CS 182/282A Scribing

Harsh Srivastav, Seunghoon Paik

Jan 30, 2023

1 Introduction

1.1 Recap

A ReLU network with single fully connected layer:

$$f(x) = W_2 \cdot \text{ReLU}(W_1 x + b_1) + b_2,$$

where all values are vectors or matrices, and ReLU is on the element-wise sense.

- Motivation: Piecewise linear functions are universal approximators.
- Some redundancy between W_1 and W_2: they both affect the slopes of the linear segment.
  - Question: how does certain redundancy in networks actually affect what you actually learn?
  - **Important question for any network**: does this network – trained on the actual data using the learning algorithms we have – **actually learn** the pattern of interests?

1.2 Today

Some tools and ways of thinking that are building towards on understanding of what actually makes the network favor certain kinds of things (in terms of what the network actually learns). Mostly about:

- gradient descent;
- OLS;
- ridge regression.

2 Regularization

2.1 Pragmatic view of the regularization

- By adding some penalty term, shape the optimization to learn something more that we like.
- Note that the (explicit) regularization does not change the expressive power of the network.
- Therefore, you only change what the optimization is favoring.

2.2 Least squares with ridge

The standard form of the regularization is

$$L_\theta = \frac{1}{n} \sum_{i=1}^{n} \ell_{\text{train}}\big(y_i, f_\theta(x_i)\big) + R(\theta).$$

As an example, observe the least squares with ridge. The task is finding w which gives $y \approx Xw$, where $X \in \mathbb{R}^{n \times d}$, $w \in \mathbb{R}^d$, and $y \in \mathbb{R}^n$. The closed form solution with the optimization cost of

$$\|y - Xw\|_2^2 + \lambda \|w\|_2^2 \tag{1}$$

is

$$\hat{w} = \left(X^\mathsf{T} X + \lambda \mathbf{I}\right)^{-1} X^\mathsf{T} y. \tag{2}$$

Throughout today's lecture, we will understand this in different ways. Some details will be omitted since they are covered by the homework.

3 Perspective 1: Gradient descent

The first part of (1) is OLS. When we perform gradient descent (with a learning rate η) to (a) OLS and (b) Ridge regression, at t-th step, we move:

(a) OLS: $2\eta \cdot X^\mathsf{T}(y - Xw_t)$. This can be understood as an adjustment (by X^T) of the residual $y - Xw_t$.

(b) Ridge: $2\eta \cdot [X^\mathsf{T}(y - Xw_t) - \lambda w_t]$. The effect of $-\lambda w_t$ term is "shrinking". This is more clear on

$$w_{t+1} = (1 - 2\eta\lambda) \cdot w_t + 2\eta X^\mathsf{T}(y - Xw_t). \tag{3}$$

$1 - 2\eta\lambda$, which is called the "weight decay", is between 0 and 1 with small η and λ. If the second term of (3) is zero (or close to 0), the weight w decays to 0 exponentially fast.

Note 1. What if $\lambda \gg 1$? Then, the optimization problem above weights $||w||^2$ heavily and makes $||w||$ smaller (moving w closer to the origin) when minimizing the objective function.

Note 2. One of the common mistakes in practice is doing both (a) adding the explicit ridge type regularizer to the loss function and (b) enabling weight decay option. This is a redundancy, and you may get more of "weight decay" effect than you originally wanted.

4 Perspective 2: Leveraging SVD for better understanding

Regarding linear algebra and Euclidean norm, a very powerful tool is observing in some different coordinate systems to see more clearly what is going on. In particular, in $\|\cdot\|_2$, the SVD is very useful.

The full SVD $X \in \mathbb{R}^{n \times d}$ can be written as

$$X = U \Sigma V^\mathsf{T}$$

where $U \in \mathbb{R}^{n \times n}$, $\Sigma \in \mathbb{R}^{n \times d}$, $V \in \mathbb{R}^{d \times d}$, U and V are orthonormal matrices, and Σ is "diagonal".

4.1 OLS

$$\begin{aligned}
Xw \approx y \quad &\Leftrightarrow \quad U\Sigma V^\mathsf{T} w \approx y \\
&\Leftrightarrow \quad \Sigma V^\mathsf{T} w \approx U^\mathsf{T} y \\
&\Leftrightarrow \quad \Sigma \tilde{w} \approx \tilde{y} \quad \text{(Changing coordinate : } \tilde{w} = V^\mathsf{T} w,\ \tilde{y} = U^\mathsf{T} y\text{)} \\
&\Leftrightarrow \quad \begin{bmatrix} \sigma_1 & & \mathbf{0} \\ & \ddots & & \mathbf{0}_{(d-n)\times n} \\ \mathbf{0} & & \sigma_n \end{bmatrix} \tilde{w} \approx \tilde{y} \quad \text{(Assume } n < d\text{)} \\
&\Leftrightarrow \quad \tilde{w}_i = \begin{cases} \frac{1}{\sigma_i}\tilde{y}_i & : i \le n \\ 0 & : \text{o.w.} \end{cases}.
\end{aligned}$$

