CM7

Results Analysis

- Runtime performance for training and testing
  The models are trained for 30 epochs and GPU is used as the dataset is large.
  With GPU,

Model 1		Model 2		Model 3	
Training per epoch	Testing per epoch	Training Per epoch	Testing Per epoch	Training Per epoch	Testing Per epoch
2-3 s	1s	3-5s	1s	3-4s	1s

These performances are depend on the architecture of the model. The fully connected model 3 takes higher time as it has more parameters to train when compared to CNN models. Model 1 takes the least runtime compared to the other 3 models.

- Comparison of the different parameters or designs you tried.

1) Experimentation with number of epochs

Epochs	Model 1		Model 2		Model 3	
	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy
10	81.84	79.74	82.95	82.50	85.39	83.24
20	86.70	82.64	86.84	87.70	90.91	86.19
30	91.69	83.49	88.89	89.68	93.46	88.47

As the number of epochs increase from 10 to 30, we can observe that the models learn and generalize better and therefore accuracy increases. But number of epochs should not be increased very high as it can lead to overfitting of the model. Number of epochs should also not be very low, as it can lead to under fitting and cannot capture the complexities of the model.

2) Experimentation with optimizers

Optimizer controls the learning rate in order to reduce the error of the model. Most commonly used optimizers are:

SGD (Stochastic Gradient Descent) - SGD differs from regular gradient descent in the way it calculates the gradient. Instead of using all the training data to calculate the gradient per epoch, it uses a randomly selected instance from the training data to estimate the gradient. Can perform best but various additions need to be made and learning rate need to be correctly tuned.
Adam (adapative moment estimate): For relatively large datasets Adam is very Robust. Converges quickly and gives pretty good performance.
Keeps track of gradient learning rates and momentum per dimension
Uses variance (2nd moment) rather than means and scales steps based on recent magnitudes.

RMSprop - The RMSprop optimizer is similar to the gradient descent algorithm with momentum. The RMSprop optimizer restricts the oscillations in the vertical direction. Therefore, we can increase our learning rate and our algorithm could take larger steps in the horizontal direction converging faster.

Optimizers	Model 1		Model 2		Model 3	
	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy
SGD	80.44	80.70	80.21	82.96	84.81	84.03
RMSProp	91.00	83.89	87.65	87.19	92.68	86.98
Adam	91.69	83.49	88.89	89.68	93.46	88.47

We can observe that among SGD, RMSProp and Adam optimizers, the accuracy performance of all the 3 models are high with Adam optimizer. It is the most commonly used optimizer and is robust to large datasets and performs well on them.

3) Experimentation with activation functions at Output

Sigmoid activation function is computationally expensive and leads to slower convergence.

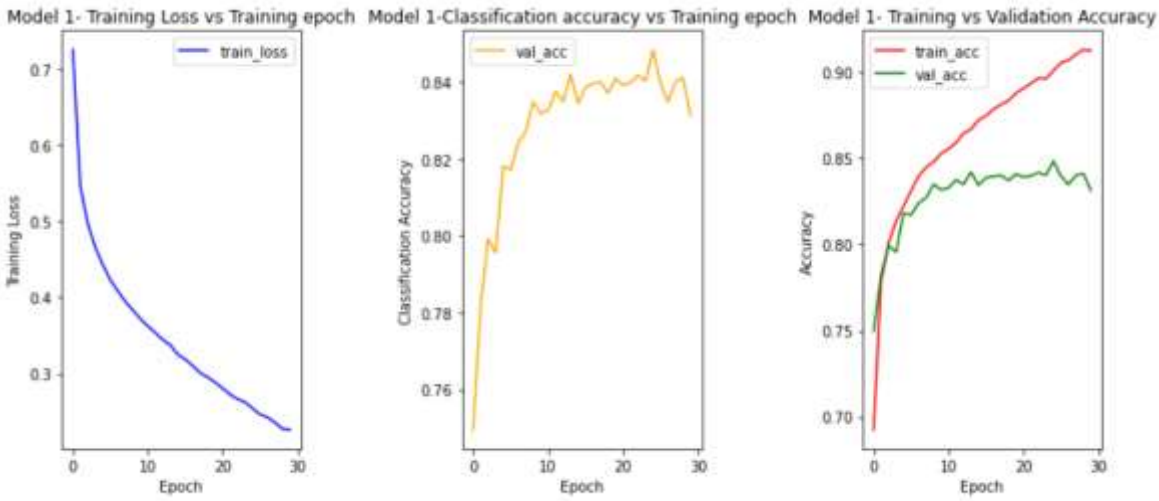
Softmax activation is a popular choice at the output for multiclass classification problems and when output labels are categorical and it is more efficient, faster and robust to large datasets compared to sigmoid.

Optimizers	Model 1		Model 2		Model 3	
	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy	Training Accuracy	Testing Accuracy
Sigmoid	90.98	82.63	88.91	88.56	93.34	87.83
Softmax	91.96	83.49	88.89	89.68	93.46	88.47

We can observe that that accuracy values of all the 3 models have better performance with softmax activation function at the output layer as FashionMNIST data is a multiclass classification problem and softmax works best with it.

- You can use any plots to explain the performance of your approach. But at the very least produce two plots, one of training loss vs. training epoch and one of classification accuracy vs. training epoch on both your training and test set.

