Convex Optimization: Homework 4

Instructor: Yuanzhi Li
Carnegie Mellon University
Due: May 8th, 2020 [No Extensions]

This homework consists of many conceptual questions. To obtain an answer, you are not allowed to use the *exact* sentences from the slides, but you can rephrase them or use the exact sentences from the course videos. Please cite lecture slides and videos appropriately.

1 Over-parameterization (30 points + 20 bonus points)

1.1 Intuition (10 points) [Cinnie]

Explain the intuition behind how over-parameterization can make a non-convex optimization problem in machine learning easier to solve.

1.2 Investigation (20 points + 20 bonus points) [Stefani]

This is a coding question: For $x \in \mathbb{R}^d$, consider the true labeling function $y(x) = \sum_{i=1}^{d} \sigma(x_i)$, where σ is the ReLU activation.

Now, suppose you want to learn it using a model $h(W, x) = \sum_{i \in [\mathbf{m}]} \sigma(\langle w_i, x \rangle)$, where $W = \{w_i\}_{i \in [d]}$ are trainable parameters. Consider the following ℓ_2 loss:

$$f(W) = \mathop{\mathbb{E}}_{x \sim \mathcal{N}(0,I)} [(y(x) - h(W, x))^2].$$

Write code to minimize $f(W)$ using (mini-batch) stochastic gradient descent, starting from a random initialization where each w_i i.i.d. $\sim \mathcal{N}(0, \frac{1}{d} I)$.

Consider two settings:

1. proper-parameterization: $d = m = 20$, and
2. over-parameterization: $d = 20, m = 200$.

Plot the function value $f(W)$ (you can calculate it approximately by randomly sampling x) v.s. number of iterations. You can pick your own mini-batch size.

Bonus (20 points): Study this problem for a larger set of m-values. Try to understand how the problem changes as a function of m (there's no single "right answer"). Provide 1 plot as a function of m (m on the x-axis, and something else - your choice - on the y-axis) with an explanation of the plot and how it reflects your improved understanding of the problem post-investigation.

2 Large learning rate (10 points) [Vishwak]

Explain the most important goal of using an initial large learning rate when training a neural network for image classification. What is the underlying mechanism?

3 Adversarial training (10 points) [Vishwak]

In the sparse coding example shown in lecture 24, explain how a neural network with ReLU activation can learn the target function, which is robust to ℓ_2 norm bounded perturbations with radius $\tau = \frac{1}{\sqrt{d}}$. Explain why a linear function can not do it.

4 Batch normalization: (10 points) [Vishwak]

Consider the batch normalization for ridge regression example shown in the slides. The objective is given by:

$$f(w) = \left\| w^* - \frac{w}{\|w\|_2} \right\|_2^2 + \lambda \|w\|_2^2.$$

Consider the case where $\lambda > 0$ is a fixed constant. $w \in \mathbb{R}^d$ and $\|w^*\|_2 = 1$.

You want to show that the geometry of the function f with batch normalization is pretty bad:

(1) Show that $f(w) \to f^*$ for $w = \epsilon w^*$ as $\epsilon \to 0^+$. Here we define $f^* = \inf_w f(w)$.

(2) Show that around $w = 0$, the function f is not Lipschitz (gradient does not have a bounded ℓ_2 norm), nor smooth (Hessian matrix does not have a bounded spectral norm).

5 Min-max optimization (35 points)

5.1 Necessity of second-order local optimal condition (15 points) [Jerry]

Consider $f(x, y) = 0.2xy - \cos(y)$ defined over $x \in [-1, 1]$, $y \in [-2\pi, 2\pi]$. Find the the global min-max optimal solutions (x^*, y^*), and explain why they are not (second order) local min-max optimal solutions.

5.2 Generative Adversarial Networks! (20 points) [Cinnie]

Use the code at this link. Train it using

1. The original setup: the learning rate is 0.0002 for both discriminator and generator, batch size is 100.
2. Discriminator too powerful: The generator's learning rate is decrease to 0.0001 and the discriminator's is increased to 0.001.
3. Low noise: The batch size is increased to 1000 (the learning rates are still 0.0002) and run for $T = 45$ epochs.

Report output of the generator after $T = 20$ epochs (except the 3rd experiment) (show some pictures). Use the principles showed in class to explain your findings.

Note: You can use Google Colab to run your experiments.