CO395 Group 57

Dongxiao Huang, Zheng Xun Phang, Yufeng Zhang

Q1

Linear layers

If each image in X is reshaped into a row vector, then the forward pass can be written as

$$z = XW + \mathbf{1}b^T$$

For a linear layer, the partial derivatives are

$$\frac{\partial z}{\partial X} = W \qquad \frac{\partial z}{\partial W} = X \qquad \frac{\partial z}{\partial b} = \mathbf{1}$$

To compute derivatives of the loss function L, we apply the chain rule

$$\frac{\partial L}{\partial X} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial X} \qquad \frac{\partial L}{\partial W} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial W} \qquad \frac{\partial L}{\partial b} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial b}$$

Of course, images are not row vectors, so they must be reshaped appropriately.

ReLU activation

The forward pass of ReLU is $\max(0, x)$.

Its derivative is 1 for $x \geq 0$ and 0 for $x < 0$. We apply the chain rule just like the linear layers.

Q2

During training, the forward pass of dropout will multiply the output of some neurons by 0, so it's effectively removing neurons. We can set the proportion p of neurons to dropout, and scale the outputs of the remaining neurons by $1/(1-p)$.

During testing, we restore all neurons.

Q3

The softmax function $\sigma : \Re^C \to [0,1]^C$ can represent a probability distribution over C classes:

$$\sigma(z_1, \dots, z_C) = \frac{1}{\exp(z_1) + \cdots + \exp(z_C)} \begin{bmatrix} \exp(z_1) \\ \vdots \\ \exp(z_C) \end{bmatrix} := \begin{bmatrix} \sigma_1 \\ \vdots \\ \sigma_C \end{bmatrix}$$

We used the "normalization trick" described in the coursework manual for numerical stability.

Derivatives of the softmax function are

$$\frac{\partial \sigma_i}{\partial z_j} = (\delta_{i,j} - \sigma_j)\,\sigma_i \qquad \forall\, i, j \in \{1, \dots, C\}$$

where $\delta_{i,j} = 0$ if $i \neq j$, otherwise $\delta_{i,j} = 1$.

The cross entropy loss of n images is

$$L = -\frac{1}{n}\sum_{i=1}^{n}\left[y_{i,1}\log\sigma_{i,1} + \cdots + y_{i,C}\log\sigma_{i,C}\right]$$

where $y_{i,k} = 1$ if image i belongs to class k, otherwise $y_{i,k} = 0$. Similarly, $\sigma_{i,k}$ is the softmax probability that image i belongs to class k.

Derivatives of the loss function for $k \in \{1, \ldots, C\}$ are

$$\begin{aligned}
\frac{\partial L}{\partial z_k} &= -\frac{1}{n}\sum_{i=1}^{n}\left[\frac{y_{i,1}}{\sigma_{i,1}}\frac{\partial\sigma_{i,1}}{\partial z_k} + \cdots + \frac{y_{i,C}}{\sigma_{i,C}}\frac{\partial\sigma_{i,C}}{\partial z_k}\right] \\
&= -\frac{1}{n}\sum_{i=1}^{n}\left[y_{i,1}\left(\delta_{1,k} - \sigma_{i,k}\right) + \cdots + y_{i,C}\left(\delta_{C,k} - \sigma_{i,k}\right)\right] \\
&= -\frac{1}{n}\sum_{i=1}^{n}\left(y_{i,1}\,\delta_{1,k} + \cdots + y_{i,C}\,\delta_{C,k} - \sigma_{i,k}\right) \qquad \text{since } y_{i,1} + \cdots + y_{i,C} = 1
\end{aligned}$$

Q4

Architecture + parameters used to overfit a small subset of the CIFAR-10 data, plots of loss and accuracy of train and test set (2 points)
Architecture + parameters used to achieve at least 50% accuracy on CIFAR-10 as well as plots of loss and accuracy of train and test set (2 points)

Q5

Step 1

Network architecture: 1 hidden layer?
stopping criterion:
Learning rate update schedule:

Step 2

We plotted validation loss against time for various learning rates: 0.05, 0.1, 0.2, etc. The best learning rate is one with lowest validation loss for a given training time i.e. it results in lowest asymptotic error and converges rapidly.

Include the plot for training and validation loss and report training and validation classification error.

Step 3

Use dropout and report if there is any improvement in the validation performance

Step 4

Use L2 and compare the performance with dropout.

Step 5

Optimise the number of hidden layers and the number of neurons in each hidden layer.

A1

No free lunch theorem

A2

No change is needed since our functions are written to accommodate an arbitrary number of classes.