HW2 Handwritten Assignment

Lecturor: Pei-Yuan Wu
TAs: Po-Yang Hsieh, Le-Rong Hsu

October 2024

Problem 1: (Automatic Differentiation) (0.5%)

Let $f = f_4 \circ f_3 \circ f_2 \circ f_1$ be a differentiable function defined as

$$\begin{aligned} f : \mathbb{R}^n &\to \mathbb{R} \\ \mathbf{x} &\mapsto f_4(f_3(f_2(f_1(\mathbf{x})))) \end{aligned}$$

where $f_1 : \mathbb{R}^n \to \mathbb{R}^{n_1}$, $f_2 : \mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$, $f_3 : \mathbb{R}^{n_2} \to \mathbb{R}^{n_3}$, and $f_4 : \mathbb{R}^{n_3} \to \mathbb{R}$. We assume $n \gg 1$ and $n \geq n_1 \geq n_2 \geq n_3 > 1$.

1. (0.2%) Write down the derivative of f with respect to $\mathbf{x}$ using **chain rule**. Next, express **forward-mode** and **reverse-mode** auto-differentiation respectively with the formula you just wrote down. The reverse-mode auto-differentiation is also called **backpropagation**. In this problem, please "assign variables" when using chain rule. For example, suppose $y = g(f(\mathbf{x}))$. Let $w_1 = f(\mathbf{x})$ and $y = w_2 = g(w_1)$. Then the partial derivative of y with respect to $\mathbf{x}$ is given by
$$\frac{\partial y}{\partial \mathbf{x}} = \frac{\partial y}{\partial w_2}\frac{\partial w_2}{\partial w_1}\frac{\partial w_1}{\partial \mathbf{x}} = 1 \cdot \frac{\partial w_2}{\partial w_1}\frac{\partial w_1}{\partial \mathbf{x}} = \frac{\partial w_2}{\partial w_1}\frac{\partial w_1}{\partial \mathbf{x}} \left(= \frac{\partial y}{\partial w_1}\frac{\partial w_1}{\partial \mathbf{x}} \right).$$
2. (0.3%) Which mode is better in terms of efficiency? Why? For the second question, you need to prove your answer. Note that if you only answer the first question, you will only get 0.01%. Hint: you may write down the size of Jacobian matrices and see how many scalar multiplications ($a \times b$, $a, b \in \mathbb{R}$) are performed in the forward-mode and reverse-mode auto-differentiation respectively.

Through this problem, we want students to tell the difference between the forward mode and the reverse mode, and to understand why the reverse mode is usually preferred for optimization over the forward mode. Some useful references:

- Automatic Differentiation in Machine Learning: a Survey
- CSC321 Lecture 10: Automatic Differentiation
- Wikipedia

Problem 2 (Batch Normalization)(1%)

Batch normalization is a popular trick for training deep networks nowadays, which aims to preserve the distribution within hidden layers and avoids vanishing gradient issue. The algorithm can be written as below (see Algorithm 1):

Algorithm 1 Batch Normalization

Input Feature from data points over a mini-batch $B = (x_i)_{i=1}^m$
Output $y_i = BN_{\gamma,\beta}(x_i)$

procedure BatchNormalize(B, γ, β)
  $\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^m x_i$ ▷ mini-batch mean
  $\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^m (x_i - \mu_B)^2$ ▷ mini-batch variance
  for $i \leftarrow 1$ to m **do**
    $\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$ ▷ normalize
    $y_i \leftarrow \gamma \hat{x}_i + \beta$ ▷ scale and shift
  end for
  return
end procedure

During training we need to backpropagate the gradient of loss ℓ through this transformation, as well as compute the gradients with respect to the parameters γ, β. Towards this end, please write down the close form expressions for $\frac{\partial \ell}{\partial x_i}$, $\frac{\partial \ell}{\partial \gamma}$, $\frac{\partial \ell}{\partial \beta}$ in terms of x_i, μ_B, σ_B^2, $\hat{x}_i$, y_i (given by the forward pass) and $\frac{\partial \ell}{\partial y_i}$ (given by the backward pass).

- Hint: You may first write down the close form expressions of $\frac{\partial \ell}{\partial \hat{x}_i}$, $\frac{\partial \ell}{\partial \sigma_B^2}$, $\frac{\partial \ell}{\partial \mu_B}$, and then use them to compute $\frac{\partial \ell}{\partial x_i}$, $\frac{\partial \ell}{\partial \gamma}$, $\frac{\partial \ell}{\partial \beta}$.

Problem 3 (Multiclass AdaBoost)(1.5%)

Let $\mathcal{X}$ be the input space, $\mathcal{F}$ be a collection of multiclass classifiers that map from $\mathcal{X}$ to $[1, K]$, where K denotes the number of classes. Let $\{(x_i, \hat{y}_i)\}_{i=1}^m$ be the training data set, where $x_i \in \mathcal{X}$ and $\hat{y}_i \in [1, K]$. Given $T \in \mathbb{N}$, suppose we want to find functions

$$g_{T+1}^k(x) = \sum_{t=1}^{T} \alpha_t f_t^k(x), \quad k \in [1, K]$$

where $f_t \in \mathcal{F}$ and $\alpha_t \in \mathbb{R}$ for all $t \in [1, T]$. Here for $f \in \mathcal{F}$, we denote $f^k(x) = \mathbf{1}\{f(x) = k\}$, where $\mathbf{1}(\cdot)$ is an indicator function, as the k'th element in the one-hot representation of $f(x) \in [1, K]$. The aggregated classifier $h : \mathcal{X} \to [1, K]$ is defined as

$$x \mapsto \underset{1 \le k \le K}{\operatorname{argmax}}\ g_{T+1}^k(x)$$

Please apply gradient boosting to show how the functions f_t and coefficients α_t are computed with an aim to minimize the following loss function

$$L((g_{T+1}^1, \cdots, g_{T+1}^K) = \sum_{i=1}^{m} \exp\left(\frac{1}{K-1} \sum_{k \neq \hat{y}_i} g_{T+1}^k(x_i) - g_{T+1}^{\hat{y}_i}(x_i)\right)$$

Problem 4 (AdaBoosting)(1%)

1. Consider training an Adaboosting classifier using decision stumps on the data set illustrated in Figure 1:

Figure 1: AdaBoost Data set

   (a) Which examples will have their weights increased at the end of the first iteration? Circle them.

