Homework 3

Spring 2020 CSCI 5525: Machine Learning

Due on March 20th 11:59 p.m.

Please type in your info:

- **Name**:Pranay Patil
- **Student ID**:5592264
- **Email**:patil122@umn.edu
- **Collaborators, and on which problems:** Prabhjot Singh Rai - 3

Homework Policy. (1) You are encouraged to collaborate with your classmates on homework problems, but each person must write up the final solutions individually. You need to fill in above to specify which problems were a collaborative effort and with whom. (2) Regarding online resources, you should **not**:

- Google around for solutions to homework problems,
- Ask for help on online.
- Look up things/post on sites like Quora, StackExchange, etc.

Submission. Submit a PDF using this LaTeX template for written assignment part and submit .py files for all programming part. You should upload all the files on Canvas.

Written Assignment

Instruction. For each problem, you are required to write down a full mathematical proof to establish the claim.

Problem 1. VC Dimension.

(**5 points**) Prove this claim formally: If $\mathcal{F}$ is finite, then $\text{VCD}(\mathcal{F}) \leq \log(|\mathcal{F}|)$.

Your answer. Restriction of F to C is defined as the set of functions from C to {0, 1} that can be derived from F.

$$F_C = \{(f(c_1), ..., f(c_m)) : f \ \epsilon \ F\}$$

If F shatters C, then F_c is the set of all functions from C to {0, 1}.
That is, $|F_c| = 2^{|C|}$
Since F_c consists only of subset of F,

$$|F_c| \leq |F|$$

Also, if $|F| < 2^{|C|}$, then F can not completely shatter C, since there won't be enough functions in F from C to {0, 1}. To successfully shatter C, F should have at least $2^{|C|}$ functions for binary classification. In other words, maximum number of points which can be shattered by F i.e. $|C|$, is constrained by $|F|$
Taking log_2 above conditions imply that,

$$log_2(|F|) \geq |C|$$
$$log_2(|F|) \geq VCD(F)$$
$$VCD(F) \leq log_2(|F|)$$

The proof is for binary classification, but can be generalized.

Problem 2. VC Dimension.

(**5 points**) Suppose the space of instances $X = \mathbb{R}^2$.

A binary classifier H_t whose decision boundary is a triangle as Fig. 1 sketches, where points inside the triangle are classified as positive labels. What is the VC dimension of H_t? Please give a proof.

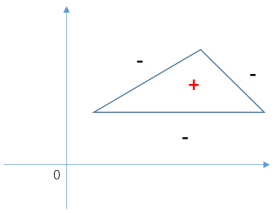


Figure 1: sketch of H_t

Your answer. VC dimension of H_t is 7.
Proof:
Consider following case with 7 points

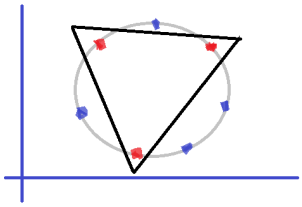


Figure 2: 7 datapoints

Datapoints upto 7 can be separated into positive and negative samples with a triangle. But 8 points can not as can be seen from the figure below. No triangle can separate 8 points into two classes, if they are organized in the following fashion.

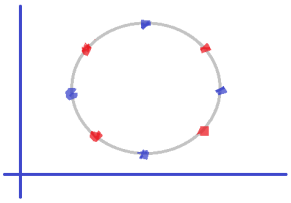


Figure 3: 8 datapoints

The reason is, for 7 datapoints, maximum numbers of groups which can be formed by both categories are 4 and 3. The later class with 3 groups can be easily separated with 3 sides of the triangle and hence 7 points can be shattered successfully.
For 8 datapoints, maximum numbers of groups which can be formed by both categories are 4 and 4. Now these 4 groups can't be separated from others with 3 sides of a triangle. And hence it can't be shattered by a triangle.

Problem 3. Uniform convergence.

In this problem, we will prove a stronger generalization error bound for the binary classification that uses more information about the distribution. Let us say that P is a distribution over (X, Y) pairs where $X \in \mathcal{X}$ and $Y \in \{+1, -1\}$. Let $\mathcal{H} \subset \mathcal{X} \to \{+1, -1\}$ be a finite hypothesis class and let ℓ denote the zero-one loss $\ell(\hat{y}, y) = 1\{\hat{y} \neq y\}$. Let $R(h) = \mathbb{E}\,\ell(h(X), Y)$ denote the risk and let $h^* = \min_{h \in \mathcal{H}} R(h)$. Given n samples, let $\hat{h}_n$ denote the empirical risk minimizer. Here, we want to prove a sample complexity bound of the form:

$$R(\hat{h}_n) - R(h^*) \leq c_1 \sqrt{\frac{R(h^*)\log(|\mathcal{H}|/\delta)}{n}} + c_2 \frac{\log(|\mathcal{H}|/\delta)}{n} \tag{1}$$

for constants c_1, c_2. If $R(h^*)$ is small, this can be a much better bound than the usual excess risk bound. In particular, if $R(h^*) = 0$, this bound recovers the $1/n$-rate.

