

CIS 520, Machine Learning, Fall 2020

## Homework 4

Due: Monday, October 12th, 11:59pm

Submit to Gradescope

Chun Chang

### 1 Neural Networks: Backpropagation

Your task is to now compute the derivatives of the loss function given in 1 with respect to  $W_1$ ,  $W_2$ ,  $b_1$  and  $b_2$  by hand, i.e.,  $\frac{\partial Loss}{\partial W_1}$ ,  $\frac{\partial Loss}{\partial b_1}$ , and  $\frac{\partial Loss}{\partial b_2}$ .

$$Loss = y * \ln(\sigma(z_2)) + (1 - y) * \ln(1 - \sigma(z_2)) \quad (1)$$

Show all the intermediate derivative computation steps. You might benefit from making a rough schematic of the backpropagation process. Also recall the derivatives of the softmax function and the tanh function:

$$\frac{d\sigma(z_2)}{dz} = \ln(\sigma(z_2)) * \ln(1 - \sigma(z_2)) \quad (2)$$

$$\frac{\partial \tanh(z_1)}{\partial z_1} = 1 - \tanh^2(z_1) \quad (3)$$

$$\begin{aligned} \frac{\partial Loss}{\partial W_1} &= y \frac{\partial \ln(\sigma(z_2))}{\partial W_1} + (1 - y) \frac{\partial \ln(1 - \sigma(z_2))}{\partial W_1} \\ &= y \frac{1}{\sigma(z_2)} \frac{\partial \sigma(z_2)}{\partial z_2} \frac{\partial z_2}{\partial W_1} + (1 - y) \frac{1}{1 - \sigma(z_2)} - \frac{\partial \sigma(z_2)}{\partial z_2} \frac{\partial z_2}{\partial W_1} \\ &= y \frac{1}{\sigma(z_2)} \sigma(z_2)(1 - \sigma(z_2)) \frac{\partial(\tanh(z_1)W_2 + b_2)}{\partial z_1} \frac{\partial z_1}{\partial W_1} \\ &\quad + (1 - y) \frac{-1}{1 - \sigma(z_2)} \sigma(z_2)(1 - \sigma(z_2)) \frac{\partial(\tanh(z_1)W_2 + b_2)}{\partial z_1} \frac{\partial z_1}{\partial W_1} \\ &= (y - \sigma(z_2))(1 - \tanh^2(z_1))W_2x \end{aligned}$$

$$\begin{aligned} \frac{\partial Loss}{\partial W_2} &= y \frac{\partial \ln(\sigma(z_2))}{\partial W_2} + (1 - y) \frac{\partial \ln(1 - \sigma(z_2))}{\partial W_2} \\ &= y \frac{1}{\sigma(z_2)} \frac{\partial \sigma(z_2)}{\partial z_2} \frac{\partial z_2}{\partial W_2} + (1 - y) \frac{1}{1 - \sigma(z_2)} - \frac{\partial \sigma(z_2)}{\partial z_2} \frac{\partial z_2}{\partial W_2} \\ &= y \frac{1}{\sigma(z_2)} \sigma(z_2)(1 - \sigma(z_2)) \frac{\partial(a_1W_2 + b_2)}{\partial W_2} \\ &\quad + (1 - y) \frac{-1}{1 - \sigma(z_2)} \sigma(z_2)(1 - \sigma(z_2)) \frac{\partial(a_1W_2 + b_2)}{\partial W_2} \\ &= (y - \sigma(z_2))a_1 \end{aligned}$$

$$\begin{aligned}
\frac{\partial Loss}{\partial b_1} &= y \frac{\partial \ln(\sigma(z_2))}{\partial b_1} + (1-y) \frac{\partial \ln(1-\sigma(z_2))}{\partial b_1} \\
&= y \frac{1}{\sigma(z_2)} \frac{\partial \sigma(z_2)}{\partial z_2} \frac{\partial z_2}{\partial b_1} + (1-y) \frac{1}{1-\sigma(z_2)} - \frac{\partial \sigma(z_2)}{\partial z_2} \frac{\partial z_2}{\partial b_1} \\
&= y \frac{1}{\sigma(z_2)} \sigma(z_2)(1-\sigma(z_2)) \frac{\partial(\tanh(z_1)W_2 + b_2)}{\partial z_1} \frac{\partial z_1}{\partial b_1} \\
&\quad + (1-y) \frac{-1}{1-\sigma(z_2)} \sigma(z_2)(1-\sigma(z_2)) \frac{\partial(\tanh(z_1)W_2 + b_2)}{\partial z_1} \frac{\partial z_1}{\partial b_1} \\
&= (y - \sigma(z_2))(1 - \tanh^2(z_1))W_2
\end{aligned}$$

