CS 1699 Special Topics: Deep Learning

First Exam: Review Guide

February 10, 2020

General info: The exam will include all lectures up until February 13 (inclusive). The format will include multiple choice and true/false questions, short answers, and applications of algorithms on small toy problems. The exam will be on February 18, at our usual class time, in our classroom. You do not need to (and shouldn’t) bring calculators or other aids.

Concepts and algorithms to review:

1. Intro
   1. How would you describe deep learning to a non-expert?
   2. What are some example deep learning (or machine learning) tasks and problems?
   3. Why are these tasks challenging?
   4. What does the overall DL/ML framework look like?
   5. What do X and Y typically denote in a DL/ML framework?
   6. What is one example approach of predicting the label of a text or image (e.g. spam/not, object category)?
   7. What would you say the hidden layers of a neural network capture?
   8. How do we test ML/DL systems? What does the evaluation pipeline look like?
   9. What are the train, validation, and test sets? How do we use them?
   10. What is cross-validation?
   11. What is a loss function?
   12. What is optimization?
   13. Why is generalization important?
   14. What is underfitting? What is overfitting?
   15. How do we measure complexity of a model, e.g. for polynomial curve fitting?
   16. What is the effect of the amount of training data available on overfitting, for a model of fixed complexity?
   17. What is regularization? What is its effect on overfitting?
   18. How do we compute the value of a regularization term?
2. Neural network basics
   1. How do we compute activations in a neural network?
   2. What are some common non-linear activation functions, and what are their advantages/disadvantages?
   3. What does it mean to train a neural network? What are the inputs and outputs? What are the steps?
   4. Why do deep networks require lots of data to train?
   5. What are hyperparameters?
   6. What are some common losses?
   7. What is gradient descent?
   8. How do we use a loss to train a deep network?
   9. What is learning rate and why is it important?
   10. How do we learn the weights in a multi-layer neural network?
   11. How do we use the chain rule to compute gradients?
   12. What does the full algorithm for training a neural network look like, given a fixed architecture, activation functions, and loss choice?
3. Training part 2
   1. What are some ways of preprocessing data for use in a neural network?
   2. What are some possible ways to initialize weights in a neural network? Why is initialization non-trivial?
   3. What is batch normalization, and how do we give the network flexibility to decide how much normalization to apply?
   4. How do we use learning curves (on the training and validation set) to gauge how well training is proceeding?
   5. How can we prevent overfitting in a neural network?
   6. What is dropout and why is it useful?
   7. What is data augmentation and why is it useful?
   8. What is transfer learning and why is it useful?
   9. What are some of the ways to do transfer learning?
   10. What variants of gradient descent are there? What is mini-batch gradient descent?
   11. What is some of the hardware needed for deep learning? What are some hardware challenges/tradeoffs affecting the speed of training?
   12. What are some conditions for successful training?
   13. What are some reasons why training may not be successful?
   14. What are some desirable and undesirable properties of the loss surface?
   15. Why are local minima problematic/not? Why are saddle points problematic?
   16. What are some desirable properties of the learning rate(s)?
   17. What is a condition number and why is it important?
   18. What are some problems with gradient descent?
   19. What are some optimization strategies beyond gradient descent? What intuitions motivate these optimization strategies?
   20. What is learning rate decay?
   21. What are some tips for choosing the values of important hyperparameters? How do we decide what is a good setting of these hyperparameters?
   22. What is a computation graph, and why is it useful?
   23. What role do different operators (e.g. plus, max) play when backpropagating gradients? What does each gate do in terms of propagating the gradient?
   24. How do we use Jacobian matrices?

Practical skills:

1. Write down some possible features (X) that could be used in the non-DL ML setting (i.e. where a network is not computing features)
2. Write down some possible (hypothetical) weights for a linear model, given X and Y
3. Write down the equations for some possible loss functions
4. Compute activations for some node in a neural network, given inputs, architecture/connections and choice of non-linear activation functions
5. Compute the loss values for a given loss function, from scores for different samples, and the ground-truth labels on the samples
6. Compute gradient numerically
7. Compute a weight update (using gradient descent)
8. Write down pseudocode for gradient descent, gradient descent + momentum, AdaGrad, and RMSProp
9. Compute a few iterations of gradient descent by hand, as in HW2 but with a simpler architecture
10. Show the terms required to compute dE/dw for some fixed weight, and show how each term can be calculated analytically
11. Show how to construct a computation graph for a simple problem
12. Using a given computation graph, compute the values of the gradients for the weights