To prove the result, we will use Bernstein's inequality, which is a sharper concentration result.

Theorem 1 *(Bernstein's inequality). Let $X_1, ..., X_n$ be i.i.d real-valued random variables with mean zero, and such that $|X_i| \leq M$ for all i. Then, for all $t > 0$*

$$\mathbb{P}[\sum_{i=1}^{n} X_i \geq t] \leq \exp\left(-\frac{t^2/2}{\sum_{i=1}^{n} \mathbb{E}[X_i^2] + Mt/3}\right) \tag{2}$$

(a) (**6 points**) Using this inequality, show that with probability at least $1 - \delta$

$$|\bar{X}| \leq \sqrt{\left(\frac{2\,\mathbb{E}[X^2]\log(2/\delta)}{n}\right)} + \frac{2M\log(2/\delta)}{3n} \tag{3}$$

where $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$ and X_i's satisfy the conditions of Bernstein's inequality.

Your answer. a For this proof, let us find the value of t in terms of δ first,

$$Let,\ \frac{\delta}{2} = exp(-\frac{\frac{t^2}{2}}{\sum_{i=1}^{n}\mathbb{E}[X_i^2] + Mt/3})$$

$$\log(\frac{\delta}{2}) = -\frac{\frac{t^2}{2}}{\sum_{i=1}^{n}\mathbb{E}[X_i^2] + Mt/3}$$

$$\log(\frac{2}{\delta}) = \frac{\frac{t^2}{2}}{\sum_{i=1}^{n}\mathbb{E}[X_i^2] + Mt/3}$$

$$\log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2] + Mt/3) = \frac{t^2}{2}$$

$$\frac{t^2}{2} - \log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2] + Mt/3) = 0$$

$$\frac{t^2}{2} - \log(\frac{2}{\delta})(M/3)t - \log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2]) = 0$$

To find t, for any quadratic equation, solution is given by $\frac{-b+\sqrt{b^2-4ac}}{2a}$

$$t = \frac{\log(\frac{2}{\delta})(M/3) + \sqrt{\log(\frac{2}{\delta})(M/3)^2 + 2(\log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2]))}}{1}$$

$$Now,\ if\ a > 0\ and\ b > 0,\ \sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$$

$$t \leq \log(\frac{2}{\delta})(M/3) + \sqrt{\log(\frac{2}{\delta})(M/3)^2} + \sqrt{2(\log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2]))}$$

$$t \leq 2\log(\frac{2}{\delta})(M/3) + \sqrt{2(\log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2]))}$$

From the problem we got,

$$P[|\sum_{i=1}^{n} X_i| > t] = \delta$$

$$P[|\sum_{i=1}^{n} X_i| \leq t] = 1 - \delta$$

So we get,

$$|\sum_{i=1}^{n} X_i| \leq t$$

$$|X_i| \leq \frac{t}{n}$$

Substituting t

$$|X_i| \leq \frac{2\log(\frac{2}{\delta})(M/3) + \sqrt{2(\log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2]))}}{n}$$

$$|X_i| \leq 2\log(\frac{2}{\delta})(M/3n) + \sqrt{\frac{2(\log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2]))}{n^2}}$$

$$|X_i| \leq 2\frac{M}{3n}\log(\frac{2}{\delta}) + \sqrt{\frac{2(\log(\frac{2}{\delta})(\sum_{i=1}^{n}\mathbb{E}[X_i^2]))}{n}}$$

$$|X_i| \leq \sqrt{\frac{2\sum_{i=1}^{n}\mathbb{E}[X_i^2]\log(\frac{2}{\delta})}{n}} + \frac{2M\log(\frac{2}{\delta})}{3n}$$

(b)(**Extra Credits 4 points**) Then, we will use Eq. (3) and the union bound, show Eq. (1)

At first, you will use the idea of Bernstein's inequality to show that with probability at least $1-\delta$, $\forall h \in \mathcal{H}$

$$|\hat{R}(h) - R(h)| \leq \sqrt{\left(\frac{R(h)2\log(2|\mathcal{H}|/\delta)}{n}\right)} + \frac{2M\log(2|\mathcal{H}|/\delta)}{3n} \tag{4}$$

where $\hat{R}(h)$ is the empirical risk and $R(h)$ is the true risk.

