Report

Author: Medha Rudra
{medha.rudra@colorado.edu}

1) Influence of Network Depth and Regularization Techniques

Methods:

Dataset used for these experiments was cifar10 from Keras. It is a small images classification dataset with 10 target classes:

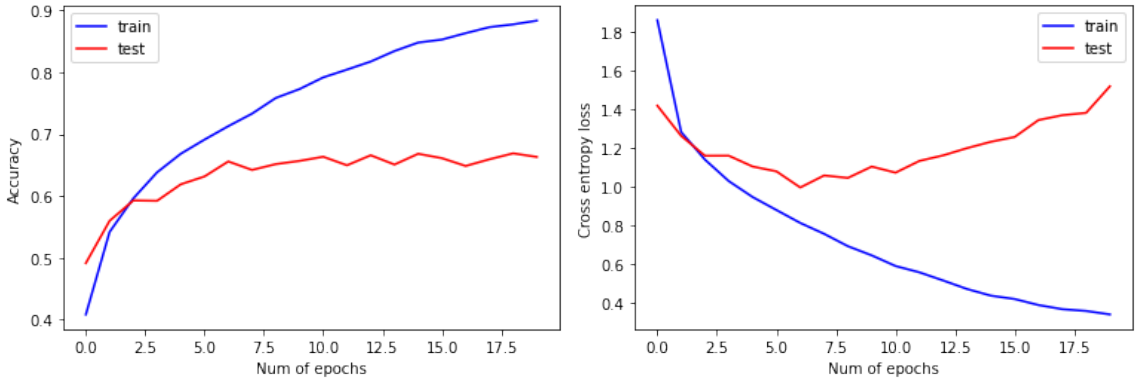
Label	Description
0	airplane
1	automobile
2	bird
3	cat
4	deer
5	dog
6	frog
7	horse
8	ship
9	truck

The dataset is already split into 50,000 training data and 10,000 test data, with each data (image) of size 32x32. This tells us that the train test split is around 83/17, which is close to an 80/20 split as asked for, in the problem. So, we take the default split for our experiments and further split the train dataset into train and validation sets such that the train/validation/test split is roughly 60/20/20.

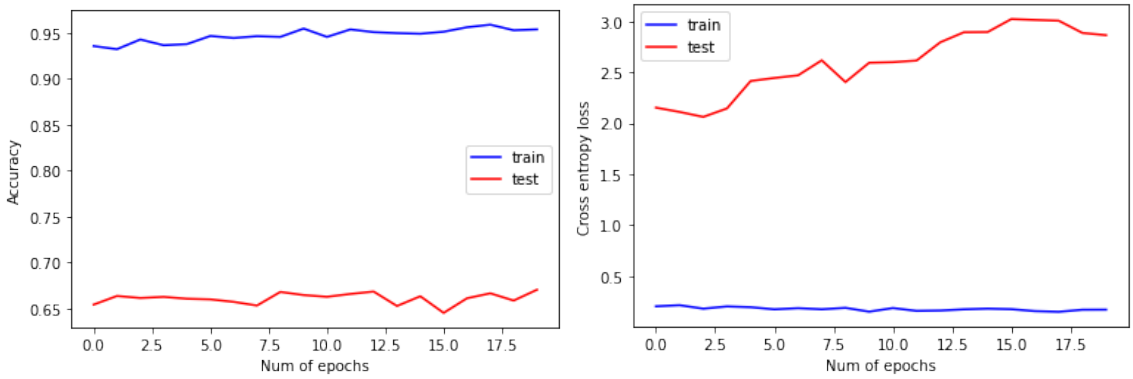
9 models were built and trained with the following specifications:

Hardware Specifications: GPU used on a system with 8 GB RAM, M1 chip.
Training Parameters that were kept constant for each model: number of epochs, same padding (zero padding), activation function in CNN layers and first fully connected layer was ReLU and activation functions in final fully connected layer was softmax, Adam optimizer, Loss = categorical cross entropy, metrics = accuracy.

