DEEP LEARNING ASSIGNMENT-1

Members:
1. Shika Rao 2019A3PS1237H
2. Akshat Agrawal 2019AAPS0264H
3. Vinita Bhat 2019A7PS1204H

Aim: To build 16 deep learning models to test on the Fashion MNIST dataset. The models are evaluated for understanding the effect of suitable hyperparameters on the performance of a model.

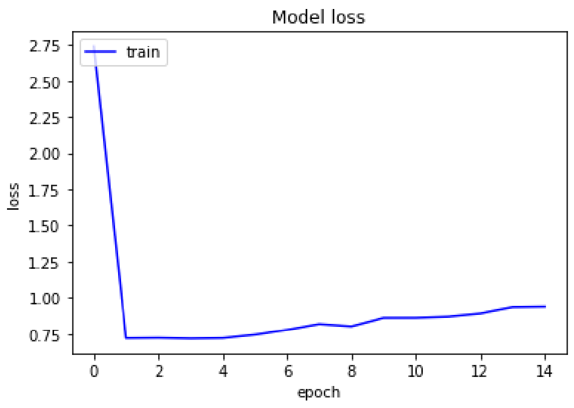
Date of Submission: 29/10/21

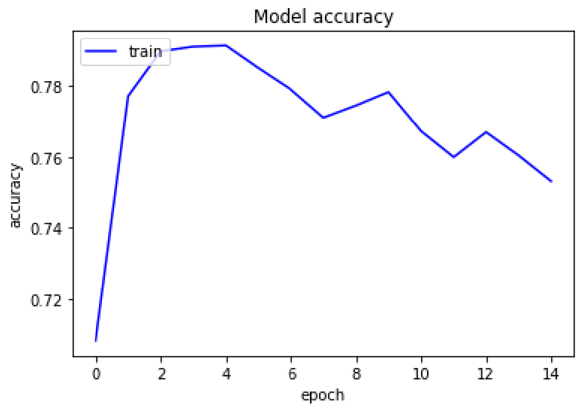
CONTENTS

ARCHITECTURAL DETAILS OF EACH OF THE MODELS BUILT

Model 1(Model 1_1)

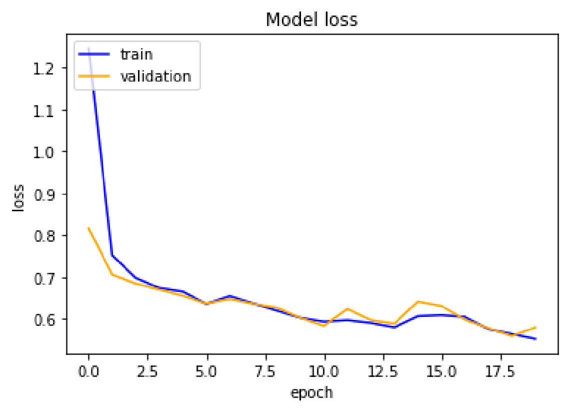
1. 3 layers
2. 128+64+10 nodes in total
3. Activation- ReLU
4. Loss- Categorical Cross Entropy Loss
5. Only testing and training
6. Time taken for 15 epochs- 61s.
7. Testing loss: `1.193249225616455`
8. Testing Accuracy: `75.26000142097473`

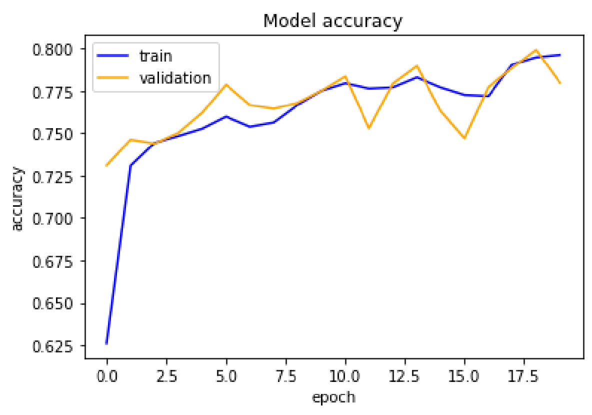




Model 2(Model 1_2)

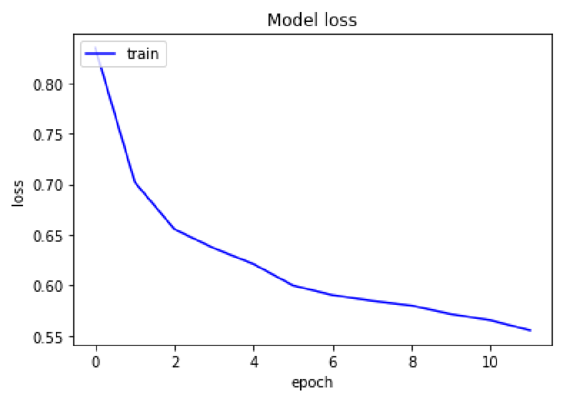
1. 4 layers
2. 256+128+32+10 nodes in total
3. Activation- Sigmoid
4. Loss- KL Divergence
5. adam optimizer
6. Testing, training, Validation (aka Hold-Out Validation)
7. Tested in batches (Not necessary cause small model anyways)
8. Time taken for 20 epochs- 41s
9. Testing loss: `0.5768556594848633`
10. Testing Accuracy: `78.68000268936157`

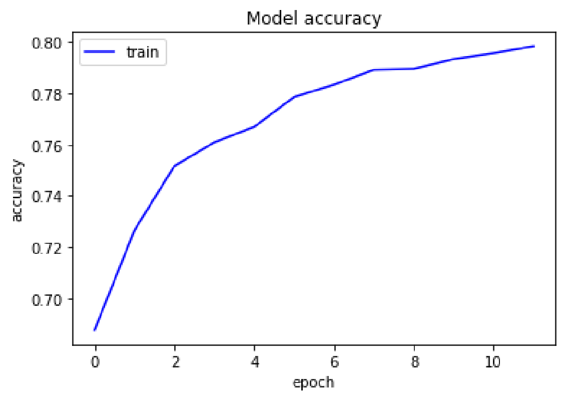




Model 3(Model 1_3)

1. 5 layers
2. 256+128+32+16+10 nodes in total
3. Activation- Tanh
4. Loss- Categorical Cross-Entropy
5. Testing, training, K-folds (5 folds) Cross-Validation
6. Time taken for 12 epochs- About 90s per fold, and five folds
7. Testing loss: `0.5573338985443115`
8. Testing Accuracy: `79.76428627967834`