(Hint: consider random variables X as the value of loss function. Pay attention to that X has zero mean.)

Your answer. b (c) (**Extra Credits 4 points**) Finally, we will use Eq. (4) to find (1)

Your answer. c $\hat{R}(h)$ is a empirical risk and $R(h)$ is true risk.

Problem 4. Bias-Variance Trade-off.

(**Total 5 points**) In the lecture, **expected test error using a machine learning algorithm, $\boldsymbol{A}$**, can be given as (with respect to squared loss):

$$E_{x,y,D}\left[(f_D(x) - y)^2\right]$$

where D represents set of training points and (x, y) pairs are test points

We can write:

$$
\begin{aligned}
&E_{x,y,D}\left[(f_D(x)-y)^2\right] \\
&= E_{x,y,D}\left[\left(\left(f_D(x)-\bar{f}(x)\right)+\left(\bar{f}(x)-y\right)\right)^2\right] \\
&= E_{x,D}\left[(f_D(x)-\bar{f}(x))^2\right] + 2\,E_{x,y,D}\left[\left(f_D(x)-\bar{f}(x)\right)\left(\bar{f}(x)-y\right)\right] + E_{x,y}\left[\left(\bar{f}(x)-y\right)^2\right]
\end{aligned}
$$

a) Prove $E_{x,y,D}\left[\left(f_D(x)-\bar{f}(x)\right)\left(\bar{f}(x)-y\right)\right]=0$

Proving above gives $E_{x,y,D}\left[(f_D(x)-y)^2\right] = \underbrace{E_{x,D}\left[(f_D(x)-\bar{f}(x))^2\right]}_{\text{Variance}} + E_{x,y}\left[\left(\bar{f}(x)-y\right)^2\right]$

where 1^{st} term is **variance**. The 2^{nd} term can further be decomposed as follows:

$$
\begin{aligned}
&E_{x,y}\left[\left(\bar{f}(x)-y\right)^2\right] = E_{x,y}\left[\left(\bar{f}(x)-\bar{y}(x)\right)+\left(\bar{y}(x)-y\right)^2\right] \\
&= \underbrace{E_{x,y}\left[\left(\bar{y}(x)-y\right)^2\right]}_{\text{Noise}} + \underbrace{E_x\left[\left(\bar{f}(x)-\bar{y}(x)\right)^2\right]}_{\text{Bias}^2} + 2\,E_{x,y}\left[\left(\bar{f}(x)-\bar{y}(x)\right)\left(\bar{y}(x)-y\right)\right]
\end{aligned}
$$

where 1^{st} term is **Noise** and 2^{nd} term is **Bias²**

b) Prove $E_{x,y}\left[\left(\bar{f}(x)-\bar{y}(x)\right)\left(\bar{y}(x)-y\right)\right]=0$

Proving above gives the magic of **Bias-Variance Decomposition**,

$$
\underbrace{E_{x,y,D}\left[(f_D(x)-y)^2\right]}_{\text{Expected Test Error}} = \underbrace{E_{x,D}\left[\left(f_D(x)-\bar{f}(x)\right)^2\right]}_{\text{Variance}} + \underbrace{E_{x,y}\left[(\bar{y}(x)-y)^2\right]}_{\text{Noise}} + \underbrace{E_x\left[\left(\bar{f}(x)-\bar{y}(x)\right)^2\right]}_{\text{Bias}^2}
$$

Your answer. a From the definition of Expected Classifier $\bar{f}$ (It is an expected value of all the classifiers trained on all Ds drawn from P^n), we know the following

$$\bar{f} = E_D[f_D] \tag{5}$$

Now consider the equation, $E_{x,y,D}[(f_D(x)-\bar{f}(x))(\bar{f}-y)]$

$$
\begin{aligned}
&E_{x,y,D}[(f_D(x)-\bar{f}(x))(\bar{f}-y)] \\
&= E_{x,y}[E_D[(f_D(x)-\bar{f}(x))(\bar{f}-y)]] \\
&= E_{x,y}[(E_D[f_D(x)]-\bar{f}(x))(\bar{f}-y)] \\
&\qquad \textit{From equation } 5 \\
&= E_{x,y}[\bar{f}(x)-\bar{f}(x))(\bar{f}-y)] \\
&= 0
\end{aligned}
$$

Your answer. b From the definition of Expected label of x (It is an expected value of all the labels for a given x over the population P), we know the following

$$\bar{y}(x) = E_P[y] \tag{6}$$

Now consider the term $\bar{y}(x) - y$ from original equation

$$\begin{gathered}
E_{x,y}[(\bar{f}(x) - \bar{y}(x))(\bar{y}(x) - y)] \\
= E_x[(\bar{f}(x) - \bar{y}(x))E_y[\bar{y}(x) - y]] \\
= E_x[(\bar{f}(x) - \bar{y}(x))(\bar{y}(x) - E_y[y])] \\
From\ equation\ 6 \\
= E_x[(\bar{f}(x) - \bar{y}(x))(\bar{y}(x) - \bar{y}(x))] \\
= E_x[(\bar{f}(x) - \bar{y}(x))(0)] \\
= 0
\end{gathered}$$

Problem 5. Neural network.