Model 1:

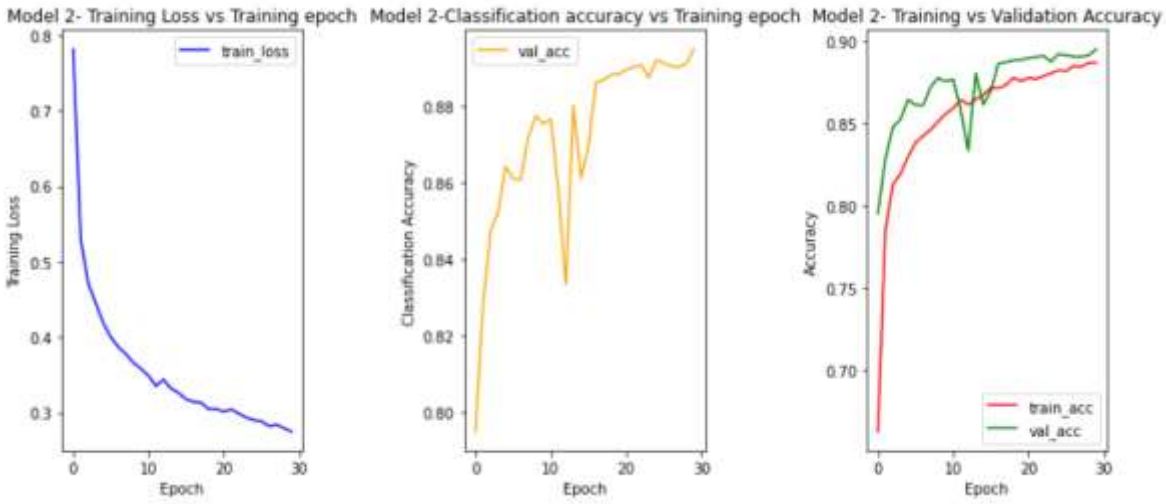


Training loss vs Training epochs and Classification vs Training epochs graphs are plotted. As the number of epochs increase, training loss decreases. The performance of the model can be better understood from the Training accuracy and validation accuracy (classification accuracy) vs epoch plot. Here we can observe that as the

number of epochs increase, training accuracy is increasing but towards the end of epoch, training accuracy is increasing but validation accuracy is not improving and showing a decreasing trend. This is not a good sign of generalization.

Since this model does not have dropout layers unlike model 2, it is prone to overfitting. Early stopping can be used to prevent overfitting or dropout layers can be included in the model.

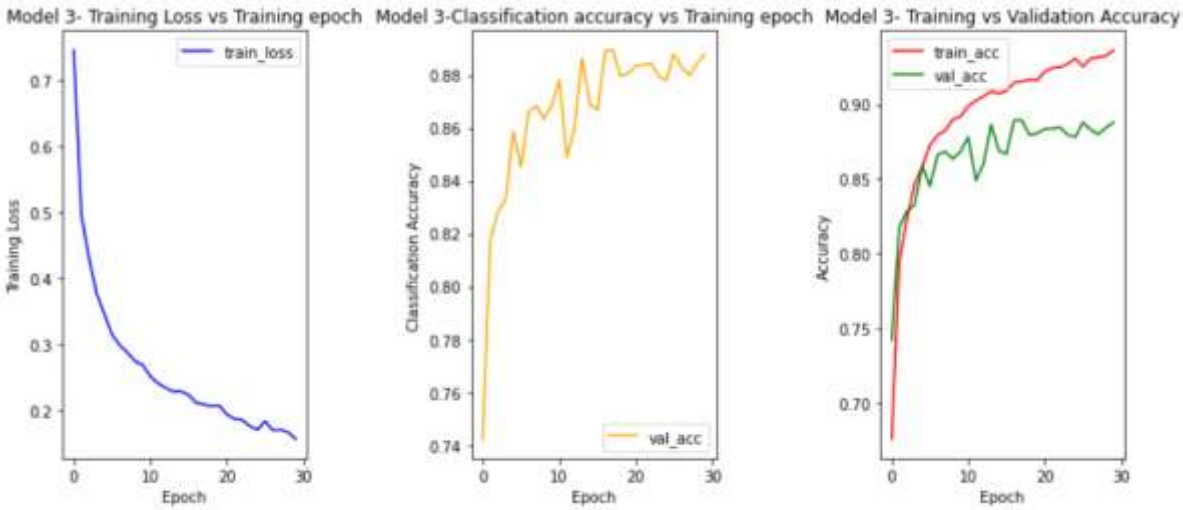
Model 2:



Training loss vs Training epochs and Classification vs Training epochs graphs are plotted. As the number of epochs increase, training loss decreases. The performance of the model can be better understood from the Training accuracy and validation accuracy (classification accuracy) vs epoch plot. Here we can observe that as the number of epochs increase, training accuracy is increasing and validation accuracy can also be observed to be increasing and is found infact to be greater than training accuracy. This means that our model is generalizing better and can perform well on new unseen data.

As this model has dropout layers, they provide regularization and keep overfitting issue at check. Model 2 has the best performance out of all the 3 in terms of generalization and has the highest test accuracy of 89.68%

Model 3:



Training loss vs Training epochs and Classification vs Training epochs graphs are plotted. As the number of epochs increase, training loss exhibits a downward trend. The performance of the model can be better understood from the Training accuracy and validation accuracy (classification accuracy) vs epoch plot. Here we can observe that as the number of epochs increase, training accuracy is increasing and validation accuracy can also be observed to be increasing initially by towards the end, the validation accuracy is almost remaining same and not improving even though train accuracy is increasing. This can be a sign of overfitting.

Since this model involves fully connected deep networks, it has lot of parameters to learn and it's prone to overfitting and is also susceptible to vanishing gradient problems.

In Conclusion, Convolutional Neural Networks work better with image data than fully connected neural networks and provide better classification.