Neural Network and Backpropagation Questions

Daeyoung Kim, Xintian Han

CDS, NYU

April 21, 2021

Question 1: Step Activation Function [1]

Suppose we have a neural network with one hidden layer.

$$f(x) = w_0 + \sum_i w_i h_i(x); \quad h_i(x) = g(b_i + v_i x),$$

where activation function g is defined as

$$g(z) = \begin{cases} 1 & \text{if } z \geqslant 0 \\ 0 & \text{if } z < 0 \end{cases}$$

Which of the following functions can be exactly represented by this neural network?

- polynomials of degree one: $l(x) = ax + b$
- hinge loss: $l(x) = max(1 - x, 0)$
- polynomials of degree two: $l(x) = ax^2 + bx + c$
- piecewise constant functions

[1] From CMU

[Solution] Question 1: Step Activation Function

Suppose we have a neural network with one hidden layer.

$$f(x) = w_0 + \sum_i w_i h_i(x); \quad h_i(x) = g(b_i + v_i x),$$

where activation function g is defined as

$$g(z) = \begin{cases} 1 & \text{if } z \geqslant 0 \\ 0 & \text{if } z < 0 \end{cases}$$

Which of the following functions can be exactly represented by this neural network?

- polynomials of degree one: $l(x) = ax + b$ **No**
  If g can be identity function, then the answer is **Yes**
- hinge loss: $l(x) = max(1 - x, 0)$ **No**
- polynomials of degree two: $l(x) = ax^2 + bx + c$ **No**
- piecewise constant functions **Yes**
  $(-c) \cdot g(x - b) + (c) \cdot g(x - a)$ can represent $l(x) = c, a \leqslant x < b$.

Question 2: Power of ReLU [2]

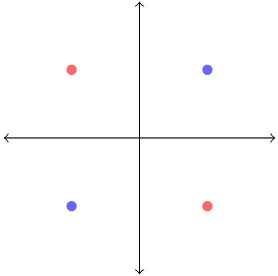
Consider the following small NN:

$$w_2^\top \mathbf{ReLU}\left(W_1 x + b_1\right) + b_2$$

where the data is 2D, W_1 is 2 by 2, b_1 is 2D, w_2 is 2D and b_2 is 1D.

$$x_1 = (1, 1)\ \ y_1 = 1; \quad x_2 = (1, -1)\ \ y_2 = -1; \quad x_3 = (-1, 1)\ \ y_3 = -1; \quad x_4 = (-1, -1)\ \ y_4 = 1$$

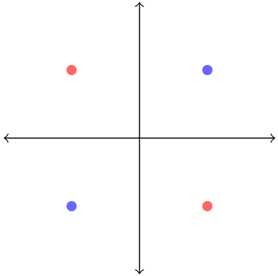
Find b_1, b_2, W_1, w_2 to solve the problem. (Separate points from class $y = 1$ and $y = -1$.)



[2] From Harvard

[Solution] Question 2: Power of ReLU [3]

$$w_2^\top \textbf{ReLU}\left(W_1 x + b_1\right) + b_2$$



One choice is

$$W_1 = \begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix}, b_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$w_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, b_2 = -1$$

Question 3: Backpropagation [4]

Suppose we have a one hidden layer network and computation is:

$$h = \text{RELU}(Wx + b1)$$
$$\hat{y} = \text{softmax}(Uh + b_2)$$
$$J = \text{Cross entropy}(y, \hat{y}) = -\sum_i y_i \log \hat{y}_i$$

The dimensions of the matrices are:

$$W \in \mathbb{R}^{m\times n} \quad x \in \mathbb{R}^n \quad b_1 \in \mathbb{R}^m \quad U \in \mathbb{R}^{k\times m} \quad b_2 \in \mathbb{R}^k$$

Use backpropagation to calculate these four gradients

$$\frac{\partial J}{\partial b_2} \quad \frac{\partial J}{\partial U} \quad \frac{\partial J}{\partial b_1} \quad \frac{\partial J}{\partial W}$$

[4] From Stanford

[Solution] Question 3: Backpropagation

$$z_2 = Uh + b2 \quad \delta_1 = \frac{\partial J}{\partial z_2} = \hat{y} - y$$

$$\frac{\partial J}{\partial b_2} = \delta_1$$

$$\frac{\partial J}{\partial U} = \delta_1 h^T$$

$$\frac{\partial J}{\partial h} = U^T \delta_1$$

$$z_1 = Wx + b_1 \quad \delta_2 = \frac{\partial J}{\partial z_1} = U^T \delta_1 \circ 1\{h > 0\}$$

