Deep Learning HW1

Yipu Li

November 2024

1 3. Back Propagation

Question 1 a

$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{W}^T$

$(\frac{dL}{d\boldsymbol{W}})_{mn} = \sum_{i,j} \frac{dL}{dY_{ij}} \frac{dY_{ij}}{dW_{mn}} = \sum_{i,j} \frac{dL}{dY_{ij}} \frac{d\sum_{p=1}^{M} X_{ip}W_{jp}}{W_{mn}} = \sum_{i,j} \frac{dL}{dY_{ij}} \delta_{jm} X_{in} = \sum_i \frac{dL}{dY_{im}} X_{in}$

Hence $\frac{dL}{d\boldsymbol{W}} = (\frac{dL}{d\boldsymbol{Y}})^T \boldsymbol{X}$

Question 1 b

$(\frac{dL}{d\boldsymbol{b}})_m = \sum_{i,j} \frac{dL}{dY_{ij}} \frac{dY_{ij}}{db_m} = \sum_{i,j} \frac{dL}{dY_{ij}} \delta_{jm} = \sum_i \frac{dL}{dY_{im}}$

Hence $\frac{dL}{d\boldsymbol{b}} = 1_S(\frac{dL}{d\boldsymbol{Y}})$ where 1_S is the row vector with S many 1.

Question 1 c $(\frac{dL}{d\boldsymbol{X}})_{mn} = \sum_{i,j} \frac{dL}{dY_{ij}} \frac{dY_{ij}}{dX_{mn}} = \sum_{i,j} \frac{dL}{dY_{ij}} \frac{d\sum_{p=1}^{M} X_{ip}W_{jp}}{X_{mn}} = \sum_{i,j} \frac{dL}{dY_{ij}} \delta_{im} W_{jn} = \sum_j \frac{dL}{dY_{mj}} W_{jn}$

Hence $\frac{dL}{d\boldsymbol{X}} = (\frac{dL}{d\boldsymbol{Y}})\boldsymbol{W}$.

Question 1 d

$(\frac{dL}{d\boldsymbol{X}})_{mn} = \sum_{i,j} \frac{dL}{dY_{ij}} \frac{dY_{ij}}{dX_{mn}} = \frac{dL}{dY_{mn}} h'(X_{mn})$.

Here h' denotes the derivative of h.

Hence $\frac{dL}{d\boldsymbol{X}} = \frac{dL}{d\boldsymbol{Y}} \circ h'(X)$

$\circ$ is the Hadamard product.

Question 1 e

Plug $\frac{dL}{d\boldsymbol{Y}}$ into $\frac{dL}{d\boldsymbol{Z}}$.

$$\frac{dL}{d\boldsymbol{Z}} = \boldsymbol{Y} \circ ((-\frac{1}{S}\frac{\boldsymbol{T}}{\boldsymbol{Y}}) - ((-\frac{1}{S}\frac{\boldsymbol{T}}{\boldsymbol{Y}} \circ \boldsymbol{Y})\boldsymbol{1}\boldsymbol{1}^T)) \tag{1}$$

$$= \frac{1}{S}(-\boldsymbol{T} + \boldsymbol{Y} \circ (\boldsymbol{T}\boldsymbol{1}\boldsymbol{1}^T)) \tag{2}$$

$\alpha = \frac{1}{S}$ and $M = (-\boldsymbol{T} + \boldsymbol{Y} \circ (\boldsymbol{T}\boldsymbol{1}\boldsymbol{1}^T))$

Question 4 a

Let $H = P\Lambda P^T$ be the eigen decomposition of H. For arbitrary eigenvalue λ of H, λ, consider an eigen vector corresponding to that eigen value, v. Then $v^T H v = (v^T P)\Lambda(vP^T)$, since eigen vectors of different eigen value are orthogonal, $v^T P = P_\lambda$, where P_λ is a vector with only m indices not equal to zero, m is the dimension of the eigenvector space corresponding to the eigen value λ. Hence $v^T H v = \lambda \sum ||v_\lambda||^2$, here $\sum ||v_\lambda||^2$ is $P_\lambda P_\lambda^T$ and by assumption it is positive, hence λ is positive.

Since λ is arbitrary, all eigen values of H is positive.

Question 4 b

Say the probability of an eigenvalue being positive is $\frac{1}{2}$, further assume that the value of each eigenvalue is independent. Then the probability that all eigenvalue being positive is around $\frac{1}{2^N}$, N is the dimension of the matrix. While the probability of some eigen value being positive and some negative is $1 - \frac{1}{2^{N-1}}$. Hence the number of saddle points are around $2^{N-1} - 1$ times more than the the number of local minimas.

Question 4 c

The gradient descend formula is

$$w' = w - \zeta \frac{df(w)}{dw}$$

around a saddle pint, the gradiant $\frac{df(w)}{dw}$ is close to zero, hence the parametres we are optimizing barely moves.

Question 5 a

$$\frac{\partial L}{\partial \gamma_i} = \frac{\partial L}{\partial y_i}\frac{\partial y_i}{\partial \gamma_i} = x_i \frac{\partial L}{\partial y_i} \tag{3}$$

$$\frac{\partial L}{\partial \beta_i} = \frac{\partial L}{\partial y_i}\frac{\partial y_i}{\partial \beta_i} = \frac{\partial L}{\partial y_i} \tag{4}$$

Question 5 b For each previous node, there are 2 normalizer parametres, hence 40 many normalizer parametres.

For each present node, there are 20 weights and 20 bias, hence there are 40^2, 1600 many parametres for linear module of the present layer.

Hence there are 1640 parametres altogether.

Question 5 c

A mini batch is essentially a subset of the data, and its mean and variance depends on the mini batch. In testing, if you apply mini-batch normalzation, you are essentially adding an unintended influence on the testing data, as how

you choose the mini batches affects the mean and variance, thus harming the overall accuracy of testing.

You can compute the global mean and variance of the training set, and apply the them as the normalizer to the test set. This should work as we assume the training data and test data comes from the same distribution.

Question 5 d

Dead neurons are neurons that have no gradient for any training input. For ReLu module, if the input of the activation function is always negative for arbitrary data input, the activation function will always return zero with zero gradiant, hence the neuron is dead. Dead neurons have no contribute to the output of the network, and hence wastes computational resource. Moreover, if the dead neurons is in the input layer, they can't be activated again and hence will remain dead.

Question 5 e

Batch normalization ensures that the input to a layer has mean approximately 0 and variance 1, hence the input of the activation function has mean around b, where b is the bias. Hence, normalization prevents extreme value from entering the activation function, thus preventing the neuron from dying.

Question 5 f

Batch normalization gives me accuracy 0.4726, which increases the accuracy as the original gives me around 0.46. The accuracy increases, as the inputs are normalized, the gradient descent step becomes more efficient and it optimizes faster. Given a fixed epoch, this gives a better performance of the model.