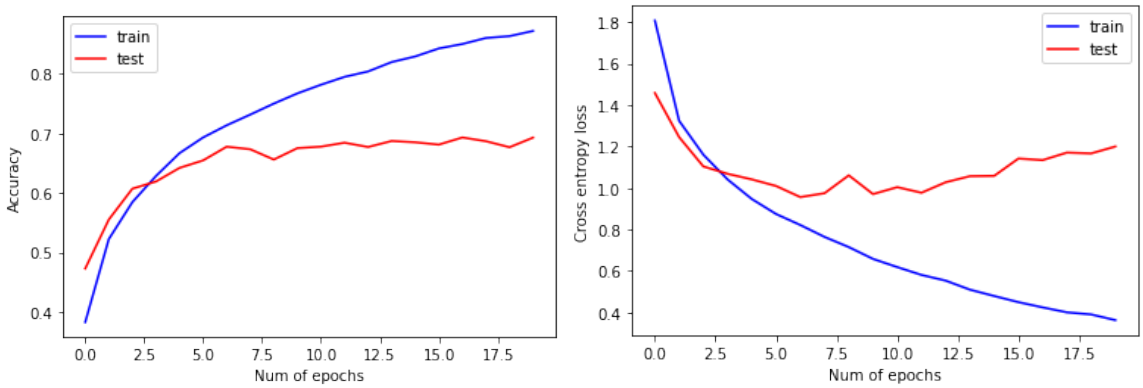
- **Model_1 :**
  Architecture: 4 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 215,474
  Trainable params: 215,474
  Total Training Time: 3.37 mins



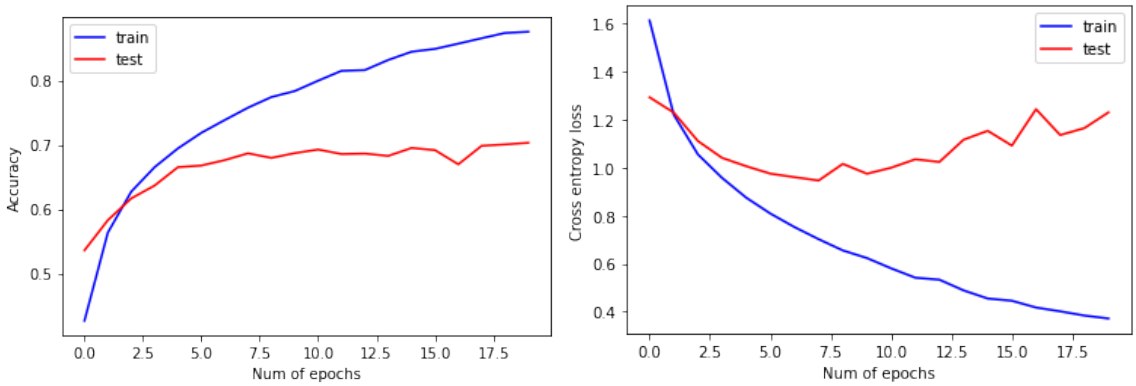
- **Model_2 :**
  Architecture: 3 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 538,002
  Trainable params: 538,002
  Total Training Time: 4.38 mins



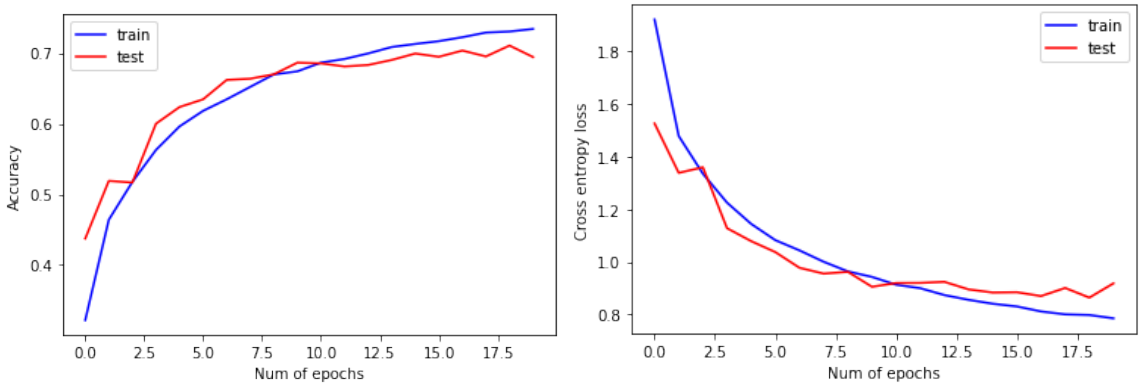
- **Model_3:**
  Architecture: 5 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 197,138
  Trainable params: 197,138
  Total Training Time: 4.4 mins



