통계학과 2017150412
이재승

STAT 433 – Midterm Part I

1. During backpropagation, when the gradient passes backward through a sigmoid activation function, the gradient will always decrease in magnitude.
   - A. True
   - B. False

   A : True.

2. Suppose that you find that your model's training error looks so good (potential overfitting). What can you do to address this issue? (Check all that apply)
   - A. Data augmentation
   - B. Dropout
   - C. Batch Normalization
   - D. RMSprop Optimizer

   A. B. C

3. Which of the following is true?
   - A. Batch Normalization is an alternative method of dropout.
   - B. Batch Normalization makes training faster.
   - C. Batch Normalization is a non-linear transformation to give nonlinearity to the network.
   - D. Batch Normalization is standardizing the data before training neural network.

   D

4. You want to make the weights sparse and smaller. How can you do that? Why?

ㅁ Pruning을 하면 된다. Pruning은 Model의 Weight들 중 중요도가 낮은 weight의 연결을 제거하여
모델의 parameter를 줄이는 방법이다. 만일 weight 값이 결과에 주는 영향이 다르다면, 상대적으로
영향력이 적은 weight는 삭제하여도 결과에 주는 영향이 적을 것이다. 따라서 최종 결과에 영향력이 적은
weight를 제거한다면 적은 parameter를 가지지만 유사한 성능을 보여주는 모델을 만들 수 있을 것이다.
즉, pruning은 네트워크의 성능이 크게 저하되지 않는 선에서 weight들을 최대한 sparse하게 만드는
방법으로 정의할 수 있다. 이는 Neural Network를 실행하는데 필요한 계산 리소스를 줄이기 위해
수행한다.
pruning에는 Magnitude pruning, sensitivity pruning 등 여러 방법이 있다.
pruning은 Parameter가 줄어들기 때문에 추론 속도가 빨라지고, Regularization이 일반화 성능을
높인다는 장점이 있다. 중요하지 않은 weight를 제거하여 Parameter를 줄여 모델의 속도를 빠르게 하기 위해 이용한다.

5. $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ has a similar performance as sigmoid function except that it is zero-centered. Write down tanh(x) in terms of $\sigma(x)$ where $\sigma(x) = 1/(1 + e^{-x})$. Show your work to get the full credit.

$$\sigma(x) = \frac{1}{1+e^{-x}} \qquad 1-\sigma(x) = 1 - \frac{1}{1+e^{-x}} = \frac{1}{1+e^{x}} = \sigma(-x) \qquad ㅁ\ 1-\sigma(x) = \sigma(-x)$$

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^x + e^{-x} - 2e^{-x}}{e^x + e^{-x}} = 1 + \frac{-2e^{-x}}{e^x + e^{-x}} = 1 - \frac{2}{e^{2x}+1}$$

$$= 1 - 2\sigma(-2x)$$
$$= 1 - 2(1 - \sigma(2x))$$
$$= 1 - 2 + 2\sigma(2x)$$
$$= 2\sigma(2x) - 1$$

$$\therefore \tanh(x) = 2\sigma(2x) - 1$$

6. You have a single layer neural network for a binary classification with a sigmoid activation function as below. (X: nXm matrix, predicted y & true label y: 1 X m)

$$z = WX + b$$
$$h = \sigma(z)$$
$$\hat{y} = h$$
$$L = -\sum_{i}^{m} y_i \log \hat{y} + (1 - y_i)\log(1 - \hat{y}_i)$$

모두 $\sigma(z) \cdot (1-\sigma(z))$를 적용한다.

What is $\frac{\partial L}{\partial W}$? Write your answer as a matrix-matrix multiplication.

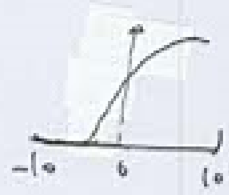
$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial h} \cdot \frac{\partial h}{\partial z} \cdot \frac{\partial z}{\partial W}$$

