Yang Liu
3323643

# Lab Exercise 5

COMP6248
yl58n21@soton.ac.uk

## 1 An initial attempt

Since this is a regression task, a simple Mean Squared Error loss function could be used to compute the loss between the computed result and target.

**Figure 1** shows the training and test loss from each training iteration, which is also a performance measure of the model. We can easily find that the model started to overfit since the early stage of the training process. This probably indicates that our model is too complex for our task(way too many parameters).

## 2 A second attempt

By using Global Max Pooling, we reduced a large number of parameters in the first fully connected layer thus reduced the complexity of our model. The problem of overfitting is solved, the two losses are now closer in **Figure 2**, but this is achieved by the rise of the training loss which is not ideal.

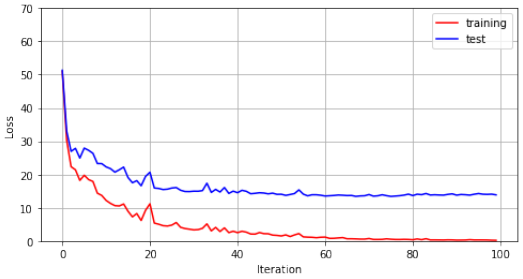


Figure 1: Initial training loss and test loss

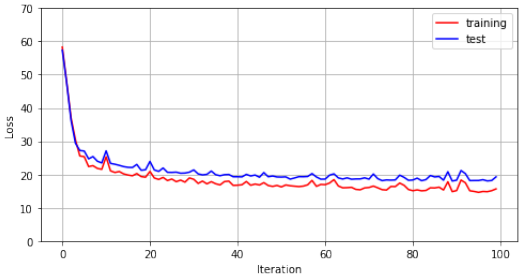


Figure 2: Second training loss and test loss

## 3 Something that actually works?

By adding two dimensions of data which are constructed by equally valued number ranging from -0.5 to 0.5 into original data, we miraculously improved the performance of the model, both the training loss and test loss converge to very small numbers in training process as **Figure 3** shows. This trick is called CoordConv. It turns out that CNN is not really good at learning a mapping between coordinates in (x,y) Cartesian space and one-hot pixel space. But after we added the coordinates channels which is the two dimensions of data we added to the original data, gave convolution access to its own input coordinates. Thus by adding the feature of coordinates of our input data, it performs much better.

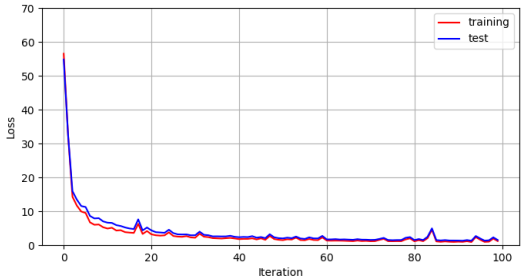


Figure 3: Training loss and test loss