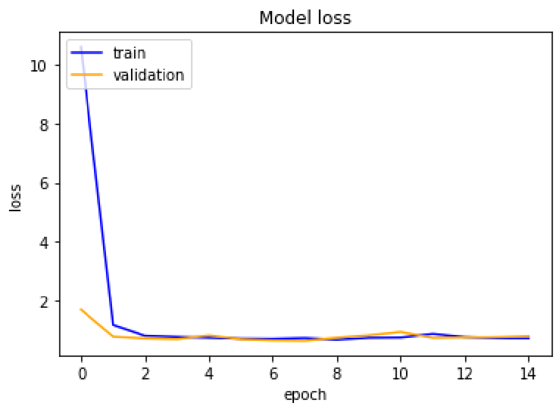


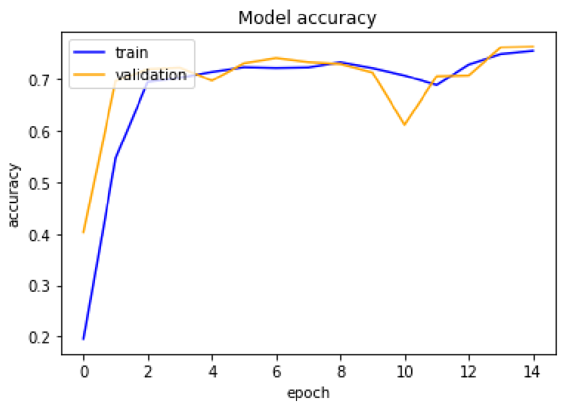


Model 4(Model 1_4)

1. 6 layers
2. 256+32+128+32+16+10 nodes in total
3. Activation- ReLU
4. Loss- KL Divergence
5. Testing, Training, validation
6. Time taken for 15 epochs- about 120s

7. Testing loss: `0.7756906747817993`
8. Testing Accuracy: `77.45000123977661`

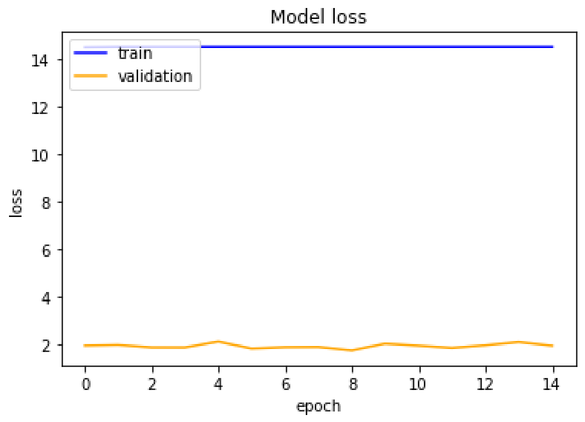


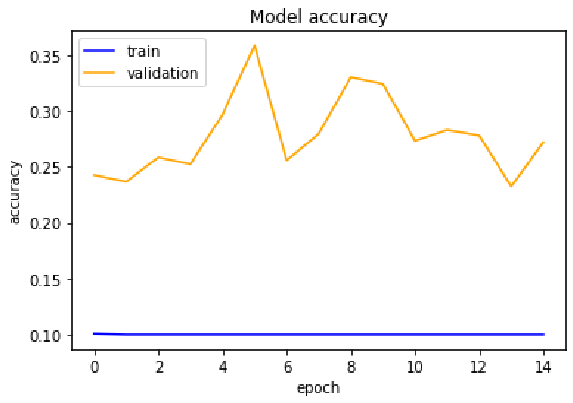


Model 5(Model 1_5)

(same as 1_4 but comparing if the model does better when the number of neurons decreases as usual)

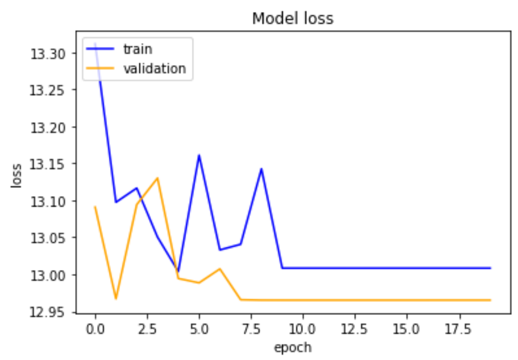
1. 6 layers
2. 256+128+64+32+16+10 nodes in total
3. Activation- ReLU
4. Loss- KL Divergence
5. Testing, Training, Validation
6. Time taken for 15 epochs- 120s
7. Testing loss: `12.989569664001465`
8. Testing Accuracy: `19.41000074148178`

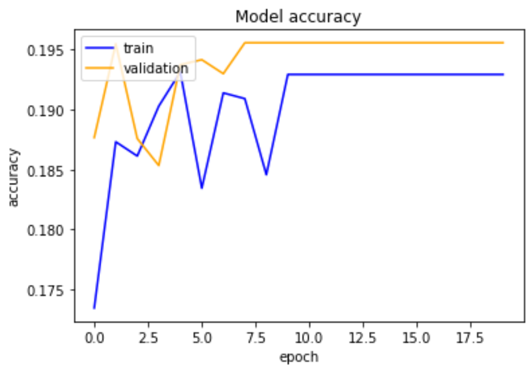




Model 6(Model 1_6)

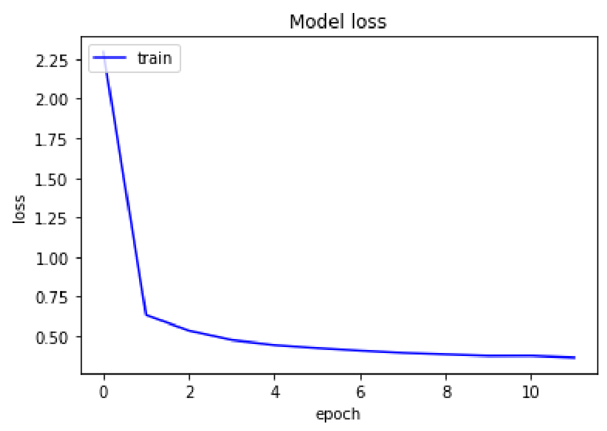
1. 4 layers
2. 128+64+32+10 nodes in total
3. Activation- Leaky_ReLU
4. Loss- KL Divergence
5. adam optimizer
6. Testing, training, Validation (aka Hold-Out Validation)
7. Tested in batches (Not necessary cause small model anyways)
8. Time taken for 20 epochs- 41s
9. Testing loss: `12.989569664001465`
10. Testing Accuracy: `19.41000074148178`

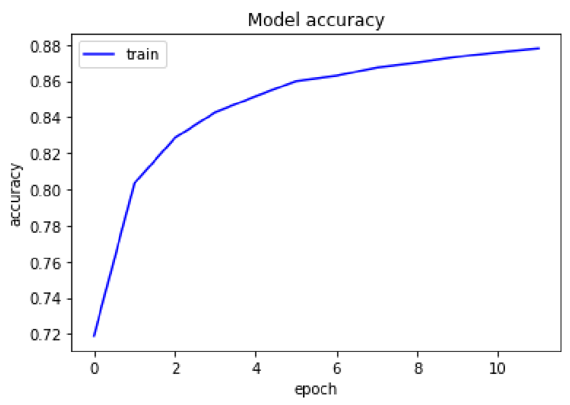




Model 7(Model 1_7)