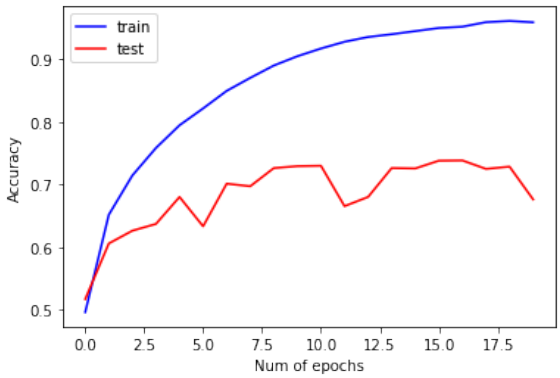
- **Model_4:**
  Architecture: 5 CNN layers with 2 fully connected layers, L2 regularization was applied
  Total params: 197,138
  Trainable params: 197,138
  Total Training Time: 4.4 mins

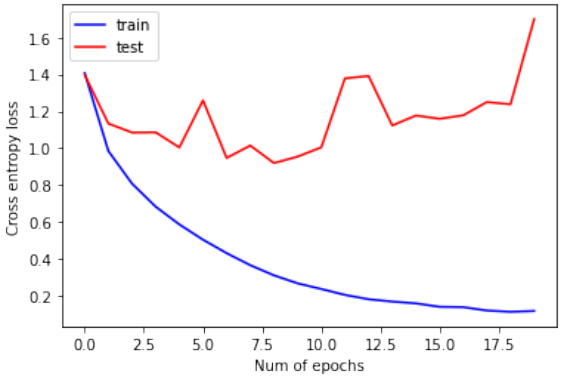


- **Model_5:**
  Architecture: 5 CNN layers with 2 fully connected layers, L2 regularization along with Dropout was applied
  Total params: 197,138
  Trainable params: 197,138
  Non-trainable params: 0
  Total Training Time: 5.42 mins

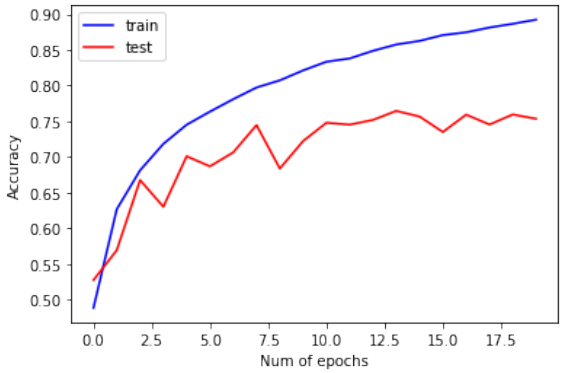


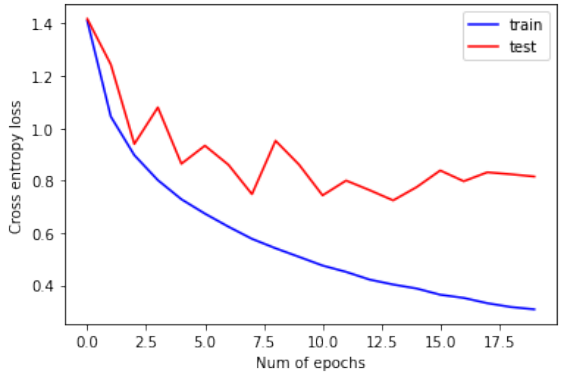
- **Model_6:**
  Architecture: 5 CNN layers with 2 fully connected layers, Batch Normalization was applied
  Total params: 199,410
  Trainable params: 198,274
  Non-trainable params: 1,136
  Total Training Time: 4.39 mins



