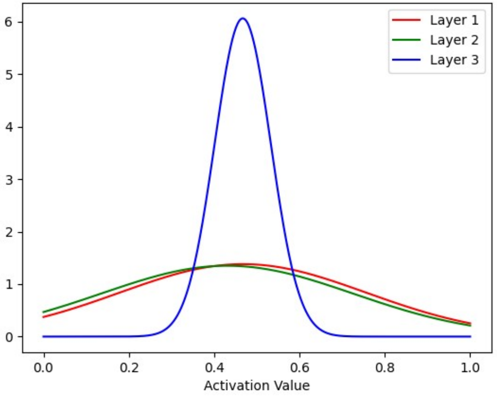
relu

Activation values in DNN

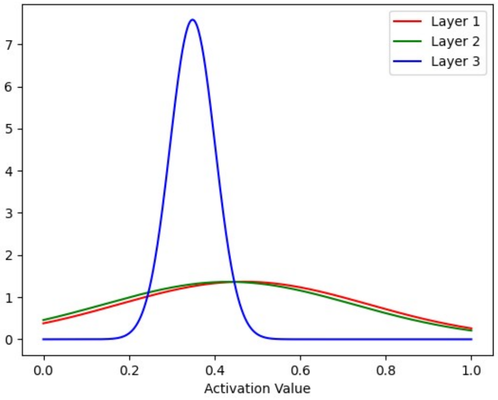
Activation Function	Layers	epochs	Learning rate	Kumar Initialization	
				Training accuracy	Testing accuracy
Sigmoid	3	500	0.1	55.76%	72.0%



Epochs 0

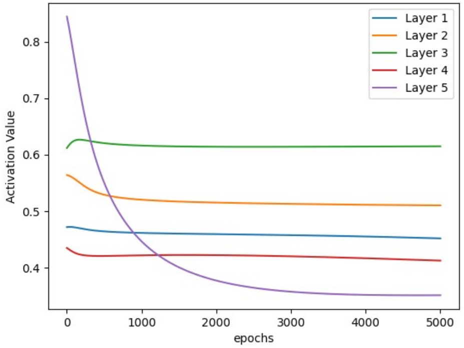


Epochs 500

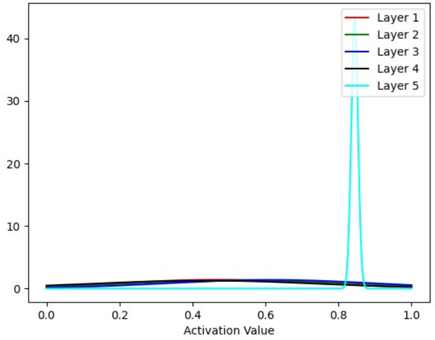


Activation values in DNN

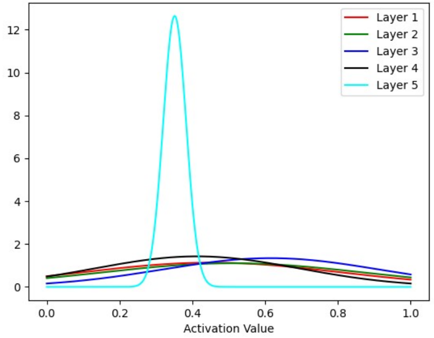
Activation Function	Layers	epochs	Learning rate	Kumar Initialization	
				Training accuracy	Testing accuracy
Sigmoid	5	5000	0.1	54.82%	66.0%