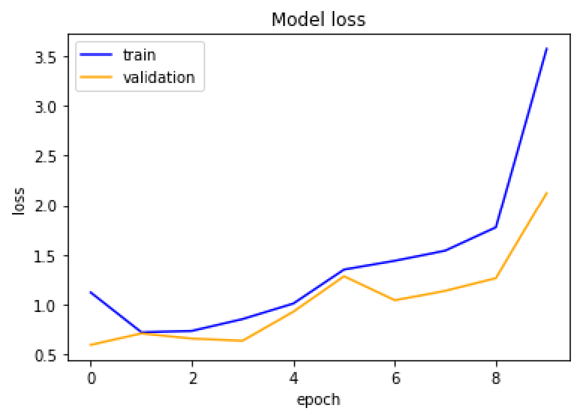
1. 5 layers
2. 256+128+32+16+10 nodes in total
3. Activation- Leaky_ReLU
4. Loss- Categorical Cross-Entropy
5. Testing, training, Validation
6. Time taken for 12 epochs- about 115s
7. Testing loss: `0.40112584829330444`
8. Testing Accuracy: `87.55000233650208`

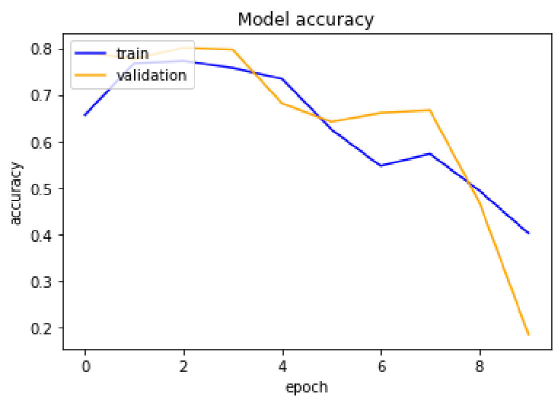




Model 8(Model 2_1)

1. 10 layers (8 hidden layers)
2. 1024+512+256+200+128+100+64+50+20+10 nodes in total
3. Activation- ReLU
4. Loss- Categorical Cross Entropy Loss
5. Dropout used
6. Training, testing, and validation
7. Time taken for 10 epochs- about 5 minutes
8. Testing loss: `2.134486675262451`
9. Testing Accuracy: `18.73999983072281`



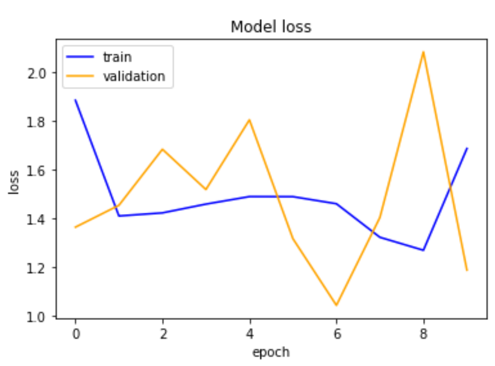


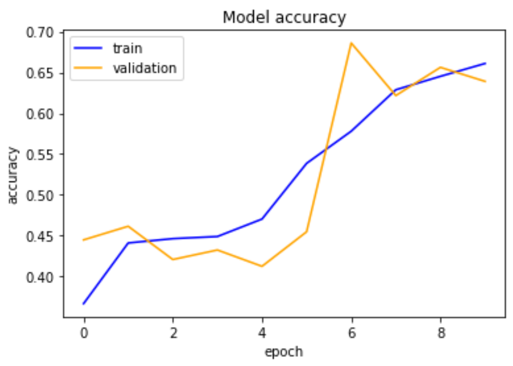
Model 9(Model 2_2)

(same as model 2_1 but removing the 1024 nodes in the first layer to check if accuracy improves)

1. 9 layers
2. 512+256+200+128+100+64+50+20+10 nodes in total

3. Activation- ReLU
4. Loss- Categorical Cross Entropy Loss
5. Dropout used
6. Training, testing, and validation
7. Time taken for 10 epochs- about 2 minutes
8. Testing loss: `1.1077382564544678`
9. Testing Accuracy: `63.66000175476074`

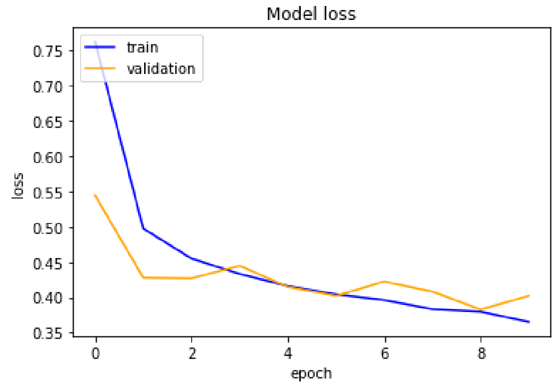


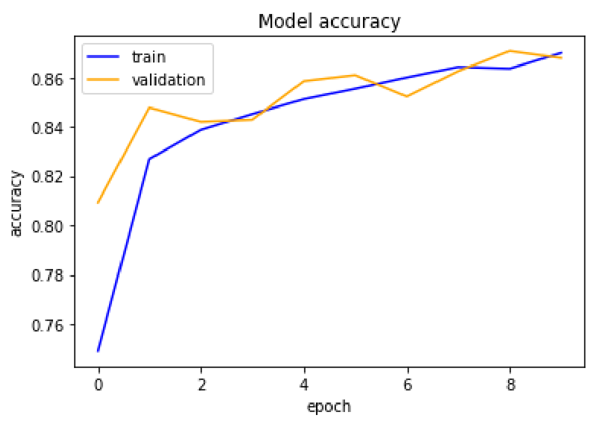


Model 10(Model 2_3)

(same as model 2_2 but adding batch normalization to check if accuracy improves)

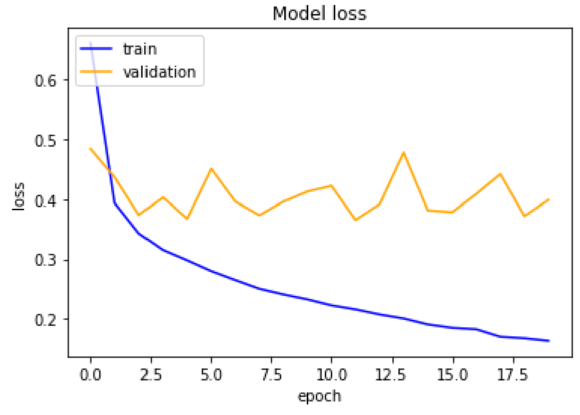
1. 9 layers
2. 512+256+200+128+100+64+50+20+10 nodes in total
3. Activation- ReLU
4. Loss- Categorical Cross Entropy Loss
5. Dropout used
6. Batch Normalization (Reducing the number of training epochs required to train deep networks.)
7. Training, testing, and validation (testing in batches)
8. Time taken for 10 epochs- about 81s
9. Testing loss: `0.38873401284217834`
10. Testing Accuracy: `86.6599977016449`