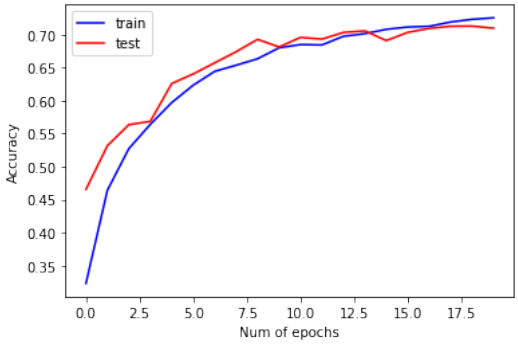


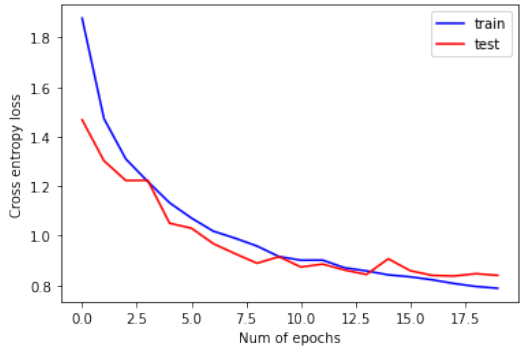
- **Model_7:**
  Architecture: 5 CNN layers with 2 fully connected layers, Batch Normalization along with Dropout was applied
  Total params: 198,034
  Trainable params: 197,586
  Non-trainable params: 448
  Total Training Time: 4.39 mins



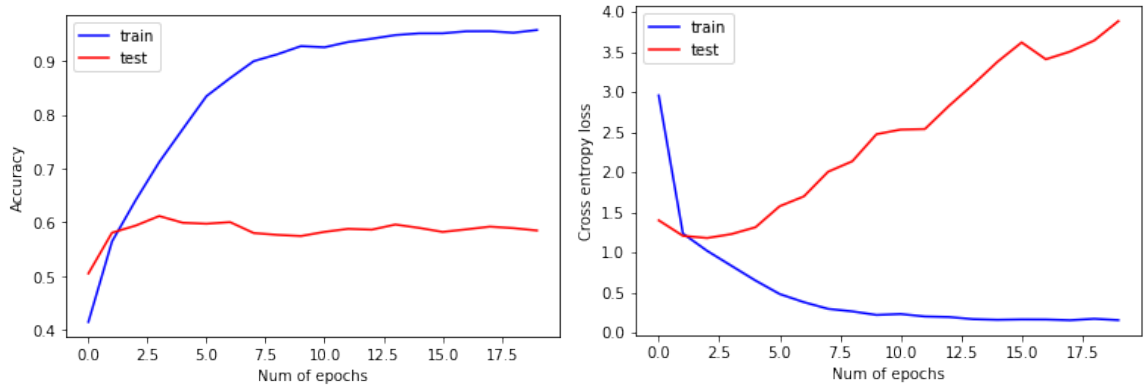


- **Model_8:**
  Architecture: 5 CNN layers with 2 fully connected layers, Dropout was applied
  Total params: 197,138
  Trainable params: 197,138
  Non-trainable params: 0
  Total Training Time: 4.38 mins





- **Model_9:**
  Architecture: 2 CNN layers with 2 fully connected layers, without regularization
  Total params: 2,122,186
  Trainable params: 2,122,186
  Non-trainable params: 0
  Total Training Time: 2.89 mins



Amongst all the above models, best performance on the validation set was shown by model 7 with an accuracy of about 75%. The accuracy on the evaluation split with this model is reported to be about 77%.

Analysis:

It is observed that when we applied regularization, the model learnt slightly better as compared to the models in which regularization was not applied. As evident from the above data, the models are trying to learn a huge number of parameters. Applying regularization on these models, helps us to keep the model complexity in check and consequently tries to prevent overfitting of the model. From the plots of the 9 models, we can see that L2 regularization was not as effective as Dropout or Batch Normalization. Randomly deactivating some neurons in the network is providing the model to learn with different architectures, but as in L2 regularization, only weights are being penalized in case of outliers. Probably that's why, using Dropout is giving better results. We also observe that using Batch Normalization is giving better results than using Dropout. This shows that Batch Normalization provides the regularization effect to an extent that dropout is not needed. In addition, we got the best performing model when we combined dropout with batch normalization, indicating that too much dropout can harm the learning of the model and that batch normalization works sufficiently well for regularization.