여기는 모두 $\sigma(z) \cdot (1-\sigma(z))$를 적용

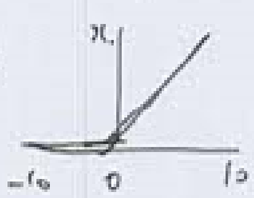
$$= \begin{bmatrix} \frac{\partial}{\partial w_1} L_1(\hat{y}) & \frac{\partial}{\partial w_2} L_1(\hat{y}) & \cdots & \frac{\partial}{\partial w_m} L_1(\hat{y}) \\ \frac{\partial}{\partial w_1} L_2(\hat{y}) & \frac{\partial}{\partial w_2} L_2(\hat{y}) & \cdots & \frac{\partial}{\partial w_m} L_2(\hat{y}) \\ \vdots & \vdots & & \vdots \\ \frac{\partial}{\partial w_1} L_n(\hat{y}) & \frac{\partial}{\partial w_2} L_n(\hat{y}) & \cdots & \frac{\partial}{\partial w_m} L_n(\hat{y}) \end{bmatrix} \cdot 1 \cdot \begin{bmatrix} \sigma(WX_{(1,1)}+b)\cdot(1-\sigma(WX_{(1,1)}+b)) & \cdots & \sigma(WX_{(1,m)}+b)\cdot(1-\sigma(WX_{(1,m)}+b)) \\ \vdots & & \vdots \\ \sigma(WX_{(n,1)}+b)\cdot(1-\sigma(WX_{(n,1)}+b)) & \cdots & \sigma(WX_{(n,m)}+b)\cdot(1-\sigma(WX_{(n,m)}+b)) \end{bmatrix} \cdot \begin{bmatrix} X_{(1,1)} & X_{(1,2)} & \cdots & X_{(1,m)} \\ X_{(2,1)} & X_{(2,2)} & \cdots & X_{(2,m)} \\ \vdots & \vdots & & \vdots \\ X_{(n,1)} & \cdots & \cdots & X_{(n,m)} \end{bmatrix}$$

7. (continued from the above question) suppose that you apply ReLU activation before sigmoid activation. i.e., $\hat{y} = \sigma(ReLU(z))$. Then you classify the object by checking if $\hat{y} \geq 0.5$ or $\hat{y} < 0.5$. What will happen? Why?

Sigmoid $\sigma(x) = \frac{1}{1+e^{-x}}$



ReLU : $\max(0, x)$



위 값은 함수 그래프 형태를 생각했을 때, ReLU를 하고 sigmoid를 적용하면 모든 prediction이 positive하게 될 것이다. 즉, $\hat{y} = \sigma(ReLU(z)) \geq 0.5$가 된다는 것을 의미한다. ReLU는 0 혹은 양수 값을 제시하기 때문에, 이를 sigmoid 함수에 넣으면 다 positive하게 되기 때문이다. 그래서 모든게 $\hat{y} \geq 0.5$로 쏠리게 될 것이다.

8. Suppose that your classmate finds an activation function that is similar to ReLU such that

$$f(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$$

Will you use this? Why?

이용하지 않을 것이다. 비록 이 활성화함수가 non-linear 하지만, discontinuous nonlinear step function이긴 하지만, 0인점을 제외한 모든곳의 gradient가 0이다. 이는 곧 Backpropagation의 과정을 거칠 때 기울기가 거의 없어지는 문제를 야기하게 된다. 기울기가 거의 없어지는 것은 최적화의 과정에서 굉장히 크리티컬하게, 부정적으로 작용한다. 그렇기 때문에 나는 이것을 이용하지 않을 것이다.

9. Provide two reasons why we are using convolutional layers instead of fully connected layers for image classification.

1) 이미지의 공간정보 소실을 막기 위해서다. Fully Connected Layer만으로 구성한 네트워크는
1차원 배열로만 이루어져 있다. 네트워크의 Input이 이미지일 때 Fully Connected Layer가 이미지를
학습하려면 (1,0)과 (2,0)이 가진 벡터의 연관관계에 대한 정보가 있어야 하는데,
1차원 배열로 되면 (1,0) (1,1) ... (1,N) (2,0)과 같이 두 벡터간의 연관관계라는 정보가
소실되어서 사람이 파악하는 것만큼 특징 파악이 어렵다. 그래서 1차원 배열로 늘어뜨리기 전에
Input을 특징 스페이스 안의 특징값들의 정보를 가지고 있는 노드로 하기 위해 Convolution Layer를
이용한다.

2) Convolutional Layer가 Fully Connected Layer보다 더 좁은 범위의 Feature를 나타내기 때문이다.
Fully Connected Layer의 Neuron은 이전 층의 모든 Neuron과 연결되기에 이전 층의 Neuron 중 하나가
변화하면 변할 수 있다. 하지만 Convolutional Layer의 Neuron은 Convolutional kernel의 width
내에서 이전 계층의 국소 neuron에만 연결된다. 이는 결과적으로 Convolutional Layer의 Neuron은
한 Neuron의 activation이 이전 layer의 대부분의 neuron 활성화에 insensitive 하다는 점에서
더 좁은 범위의 feature를 나타낼 수 있다.

10. Consider to build a CNN for an image classification problem in which the layers are defined by the left column below. Fill the table below. Assume that width & height of the kernels (for Conv, Pool) are the same. Stride 1 Pad 1 for convolving layers. Stride 2 Pad 0 for Pooling layers. FC: a fully-connected layer.

Layer	Output Size		Layer		Number of parameters
	C	H/W	filters	kernel	
Input	3	32	-	-	0
Conv	16	32	16	3	448
ReLU	16	32	-	-	0
Pool	16	16	-	2	0
BatchNorm	16	16	-	-	32
Conv	16	16	16	3	2320
ReLU	16	16	-	-	0
Pool	16	8	-	2	0
Flatten	16·8·8	-	-	-	0
FC	10	-	-	-	10250