DS-GA 1008 HW 1

Long Chen
Center for Data Science, New York University
lc3424@nyu.edu

Fall 2021

1 Theory

1.1 Two-Layer Neural Nets

1.2 Regression Task

(a)

- Load data to model
- Initialize parameters (random normalized number, or zero)
- Forward step. Compute loss of training
- Backward step. Calculate derivatives
- Update parameters

(b)

- Linear_1:
  - Input: x
  - Output: $z_1 = W^{(1)}x + b^{(1)}$
- f (ReLU):
  - Input: $W^{(1)}x + b^{(1)}$
  - Output: $z_2 = \max(0, W^{(1)}x + b^{(1)})$
- Linear_2:
  - Input: $\max(0, W^{(1)}x + b^{(1)})$
  - Output: $z_3 = W^{(2)} \max(0, W^{(1)}x + b^{(1)}) + b^{(2)}$

- g (identity):
  - Input: $W^{(2)}\max(0, W^{(1)}x + b^{(1)}) + b^{(2)}$
  - Output: $\hat{y} = W^{(2)}\max(0, W^{(1)}x + b^{(1)}) + b^{(2)}$

(c)

- $\frac{\delta \ell}{\delta z_3} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3}$
- $\frac{\delta \ell}{\delta W^{(2)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \frac{\delta z_3}{\delta W^{(2)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \max(0, W^{(1)}x + b^{(1)})$
- $\frac{\delta \ell}{\delta b^{(2)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \frac{\delta z_3}{\delta b^{(2)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3}$
- $\frac{\delta \ell}{\delta W^{(1)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \frac{\delta z_3}{\delta z_2} \frac{\delta z_2}{\delta z_1} \frac{\delta z_1}{\delta W^{(1)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} W^{(2)} \frac{\delta z_2}{\delta z_1}\ x$
- $\frac{\delta \ell}{\delta b^{(1)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \frac{\delta z_3}{\delta z_2} \frac{\delta z_2}{\delta z_1} \frac{\delta z_1}{\delta b^{(1)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} W^{(2)} \frac{\delta z_2}{\delta z_1}$

(d)

- $\frac{\delta \ell}{\delta \hat{y}} = 2\,(\hat{y} - y)$
- $\frac{\delta \hat{y}}{\delta z_3} = 1$
- $\frac{\delta z_2}{\delta z_1} = 0$ if $z_1 \le 0$ and 1 if $z_1 > 0$

1.3 Classification Task

a)

- Linear_1:
  - Input: x
  - Output: $W^{(1)}x + b^{(1)}$
- f (Sigmoid):
  - Input: $W^{(1)}x + b^{(1)}$
  - Output: $\frac{1}{1+\exp\{-(W^{(1)}x+b^{(1)})\}}$
- Linear_2:
  - Input: $\frac{1}{1+\exp\{-(W^{(1)}x+b^{(1)})\}}$
  - Output: $W^{(2)}\frac{1}{1+\exp\{-(W^{(1)}x+b^{(1)})\}} + b^{(2)}$
- g (Sigmoid):
  - Input: $z_2 = W^{(2)}\frac{1}{1+\exp\{-(W^{(1)}x+b^{(1)})\}} + b^{(2)}$
  - Output: $\frac{1}{1+\exp\{-z_2\}}$

Note For derivatives, no much is changed from regression tasks as we are taking derivatives w.r.t. W & b

- $\frac{\delta \ell}{\delta z_3} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3}$
- $\frac{\delta \ell}{\delta W^{(2)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \frac{\delta z_3}{\delta W^{(2)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} z_2$
- $\frac{\delta \ell}{\delta b^{(2)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \frac{\delta z_3}{\delta b^{(2)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3}$
- $\frac{\delta \ell}{\delta W^{(1)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \frac{\delta z_3}{\delta z_2} \frac{\delta z_2}{\delta z_1} \frac{\delta z_1}{\delta W^{(1)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} W^{(2)} \frac{\delta z_2}{\delta z_1} \; x$
- $\frac{\delta \ell}{\delta b^{(1)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} \frac{\delta z_3}{\delta z_2} \frac{\delta z_2}{\delta z_1} \frac{\delta z_1}{\delta b^{(1)}} = \frac{\delta \ell}{\delta \hat{y}} \frac{\delta \hat{y}}{\delta z_3} W^{(2)} \frac{\delta z_2}{\delta z_1}$

- $\frac{\delta \ell}{\delta \hat{y}} = 2\,(\hat{y} - y)$
- $\frac{\delta \hat{y}}{\delta z_3} = \sigma(z_3)(1 - \sigma(z_3))$
- $\frac{\delta z_2}{\delta z_1} = \sigma(z_1)(1 - \sigma(z_1))$

Where $\sigma(x) = \frac{1}{1+\exp\{-x\}}$

(b) Gradient vanishment issues with sigmoid. As $x \longrightarrow \infty$, gradient of sigmoid converges to zero, which does not happen for ReLU. Therefore, in the intermediate layers of a deeper network, we prefer ReLU over sigmoid.

1.4 Conceptual Questions

(a) because softmax is a smooth approximation of the argmax function, as it returns (normalized) index of the maximum value
(b) When there are outliers (very very large numbers relatively), softmax could incur underflow issues where values of relatively small numbers are observed (by the computer) as 0's.
(c) Two consecutive linear layers are essentially ONE linear layer, as could be very easily checked by looking at forward steps. Therefore, some non-linear function (activation function) need to be added.
(d)

- ReLU
  - Pro: Computationally cheap. No vanishing gradient issue.
  - Con: If input is negative, the cell is completely inactive, which could be problematic in some situations.
- Tanh
  - Pro: Smooth gradient, differentiable

  - Con: Vanishing tradient problem.

- Sigmoid:
  - Pro: Normalized to $[-1, 1]$.
  - Con: Non-zero mean output.

- LeakyReLU:
  - Pro: When negative input is observed, the cell does not go to complete inactivity, which could be useful in some situations.
  - Con: Computationally more expensive than ReLU (multiplication needed).

(e) Rotation, skewing (sheer), projection, scaling

For linear: efficiently combines output from nodes for a mixed node, thus discovering inter-relationships.

For non-linear: captures much complex signal (information) of the dataset for better models.

(f) Increase proportionally w.r.t batch size.