Considering the number of CNN layers, we see that the model performs better with 4 or 5 CNN layers as compared to 2 or 3 CNN layers. This is because, with low number of CNN layers, the model is having to learn a huge number of parameters and eventually leads to overfitting. As we add more layers, it has to learn fewer parameters, and this helps control overfitting of the model.

The model takes more time to train when there are more number of convolution layers. It also takes more time to train if regularization is applied to the model as compared to when regularization is not applied. This is because the complexity of the model architecture increases. but considering that we want our model to learn optimally, we can use batch normalization over L2 regularization and instead of having 5-6 CNN layers, we can have 4 CNN layers. This will help us keep an average training time of 3.5-4mins and help the model learn better.

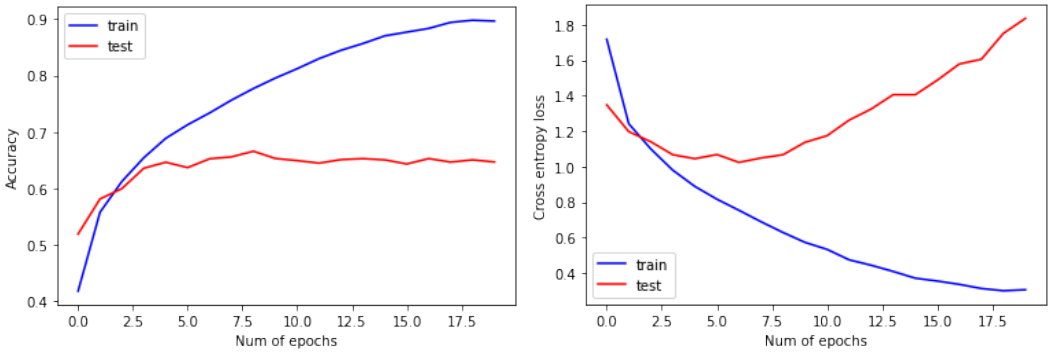
The top performing model performed better when trained on both train and validation data as compared to when trained only on the train data. This is because the train and validation sets together account for 80% of the total dataset and only using train data would account for 60% of the entire dataset. When we trained on 80% of the data, the model was able to learn more as compared to when it learned from 60% of the data – implying that, more the training data, better was the performance because the model had more information to identify the patterns of the dataset and the data it learnt from was relevant and valid.

2) Impact of Training Data Amount (Size of Dataset)

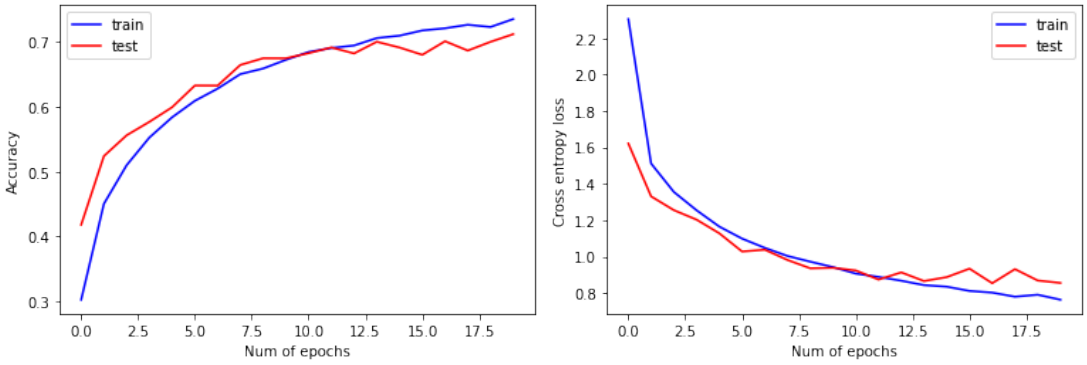
6 models were trained with the following specifications:

Hardware Specifications: GPU used on a system with 8 GB RAM, M1 chip.
Training Parameters that were kept constant for each model: number of epochs, same padding (zero padding), activation function in CNN layers and first fully connected layer was ReLU and activation functions in final fully connected layer was softmax, Adam optimizer, Loss = categorical cross entropy, metrics = accuracy.

