Name: ______________________

University of Illinois Spring 2020

CS 446/ECE 449 Machine Learning
Homework 5: Deep Net

Due on Thursday March 5 2020, noon Central Time

1. **[25 points]** Deep Net

We want to train a simple deep net $f(w_3, w_2, w_1, x) = w_3\sigma_2(\underbrace{w_2\sigma_1(\underbrace{w_1 x}_{x_1})}_{x_2})$ where $w_1, w_2, w_3, x \in \mathbb{R}$ are real valued and $\sigma_1(u) = \sigma_2(u) = \frac{1}{1+\exp(-u)}$ is the sigmoid function.

(a) (1 point) Draw the computation graph that is specified by the function f.

Solution:
f connected to w_2 and σ_2; σ_2 connected to w_2 and σ_1; σ_1 connected to w_1 and x

(b) (2 points) Compute $\frac{\partial\sigma_1}{\partial u}$ and provide the answer (1) using only u, the exp-function and the square function, and (2) using only $\sigma_1(u)$.

Solution:
(1) $\frac{\partial\sigma_1}{\partial u} = \frac{\exp(-u)}{(1+\exp(-u))^2}$
(2) $\frac{\partial\sigma_1}{\partial u} = \sigma_1(u)\,(1-\sigma_1(u))$

(c) (2 points) Describe briefly what is meant by a 'forward pass' and a 'backward pass'?

Solution:
Forward pass: given x and current parameters w we compute f from data to final result
Backward pass: given the final and intermediate results we back-propagate the error to obtain the derivatives

(d) (2 points) Compute $\frac{\partial f}{\partial w_3}$. Which result should we retain from the forward pass in order to make computation of this derivative easy?

Solution:
$\frac{\partial f}{\partial w_3} = \sigma_2(w_2\sigma_1(w_1 x))$; retain $\sigma_2(w_2\sigma_1(w_1 x))$ from forward pass to avoid having to recompute it

(e) (3 points) Compute $\frac{\partial f}{\partial w_2}$. Make use of the second option obtained in part (b). Which results should we retain from the forward pass in order to make computation of this derivative easy?

Solution:

$$\frac{\partial f}{\partial w_2} = \underbrace{\frac{\partial f}{\partial \sigma_2}}_{w_3} \cdot \underbrace{\frac{\partial \sigma_2}{\partial x_2}}_{\sigma_2(x_2)(1-\sigma_2(x_2))} \cdot \underbrace{\frac{\partial x_2}{\partial w_2}}_{\sigma_1(w_1 x)}$$

Retain σ_2 or x_2 and σ_1 or x_1

(f) (5 points) Compute $\frac{\partial f}{\partial w_1}$. Make use of the second option obtained in part (b). Which results should we retain from the forward pass in order to make computation of this derivative easy? In what order should we compute the derivatives $\frac{\partial f}{\partial w_3}$, $\frac{\partial f}{\partial w_2}$ and $\frac{\partial f}{\partial w_1}$ in order to obtain the result as early as possible and in order to reuse as many results as possible. How is this order related to the forward pass?

Solution:

$$\frac{\partial f}{\partial w_2} = \underbrace{\underbrace{\frac{\partial f}{\partial \sigma_2}}_{w_3} \cdot \underbrace{\frac{\partial \sigma_2}{\partial x_2}}_{\sigma_2(x_2)(1-\sigma_2(x_2))}}_{\text{already computed in part (e)}} \cdot \underbrace{\frac{\partial x_2}{\partial \sigma_1}}_{w_2} \cdot \underbrace{\frac{\partial \sigma_1}{\partial x_1}}_{\sigma_1(x_1)(1-\sigma_1(x_1))} \cdot \underbrace{\frac{\partial x_1}{\partial w_1}}_{x}$$

retain/use: results from part (e) and also keep σ_1 or x_1

reverse order of forward pass is computationally most effective, *i.e.*, $\frac{\partial f}{\partial w_3}, \frac{\partial f}{\partial w_2}, \frac{\partial f}{\partial w_1}$

(g) (2 points) We now want to train a convolutional neural net for 10-class classification of MNIST images which are of size 28 × 28. As a first layer we use 20 2d convolutions each with a filter size of 5 × 5, a stride of 1 and a padding of 0. What is the output dimension after this layer? Subsequently we apply max-pooling with a size of 2 × 2. What is the output dimension after this layer?

Solution:

Output after convolution: 24 × 24

Output after subsequent max-pooling: 12 × 12

(h) (4 points) After having applied the two layers (convolution + pooling) designed in part (g) we want to use a second convolution + max-pooling operation. The max-pooling operation has a filter size of 2 × 2. The desired output should have 50 channels and should be of size 4 × 4. What is the filter size, the stride, and the channel dimension of the second convolution operation, assuming that padding is 0?

Solution:

Desired output: 4 × 4

Input to max-pooling which results in desired output: 8 × 8

Input to convolution was: 12 × 12

Consequently: filter size is 5 × 5; stride is 1; padding is 0; channels is 50

(i) (4 points) Complete `A5_DeepNet.py` by first implementing the two operations, where each operation is convolution+pooling. We also want to apply two linear layers, which you must implement. The first one maps from a $50 \cdot 4 \cdot 4$ dimensional space to a 500 dimensional one. After each convolution and after the first linear layer, we also want to apply ReLU non-linearities. Provide your entire code pertaining to the "Net" class here. What is the best test set accuracy that you observed during training with this architecture? How many parameters does your network have (including biases)?

Name:

Solution:

```
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(1, 20, 5, 1)
        self.conv2 = nn.Conv2d(20, 50, 5, 1)
        self.fc1 = nn.Linear(4*4*50, 500)
        self.fc2 = nn.Linear(500, 10)

    def forward(self, x):
        x = F.relu(self.conv1(x))
        x = F.max_pool2d(x, 2, 2)
        x = F.relu(self.conv2(x))
        x = F.max_pool2d(x, 2, 2)
        x = x.view(-1, 4*4*50)
        x = F.relu(self.fc1(x))
        return self.fc2(x)
```

Number of parameters: 431,080

Best test set accuracy: 99.15%