$$\begin{aligned}
\frac{\partial Loss}{\partial W_2} &= y \frac{\partial \ln(\sigma(z_2))}{\partial b_2} + (1-y) \frac{\partial \ln(1-\sigma(z_2))}{\partial b_2} \\
&= y \frac{1}{\sigma(z_2)} \frac{\partial \sigma(z_2)}{\partial z_2} \frac{\partial z_2}{\partial b_2} + (1-y) \frac{1}{1-\sigma(z_2)} - \frac{\partial \sigma(z_2)}{\partial z_2} \frac{\partial z_2}{\partial b_2} \\
&= y \frac{1}{\sigma(z_2)} \sigma(z_2)(1-\sigma(z_2)) \frac{\partial(a_1W_2 + b_2)}{\partial b_2} \\
&\quad + (1-y) \frac{-1}{1-\sigma(z_2)} \sigma(z_2)(1-\sigma(z_2)) \frac{\partial(a_1W_2 + b_2)}{\partial b_2} \\
&= y - \sigma(z_2)
\end{aligned}$$

## 2 Programming

### 2.1 Random Forests

Tabulate the prediction results on the test set in Table 1

<b>n_estimators</b>	<b>Accuracy (%)</b>
1	73.7%
5	92.5%
10	95.2%
50	96.6%
100	97.6%
500	97.6%

Table 1: Accuracy for the Random Forests classification problem on the test set

### 2.2 Kernel SVM

Tabulate the prediction results on the test set in Table 2.

<b>kernel</b>	<b>Accuracy (%)</b>
Linear	98.3%
Poly	99.3%
RBF	98.6%

Table 2: Accuracy for the kernel SVM classification problem on the test set

## 2.3 Multi Layer Perceptron

Tabulate the prediction results on the test set in Table 3.

Network Architecture	Accuracy (%)
(3)	85.8%
(10)	94.6%
(10,10,10)	92.92%
(20,50,20)	96.2%

Table 3: Accuracy for the MLP classification problem on the test set

## 2.4 AdaBoost

Tabulate the prediction results on the test set in Table 4.

n_estimators	Accuracy (%)
1	43.7%
5	82.4%
10	83.1%
50	89.8%
100	92.9%
150	92.6 %

Table 4: Accuracy for the AdaBoost classification problem on the test set

## 2.5 Short Answer

For random forest trees, if the features are not correlated, the model could predict better as the number of estimators increases; otherwise, as the the number of estimators increases, the model trained with correlated data tends to overfitting, but it is really rare since we pick random features in each iteration.

For MLP, more the total numbers of neurons or layers, more complicated the model is. The complicated model tends to perform better since the training process extracts more classification details from the data. MLP is still like nerual network, more complicated the model is, higher the variance we tend to get.

For SVM, different kernal uses different mapping functions to transform the features to another space. For nolinear dataset, we want to use Rbf or poly to fit the data. For linear separable data, we could get good model by linear kernal.

For adaboost, it should be really similar to random forest tree. However, adaboost is more possible to overfit than random forest trees. The more the estimator we have trained from the uncorrelated dataset, the lower the bias of the final estimator.

# 3 Convolutional Neural Networks

## 3.1 Theory

1. The output volume is:

$$\frac{86 + 4 - 6}{3} + 1 = 29$$

output:29x29x64

2. Including bias parameters, this hidden layer has the following number of parameters:

$$450 * 450 * 3 * 200 + 200 = 121500200$$

3. The output from the simple convolutional layer for the given filter and input is:

Row	Column	Filter	Value
1	1	1	6
1	1	2	17
1	2	1	-10
2	1	1	-7

Table 5: Output from convolution

4. The total number of trainable parameters in this network is:

$$40 * 25 + 25^2 * 6 + 25 + 25 * 7 + 1 = 4951$$

5. State two advantages of convolutional neural networks over fully connected networks.

The CNN has fewer weights to train therefore avoiding the overfitting, large memory usage and long time computation

## 3.2 Programming

For this question, refer to the Jupyter Notebook. You will be using PyTorch to implement a convolutional neural network – the notebook will have detailed instructions. We will be using the fashion MNIST dataset for a classification task.