5.1 (**6 points**) Consider a neural neural network with input $x \in R^{d_i,1}$ and the number of classes being d_o. The network can be given by $\hat{y} = \text{softmax}(W_2(ReLU(W_1x + b_1)) + b_2)$ where $W_1 \in R^{d_1,d_i}$, $b_1 \in R^{d_1,1}$, $W_2 \in R^{d_o,d_1}$, $b_2 \in R^{d_o,1}$ and Loss $L = \text{cross\_entropy}(y, \hat{y})$

Cross entropy loss between $y, \hat{y}$ is given by $-\sum_i^m y_i \log(\hat{y_i})$ and softmax clamps x to [0,1] range using $\frac{e^{x_i}}{\sum_i e^{x_i}}$.

Derive expressions for partial derivatives (be precise and clear) $\frac{\partial L}{\partial W_2}, \frac{\partial L}{\partial b_2}, \frac{\partial L}{\partial W_1}, \frac{\partial L}{\partial b_1}$

Your answer. Forward pass:

$$z_1 = W_1x + b1 \tag{7}$$
$$x_2 = ReLU(z_1) \tag{8}$$
$$z_2 = w_2x_2 + b2 \tag{9}$$
$$\hat{y} = softmax(z_2) \tag{10}$$
$$L = -\sum y_i \log \hat{y}_i \tag{11}$$

From equations 7, 8, 9 we get following derivatives

$$\frac{\partial z_1}{W_1} = x \tag{12}$$
$$\frac{\partial z_1}{x} = W_1 \tag{13}$$
$$\frac{\partial z_1}{b_1} = 1 \tag{14}$$
$$\frac{\partial x_2}{z_1} = 1 \; if \; z_1 > 0 \tag{15}$$
$$\frac{\partial x_2}{z_1} = 0 \; if \; z_1 \leq 0 \tag{16}$$
$$\frac{\partial z_2}{W_2} = x_2 \tag{17}$$
$$\frac{\partial z_2}{x_2} = W_2 \tag{18}$$
$$\frac{\partial z_2}{b_2} = 1 \tag{19}$$

From equations 10, 11 we can compute following derivatives

$$\frac{\partial L}{\partial \hat{y}_i} = \frac{\partial}{\partial \hat{y}_i}(-\sum y_k \log \hat{y_k})$$
$$For \; i \neq k, \; it \; is \; 0$$
$$\frac{\partial L}{\partial \hat{y}_i} = \frac{-y_i}{\hat{y}_i}$$

i.e.

$$\frac{\partial L}{\partial \hat{y}} = \frac{-y}{\hat{y}} \tag{20}$$

Consider softmax function

$$\hat{y} = softmax(z_2)$$
$$for\ a\ single\ entry$$
$$\hat{y_j} = \frac{e^{z_{2j}}}{\sum_k e^{z_{2k}}}$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \frac{\partial \frac{e^{z_{2j}}}{\sum_k e^{z_{2k}}}}{\partial z_{2i}}$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \frac{\sum_k e^{z_{2k}} \frac{\partial e^{z_{2j}}}{\partial z_{2i}} - e^{z_{2j}} \frac{\partial \sum_k e^{z_{2k}}}{\partial z_{2i}}}{(\sum_k e^{z_{2k}})^2}$$
$$For\ i \neq j$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \frac{\sum_k e^{z_{2k}} * 0 - e^{z_{2j}} e^{z_{2i}}}{(\sum_k e^{z_{2k}})^2}$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \frac{-e^{z_{2j}} e^{z_{2i}}}{(\sum_k e^{z_{2k}})(\sum_k e^{z_{2k}})}$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = -\hat{y_j}\hat{y_i}$$
$$And\ for\ i = j$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \frac{\sum_k e^{z_{2k}} e^{z_{2j}} - e^{z_{2j}} e^{z_{2j}}}{(\sum_k e^{z_{2k}})^2}$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \frac{e^{z_{2j}} (\sum_k e^{z_{2k}} - e^{z_{2j}})}{(\sum_k e^{z_{2k}})^2}$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \frac{e^{z_{2j}} (\sum_k e^{z_{2k}} - e^{z_{2j}})}{(\sum_k e^{z_{2k}})(\sum_k e^{z_{2k}})}$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \hat{y_j}(1 - \hat{y_j})$$