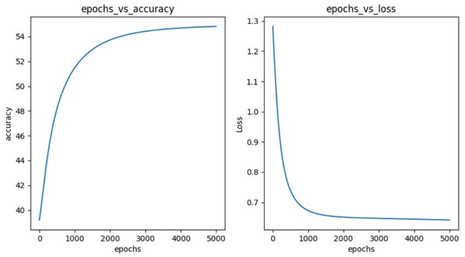


Epochs 0



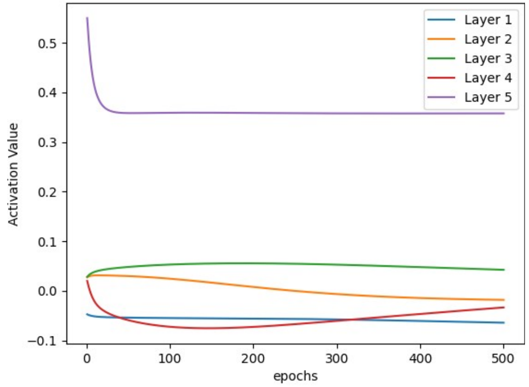
Epochs 500



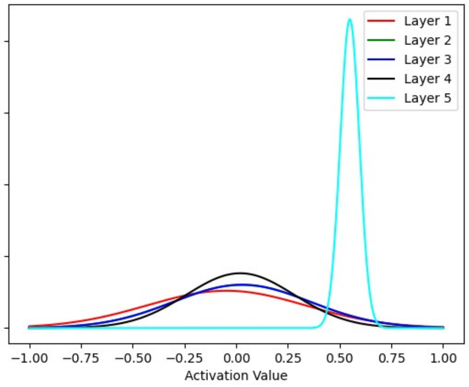


Activation values in DNN

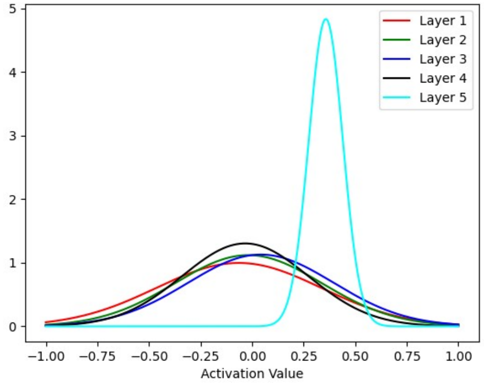
Activation Function	Layers	epochs	Learning rate	Xavier Initialization	
				Training accuracy	Testing accuracy
Tanh	5	500	0.1	58.66%	40.0%



Epochs 0

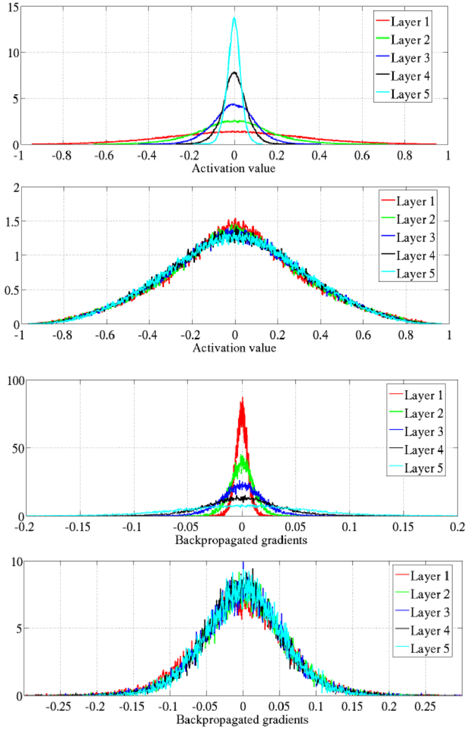


Epochs 500

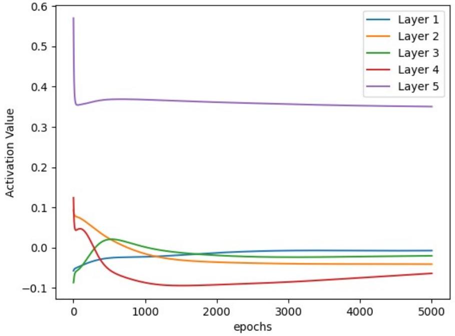


Activation values in DNN

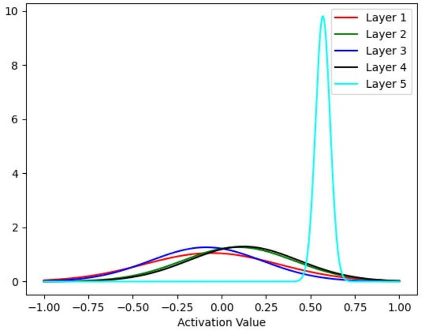
Activation Function	Layers	epochs	Learning rate	Xavier Initialization	
				Training accuracy	Testing accuracy
Tanh	5	5000	0.1	88.00%	76.0%



