**Week 2 – Notes**

**Error Analysis**

**Carrying Out Error Analysis**

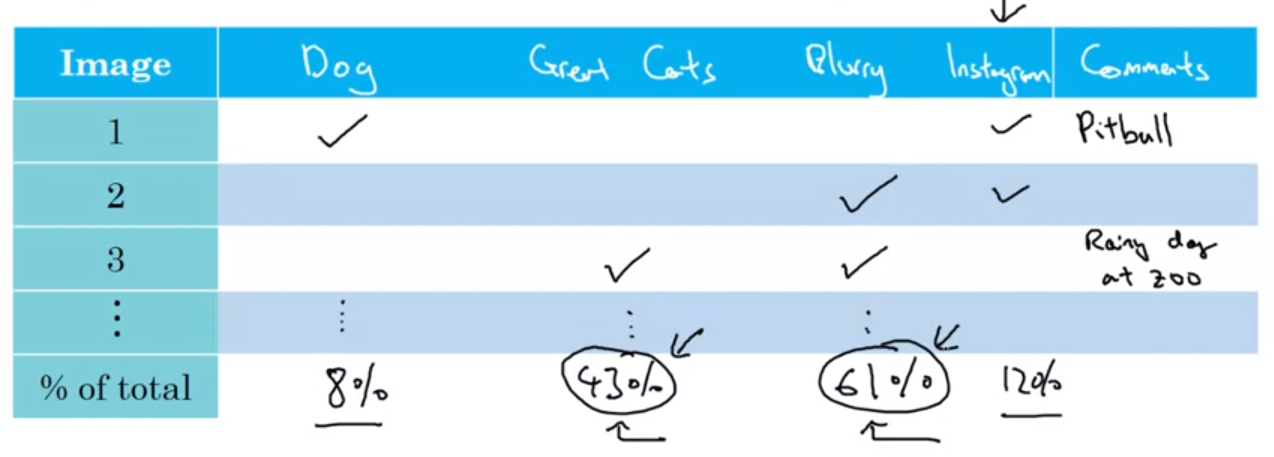
Before trying to fix the error rate by going down one path, you should perform an error analysis

Get ~100 dev set examples that were mislabeled by the classifier

Count how many include the category you want to fix (e.g.: optimize on dogs while building a cat detector)

For example, let’s say that the error is 10% and you find 5 / 100 mislabeled examples that are dogs => the ceiling (maximum improvement) is of 0.5%

If you have multiple improv. ideas, create a table for all of them and perform the error analysis



In this way, based on a 5–10-minute effort, you can take an informed decision

**Cleaning Up Incorrectly Labeled Data**

Usually, deep learning algorithms are quite robust to random errors in the training set and avoids learning from them

However, this error should not be systematic (the labeler mislabeled only white dogs as cats) because the algorithm will learn that white dogs should be labeled as cats

When it comes to the dev data set it’s more sensible

Let’s assume that the overall dev set error is 10%; if the error due to incorrect labels is 0.6% it’s not that much, considering that 9.4% represent errors from other causes

However, if the overall dev set error is 2% and the incorrect labels represent 0.6%, then this would restrict a proper evaluation of different models (the uncertainty of the quality of models is too high)

If we want to correct the incorrect labels of dev and test:

Apply the same process to these data sets so that they continue to come from the same distrib.

Also, consider examining examples your algorithm got right as well

If you correct only dev and test data sets, they can have slightly different distributions compared to the train data set

**Build your First System Quickly, then Iterate**

For a new problem you don’t know with what edge cases to start, so the best approach for new problems is to:

Set up dev / test set and metric

Build initial systems quickly

Use Bias / Variance analysis and Error analysis to prioritize next steps

If you deal with well-known problems, with a lot of literature, then you can start by building something more complex

More teams are overthinking compared to the ones which underestimate the problem (and then build to simple solutions)

**Mismatched Training and Dev/Test Set**

**Training and Testing on Different Distributions**

Let’s day you build a classifier for a mobile app and you have 2 types of images: from webpages (200k) and from mobile app (10k)

You can create your train / dev / test sets in 2 ways:

1. Combine the data sources, shuffle the images and create the sets
2. Use all web images + 5k from users for training, and the rest of 5k from users for dev and test

The first case isn’t ideal because you optimize your model to predict only a small portion of images that actually will be seen in production

The second option is better, because you optimize your model to classify images that actually will be in production, even though the train and dev / test distributions are different; we rely on the model’s capability to generalize; additionally, there’s the option to use all images from user only for dev and test sets