The derivative for softmax function is given as,

$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = \hat{y_j}(1 - \hat{y_j})\ for\ i == j \tag{21}$$
$$\frac{\partial \hat{y_j}}{\partial z_{2i}} = -\hat{y_j}\hat{y_i}\ for\ i \neq j \tag{22}$$

Consider $\frac{\partial L}{\partial z_2}$

$$\frac{\partial L}{\partial z_2 i} = \frac{\partial}{\partial z_2 i}(-\sum y_k \log \hat{y_k})$$
$$\frac{\partial L}{\partial z_2 i} = -\sum_k^m y_k \frac{\partial \log \hat{y_k}}{\partial z_2 i}$$
$$\frac{\partial L}{\partial z_2 i} = -\sum_k^m y_k \frac{1}{\hat{y_k}} \frac{\partial \hat{y_k}}{\partial z_2 i}$$

From 21 *and* 22

$$\frac{\partial L}{\partial z_2 i} = -y_i \frac{1}{\hat{y_k}} \hat{y_k}(1 - \hat{y_i}) - \sum_{k \neq i}^m y_k \frac{1}{\hat{y_k}} - \hat{y_k}\hat{y_i}$$
$$\frac{\partial L}{\partial z_2 i} = -y_i(1 - \hat{y_i}) + \sum_{k \neq i}^m y_k \hat{y_i}$$
$$\frac{\partial L}{\partial z_2 i} = -y_i + y_i\hat{y_i} + \sum_{k \neq i}^m y_k \hat{y_i}$$
$$\frac{\partial L}{\partial z_2 i} = -y_i + \hat{y_i}(y_i + \sum_{k \neq i}^m y_k)$$
$$\frac{\partial L}{\partial z_2 i} = -y_i + \hat{y_i}(\sum_k^m y_k)$$
$$\frac{\partial L}{\partial z_2 i} = -y_i + \hat{y_i}(1)$$
$$\frac{\partial L}{\partial z_2 i} = \hat{y_i} - y_i$$

i.e.

$$\frac{\partial L}{\partial z_2} = \hat{y} - y \tag{23}$$

1. $\frac{\partial L}{\partial W_2}$

By chain rule

$$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial z_2} \frac{\partial z_2}{\partial w_2}$$

From equations 23 *and* 17

$$\frac{\partial L}{\partial W_2} = (\hat{y} - y)x_2$$

2. $\frac{\partial L}{\partial b_2}$

$$\begin{gathered}
By\ chain\ rule \\
\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial b_2} \\
From\ equations\ 23\ and\ 19 \\
\frac{\partial L}{\partial b_2} = (\hat{y} - y)
\end{gathered}$$

3. $\frac{\partial L}{\partial W_1}$

$$\begin{gathered}
By\ chain\ rule \\
\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial x_2}\frac{\partial x_2}{\partial z_1}\frac{\partial z_1}{\partial W_1} \\
From\ equations\ 23, 18, 15, 16 and 12 \\
\frac{\partial L}{\partial W_1} = (\hat{y} - y)W_2(1)x\ if\ z_1 > 0 \\
\frac{\partial L}{\partial W_1} = (\hat{y} - y)W_2 x\ if\ z_1 > 0 \\
\frac{\partial L}{\partial W_1} = 0\ if\ z_1 \leq 0
\end{gathered}$$

4. $\frac{\partial L}{\partial b_1}$

$$\begin{gathered}
By\ chain\ rule \\
\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial z_2}\frac{\partial z_2}{\partial x_2}\frac{\partial x_2}{\partial z_1}\frac{\partial z_1}{\partial b_1} \\
From\ equations\ 23, 18, 15, 16 and 14 \\
\frac{\partial L}{\partial b_1} = (\hat{y} - y)W_2(1)(1)\ if\ z_1 > 0 \\
\frac{\partial L}{\partial b_1} = (\hat{y} - y)W_2\ if\ z_1 > 0 \\
\frac{\partial L}{\partial b_1} = 0\ if\ z_1 \leq 0
\end{gathered}$$

5.2 (**6 points**) Consider a convolutional neural network architecture given in figure 4 for MNIST to classify digits (10 digits). Given that the input images are reduced to size 10×10 with only 1 channel (represented as a matrix in $\mathbb{R}^{10\times 10}$). The covolutional layer ues a 3×3 filter, $\mathbf{W}_{conv}$, with stride 2 and use zero padding to make it fit. The dimensions of the outputs of each layer are shown.