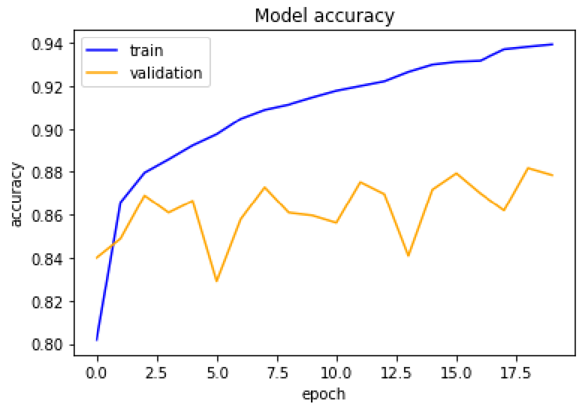




Model 11(Model 4_1)

1. 5 layers
2. 128+64+32+16+10 nodes in total
3. Activation- LeakyReLU
4. Categorical Cross-Entropy
5. adam optimizer
6. Testing, training, Validation (aka Hold-Out Validation)
7. Tested in batches (Not necessary cause small model anyways)
8. Time taken for 20 epochs- 41s
9. Testing loss: `0.37831899523735046`
10. Testing Accuracy: `88.16999793052673`

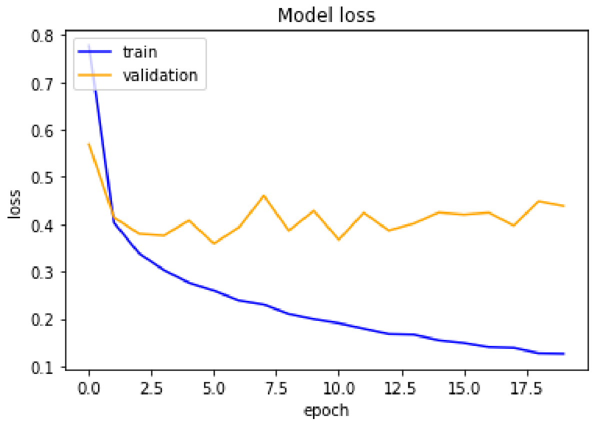


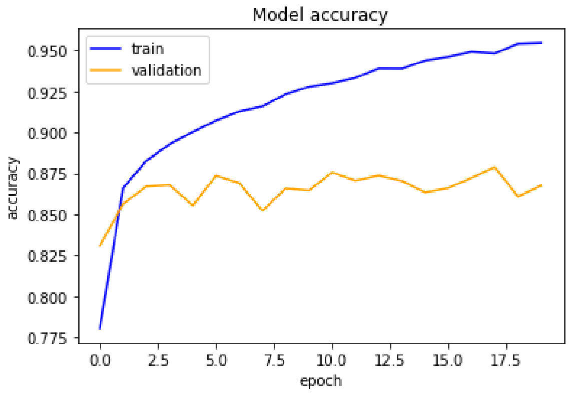


Model 12(Model 4_2)

1. 5 layers
2. 128+64+32+16+10 nodes in total
3. Activation- ReLU

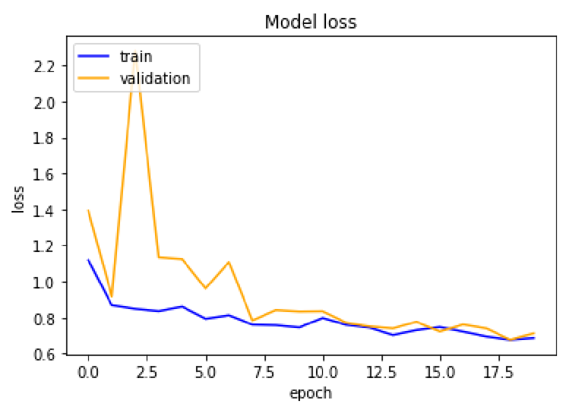
4. Categorical Cross-Entropy
5. adam optimizer
6. Testing, training, Validation (aka Hold-Out Validation)
7. Tested in batches (Not necessary cause small model anyways)
8. Time taken for 20 epochs- 41s
9. Testing loss: `0.4427981376647949`
10. Testing Accuracy: `86.79999709129333`

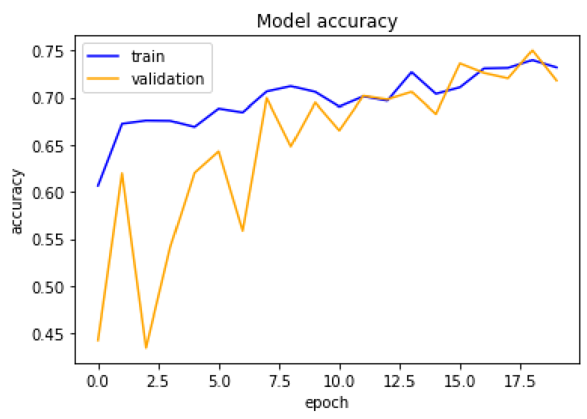




Model 13(Model 4_3)

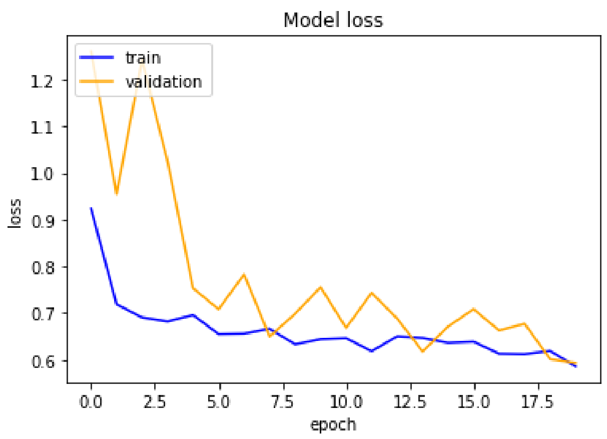
1. 5 layers
2. 128+64+32+16+10 nodes in total
3. Activation- Tanh
4. Categorical Cross-Entropy
5. adam optimizer
6. Testing, training, Validation (aka Hold-Out Validation)
7. Tested in batches (Not necessary cause small model anyways)
8. Time taken for 20 epochs- 41s
9. Testing loss: `0.7266266942024231`
10. Testing Accuracy: `70.59999704360962`

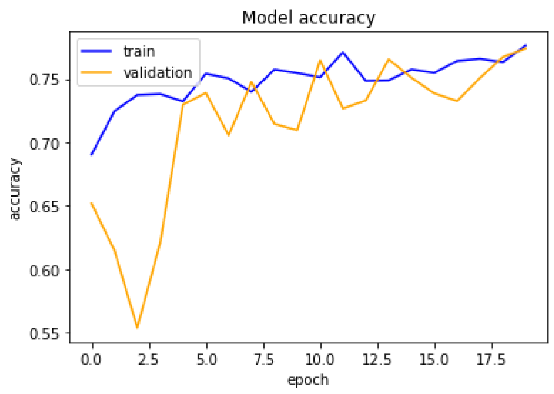




Model 14(Model 4_4)

