Structuring Machine Learning Projects

Week 1:

1. Chain of assumption in ML:
    a. Fit training set well on cost function.(Human level perfomacne)
        i. Bigger network or optimization
    b. Fit dev set well on cost funct
        i. Regularize or bigger training set
    c. Fit test set well on cost func
        i. Bigger dev set
    d. Performs well in real world
        i. Change dev set or cost function
2. Single number eval metric:
    a. So if classifier A has 95% precision, this means that when classifier A says something is a cat, there's a 95% chance it really is a cat. And recall is, of all the images that really are cats, what percentage were correctly recognized by your classifier? So what percentage of actual cats, Are correctly recognized?
    b. So if classifier A is 90% recall, this means that of all of the images in, say, your dev sets that really are cats, classifier A accurately pulled out 90% of them.
    c. F1 score – Average of precision and recall(Harmonic mean)
3. Train/dev/test distribution:
    a. Dev and test must come from the same distribution
    b. Choose a dev set and test set to reflect your data you expect to get in future and consider important to well on
4. Size of the dev and test sets:
    a. Set your test set to be big enough to give high confidence in the overall performance of your system
5. When to change dev/test set and metrics:
    a. But the high level of take away is, if you find that evaluation metric is not giving the correct rank order preference for what is actually better algorithm,
6. Orthogonalization:
    a. Define metrics
    b. Worry separately about how to do well on this metric
7. The two fundamental assumption of supervised learning:
    a. You can fit the training set pretty well
    b. The training set performance generalizes pretty well to the dev/test set
8. Reducing avoidable bias and variance:
    a. Reduce avoidable bias:
        i. Train big model
        ii. Train longer or better optimization algorithms
        iii. Find better set of hyperparameters
    b. Reduce variance:
        i. More data
        ii. Regularization (L2, dropout, data augmentation)
        iii. Search better hyperparameters

Week 2:

9. Correcting incorrect dev/test set examples:

    a. Apply some process to your dev and test sets to make sure the come from the same distribution
    b. Consider examining examples of your algorithm got right as well as ones it got wrong
10. Build your system quickly, then iterate:
    a. Set up dev/test set and metric
    b. Build initial system quickly
    c. Use Bias/Variance analysis and error analysis to prioritize next steps
11. Addressing data mismatch:
    a. Carry out manual error analysis to try to understand difference between training and dev/test sets
    b. Make training data more similar or collect more data similar to dev/test sets
12. Transfer learning:
    a. Transfer learning is a machine learning technique where a model trained on one task is re-purposed on a second related task.
    b. Transfer learning makes sense when you have a lot of data for the problem you're transferring from and usually relatively less data for the problem you're transferring to.
13. Multi-task learning:
    a. Training on a set of tasks that could benefit from having shared low-level features
    b. Amount of data you have for each task is quite similar
    c. Enables you to train an NN doing with many tasks
14. Pros and cons of End-to-end deep learning:
    a. Pros:
        i. Let data speak
        ii. Less hand-designing of components needed
    b. Cons:
        i. Need a lot of data
        ii. Exclude potentially useful hand-designing things
    c. Applying E2E:
        i. Key question: Do you have sufficient data to learn a function of the complexity needed to map x to y