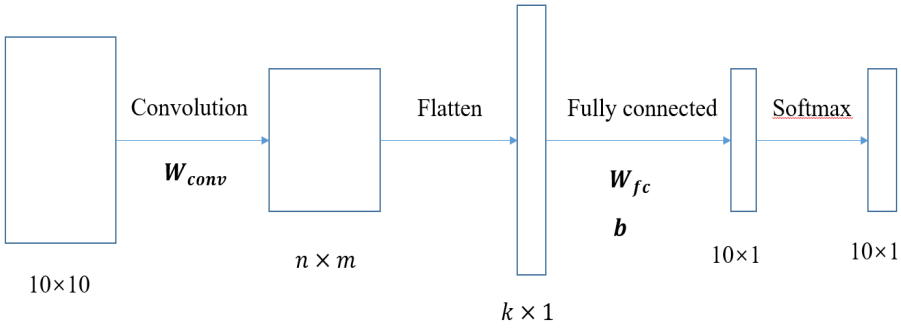


Figure 4: A toy CNN

(a) What is n, m, k in the graph?
(b) What is the size of $\mathbf{W}_{conv}$, $\mathbf{W}_{fc}$, and $\mathbf{b}$?

Your answer. a

$$n = 4$$
$$m = 4$$
$$k = 16$$

Your answer. b

$$W_{conv} = 3 \times 3 \times 1 \times 1$$
$$W_{fc} = 10 \times 16$$
$$b = 10 \times 1$$

Programming Assignment

Instruction. For each problem, you are required to report descriptions and results in the PDF and submit code as python file (.py) (as per the question).

- **Python** version: Python 3.
- Please follow PEP 8 style of writing your Python code for better readability in case you are wondering how to name functions & variables, put comments and indent your code
- **Packages allowed**: pytorch, numpy, pandas, matplotlib
- **Submission**: Submit report, description and explanation of results in the main PDF and code in .py files.
- Please PROPERLY COMMENT your code in order to have utmost readability
- Please provide the required functions for each problem.
- There shall be NO runtime errors involved.
- There would be PENALTY if any of the above is not followed

Programming Common

This programming assignment focuses on implementing neural network for handwritten digits. This problem is about multi class classification and you will use MNIST dataset. You already have an idea of the dataset by now.

You will use pytorch to implement neural network. The implementation has to be for the CPU version only - No GPU's or MPI parallel programming is required for this assignment. Follow the installation instructions at https://pytorch.org in case you want to use your local machine, we recommend using conda environment.

Here is the pytorch tutorial. If you are new to pytorch, we recommend you to go through the tutorial here. https://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html

Problem 6. Neural Network Implementation.

(**27 Points**)

1. (**No Points**) You can directly download MNIST in pytorch using torchvision.datasets.MNIST. It will allow you to

   ```
   trainset = torchvision.datasets.MNIST(root='./data', train=True, download=True,
   transform=transform)
   testset = torchvision.datasets.MNIST(root='./data', train=False, download=True,
   transform=transform)
   ```

2. (**9 Points**) First, implement a multi-layer fully connected network:
   - Input: 1-channel input, size 28x28
   - Keep batch size as 32.
   - Fully connected layer 1: Input with bias; output - 128

- ReLu Layer
- Fully connected layer 2: input - 128; output - 10
- Softmax layer
- Use cross entropy as loss function
- Use SGD as optimizer.

Train using mini-batches of the given batch size. Plot loss and training accuracy for every epoch. At the end of the training, save your model with the name "mnist-fc". Load the saved model and report testing accuracy on the reloaded model. We should be able to reload your trained model and test.

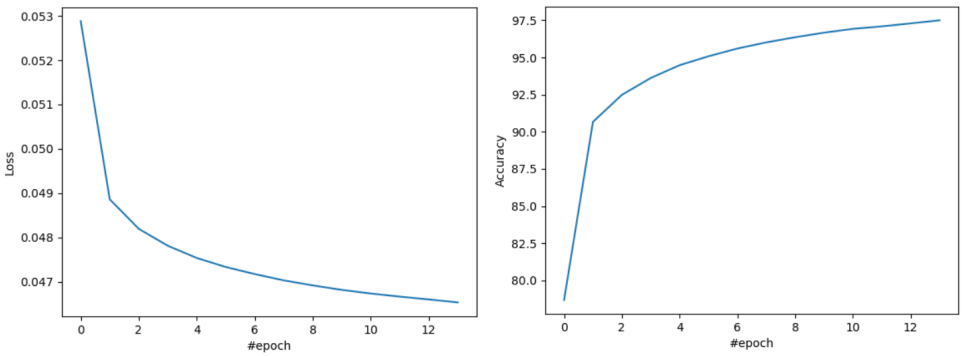
epoch: An epoch tells you the number of times the learning algorithm sees the whole training data set. When it has seen all the training samples, you say 1 epoch is over and you start iterating in the next epoch.