1. 5 layers
2. 128+64+32+16+10 nodes in total
3. Activation- sigmoid
4. Categorical Cross-Entropy
5. adam optimizer
6. Testing, training, Validation (aka Hold-Out Validation)
7. Tested in batches (Not necessary cause small model anyways)
8. Time taken for 20 epochs- 41s
9. Testing loss: `0.5969304442405701`
10. Testing Accuracy: `77.49000191688538`

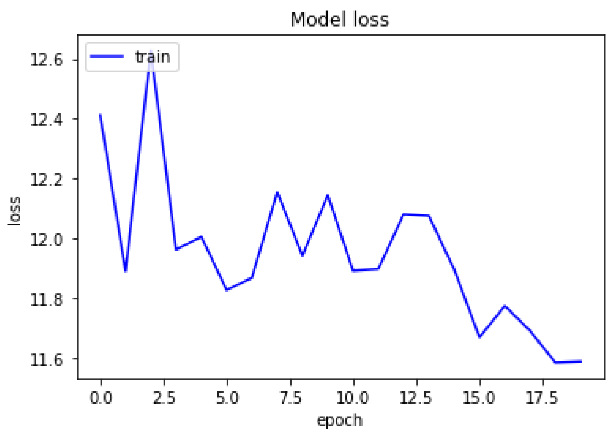


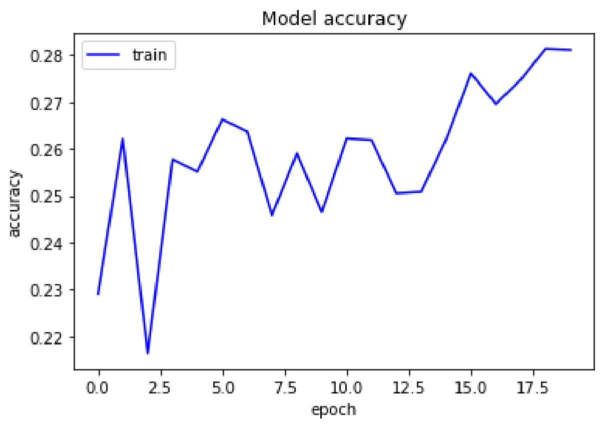


Model 15 (Model 4_5)

1. 4 layers
2. 128+64+32+10 nodes in total
3. Activation- ReLU

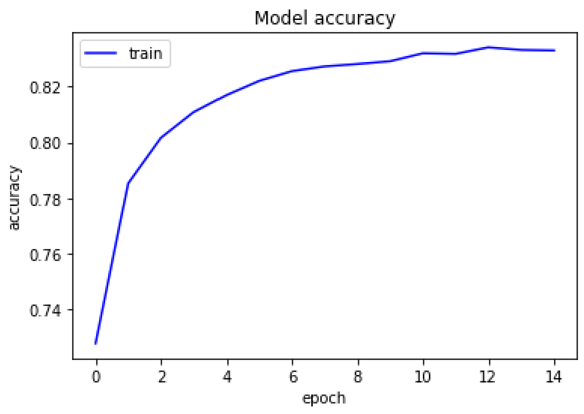
4. Loss- KL Divergence
5. adam optimizer
6. Testing, training
7. Time taken for 20 epochs- 92s
8. Testing Loss: `11.604256629943848`
9. Testing Accuracy: `28.00000011920929`

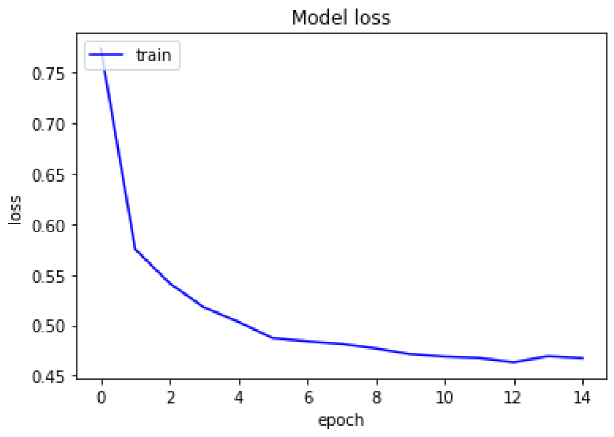




Model 16 (Model 4_6)

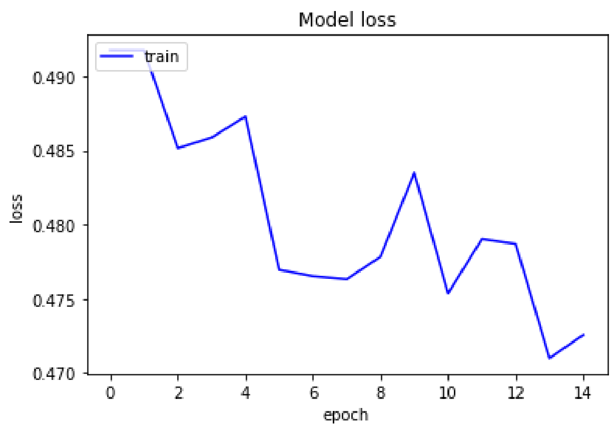
1. 3 layers
2. 128+64+10 nodes in total
3. Activation- Sigmoid
4. Loss- Categorical Cross Entropy Loss
5. Only testing and training
6. Time taken for 15 epochs- 77s
7. Testing Loss: `0.4783090353012085`
8. Testing Accuracy: `83.3100020885467`

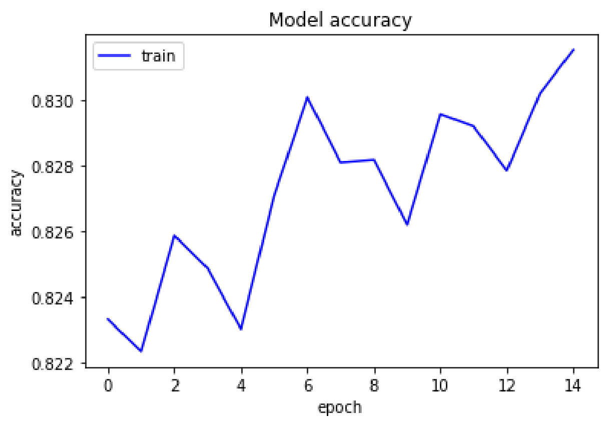




Model 17 (Model 4_7)

1. 3 layers
2. 128+64+10 nodes in total
3. Activation- TanH
4. Loss- Categorical Cross Entropy Loss
5. Only testing and training
6. Time taken for 15 epochs- 45s
7. Testing Loss: `0.47423964738845825`
8. Testing Accuracy: `83.05000066757202`