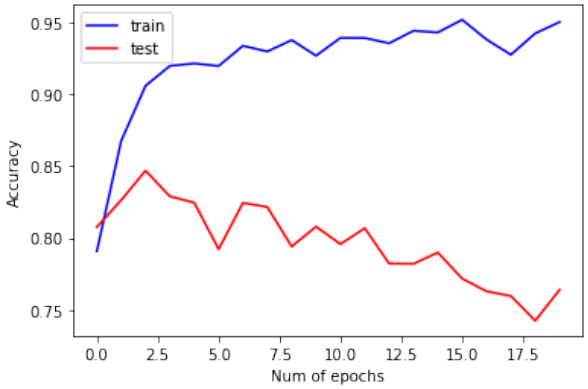
- **Model_1:**
  Architecture: 4 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 370,282
  Trainable params: 370,282
  Non-trainable params: 0
  Training data: 100%
  Total Training Time: 4.4 mins

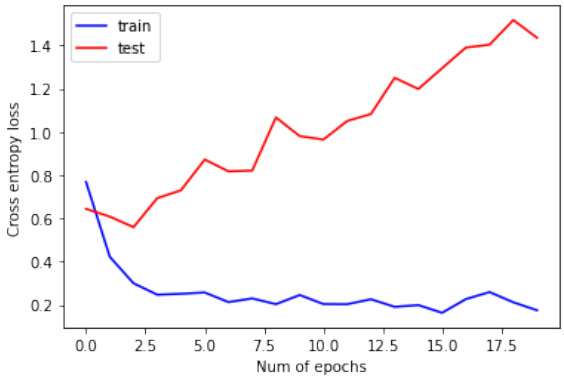


- **Model_2:**
  Architecture: 4 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 370,282
  Trainable params: 370,282
  Non-trainable params: 0
  Training data: 100%
  Total Training Time: 4.4 mins

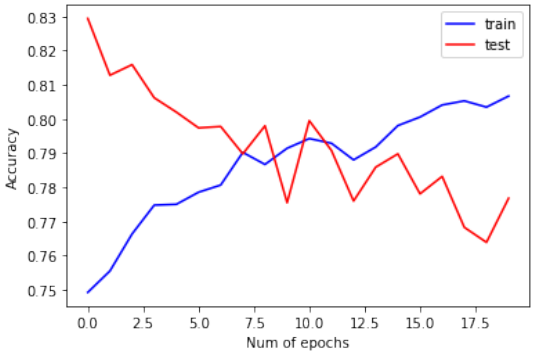


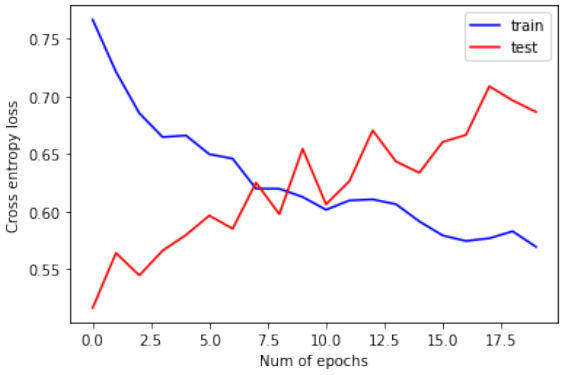
- **Model_3:**
  Architecture: 4 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 370,282
  Trainable params: 370,282
  Non-trainable params: 0
  Training data: 75%
  Total Training Time: 2.4 mins