Your answer.

a) Learning rate - 0.1

b) Termination condition $\Delta Loss \leq 0.00005$

c) Testing accuracy - 96.96%



(a) Loss vs epoch (b) Accuracy vs epoch

3. (**10 Points**) Implement a convolutional neural network with the following specifications.
   - Input: 1-channel input, size 28x28
   - Keep batch size as 32.
   - Convolution layer: Convolution kernel size is (3, 3) with stride as 1. Input channels - 1; Output channels - 20
   - Max-pool: 2x2 max pool
   - ReLu Layer
   - Flatten input for feed to fully connected layers
   - Fully connected layer 1: flattened input with bias; output - 128
   - ReLu Layer
   - Fully connected layer 2: input - 128; output - 10

- Softmax layer as above
- Use cross entropy as loss function
- Use SGD as optimizer.

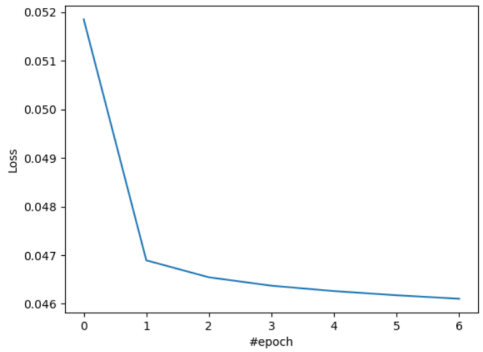
Train using mini-batches of the given batch size. Plot loss and training accuracy for every epoch. At the end of the training, save your model with the name "mnist-cnn". Load the saved model and report testing accuracy on the reloaded model. We should be able to reload your trained model and test.

Your answer.

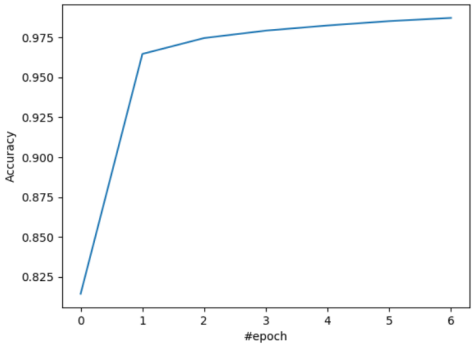
a) Learning rate - 0.1

b) Termination condition $\Delta Loss \leq 0.00005$

c) Testing accuracy - 98.39%



(a) Loss vs epoch



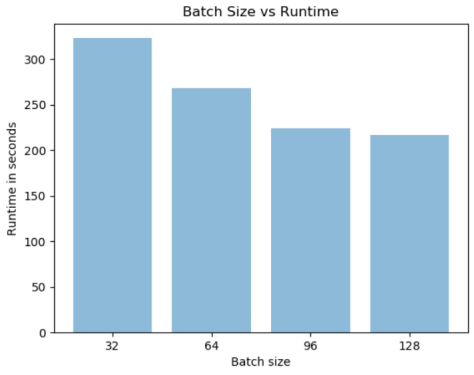
(b) Accuracy vs epoch

4. (**4 Points**) For the convolutional network implemented above, vary the batch sizes as [32, 64, 96, 128] and plot the convergence run time vs batch sizes.

Your answer.