The last relation comes from approximating $\Sigma\tilde{w} \approx \tilde{y}$ using the min-norm

$$\tilde{w} = \Sigma^T(\Sigma\Sigma^T)^{-1}\tilde{y} = \begin{bmatrix} \sigma_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \sigma_d \\ & \mathbf{0}_{(n-d)\times d} & \end{bmatrix} \begin{bmatrix} \frac{1}{\sigma_1^2} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \frac{1}{\sigma_d^2} \end{bmatrix} \tilde{y} = \begin{bmatrix} \frac{1}{\sigma_1} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \frac{1}{\sigma_d} \\ & \mathbf{0}_{(n-d)\times d} & \end{bmatrix} \tilde{y}$$

If instead $n \geq d$, then Σ is a tall matrix instead, which gives, using the least squares solution of $\Sigma\tilde{w} \approx \tilde{y}$ approximated by $\tilde{w} = (\Sigma^T\Sigma)^{-1}\Sigma^T\tilde{y}$,

$$\begin{bmatrix} \sigma_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \sigma_d \\ & \mathbf{0}_{d\times(n-d)} & \end{bmatrix} \tilde{w} \approx \tilde{y} \quad (\text{Assume } n \geq d) \Leftrightarrow \tilde{w}_i = \frac{1}{\sigma_i}\tilde{y}_i, i \leq d$$

Both cases above can be combined for brevity as

$$\tilde{w}_i = \begin{cases} \frac{1}{\sigma_i}\tilde{y}_i & : i \leq \min(n,d) \\ 0 & : \text{o.w.} \end{cases}.$$

4.2 Ridge regression

With similar observation as the OLS case, since U and V are orthonormal matrices, the ridge regression of the form (1) is equivalent to

$$\|\tilde{y} - \Sigma\tilde{w}\|_2^2 + \lambda\|w\|_2^2.$$

This optimization problem can be decoupled into n scalar ridge problems for each w_i, since Σ is "diagonal". Substituting the SVD simplification into the Ridge solution,

$$\begin{aligned} \hat{w} = \left(\Sigma^\mathsf{T}\Sigma + \lambda\mathbf{I}\right)^{-1}\Sigma^\mathsf{T}\tilde{y} &= \begin{bmatrix} \frac{1}{\sigma_1^2+\lambda} & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \frac{1}{\sigma_d^2+\lambda} \end{bmatrix} \begin{bmatrix} \sigma_1 & & \mathbf{0} & \\ & \ddots & & \mathbf{0}_{d\times(n-d)} \\ \mathbf{0} & & \sigma_d & \end{bmatrix} \tilde{y} \\ &= \begin{bmatrix} \frac{\sigma_1}{\sigma_1^2+\lambda} & & \mathbf{0} & \\ & \ddots & & \mathbf{0}_{d\times(n-d)} \\ \mathbf{0} & & \frac{\sigma_d}{\sigma_d^2+\lambda} & \end{bmatrix} \tilde{y} \end{aligned}$$

Then, by solving each of n problems, the solution is (with the same reasoning as above using the least squares solutions):

$$\hat{w}_i = \begin{cases} \frac{\sigma_i}{\sigma_i^2+\lambda}\tilde{y}_i & : i \leq \min(n,d) \\ 0 & : \text{o.w.} \end{cases}.$$

From here, let's observe the role of λ.

- High level perspective on closed form solution (2): even though $X^\mathsf{T}X$ is not invertible or barely invertible, adding $\lambda\mathbf{I}$ makes $X^\mathsf{T}X + \lambda\mathbf{I}$ invertible.

- If $\lambda \ll \sigma_i$, then we have
$$\hat{w}_i \sim \frac{1}{\sigma_i} y_i$$
which gives the usual solution from ordinary least squares. When $\lambda \gg \sigma_i$ instead, we obtain
$$\hat{w}_i \sim \frac{\sigma_i}{\lambda} y_i$$
and we prevent the case of blowup for small eigenvalues, creating overfitting and/or solutions that are difficult to believe. Ridge essentially makes weights associated with smaller singular values close to 0, while leaving larger singular values which give the system the greatest variance, unchanged. In many such systems, in practice, we have a wide, dynamic range of singular values, so this is a common technique.

- Note: Ridge in this sense is similar to projecting the data onto a low dimensional space of the higher singular values, but has the benefit of not requiring the computational expense of finding the SVD explicitly.

