Neural Networks (Part II)

Training:

Setup: Given $\mathcal{D} := \{(x_1, y_1), \dots, (x_n, y_n)\}$, $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}^s$

Likelihood: $y_i \overset{i.i.d}{\sim} \mathcal{N}(y_i \mid f_\theta(x_i), \sigma^2) \iff y_i = f_\theta(x_i) + \varepsilon$

$\varepsilon \sim \mathcal{N}(0, \sigma$

Forward Pass

$$\begin{cases} f_\theta(x_i) = H^{(L)} W^{(0)} + b^{(0)} & \text{(regression output layer)} \\ H^{(L)} = \sigma(H^{(L-1)} W^{(L)} + b^{(L)}) & \\ \vdots & \\ H^{(1)} = \sigma(X W^{(1)} + b^{(1)}) & \end{cases}$$

Model parameters to be traine

$\Theta := \{\theta, \sigma^2\}$

$\Theta := \{(W^{(0)}, W^{(L)}, \dots, W^{(1)}), (b^{(0)}, b^{(L)}, \dots, b^{(1)}), \sigma^2\}$

Goal: Identify a set of "optimal" parameters Θ^*, such that:

$$\Theta^* = \arg\max_{\Theta} p(y|X, \Theta) = \arg\min_{\Theta} -\log p(y|X, \Theta)$$

$$\mathcal{L}(\Theta) := -\log p(y|X, \Theta) = \frac{1}{2\sigma^2} \sum_{i=1}^{n} [\underbrace{f_\theta(x_i)}_{} - y_i]^2 + \frac{n}{2} \log(2\pi$$

↳ not convex

because f_θ is highly non-linear

(*) **Remark**: Typically we are only interested in estimating the neural network parameters θ. For that, it suffices to minimize the sum of square errors:

(regression): $\mathcal{L}(\theta) := \frac{1}{2} \sum_{i=1}^{n} [f_\theta(x_i) - y_i]^2$ (sum of square errors)

(binary classification): $\mathcal{L}(\theta) := -\sum_{i=1}^{n} y_i \log f_\theta(x_i) + (1 - y_i) \log(1 - f_\theta$

(multi-class classification) $\mathcal{L}(\theta) := - \sum_{i=1}^{n} \sum_{c=1}^{C} y_{ic} \log f_\theta(x_i)$

We will "train" our model via gradient descent : $\theta_{n+1} = \theta_n - \eta \nabla_\theta \mathcal{L}(\theta_n)$

To do so, we need to compute $\nabla_\theta \mathcal{L}(\theta_n)$! ... we will do so via back-propagation !

Overfitting and Regularization

1.) L_2 parameter regularization / weight decay :

Idea : Modify the loss function :

regularization weight $(10^{-2} - 10^{-5})$

$$\mathcal{L}(\theta) = \underbrace{\underbrace{\frac{1}{2} \sum_{i=1}^{n} [y_i - f_\theta(x_i)]^2}_{\text{MLE}} + \frac{\lambda}{2} \theta^T \theta}_{\text{MAP}} , \quad \|\theta\|_2^2 := \theta^T \theta$$

corresponds to assuming $\theta \sim \mathcal{N}(0, I)$

drive the weights and bias closer to the origin.

2.) L_1 parameter regularization :

need to be tuned $(10^{-2} - 10^{-6})$

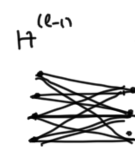
corresponds to assuming a Laplace prior

$$\mathcal{L}(\theta) = \frac{1}{2} \sum_{i=1}^{n} [y_i - f_\theta(x_i)]^2 + \alpha \|\theta\|_1 , \quad \|\theta\|_1 := \sum_i |\theta_i|$$

The L_1 penalty promotes sparsity in θ, i.e. discards features that are not needed.

(*) In practice, we often use a combination of L_1/L_2 regularization.

3.) Dropout regularization :



Recall : $H^{(l)} = \sigma(H^{(l-1)} W^{(l)} + b^{(l)})$

With dropout, this becomes :

$r_j^{(l)} \sim \text{Bernaulli}(p)$

$z^{(l)} = r^{(l)} \odot H^{(l-1)}$ (element-wise multiplication will discard some features in $H^{(l-}$

$H^{(l)} = f(z^{(l)} W^{(l)} + b^{(l)})$