a) Learning rate - 0.1

b) Termination condition $\Delta Loss \leq 0.00005$



(a) Runtime in minutes vs Batch size

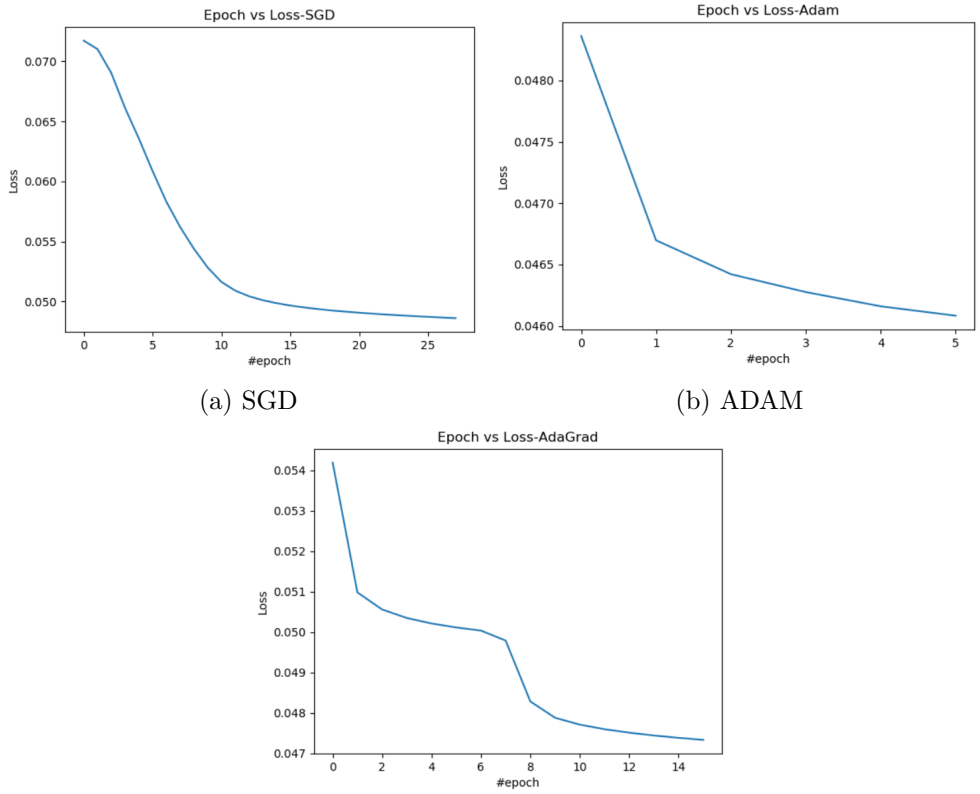
The performance difference is not very impacting.

5. (**4 Points**) "torch optim" provides many optimizers. Try the following optimizers in your last implementation - "SGD", "ADAM", "ADAGRAD". (SGD is already covered in the class, for ADAM and ADAGRAD refer to https://arxiv.org/abs/1412.6980and http://www.jmlr.org/papers/volume12/duchi11a/duchi11a.pdf respectively). Plot loss vs epochs for each optimizer. Briefly describe.

Your answer.

a) Learning rate - 0.001

b) Termination condition $\Delta Loss \leq 0.00005$

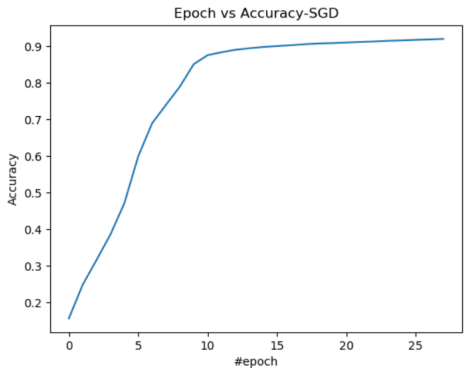


(a) SGD

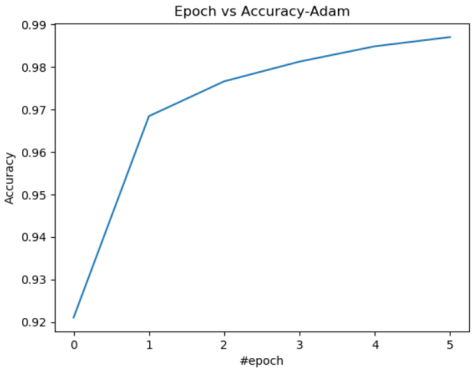
(b) ADAM

(c) ADAGRAD

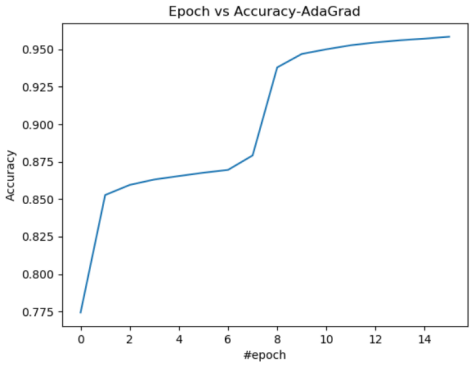
Figure 8: Loss vs Epoch



(a) SGD



(b) ADAM



(c) ADAGRAD

Figure 9: Accuracy vs Epoch

Few notable points:

a) Convergence for ADAM was fastest (5 iterations), and slowest for SGD (25 iterations)

b) Accuracy for ADAM was best and worst for SGD

c) ADAGRAD had stuck in local optimal for few iterations