Li-Jintao 2201213292 lijintao@stu.pku.edu.cn
Prof. Mu-Yadong's Deep Learning: Algorithms and Applications
May 10th, 2023

Homework-1

Question 1. MLP for Classification: In a multi-class classification problem, e.g. a problem with M classes, we use a model f_θ to produce the output $\hat{y}$. $\hat{y}$ is a vector of dimension M with sum 1, whose components indicate the probability of x belong to each class.

In this section, you will use a simple model – the 3-layer MLP – to tackle this problem. Assume that we have K labeled data (x_k, y_k) with $k = 1, 2, \ldots, K$ where $x_k \in \mathbb{R}^N$ and $y_k \in \mathbb{R}^M$ is a M dimensional one-hot vector.

a) Define the cross entropy loss for a multi-class classification problem.

b) A 3-layer MLP consists of an input layer, a hidden layer, and an output layer with two learned parameter matrices W^1, W^2 between successive layers. Please note that there is generally a Softmax layer after the output layer to scale the output, which we omitted in this sub-question for simplicity. When given an input x, this model performs the following forward computations sequentially:

$$\begin{aligned} a_1 &= \mathrm{W}^1 x, \\ h &= \sigma\left(a_1\right), \\ a_2 &= \mathrm{W}^2 h, \\ \hat{y} &= \sigma\left(a_2\right). \end{aligned}$$

where $\mathrm{W}^1 \in \mathbb{R}^{D\times N}, \mathrm{W}^2 \in \mathbb{R}^{M\times D}$ are parameter matrices and $\sigma(z) = \frac{1}{1+e^{-z}}$ is the element-wise Sigmoid function. Use the loss function you defined in sub-question a), apply an SGD update to the parameters W^1 and W^2 with learning rate η via backpropagation. Must identify the closed-form gradient of each parameter, which is $\frac{\partial \mathcal{L}}{\partial \mathrm{W}^2(m,d)}$ and $\frac{\partial \mathcal{L}}{\partial \mathrm{W}^1(d,n)}$ in your derivation.

HINT: 1) For simplicity, you can assume that the SGD program uses only one training data per batch. 2) Use formula $\sigma'(z) = \sigma(z)(1 - \sigma(z))$ for a scalar z.

SOLUTION:

a) The cross-entropy loss for a multi-class classification problem is defined as:

$$\mathcal{L} = -\frac{1}{B}\sum_{b=1}^{B} y_b^T \log \hat{y}_b$$

where B stands for the batch-size of training data, and y_b is the M dimensional one-hot vector for the b-th training example, and $\hat{y}_b$ is the output vector predicting the b-th example.

According to the **HINT**, we assume that the SGD program uses only one training data per batch. Then we have $B = 1$, and the loss is:

$$\mathcal{L} = y^T \log \hat{y}$$

b) To compute the gradients of the loss function with respect to W^1 and W^2 using back-propagation and SGD, we first need to compute the derivative of the loss function with respect to $\hat{y}(m)$, where $m \in \{1, 2, \ldots, M\}$ denotes the m-th neuron in the output layer. And it is given by:

$$\frac{\partial \mathcal{L}}{\partial \hat{y}(m)} = -\frac{y(m)}{\hat{y}(m)}$$

Using the chain rule, we can compute the derivative of the loss function with respect to a_2:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial a_2(m)} &= \frac{\partial \mathcal{L}}{\partial \hat{y}(m)}\frac{\partial \hat{y}(m)}{\partial a_2(m)}\\ &= -\frac{y(m)}{\hat{y}(m)}\sigma(a_2(m))\left(1-\sigma(a_2(m))\right)\\ &= y(m)\left(\hat{y}(m)-1\right)\end{aligned}$$

