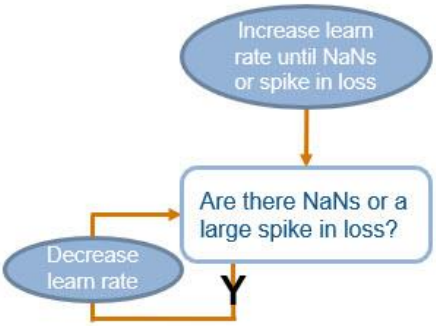
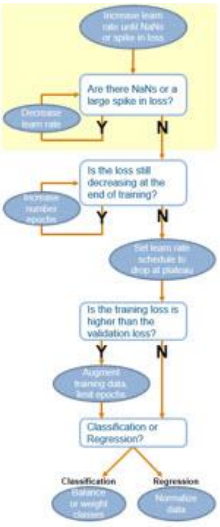
Guidelines for Training Options

1.



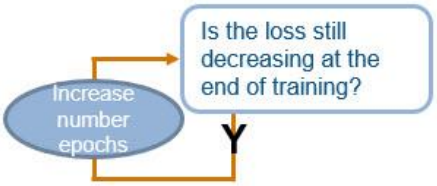


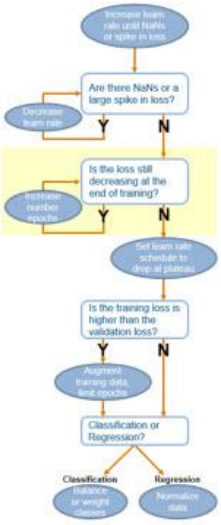
The ideal learning rate is the largest value such that the network trains effectively.

- *If the learning rate is too small, it will take a very long time to train your network*
- *If the learning rate is too big, your loss may evaluate to NaN or you may see a large rapid increase in your loss at the beginning of training*

To find this learning rate, you should increase the learning rate until you see the loss "explode". Then, decrease the loss until training proceeds appropriately.

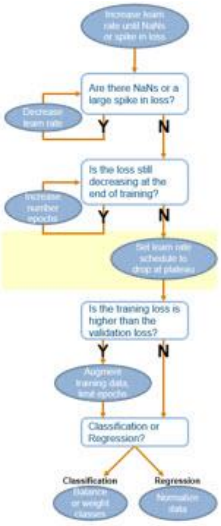
2.





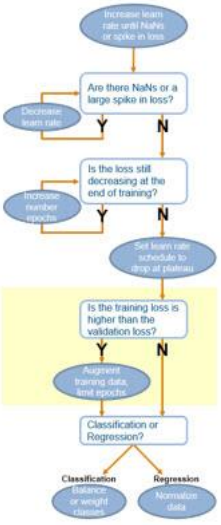
Increasing the maximum epochs ensures that your network finishes learning. If the accuracy and loss have plateaued for many epochs, you can also decrease the number of epochs to decrease training time.

3.



Once the training has converged, you can set the `'LearnRateSchedule'` and `'LearnRateDropPeriod'` to decrease the learning rate at this point. Make sure the training has converged before the learning rate is lowered.

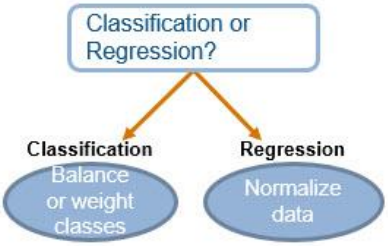
4.

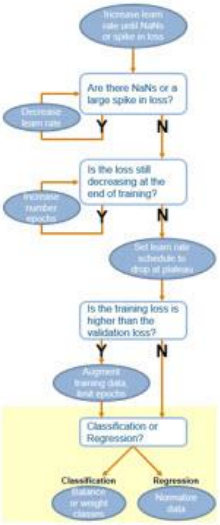


Validation is key to determining if your network is overfitting. If so, adding data augmentation or limiting the number epochs can help.

You can also increase the factor L_2 regularization. This option decays the weights.

5.





If you have very unbalanced classes, try splitting the data so that the training images have an equal number of samples from each class.

If you are performing regression, normalizing can help stabilize and speed up the network training.

Additional Considerations when Designing an Architecture

- If you are training from scratch, you can also try modifying your network architecture to prevent overfitting. Try adding a dropout layer before your last fully connected or convolutional layer.
- It is also possible to *underfit* if your training and validation accuracy are equal. In this case, try increasing the depth of your network.