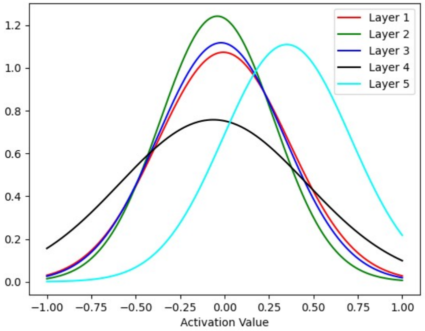
Mean activation values



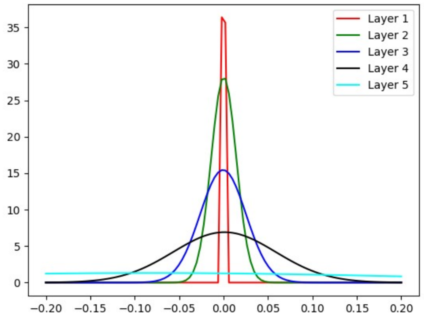
Epochs 0



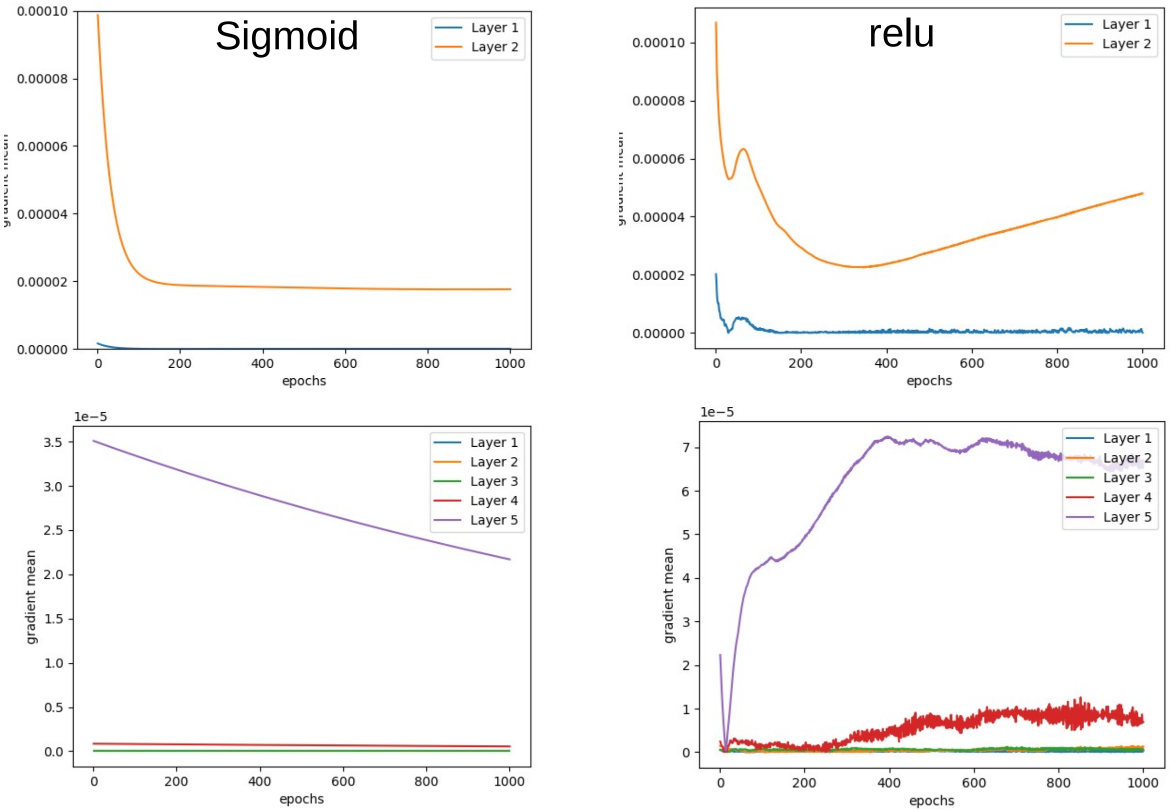
Epochs 5000



Backpropogation gradients

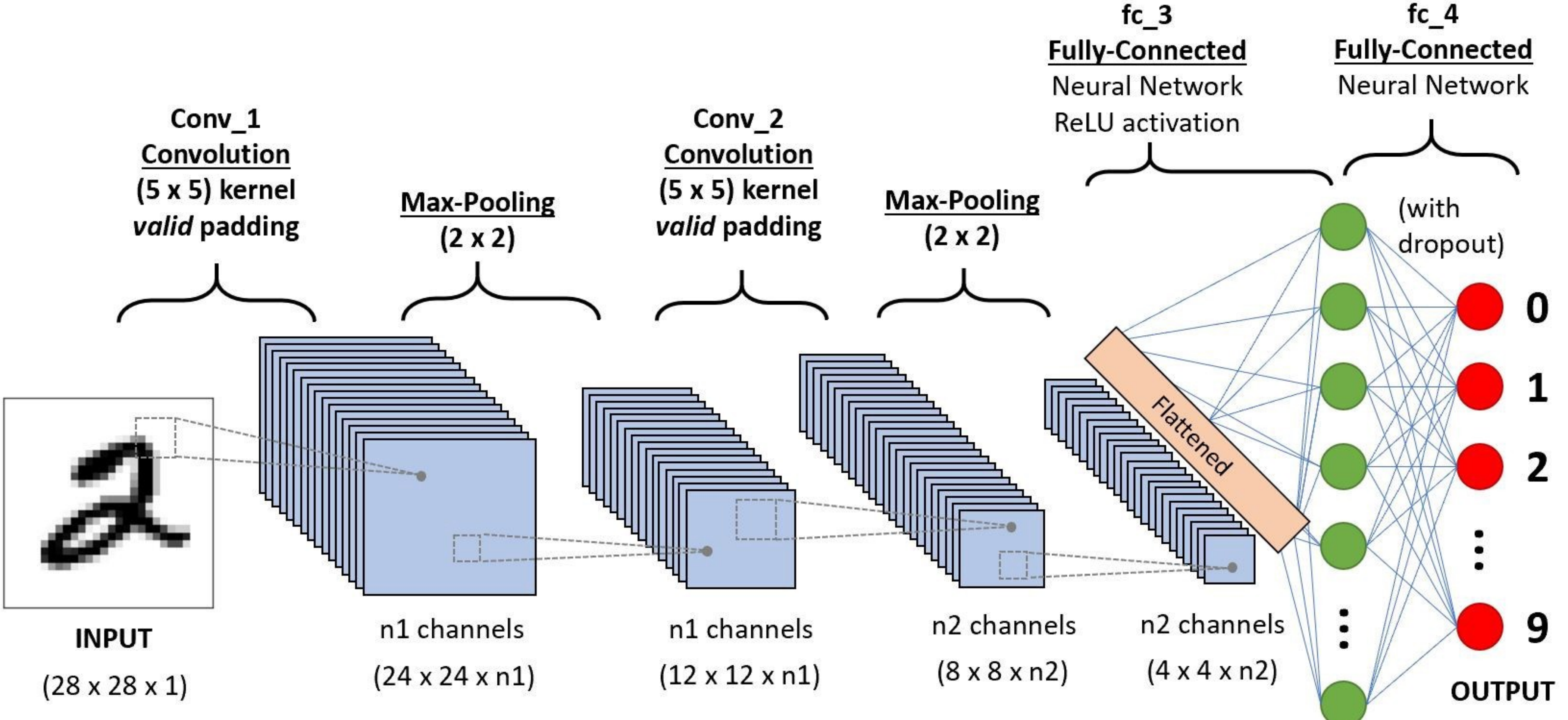


Backpropogation Gradient mean



Vanishing Gradient Problem

Convolution Neural Network



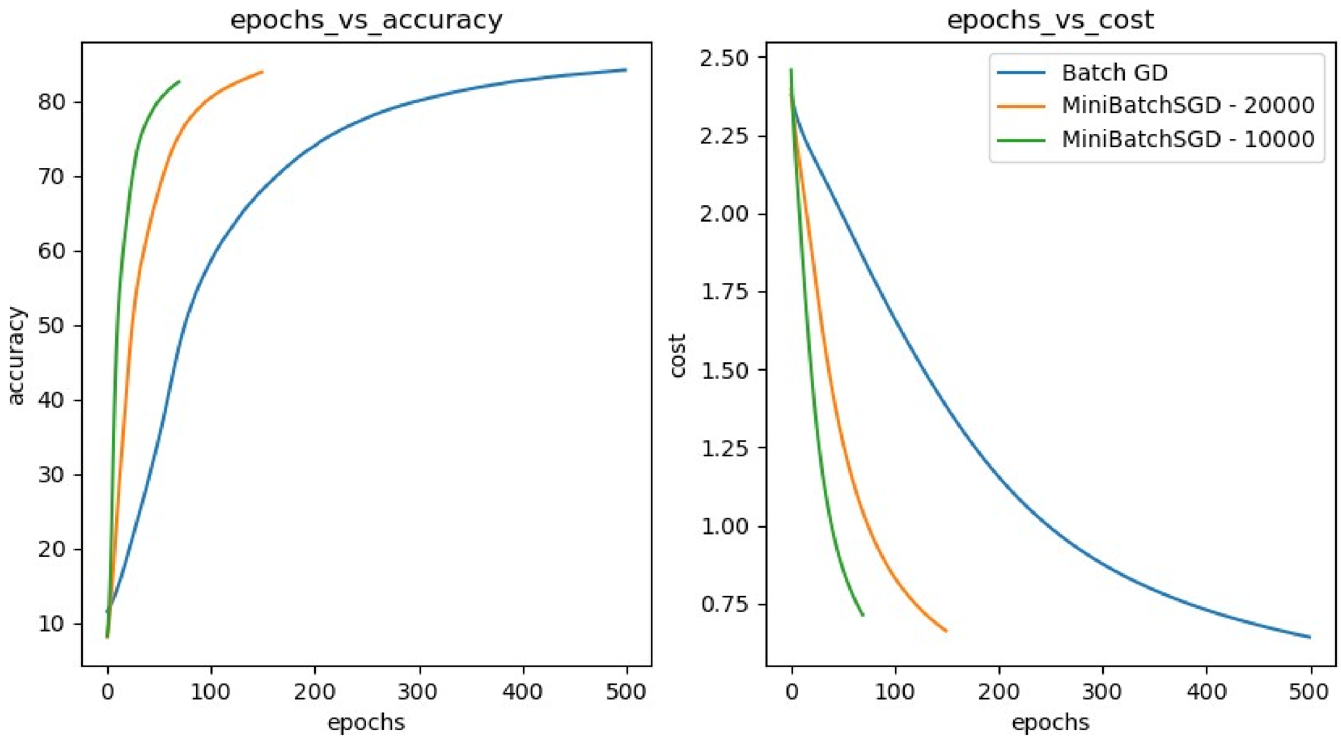
Neural Network - MNIST

Activation Function	GD	Layers	epochs	Lr	Kumar Initialization		RunTime (s)
					Training accuracy	Testing accuracy	
relu	Batch	2	500	0.01	84.67%	85.7%	187.62
relu	Mini batch 20000	2	139	0.01	82.79%	84.44%	151.09
relu	Mini batch 10000	2	70	0.01	82.93%	84.45%	151.78

Gradient descent variants
Batch, Stocastic and mini batch GD