Using the same chain rule, we can compute the derivative of the loss function with respect to a_1:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial a_1(d)} &= \sum_{m=1}^{M}\frac{\partial \mathcal{L}}{\partial a_2(m)}\frac{\partial a_2(m)}{\partial h(d)}\frac{\partial h(d)}{\partial a_1(d)}\\ &= -\sum_{m=1}^{M}\frac{y(m)}{\hat{y}(m)}\sigma(a_2(m))\left(1-\sigma(a_2(m))\right)\mathrm{W}^2(m,d)\sigma(a_1(d))\left(1-\sigma(a_1(d))\right)\\ &= \sum_{m=1}^{M} y(m)\left(\hat{y}(m)-1\right)\mathrm{W}^2(m,d)h(d)(1-h(d))\end{aligned}$$

Finally, we can compute the gradients of the loss function with respect to the parameter matrices:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathrm{W}^2(m,d)} &= \frac{\partial \mathcal{L}}{\partial \hat{y}(m)}\frac{\partial \hat{y}(m)}{\partial a_2(m)}\frac{\partial a_2(m)}{\partial \mathrm{W}^2(m,d)}\\ &= \frac{\partial \mathcal{L}}{\partial a_2(m)}\frac{\partial a_2(m)}{\partial \mathrm{W}^2(m,d)}\\ &= -\frac{y(m)}{\hat{y}(m)}\sigma(a_2(m))(1-\sigma(a_2(m)))h(d)\\ &= y(m)(\hat{y}(m)-1)h(d)\end{aligned}$$

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathrm{W}^1(d,n)} &= \frac{\partial \mathcal{L}}{\partial \hat{y}(m)}\frac{\partial \hat{y}(m)}{\partial a_2(m)}\frac{\partial a_2(m)}{\partial h(d)}\frac{\partial h(d)}{\partial a_1(d)}\frac{\partial a_1(d)}{\partial \mathrm{W}^1(d,n)}\\ &= -\sum_{m=1}^{M}\frac{y(m)}{\hat{y}(m)}\sigma(a_2(m))(1-\sigma(a_2(m)))\mathrm{W}^2(m,d)\sigma(a_1(d))(1-\sigma(a_1(d)))x(n)\\ &= \sum_{m=1}^{M} y(m)\left(\hat{y}(m)-1\right)\mathrm{W}^2(m,d)h(d)(1-h(d))x(n)\end{aligned}$$

Therefore, the update rule for the parameters W^2 and W^1 are:

$$\begin{aligned}\mathrm{W}^1(d,n) &\leftarrow \mathrm{W}^1(d,n) - \eta\sum_{m=1}^{M} y(m)\left(\hat{y}(m)-1\right)\mathrm{W}^2(m,d)h(d)(1-h(d))x(n)\\ \mathrm{W}^2(m,d) &\leftarrow \mathrm{W}^2(m,d) - \eta(\hat{y}(m)-1)y(m)h(d)\end{aligned}$$

where η is the learning rate.

Question 2. Dropout and Regularization

Dropout is a well-known way to prevent neural networks from overfitting. In this section, you will show this regularization explicitly for linear regression. Recall that linear regression optimizes $w \in \mathbb{R}^d$ to minimize the following MSE objective:

$$\mathcal{L}(w) = \|y - \mathrm{X}w\|^2$$

where $y \in \mathbb{R}^n$ is the response to the design matrix $\mathrm{X} \in \mathbb{R}^{n\times d}$. One way to use dropout during training on the d-dimensional input features x_i involves keeping each feature at random with probability p (and zero out it if not kept).

a) Show that when we apply such dropout, the learning objective becomes

$$\mathcal{L}(w) = \mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}\|y - (\mathrm{M} \odot \mathrm{X})w\|^2$$

where $\odot$ denotes the element-wise product and $\mathrm{M} \in \{0,1\}^{n\times d}$ is a random mask matrix whose element $m_{i,j}$ have *i.i.d.* Bernoulli distribution with success probability p.

b) Show that we can manipulate the dropout learning objective to an explicit regularized objective:

$$\mathcal{L}(w) = \|y - p\mathrm{X}w\|^2 + p(1-p)\|\Gamma w\|^2$$

and define a suitable matrix Γ.