### 3.2.1 Convolutional Neural Network

Add the accuracy and the loss curve from tensorboard in this report:

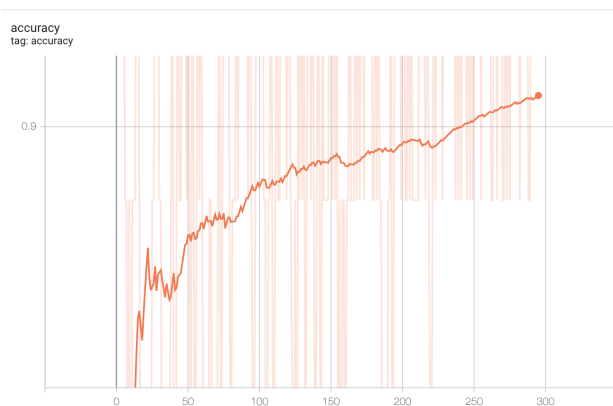


Figure 1: Accuracy curve

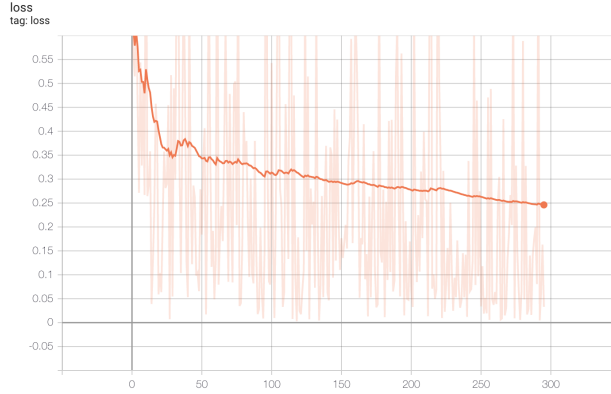


Figure 2: Loss curve

### 3.2.2 Accuracy

Report the overall accuracy and the per-class accuracy:

Overall Accuracy	91%
------------------	-----

Table 6: Overall Accuracy for Convolutional Neural Network

Class	Accuracy
T-shirt/top	92%
Trouser	96%
Pullover	87%
Dress	94%
Coat	85%
Sandal	98%
Shirt	68%
Sneaker	98%
Bag	98%
Ankle boot	96%

Table 7: Per Class Accuracy for Convolutional Neural Network

Identify the problematic classes and list the possible reasons as to why these classes may have significantly lower accuracy compared to other classes.

My CNN did poor on classifying Shirt. The reason might be the the shirt, t-shirt and several other clothes shared the global shape, then the CNN failed to classify due to the similar contours.

## 4 Multiclass Adaboost and Support Vector Machines

### 4.1 Adaboost: Theory

(a) Show that

$$\begin{aligned}
D_{T+1}(i) &= D_T * \frac{\exp(\alpha_t \tilde{h}_{t,y_i}(x_i))}{Z_t} \\
&= D_{T-1} * \frac{\exp(\alpha_{t-1} \tilde{h}_{t-1,y_i}(x_i))}{Z_{t-1}} * \frac{\exp(\alpha_t \tilde{h}_{t,y_i}(x_i))}{Z_T} \\
&= D_1 * \frac{\exp(\alpha_1 \tilde{h}_{1,y_i}(x_i))}{Z_1} * \dots * \frac{\exp(\alpha_{T-1} \tilde{h}_{T-1,y_i}(x_i))}{Z_{T-1}} * \frac{\exp(\alpha_T \tilde{h}_{T,y_i}(x_i))}{Z_T} \\
&= \frac{\frac{1}{m} \exp(-\sum_{t=1}^T \alpha_t \tilde{h}_{t,y_i}(x_i))}{\prod_{t=1}^T Z_t} \\
&= \frac{\frac{1}{m} e^{-F_{T,y_i}(x_i)}}{\prod_{t=1}^T Z_t}
\end{aligned}$$

where

$$F_{T,y_i}(x_i) = \sum_{t=1}^T \alpha_t \tilde{h}_{t,y_i}(x_i)$$

(b) Show that

$$\mathbf{1}(H(x_i) \neq y_i) \leq \mathbf{1}(F_{T,y_i}(x_i) < 0)$$

For case:  $H(x_i) = y_i \rightarrow \mathbf{1}(H(x_i) = y_i) = 0$

$$1: \mathbf{1}(F_{T,y_i}(x_i) < 0) > \mathbf{1}(H(x_i) = y_i) = 0$$

$$2: \mathbf{1}(F_{T,y_i}(x_i) > 0) = \mathbf{1}(H(x_i) = y_i) = 0$$

For case:  $H(x_i) \neq y_i \rightarrow \mathbf{1}(H(x_i) \neq y_i) = 1$

$$\begin{aligned}
F_{T,y_i}(x_i) &= \sum_{t=1}^T \alpha_t * \tilde{h}_{y_i,t}(x_i) = \sum_{k=1}^K \sum_{t=1}^T \alpha_t * \tilde{h}_{k,t}(x_i) - \sum_{k=1, k \neq y_i}^K \sum_{t=1}^T \alpha_t * \tilde{h}_{k,t}(x_i) \\
&= -(K-2) \sum_{t=1}^T \alpha_t - (K-3) \sum_{t=1}^T \alpha_t \\
&= -\sum_{t=1}^T \alpha_t < 0 \\
&\rightarrow F_{T,y_i}(x_i) < 0 \rightarrow \mathbf{1}(H(x_i) \neq y_i) = 1
\end{aligned}$$