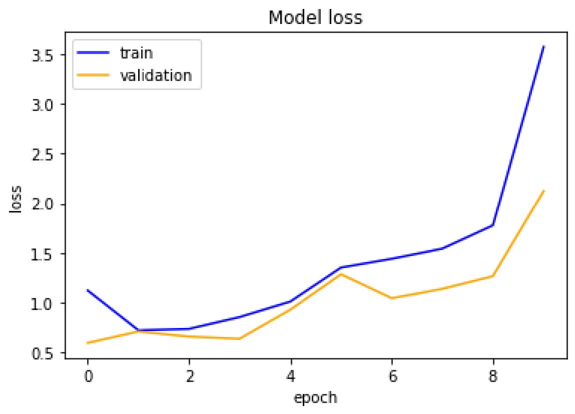


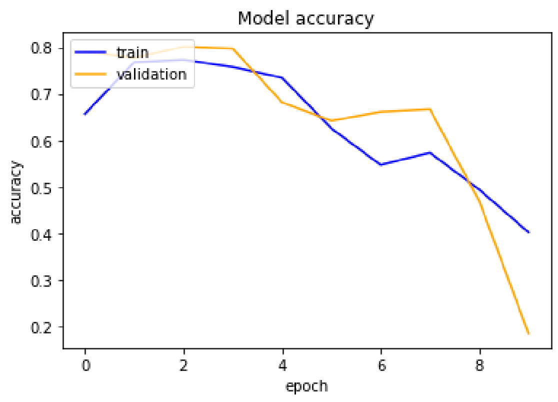
EXPLANATION OF THE ARCHITECTURAL CHOICE BEHIND THE MODELS

Model 8, 9, 10 (2_1, 2_2, 2_3) comparison: *(10, 9, 9 layers)*

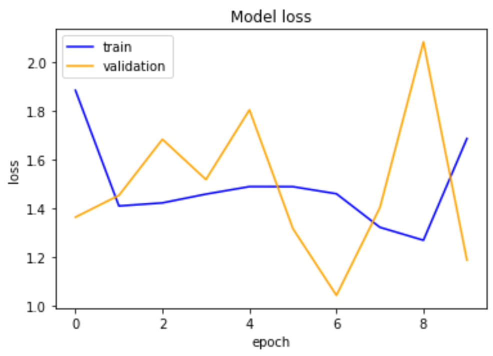
Model 8, 9, and 10 are the same in most aspects. Model 8 has 10 layers, uses dropout. *Improvement to model 8 is model 9.* It uses 9 layers and dropouts. *Improvement to model 9 is model 10.* It uses 9 layers, dropout, and batch normalization.

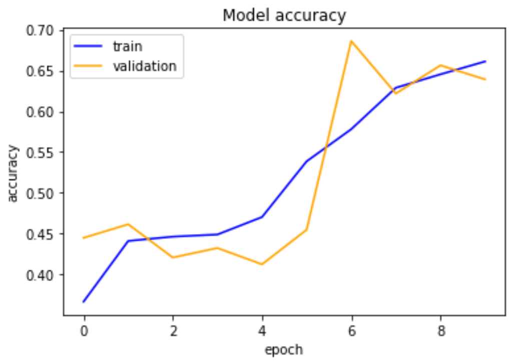
In model 8, the number of layers is 10, with 1024 units in the input layer when the input size is 784. Dropout was adopted in the hope that the model's performance would improve, but the testing accuracy is 18%. The training loss curves are increasing, and the training accuracy curves are decreasing. This indicates that this is an inferior model even though it is a deeper model with more nodes. The accuracy is also very random every time we run it (without using seed), and it is thus not a very reliable model. The model is not constrained due to the high number of nodes and layers which is not necessary for the given dataset. In addition, the time taken is about 5 minutes for 10 epochs which is not worth it since the accuracy is also bismal.



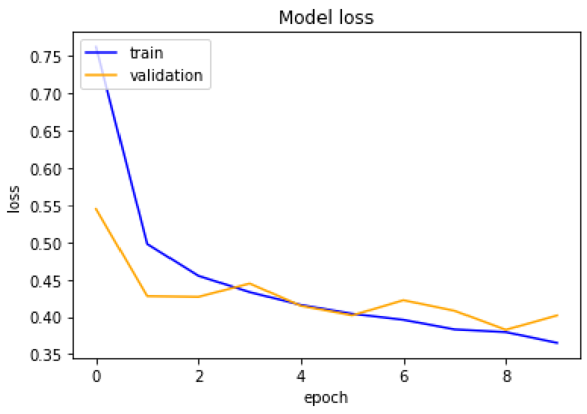


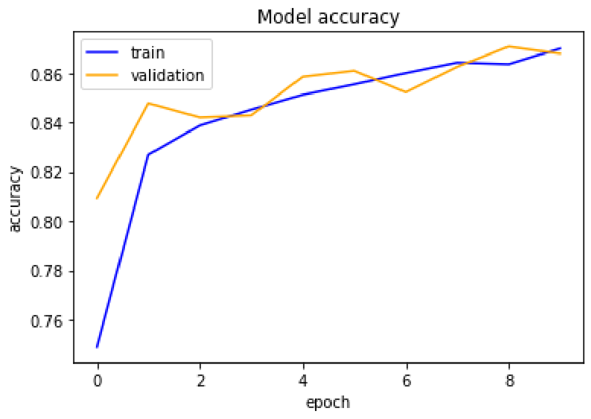
In model 9, the number of layers is 9, with 512 units in the input layer. The rest of the model architecture adopted was the same as model 8. The testing accuracy improves, and it is 63%. However, the training and validation loss curves indicate that the loss decreases and increases; it's not a predictable curve. The accuracy curves are also not strictly increasing. The accuracy is also very random every time we run it (without using seed), and it is thus not a very reliable model. There is, however, an improvement compared to model 8. This could be because this dataset doesn't need a deeper neural network architecture, and in model 8, 1024 nodes are used, which is not needed for this dataset. The time taken is about 2 minutes for 10 epochs, again indicating that this model is unnecessary for this dataset when smaller models perform better.





In model 10, the same architecture as model 9 is adopted. However, batch normalization along with dropout is used. This improves the model significantly, and it consistently gives a testing accuracy of above 85%, which indicates that it performs very well. The training and validation loss curves and accuracy curves are also strictly decreasing and strictly increasing, respectively suggesting that it is a good model. The time taken is also just about 1.5 minutes. At the same time, though the accuracy is excellent and the time taken is also not too much, the number of layers taken in this model is not necessary for this dataset when other models with lesser layers, no dropout used, and no batch normalization are performing similarly (they have around 75% accuracy.)





Thus, more layers do not indicate a better model.