Batch GD	Stocastic GD	Mini batch GD
+ In Batch Gradient Descent, all the training data is taken into consideration to take a single step. + We take the average of the gradients of all the training examples and then use that mean gradient to update our parameters. So that’s just one step of gradient descent in one epoch	+ In Stochastic Gradient Descent (SGD), we consider just one example at a time to take a single step. - we cannot implement the vectorized implementation on it - Noisy response - computationally expensive	+ combination of batch GD and stocastic GD. + more frequency updates as in stocastic GD and even we can vectorize the data as in batch GD

Results


epochs_vs_accuracy
accuracy
80
70
60
50
40
30
20
10
0
100
200
300
400
500
epochs
epochs_vs_cost
cost
2.50
2.25
2.00
1.75
1.50
1.25
1.00
0.75
0
100
200
300
400
500
epochs
Batch GD
MiniBatchSGD - 20000
MiniBatchSGD - 10000

Mini Batch GD

Advantages of using a batch size < number of all samples:

1. **It requires less memory.** Since you train the network using fewer samples, the overall training procedure requires less memory. That's especially important if you are not able to fit the whole dataset in your machine's memory.

2. **Typically networks train faster with mini-batches**. That's because we update the weights after each propagation. For each batch we've updated our network's parameters. If we used all samples during propagation we would make only 1 update for the network's parameter.