- Graphically, on a log scale, λ changes the singular values as such

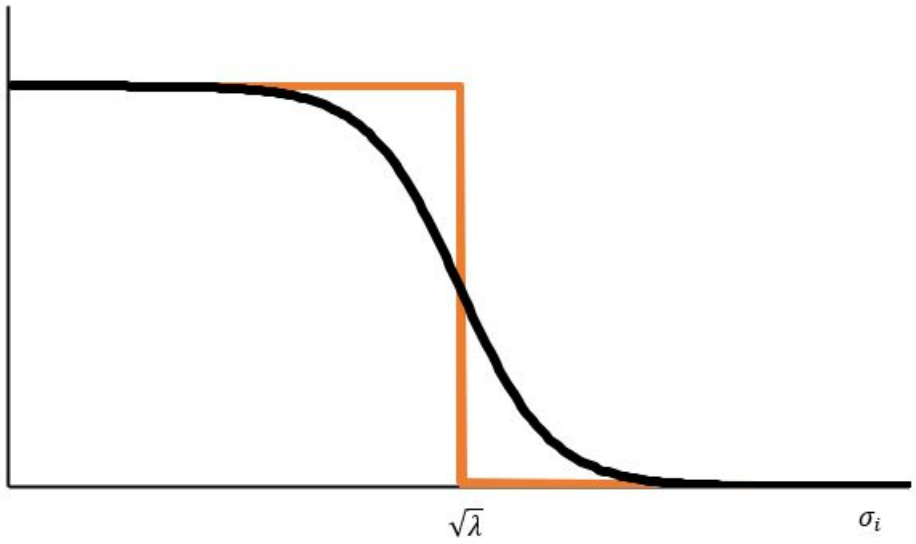

Figure 1: Image created in excel to depict the factor multiplying σ_i in the Ridge Regression solution. The goal is to have a sharp decrease increasing to higher sigma values so that Ridge Regression (black) approximates a step (orange) well.

4.3 Gradient Descent

Looking at GD in SVD coordinates, we have the update in between w_{t-1} and w_t has the form
$$w_t = w_{t-1} + 2\eta\Sigma^T(\tilde{y} - \Sigma\tilde{w}_t)$$
looking at each coordinate, this update becomes
$$w_t[i] = w_{t-1}[i] + 2\eta\sigma_i(\tilde{y}[i] - \sigma_i\tilde{w}_t[i])$$
This gives the interpretation that for large singular values, gradient descent moves much more and faster than smaller singular values, which can take longer to converge. So, we first fit in the largest σ_i direction, then the next and so on. When working with very small singular values that have the propensity to give large solutions, we may converge to something very large, but slowly, especially without regularization.

- Solution: Use gradient descent, but limit the number of steps to a finite number to avoid overtraining. This is sometimes called early stopping.

5 Implicit Regularization

There are a few different tricks that can be used to avoid explictly including Ridge regularization while still accomplishing the same result, either by adding extra entries or extra features to the data matrix.

5.1 Adding Extra Data

We can augment the data matrix X and the vector y as follows:

$$\hat{X} = \begin{bmatrix} X \\ \sqrt{\lambda} I_d \end{bmatrix}, \hat{y} = \begin{bmatrix} y \\ \mathbf{0}_d \end{bmatrix}$$

Using this new data in the ordinary least squares solution, we obtain

$$\hat{w} = (\hat{X}^T \hat{X})^{-1} \hat{X}^T \hat{y} = (X^T X + \lambda I_d)^{-1} X^T y$$

which gives the exact same solution as Ridge Regression.

5.2 Adding Extra Features

Instead of adding extra data, we can instead add additional features to make our matrix wide and have infinitely many solutions.

$$\hat{X} = \begin{bmatrix} X & \sqrt{\lambda} I_n \end{bmatrix}$$

Here, if we solve the min-norm problem

$$\min_{\hat{w}: \hat{X}\hat{w} = y} ||\hat{w}||^2$$

Then, we have the closed form solution of the min-norm problem given by

$$\begin{bmatrix} \hat{w} \\ f \end{bmatrix} = \hat{X}^T (\hat{X} \hat{X}^T)^{-1} y = \begin{bmatrix} X^T \\ \sqrt{\lambda} I_n \end{bmatrix} (X X^T + \lambda I_n)^{-1} y = (X^T X + \lambda I_d)^{-1} \begin{bmatrix} X^T \\ \sqrt{\lambda} I_n \end{bmatrix} y$$

where the first n entries give the same solution as Ridge Regression once again.

6 Conclusion - Tying Back to Neural Networks

We are often interested in minimizing the loss function of a neural network $f_\theta(x)$, but our optimization algorithms require care to ensure the minimization obtained is in fact the one we want. With nonlinear problems, we can often linearize around the operating point by taking

$$f_\theta(x) = f_{\theta_0}(x) + \frac{\partial f}{\partial x}\Big|_{\theta_0} \cdot \Delta\theta$$

and linearize around every point as part of a Generalized Linear Model (GLM). In minimizing the loss function, we face the following characteristics:

- Gradient Descent is the tool that we understand the most and is often our first choice
- The loss function will typically contain training data with labels that we would like to predict with great accuracy

- The singular values of the data matrices involved greatly matter and can lead to overfitting, slow convergence times, and unreasonably large parameters
- In solving this problem, we usually add regularization, which we can understand through SVD as bounding the solution to the smaller singular values while leaving the solution of the larger singular values unchanged