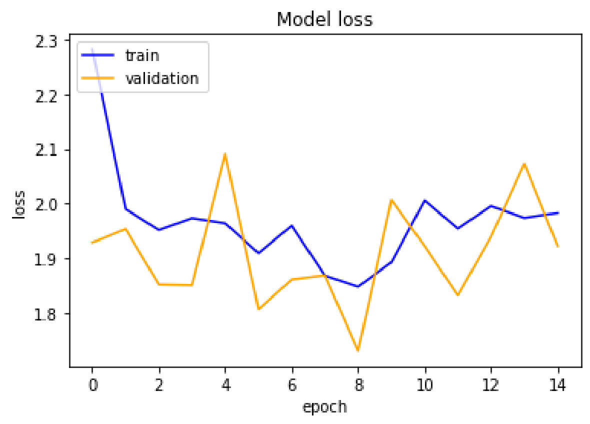
Model 4 and 5 (1_4 and 1_5) comparison: *(6, 6 layers)*

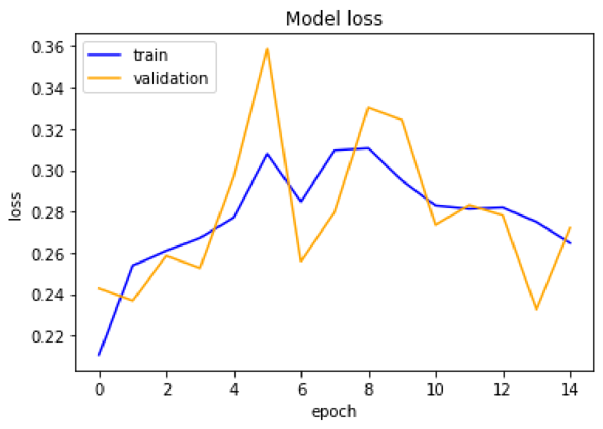
Model 4 and Model 5 are the same in all aspects except the number of nodes in each layer.

Model 4 has 256+32+128+32+16+10 nodes in the 6 layers. In model 4, the number of nodes decreases, then increasing and then decreasing again. The training loss and accuracy curves are decreasing and increasing respectively. However, the validation loss and accuracy curves are very random and choppy. Also, the testing accuracy is also 77.45%. However, this model is not reliable and changes every time we run it. The first time we ran it, the testing accuracy and validation accuracy was low because it got affected by overfitting and the generalization accuracy of the model decreased and thus it didn't perform well on unseen data though the training curves indicated a good model. The graphs attached are from another run of the model and it seems to have performed well this time. Theoretically though, if the number of nodes are increased and then decreased, especially in the middle hidden layers, the model should get more adaptive, so it can learn smaller details.

Old testing Accuracy: 26.94%

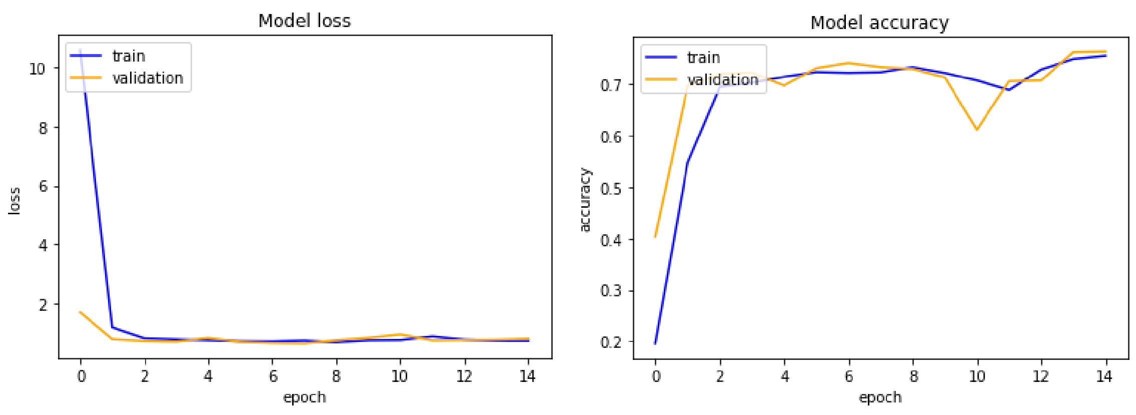
Old plots:



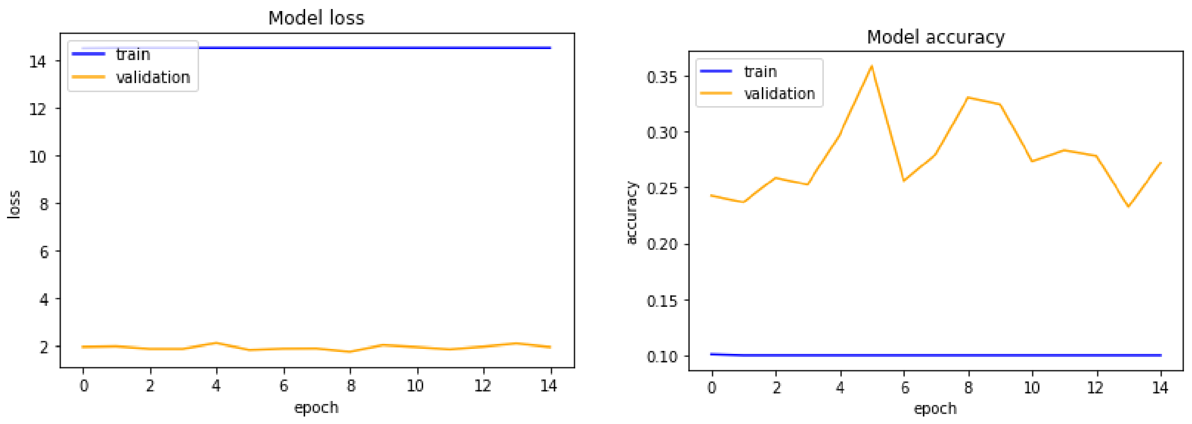


New testing Accuracy: 77.45%

New Plots:



Model 5 has 256+128+64+32+16+10 nodes in the 6 layers. The same, commonly used trend of decreasing number of nodes in the layers is followed. However it is consistently seen (it was run about 6 times in the hope that it'll work) that the model is just not learning. The training loss and accuracy curves are straight lines indicating that the model is not learning or improving at all. The validation loss and accuracy curves are however as expected. But this doesn't matter as the model has not learnt at all. The same procedure was followed for construction of this model as compared to the other models, but it still hasn't learnt. That is why we concluded that we don't need to build a model with 6 or more layers for this dataset. Also the model isn't constrained due to high number of nodes in the layers which is the reason why the model hasn't learnt.

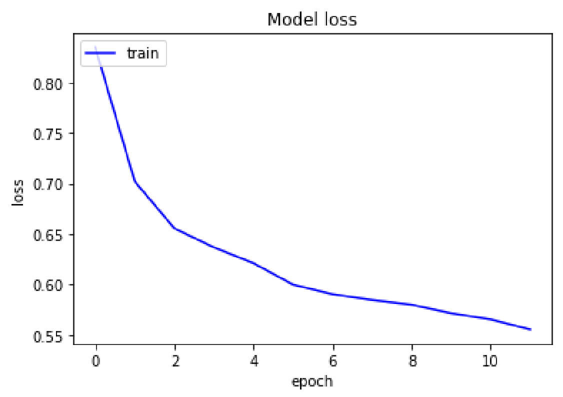


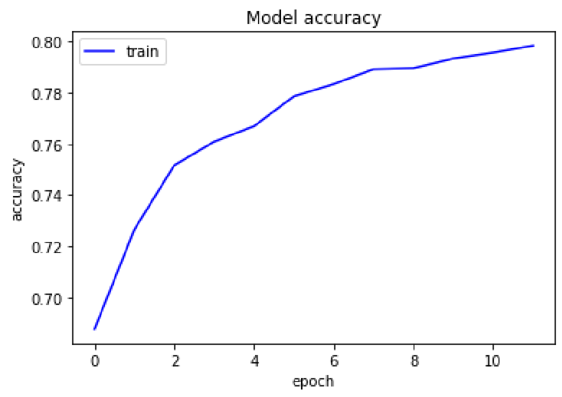
Model 3 explanation:

Model 3(Model 1_3): *(5 layers)*

In model 3, 5 layers were used to build the model. The testing accuracy is 79.76 % and the training loss and accuracy are also strictly decreasing and strictly increasing curves.