   (b) How many iterations will it take to achieve zero training error? Justify your answers and show each iteration step.

2. Suppose AdaBoost is run on N training examples, and suppose on each round that the weighted training error ϵ_t of the t'th weak hypothesis is at most $1/2-\gamma$, for some number $0 < \gamma < 1/2$. After how many iterations, T, will the combined hypothesis be consistent with the N training examples, i.e., achieves zero training error? Your answer should only be expressed in terms of N and γ. (Hint: Recall that exponential loss is an upper bound for 0-1 loss. What is the training error when 1 example is misclassified?)

Problem 5 (Gradient Descent Convergence) (2%)

Suppose the function $f : \mathbb{R}^n \to \mathbb{R}$ is differentiable. Also, f is β-smoothness and α-strongly convex.

$$\beta\text{-smoothness} : \beta > 0, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, \|\nabla f(\boldsymbol{x}) - \nabla f(\boldsymbol{y})\|_2 \leq \beta \|\boldsymbol{x} - \boldsymbol{y}\|_2$$

$$\alpha\text{-strongly convex} : \alpha > 0, \forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, f(\boldsymbol{x}) - f(\boldsymbol{y}) - \nabla f(\boldsymbol{y})^T(\boldsymbol{x} - \boldsymbol{y}) \geq \frac{\alpha}{2}\|\boldsymbol{x} - \boldsymbol{y}\|_2^2$$

Then we propose a gradient descent algorithm

1. Find a initial $\boldsymbol{\theta}^0$.
2. Let $\boldsymbol{\theta}^{n+1} = \boldsymbol{\theta}^n - \eta \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^n)\ \forall n \geq 0$, where $\eta = \frac{1}{\beta}$.

The following problems lead you to prove the gradient descent convergence.

1. Prove the property of β-smoothness function

$$\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n, f(\boldsymbol{y}) - f(\boldsymbol{x}) - \nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) \leq \frac{\beta}{2}\|\boldsymbol{y} - \boldsymbol{x}\|_2^2$$

   (a) Define $g : \mathbb{R} \to \mathbb{R}, g(t) = f(\boldsymbol{x} + t(\boldsymbol{y} - \boldsymbol{x}))$. Show that $f(\boldsymbol{y}) - f(\boldsymbol{x}) = \int_0^1 g^{'}(t)\, dt$.

   (b) Show that $g^{'}(t) = \nabla f(\boldsymbol{x} + t(\boldsymbol{y} - \boldsymbol{x}))^T(\boldsymbol{y} - \boldsymbol{x})$.

   (c) Show that $|f(\boldsymbol{y}) - f(\boldsymbol{x}) - \nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x})| \leq \int_0^1 |(\nabla f(\boldsymbol{x} + t(\boldsymbol{y} - \boldsymbol{x})) - \nabla f(\boldsymbol{x}))^T(\boldsymbol{y} - \boldsymbol{x})|\, dt$.

   (d) By Cauchy-Schwarz inequality and the definition of β-smoothness, show that $|f(\boldsymbol{y}) - f(\boldsymbol{x}) - \nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x})| \leq \frac{\beta}{2}\|\boldsymbol{y} - \boldsymbol{x}\|_2^2$, hence we get

$$f(\boldsymbol{y}) - f(\boldsymbol{x}) - \nabla f(\boldsymbol{x})^T(\boldsymbol{y} - \boldsymbol{x}) \leq \frac{\beta}{2}\|\boldsymbol{y} - \boldsymbol{x}\|_2^2$$

2. Let $\boldsymbol{y} = \boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})$ and use 1., prove that

$$f\left(\boldsymbol{x} - \frac{1}{\beta}\nabla f(\boldsymbol{x})\right) - f(\boldsymbol{x}) \leq -\frac{1}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2$$

   and

$$f(\boldsymbol{x}^*) - f(\boldsymbol{x}) \leq -\frac{1}{2\beta}\|\nabla f(\boldsymbol{x})\|_2^2,$$

   where $\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}} f(\boldsymbol{x})$.

3. Show that $\forall n \geq 0$,

$$\|\boldsymbol{\theta}^{n+1} - \boldsymbol{\theta}^*\|_2^2 = \|\boldsymbol{\theta}^n - \boldsymbol{\theta}^*\|_2^2 + \eta^2\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^n)\|_2^2 - 2\eta\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}^n)^T(\boldsymbol{\theta}^n - \boldsymbol{\theta}^*),$$

   where $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$.

4. Use 2. and the definition of α-strongly convex to prove $\forall n \geq 0$

$$\|\boldsymbol{\theta}^{n+1} - \boldsymbol{\theta}^*\|_2^2 \leq (1 - \frac{\alpha}{\beta})\|\boldsymbol{\theta}^n - \boldsymbol{\theta}^*\|_2^2,$$

   where $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$.

5. Use the above inequality to show that $\boldsymbol{\theta}^n$ will converge to $\boldsymbol{\theta}^*$ when n goes to infinity.

Version Description

1. First Edition: Finish Problem 1 to 5