(c) Show that

$$\text{er}_S[H] \leq \frac{1}{m} \sum_{i=1}^m e^{-F_{T,y_i}(x_i)} = \prod_{t=1}^T Z_t$$

*Hint:* For the inequality, use the result of part (b) above, and the fact that  $\mathbf{1}(u < 0) \leq e^{-u}$ ; for the equality, use the result of part (a) above.

$$\begin{aligned}
\text{er}_S[H] &= \frac{1}{m} \sum_{i=1}^m \mathbf{1}(H(x_i) \neq y_i) \leq \frac{1}{m} \sum_{i=1}^m \mathbf{1}(F_{T,y_i}(x_i) < 0) \\
\text{because: } \mathbf{1}(F_{T,y_i}(x_i) < 0) &\leq e^{-F_{T,y_i}(x_i)} \\
\text{er}_S[H] &= \frac{1}{m} \sum_{i=1}^m \mathbf{1}(H(x_i) \neq y_i) \leq \frac{1}{m} \sum_{i=1}^m \mathbf{1}(F_{T,y_i}(x_i) < 0) \leq \frac{1}{m} \sum_{i=1}^m e^{-F_{T,y_i}(x_i)} \\
&= \sum_{i=1}^m D_{T+1}(i) \prod_{t=1}^T Z_t = \prod_{t=1}^T Z_t
\end{aligned}$$

(d) Show that for the given choice of  $\alpha_t$ , we have

$$Z_t = 2\sqrt{\text{er}_t(1 - \text{er}_t)}.$$


$$\begin{aligned}
Z_t &= \sum_{i=1}^m D_t(i) \exp\{-\alpha_t \tilde{h}_{y_i,t}(x_i)\} = \sum_{i=1, \tilde{h}_{y_i,t}(x_i) \neq y_i}^m D_t(i) \exp\{\alpha_t\} + \sum_{i=1, \tilde{h}_{y_i,t}(x_i) = y_i}^m D_t(i) \exp\{-\alpha_t\} \\
&= \exp\{-\alpha_t\} * (1 - \text{er}_t) + \exp\{\alpha_t\} * \text{er}_t \\
\text{given: } \alpha_t &= \frac{1}{2} \ln\left(\frac{1 - \text{er}_t}{\text{er}_t}\right) \\
Z_t &= \sqrt{(1 - \text{er}_t)\text{er}_t} + \sqrt{(1 - \text{er}_t)\text{er}_t} = 2\sqrt{(1 - \text{er}_t)\text{er}_t}
\end{aligned}$$

(e) Suppose  $\text{er}_t \leq -\gamma$  for all  $t$  (where  $0 < \gamma \leq$ ). Then show that

$$\text{er}_S[H] \leq e^{-2T\gamma^2}.$$

$$\begin{aligned}
&\text{plug in the inequality: } \text{er}_t \leq \frac{1}{2} - \gamma \text{ to part(d)} \\
Z_t &= 2\sqrt{\left(\frac{1}{2} - \gamma\right) * \left(\frac{1}{2} + \gamma\right)} \\
&= \sqrt{1 - 4\gamma^2} \\
&\text{applied the inequality: } 1 + x \leq e^x \\
Z_t &\leq \exp\{-\gamma^2\} \\
\prod_{t=1}^T Z_t &\leq \exp\{-T\gamma^2\} \\
\text{er}_S[H] &\leq \prod_{t=1}^T Z_t \leq \exp\{-T\gamma^2\}
\end{aligned}$$

## 4.2 Support Vector Machine



HW4/questions/SVM.png

What is the largest number of data points that can be removed from the training set without changing the hard margin SVM solution? Explain your solution.

17 points can be removed since the support vector machine only needs two data points one border( $wx+b = + - 1$ ) and one on the other to keep the hyperplane. In our case, we can pick two negative samples on the border of margin and one positive on the border of the margin, or the other way, two positive on the border and one negative on the border, works as well.