It seems like 5 layers and the other hyperparameters used are pretty ideal for this dataset considering the time taken and the high accuracy obtained with minimal number of layers and nodes. The number of epochs this model is trained for is 12 epochs. However, from the information in the training loss and accuracy curves, we can see that the model is still learning. The loss and accuracy haven't plateaued yet indicating that this model could perform much much better if it was trained for more time.





Model 3, 7, 11, 12, 13, and 14 (1_3, 1_7, 4_1, 4_2, 4_3, and 4_4) comparison: *(5, 5, 5, 5, 5, 5 layers with different activation function)*

Model 3 and 13 are both using tanh. The number of layers are also the same, 5. Although Model 13 has been trained longer for 20 epochs, Model 3 has performed better by almost 10% as compared to Model 13 as the number of nodes in each layer were decreased. It constrained the model and hence performed better.

Model 7 and 11 are both using Leaky ReLU. The number of layers are also the same, 5. Model 11 has performed better by almost 1% as compared to Model 7 as the number of nodes in each layer were decreased. It constrained the model and hence performed better.

Model 12 and Model 14 use exactly same architectures, but varying activation functions, we observe that when we ReLU function instead of Sigmoid, there is a 10% increase testing accuracy, ReLU activation function hence has performed better.

Thus it is concluded that for this dataset, 5 layers is the ideal number of layers for any model with ReLU/Leaky ReLU activation function to perform well.

ALL THE MODELS GROUPED USING ACTIVATION FUNCTION AS THE METRIC FOR ANALYSING ACCURACY

Sigmoid:

Model 2(Model 1_2)

1. Layers: 4
2. Testing Loss: 0.58
3. Testing accuracy: 78.68 %
4. Epochs: 20

Model 14(Model 4_4)

1. Layers: 5
2. Testing Loss: 0.60
3. Testing accuracy: 77.49%
4. Epochs:

Model 16(Model 4_6)

1. Layers: 3
2. Testing Loss: 0.48
3. Testing accuracy: 83.31%
4. Epochs: 15

Model Number	Layers	Accuracy
Model 2	4	78.68 %
Model 14	5	77.49%
Model 16	3	83.31%

Tanh:

Model 3(Model 1_3)

1. Layers: 4
2. Testing Loss: 0.56

3. Testing accuracy: 79.76%
4. Epochs: 12

Model 13(Model 4_3)

1. Layers: 5
2. Testing Loss: 0.73
3. Testing accuracy: 70.60%
4. Epochs:

Model 17(Model 4_7)

1. Layers: 3
2. Testing Loss: 0.47
3. Testing accuracy: 83.05%
4. Epochs:15

Model Number	Layers	Accuracy
Model 3	4	79.76%
Model 13	5	70.60%
Model 17	3	83.05%

Leaky ReLu:

Model 6(Model 1_6)

1. Layers: 4
2. Testing Loss: 12.99
3. Testing accuracy: 19.41%
4. Epochs: 20

Model 7(Model 1_7)

1. Layers: 5
2. Testing Loss: 0.40
3. Testing accuracy: 87.55%

4. Epochs: 12

Model 11(Model 4_1)

1. Layers: 5
2. Testing Loss: 0.38
3. Testing accuracy: 88.17
4. Epochs:

Model Number	Layers	Accuracy
Model 6	4	19.41%
Model 7	5	87.55%
Model 11	5	88.17%

ReLU:

Model 1(Model 1_1)

1. Layers: 3
2. Testing Loss: 1.19
3. Testing accuracy: 75.26%
4. Epochs: 15

Model 4(Model 1_4)

1. Layers: 6
2. Testing Loss: 1.95
3. Testing accuracy: 77.45%
4. Epochs: 15

Model 5(Model 1_5)

1. Layers: 6
2. Testing Loss: 14.51
3. Testing accuracy: 19.0%
4. Epochs: 15

Model 8(Model 2_1)

1. Layers: 10
2. Testing Loss: 2.13
3. Testing accuracy: 18.73%
4. Epochs: 10

Model 9(Model 2_2)

1. Layers: 9
2. Testing Loss: 1.1
3. Testing accuracy: 63.66%
4. Epochs: 10

Model 10(Model 2_3)

1. Layers: 9
2. Testing Loss: 0.39
3. Testing accuracy: 86.66 %
4. Epochs: 10

Model 12(Model 4_2)

1. Layers: 5
2. Testing Loss: 0.44
3. Testing accuracy: 86.80
4. Epochs:

Model 15(Model 4_5)

5. Layers: 4
6. Testing Loss: 11.60
7. Testing accuracy: 28.00
8. Epochs: 20

Model Number	Layers	Accuracy
Model 1	3	75.26%
Model 4	6	26.95%
Model 5	6	19.0%

Model 8	10	18.73%
Model 9	9	63.66%
Model 10	9	86.66%
Model 12	5	86.80%
Model 15	4	28.00%

FINAL INFERENCE TABLE

MODEL NUMBER	Activation Function	Layers	Test Accuracy	Class
1	ReLU	3	75.26	Medium
2	Sigmoid	4	78.68	Medium
3	tanh	5	79.76	Medium
4	ReLU	6	77.45	Medium
5	ReLU	6	19.41	Inferior
6	Leaky ReLU	4	19.41	Inferior
7	Leaky ReLU	5	87.55	Superior
8	ReLU	10	18.73	Inferior
9	ReLU	9	63.66	Medium
10	ReLU	9	86.66	Superior
11	Leaky_ReLU	5	88.17	Superior
12	ReLU	5	86.80	Superior
13	tanh	5	70.60	Medium
14	Sigmoid	5	77.49	Medium
15	ReLU	4	28.00	Inferior
16	Sigmoid	4	83.31	Superior
17	Tanh	4	83.05	Superior

CONCLUSIONS

1. **No one hyperparameter is solely responsible for the performance of a model.** All of them taken together were used for our analysis.

2. Out of the activation functions used and our analysis for the given dataset, the performance is as follows:
   Leaky ReLU>ReLU>Sigmoid>Tanh
   However, sigmoid and tanh perform well on the models as well.

3. The models with a **total number of layers <=5** seems to perform well on the given dataset (even without dropout or batch normalization to reduce the factors of variation) and **5 layers seems to be ideal**.
   However, a model with 9 layers has consistently performed well too, but that is because dropout and batch normalization was used. This is an exception.
   Thus, more nodes and more layers don't necessarily mean a better model.

4. Normalizing the outputs of the hidden layer with Batch Normalisation enhanced the performance of the model to a great extent.