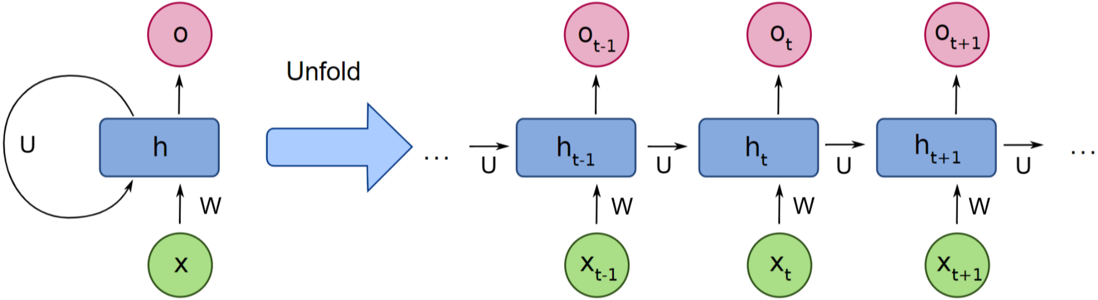
$$\frac{\partial J}{\partial b_1} = \delta_2$$

$$\frac{\partial J}{\partial W} = \delta_2 x^T$$

Coding Exercise

- Computation graph hands-on

[Optional] Recurrent Neural Networks



Source: https://medium.com/deeplearningbrasilia/deep-learning-recurrent-neural-networks-f9482a24d010

[Optional]: Backpropagation in RNN

Suppose we have a recurrent neural network (RNN). The recursive function is:

$$\begin{aligned} \boldsymbol{z}_{t-1} &= \boldsymbol{W}\boldsymbol{x}_{t-1} + \boldsymbol{U}\boldsymbol{h}_{t-1}, \\ \boldsymbol{h}_t &= g(\boldsymbol{z}_{t-1}), \end{aligned}$$

where $\boldsymbol{h}_t$ is the hidden state and $\boldsymbol{x}_t$ is the input at time step t. $\boldsymbol{W}$ and $\boldsymbol{U}$ are the weighted matrix. g is an element-wise activation function. And $\boldsymbol{h}_0$ is a given fixed initial hidden state.

- Assume loss function $\mathcal{L}$ is a function of $\boldsymbol{h}_T$. Given $\partial\mathcal{L}/\partial\boldsymbol{h}_T$, calculate $\partial\mathcal{L}/\partial\boldsymbol{U}$ and $\partial\mathcal{L}/\partial\boldsymbol{W}$.
- Suppose g' is always greater than λ and the smallest singular value of U is larger than $1/\lambda$. What will happen to the gradient $\partial\mathcal{L}/\partial\boldsymbol{U}$ and $\partial\mathcal{L}/\partial\boldsymbol{W}$?
- Suppose g' is always smaller than λ and the largest singular value of U is smaller than $1/\lambda$. What will happen to the gradient $\partial\mathcal{L}/\partial\boldsymbol{U}$ and $\partial\mathcal{L}/\partial\boldsymbol{W}$?

[Solution] [Optional]: Backpropagation in RNN

- $$\frac{\partial \mathcal{L}}{\partial U} = \sum_{t=1}^{T} \left(\Pi_{k=t-1}^{T-1} (\boldsymbol{U}^T D_k)\right) \frac{\partial \mathcal{L}}{\partial \boldsymbol{h}_T} \boldsymbol{h}_{t-1}^T$$
  $$\frac{\partial \mathcal{L}}{\partial W} = \sum_{t=1}^{T} \left(\Pi_{k=t-1}^{T-1} (\boldsymbol{U}^T D_k)\right) \frac{\partial \mathcal{L}}{\partial \boldsymbol{h}_T} \boldsymbol{x}_{t-1}^T$$
  $D_k = \mathrm{diag}(g'(\boldsymbol{z}_k))$ is the Jacobian matrix of the element-wise activation function.
- The smallest singular value of the $\boldsymbol{U}^T D_{k-1}$ will be greater than one. So the smallest singular value of the gradient $\frac{\partial h_s}{\partial h_t}$ will be larger than a^{s-t} for some $a > 1$. So the gradient is going to be exponentially large. This is called exploding gradient.
- The largest singular value of the $\boldsymbol{U}^T D_{k-1}$ will be smaller than one. So the largest singular value of the gradient $\frac{\partial h_s}{\partial h_t}$ will be smaller than a^{s-t} for some $a < 1$. So the gradient is going to be exponentially small. This is called vanishing gradient.