Question 1:

(d)

There are quite many choices in TensorFlow, I choose AdamOptimizer.

According to the paper which introduces Adam Algorithm:

Adam is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments. The method is straightforward to implement, is computationally efficient, has little memory requirements, is invariant to diagonal rescaling of the gradients, and is well suited for problems that are large in terms of data and/or parameters. The method is also appropriate for non-stationary objectives and problems with very noisy and/or sparse gradients.

I try to tune AdamOptimizer's parameters, the result is:

For fully connected model, AdamOptimizer can beat RMSPropOptimizer, but it can not beat GradientDescentOptimizer as well
as MomentumOptimizer.

For convolutional model, after I tune AdamOptimizer's
parameters, AdamOptimizer can beat all three other optimizers.

(e)
For fully connected model:

The train loss, validation error and test error MomentumOptimizer gets finally are 29.4285, 68, 147 respectively. All these three values are the minimum values among four different optimizers, so MomentumOptimizer is the best.

RMSPropOptimizer gets the worst result. The test error RMSPropOptimizer gets finally is 188, which is the maximum values among four different optimizers.

Training and testing performance are almost monotonically related. Training and testing performance achieved
by GradientDescentOptimizer, MomentumOptimizer

and RMSPropOptimizer are monotonically related. Only one exceptional case is that the test error AdamOptimizer gets finally is a little bit smaller than RMSPropOptimizer, but its train loss is larger than the latter one.

(f)

For convolutional model:

AdamOptimizer does the best job. The train loss, validation error and test error gained by AdamOptimizer are the minimum values among four different optimizers. The other four optimizers get almost the same validation error and test error, but their train loss are different. Therefore, training and performance are not monotonically related.

(g)

When compare the results between the fully connected and convolutional models, one interesting phenomena is that all convolutional models with different optimizers get better results (lower error) in validation and test data compared to fully connected models. However, the train loss gained by convolutional models are significantly larger than their corresponding fully connected models except AdamOptimizer, which gets smaller train loss as well as smaller test error.