Neural Network - MNIST

Activation Function	GD	Layers	epochs	Lr	Kumar Initialization		RunTime (s)
					Training accuracy	Testing accuracy	
relu	Mini batch 10000	2	70	0.01	82.67%	83.61%	143.91
relu	Mini batch 10000	2	30	Exp decay	89.18%	89.9%	64.898

Learning rate - Neural Network

Activation Function	GD	Layers	epochs	Lr	Kumar Initialization		RunTime (s)
					Training accuracy	Testing accuracy	
relu	Mini batch 10000	2	15	Exp decay	86.98%	87.55%	36.43
relu	Mini batch 10000	2	15	Time based decay	86.62%	87.16%	32.372

Learning rate schedule: exponential decay, time based decay etc.

Gradient descent optimization algorithms

First order methods

- Momentum
- Nesterov accelerated gradient
- Adagrad
- Adadelta
- **RMSprop**
- **Adam**
- **AdaMax**
- Nadam
- AMSGrad

Second order methods

- Newton method
- Conjugate gradient
- Qausi Newton method
- Levenberg-Marquardt algorithm.

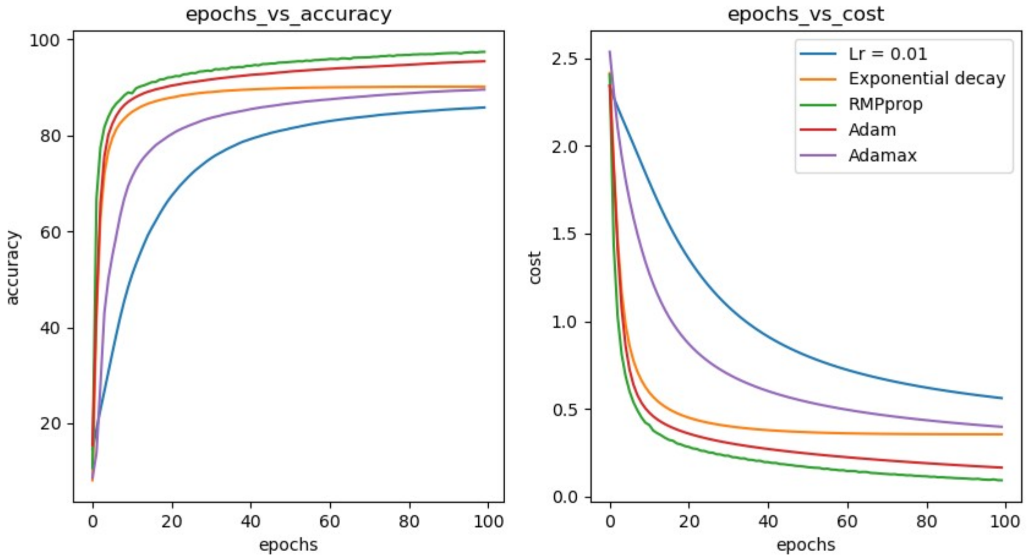
Learning rate - Neural Network

Activation Function	GD	Layers	epochs	Lr	Kumar Initialization		RunTime (s)
					Training accuracy	Testing accuracy	
relu	Mini batch 10000	2	30	Exp decay	89.18%	89.9%	64.898
relu	Mini batch 10000	2	10	RMSprop	89.13%	89.42%	25.558
relu	Mini batch 10000	2	10	Adam	87.59%	88.44%	21.992

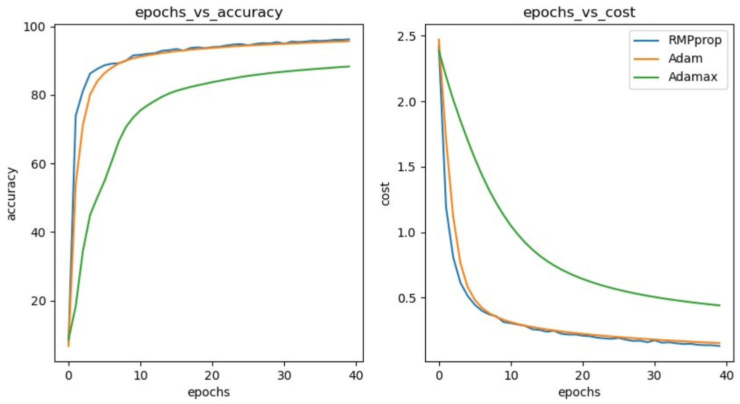
Adaptive learning rate: Adagrad, Adadelta, RMSprop, and Adam etc.

Results – MNIST

2 Layers



3 Layers


epochs_vs_accuracy
accuracy
epochs
epochs_vs_cost
cost
epochs
RMPprop
Adam
Adamax