SOLUTION:

a) To incorporate dropout, we multiply the input features X element-wise with a mask matrix M which has values 1 with probability p and 0 with probability $1-p$. This randomly masks out features during training. Now, let's compute the objective function:

$$\begin{aligned}
\mathcal{L}(w) &= \|y - (\mathrm{M} \odot \mathrm{X})w\|^2 \\
&= (y - (\mathrm{M} \odot \mathrm{X})w)^T(y - (\mathrm{M} \odot \mathrm{X})w) \\
&= y^Ty - y^T(\mathrm{M} \odot \mathrm{X})w - w^T(\mathrm{M} \odot \mathrm{X})^Ty + w^T(\mathrm{M} \odot \mathrm{X})^T(\mathrm{M} \odot \mathrm{X})w \\
&= y^Ty - 2w^T(\mathrm{M} \odot \mathrm{X})^Ty + w^T(\mathrm{M} \odot \mathrm{X})^T(\mathrm{M} \odot \mathrm{X})w
\end{aligned}$$

Now, we take the expectation over the mask matrix M drawn from the Bernoulli distribution:

$$\begin{aligned}
\mathcal{L}(w) &= \mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}[y^Ty - 2w^T(\mathrm{M} \odot \mathrm{X})^Ty + w^T(\mathrm{M} \odot \mathrm{X})^T(\mathrm{M} \odot \mathrm{X})w] \\
&= \mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}[y^Ty] - 2w^T\mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}[(\mathrm{M} \odot \mathrm{X})^Ty] + w^T\mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}[(\mathrm{M} \odot \mathrm{X})^T(\mathrm{M} \odot \mathrm{X})]w \\
&= \|y - \mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}[(\mathrm{M} \odot \mathrm{X})w]\|^2 \\
&= \mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}\|y - (\mathrm{M} \odot \mathrm{X})w\|^2
\end{aligned}$$

Therefore, the learning objective with dropout becomes $\mathcal{L}(w) = \mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}\|y - (\mathrm{M} \odot \mathrm{X})w\|^2$.

b) Acknowledgement: This answer refers to UC-Berkeley's CS182[1]. Starting with the dropout learning objective:

$$\mathcal{L}(w) = \mathbb{E}_{M\sim\text{Bernoulli}(p)}\|y - (\mathrm{M} \odot \mathrm{X})w\|^2$$

Let $\mathrm{P} = \mathrm{M} \odot \mathrm{X}$ where $\odot$ is the element-wise multiplication. Therefore, we have:

$$\|y - \mathrm{P}w\|_2^2 = y^Ty - 2w^T\mathrm{P}^Ty + w^T\mathrm{P}^T\mathrm{P}w$$

That is:

$$\mathbb{E}_{\mathrm{M}\sim\text{Bernoulli}(p)}\left[\|y - \mathrm{M} \odot \mathrm{X}w\|^2\right] = \mathbb{E}_{\mathrm{M}}\left[y^Ty - 2w^T\mathrm{P}^Ty + w^T\mathrm{P}^T\mathrm{P}w\right]$$

Since the expected value of a matrix is the matrix of the expected value of its elements, we have that

$$\mathbb{E}_{\mathrm{M}}[\mathrm{P}]_{ij} = \mathbb{E}_{\mathrm{M}}\left[(\mathrm{M} \odot \mathrm{X})_{ij}\right] = \mathrm{X}_{ij}\mathbb{E}_{\mathrm{M}}\left[\mathrm{M}_{ij}\right] = p\mathrm{X}_{ij}$$

It follows that:

$$\mathbb{E}_{\mathrm{M}}\left[2w^T\mathrm{P}^Ty\right] = 2pw^T\mathrm{X}^Ty$$

and:

$$\left(\mathbb{E}_{\mathrm{M}}\left[\left(\mathrm{P}^T\mathrm{P}\right)\right]\right)_{ij} = \Sigma_{k=1}^n\mathbb{E}_{\mathrm{M}}\left[\mathrm{M}_{ki}\mathrm{M}_{kj}\mathrm{X}_{ki}\mathrm{X}_{kj}\right]$$

where:

$$\mathbb{E}_{\mathrm{M}}\left[\left(\mathrm{P}^T\mathrm{P}\right)\right]_{ij} = \begin{cases} \sum_{k=1}^n \mathbb{E}_{\mathrm{M}}\left[\mathrm{M}_{ki}\mathrm{M}_{kj}\mathrm{X}_{ki}\mathrm{X}_{kj}\right] = \sum_{k=1}^n \mathbb{E}_{\mathrm{M}}\left[\mathrm{M}_{ki}\right]\mathbb{E}_{\mathrm{M}}\left[\mathrm{M}_{kj}\right]\mathrm{X}_{ki}\mathrm{X}_{kj} = p^2\left(\mathrm{X}^T\mathrm{X}\right)_{ij} & \text{if } i \neq j \\ \sum_{k=1}^n \mathbb{E}_{\mathrm{M}}\left[\mathrm{M}_{ki}^2\mathrm{X}_{ki}\mathrm{X}_{kj}\right] = \sum_{k=1}^n \mathbb{E}_{\mathrm{M}}\left[\mathrm{M}_{ki}^2\right]\mathrm{X}_{ki}\mathrm{X}_{kj} = p\left(\mathrm{X}^T\mathrm{X}\right)_{ij} & \text{if } i = j \end{cases}$$

Finally, we note that:

$$\left(\mathbb{E}_{\mathrm{M}}\left[\left(\mathrm{P}^T\mathrm{P}\right)\right]\right)_{ij} - p^2\left(\mathrm{X}^T\mathrm{X}\right)_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ \left(p - p^2\right)\left(\mathrm{X}^T\mathrm{X}\right)_{ij} & \text{if } i = j \end{cases}$$

We now can put everything together as follow:

$$
\begin{aligned}
\mathcal{L}(w) &= \mathbb{E}_{\mathrm{M}}\left[\|y - \mathrm{M} \odot \mathrm{X}w\|^2\right] \\
&= \mathbb{E}_{\mathrm{M}}\left[y^T y - 2w^T \mathrm{P}^T y + w^T \mathrm{P}^T \mathrm{P} w\right] \\
&= y^T y - 2pw^T \mathrm{X}^T y + p^2 w^T \mathrm{X}^T \mathrm{X} w - p^2 w^T \mathrm{X}^T \mathrm{X} w + w^T \mathbb{E}_{\mathrm{M}}\left[\mathrm{P}^T \mathrm{P}\right] w \\
&= \|y - p\mathrm{X}w\|^2 + \left(w^T \mathbb{E}_{\mathrm{M}}\left[\mathrm{P}^T \mathrm{P}\right] w - p^2 w^T \mathrm{X}^T \mathrm{X} w\right) \\
&= \|y - p\mathrm{X}w\|^2 + w^T \left(\mathbb{E}_{\mathrm{M}}\left[\mathrm{P}^T \mathrm{P}\right] - p^2 \left(\mathrm{X}^T \mathrm{X}\right)\right) w \\
&= \|y - p\mathrm{X}w\|^2 + \left(p^2 - p\right) w^T \left(\mathrm{diag}\left(\mathrm{X}^T \mathrm{X}\right)\right) w \\
&= \|y - p\mathrm{X}w\|^2 + p(1-p) w^T \left(\mathrm{diag}\left(\mathrm{X}^T \mathrm{X}\right)\right) w \\
&= \|y - p\mathrm{X}w\|^2 + p(1-p)\|\Gamma w\|^2
\end{aligned}
$$

where $\mathrm{diag}\left(\mathrm{X}^T \mathrm{X}\right)$ refers to the matrix where the non-diagonal elements of $\mathrm{X}^T \mathrm{X}$ are set to 0, and $\Gamma = \left(\mathrm{diag}\left(\mathrm{X}^T \mathrm{X}\right)\right)^{1/2}$, which exists as $\mathrm{X}^T \mathrm{X}$ is positive-semidefinite(PSD) and therefore has non-negative diagonal elements.

References

[1] CS 182: Deep neural networks, spring 2023. [Online]. Available: https://inst.eecs.berkeley.edu/~cs182/sp23/