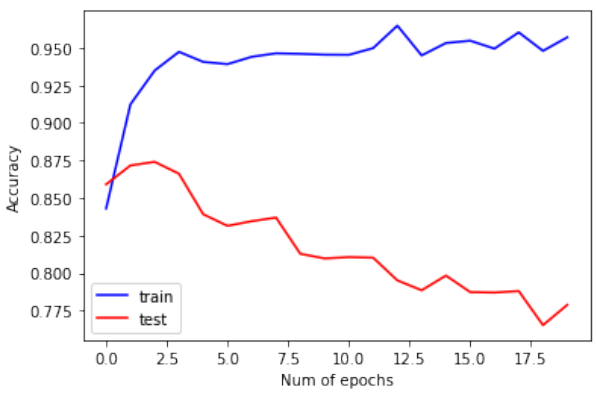


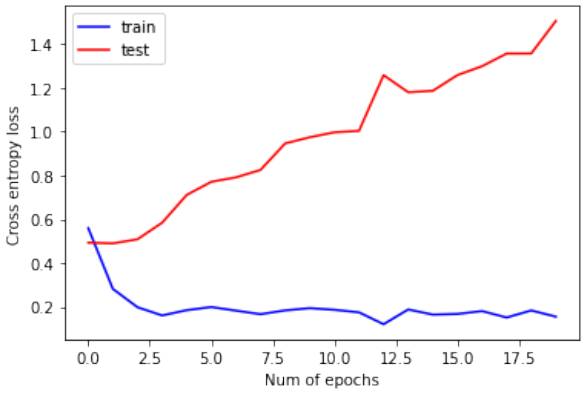
- **Model_4:**
  Architecture: 4 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 370,282
  Trainable params: 370,282
  Non-trainable params: 0
  Training data: 75%
  Total Training Time: 3.37 mins



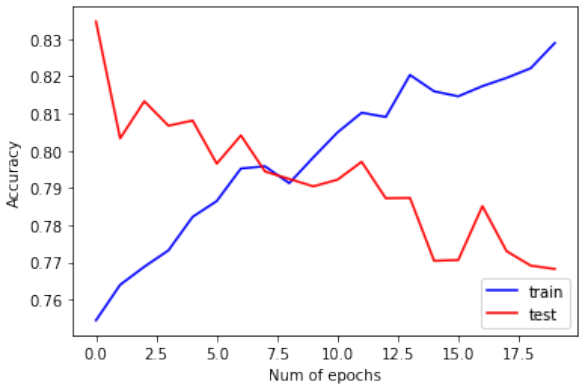


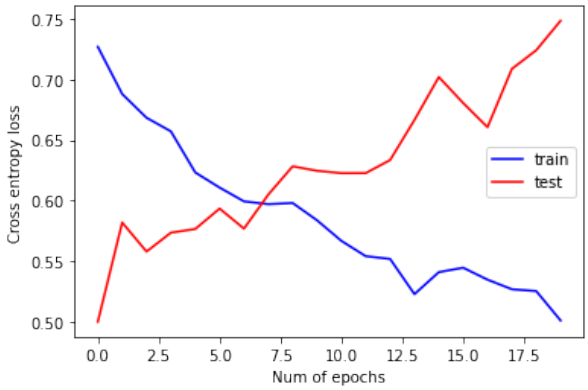
- **Model_5:**
  Architecture: 4 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 370,282
  Trainable params: 370,282
  Non-trainable params: 0
  Training data: 50%
  Total Training Time: 1.79 mins





- **Model_6:**
  Architecture: 4 CNN layers with 2 fully connected layers, no regularization was applied
  Total params: 370,282
  Trainable params: 370,282
  Non-trainable params: 0
  Training data: 50%
  Total Training Time: 2.36 mins





Analysis:

On a general note, we can see that the performance is decreasing with the decrease in the amount of training data available. While the performance is steadily decreasing for models without regularization, the performance decrease is getting subdued if regularization is applied to the model. This is because, regularization is preventing underfitting that would occur due to the model being trained on less amount of data.

From the plots above, we observe that as the training data decreases, the model is unable to learn well. More so, it is underfitting especially when 50% of the training data is being used. This is evident from the curves that show that the model is gradually unable to train with good accuracy and is unable to perform well on test data.

It is observed that when 100% of the training data is used, the model performs best but takes more time. Again, if 50% of the training data is used, the model performs worst but takes less time. Considering the tradeoff between the performance and training time, it would be more prudent to train with 75% of training data as it takes less time and performs near about as good as when trained with 100% train data. We don't want to overfit or underfit the model and want to keep the training time low, hence the choice.