# Final Project in Advanced Lectures in Learning Theory

Roi Livni

January 3, 2022

## 1 Part I

In this part we will revisit and try to experimentally study the role of regularization, implicit bias, and optimization in learning. For this exercise you will need to download MNIST data set `http://yann.lecun.com/exdb/mnist/` on which we will run our experiments.

**Remark.** *Feel free to choose any other data set as long as you properly describe it as well as any preprocessing procedure you perform over it.*

- The original data-set is divided to train and test, do not use this partition, just construct one big data set (which you will later divide to train-test by yourself.

  Many of the results we depicted in this lecture assume the data is bounded in a ball of diameter D

  - You are allowed, and should, at this point perform any preprocessing that you believe will improve the results in the next section (scaling, shifting etc..). But make sure you describe every type of preprocessing you propose and did on the dataset.
  - It is also okay to reparameterize the original features and use any other representation of the data (for example, consider the output of the last hidden layer of some architecture you want). As long as you explain what you did, how you choose the features and how your choice effects the experiments.

- Next, we will want (for simplicity) to turn the above problem to binary. Generate a binary problem from the MNIST dataset (for example, by choosing two figures against another, or by clustering the figures to two sets. Create 3 different classification problem this way to experiment with. In each of these experiments we will want to learn a classifier by minimizing the loss

$$\mathcal{L}(w) = \mathop{\mathbb{E}}_{(x,y)\sim D}[\ell(\mathbf{w}\cdot\mathbf{x}, y)].$$

Where $D$ is a distribution that uniformly picks a point from the dataset and $\ell$ is some **convex** loss function for your choice (e.g. square loss, hinge loss, log-loss etc...).

- We will study the performance of the following optimization algorithms: GD, Constrained GD, Regularized GD and SGD. Recall that for a loss function $F$:

  1. For GD we consider the update rule:
  $$w_t - \eta \nabla F(w_t).$$
  2. For constrained GD we consider the update rule[1]:
  $$w_{t+1/2} = w_{t+1} - \eta \nabla F(w_t) \quad w_{t+1} = \Pi_K(w_{t+1/2}).$$
  3. For regularized GD we perform GD over the following regularized objective
  $$\lambda \|w\|^2 + F(w).$$
  4. For SGD we, at each iteration, draw a fresh example $(x, y) \sim D_S$ from the data set, without repetitions, and perform an update step:
  $$w_{t+1} = w_t - \eta \nabla \ell(w \cdot x, y).$$
  (here we assume $F(w) = \frac{1}{t} \sum_{i=1}^{t} \ell(w \cdot x_i, y_i)$)

  We will want to compare the performance of each of these algorithms in terms of generalization and optimization. Divide your set into a *train* set and a *test* set and run the following experiments: Repeat the experiments 10 times, by considering different random partitions (all graphs should show the *averaged behavior*: for example if you have a graph that depicts the optimization error at time $t$, then the graph should show the *average* optimization error over all 10 realizations.

  **Experiment A: Optimization** First, we will want to compare the running time of each of these algorithms. For this experiment use *only* the training set.

  1. Using the lecture notes and what we studied in class, choose "theoretically justified" learning rates and other parameters (such as $\lambda$ in regularized GD). State your choice and provide the justification.
  2. Compare the running time of each algorithm with the choice of parameters: Namely provide a curve that shows the optimization error (i.e. *training error*) vs. number of iterations with the give choice of hyperparameters (i.e. $\lambda$, $\eta$ etc..).

[1] You might want to choose $K = \{w : \|w\| \leq B\}$ for some set $B$ but you can choose any other constraint. The parameter $B$ depends on your preprocessing and discuss your choice (you can experiment with different choices)

3. Consider regularized-GD and improve the training time: For example, experiment with different learning rates or with different regularization coefficient.
4. Discuss the results and conclude, are all algorithms equal or do some perform better?

**Experiment B: Generalization** Here, we will divide our dataset into two: test set and train set. We will denote the train set by $S$ and we will consider the empirical loss

$$\mathcal{L}_S(w) = \frac{1}{t}\sum_{i=1}^{t} \ell(\mathbf{w}\cdot\mathbf{x}, y).$$

- Repeat the last experiments with both the theoretically justified parameters as well as those you found to be empirically optimal for optimization.
- Compare the performance of each model in terms of *test*-error.
- Are the parameters that lead to optimal optimization error also lead to optimal test error?
- Analyze also the 0-1 loss over the test set (namely, measure the performance of classification).

# 2 Part III

In this part we will review Theorem 9.12 from the lecture (due to `https://arxiv.org/pdf/2006.06914.pdf`)

In this paper the authors show that, if $F = \mathbb{E}_{z\sim D}[f(w,z)]$ is a stochastic convex function, and we choose $w_S$ by running GD for $T$ steps with step size $\eta$ over the empirical risk (i.e. $\frac{1}{|S|}\sum_{i=1}^{m} f(w,z_i)$ then for any step size, running GD leads to the following generalization bound (see section 3.4 therein):

$$\mathop{\mathbb{E}}_{S\sim D} F(w_S) \leq 4L^2\sqrt{T}\eta + \frac{4L^2T\eta}{m} + \frac{R^2}{2\eta T} + \frac{\eta L^2}{2}.$$

- Choose sample complexity, $m$ a step size, $\eta$, no. of Oracle calls, $T$, that will lead to meaning-full guarantees. In other words, how large should $m$ be and how should we choose $\eta$ and $T$ as functions of $\epsilon$ if we want to obtain a test error of at most $\epsilon$?
- Discuss the variant of GD that this result induces, compare it to the complexity of SGD in terms of sample complexity and running time?
- Run GD (unregulrized) on several instances (you can use the mnist data set) and over a **non-smooth**, **non-strongly convex** loss function (e.g. hinge-loss) and verify what is the optimal learning rate to choose if we wish to minimize the test error?

- Compare the empirical performance of GD to the that of Regularized GD. Which one is faster in terms of iteration complexity?