COL 341 Major

Chinmay Mittal

TOTAL POINTS

67 / 70

QUESTION 1

1 Conv Layer 8 / 8

✓ **+ 2 pts** *part a correct*

+ 1 pts part a : answer is correct but explanation is wrong

+ 0 pts part a: not attempted/ incorrect

✓ **+ 2 pts** *part b correct*

+ 1 pts part b : answer is correct but explanation is wrong

+ 0 pts part b: not attempted/ incorrect

✓ **+ 2 pts** *part c correct*

+ 1 pts part c : answer is correct but explanation is wrong

+ 0 pts part c: not attempted/ incorrect

✓ **+ 2 pts** *part d correct*

+ 1 pts part d : answer is correct but explanation is wrong

+ 0 pts part d: not attempted/ incorrect

QUESTION 2

2 Neural Net Training 6 / 6

✓ **+ 6 pts** *Identified the bias issue (under-fitting) and proposed correct solution*

+ 4 pts Bugs/ Incomplete explanation

+ 2 pts Some reasonable answer/ Incomplete

+ 0 pts Incorrect/ Not attempted

QUESTION 3

3 Benefits of Conv Layer 4 / 4

+ 2 pts 1 Correct

✓ **+ 4 pts** *2 Correct*

+ 0 pts None correct

QUESTION 4

4 ReLU 4 / 4

✓ **+ 4 pts** *Correct*

+ 0 pts Incorrect/Not attempted

+ 2 pts Partial Explanation

QUESTION 5

5 Non-Linearity 4 / 4

Click here to replace this description.

✓ **- 0 pts** *Correct*

- 4 pts Incorrect/Unattempted

- 2 pts The resultant of linearly activated nn being equivalent to a single linear layer not mentioned.

QUESTION 6

6 Softmax 8 / 8

+ 0 pts Incorrect or not attempted

✓ **+ 8 pts** *Fully correct*

- 2 pts Missing steps or Mistakes

+ 4 pts One of the derivative is correct

- 0.5 pts One of the steps isn't clear

QUESTION 7

7 MLE 8 / 8

✓ **+ 8 pts** *Correct*

- 2 pts likelihood is not written/ is incorrect

- 2 pts log likelihood is not written/ is incorrect

- 2 pts Case when X_1<= mu <= X_2 is not handled

- 1 pts Case when mu< min(X_1, X_2) is not handled

- 1 pts Case when mu> max(X_1, X_2) is not handled

+ 0 pts not attempted/incorrect

+ 1.5 pts partially correct

- 1 pts minor mistakes/ step missing

QUESTION 8

8 PCA 6 / 8

+ 8 pts All Correct

✓ **+ 6 pts** *Any three are correct*

+ 4 pts Any two are correct

+ 2 pts Any one is correct

+ 0 pts All incorrect/ Not attempted

QUESTION 9

9 AdaBoost 8 / 8

✓ **+ 2 pts** *getting that each features has 2k parameter*

+ 0 pts Incorrect

✓ **+ 2 pts** *Prior term*

✓ **+ 2 pts** *multiplying t with parameters in each iteration*

✓ **+ 2 pts** *adding parameter for alpha*

QUESTION 10

10 SVM 4 / 4

✓ **+ 4 pts** *Correct*

+ 2 pts partially correct

+ 0 pts Incorrect/not_attempted

QUESTION 11

11 Linear Classifier 4 / 4

Click here to replace this description.

✓ **- 0 pts** *Correct*

- 4 pts Wrong/Unattempted

QUESTION 12

12 K-Means 3 / 4

✓ **+ 2 pts** *Correct explanation of why revisit won't occur*

✓ **- 1 pts** *error decreases or remains the same at each iteration isn't proved OR is proved incorrectly*

✓ **+ 2 pts** *Correct proof for convergence*

- 0.5 pts Why finite number of configurations isn't mentioned or incorrectly calculated- striling number S(n, k)

+ 0 pts Incorrect totally/ not attempted

- 0.5 pts Definition of variables used is not present

- 0.5 pts Minor mistakes

Student Name: CHINMAY MITTAL	Entry Number: 2020CS10336

Department of Computer Science and Engineering
Indian Institute of Technology Delhi

COL341: Fundamentals of Machine Learning

Major Exam

Time: 120 minutes | Maximum Marks: 70 | Number of Questions: 12

Instructions:

1. This is a closed book examination, you must not have any study material with you while taking exam. All electronic devices must be stored away from you during the exam.
2. Please attempt all questions.
3. If you feel any question/statement is ambiguous, please write your assumptions clearly, and then answer as per your assumptions.
4. You must answer the question in the box assigned for each question. Any text outside the given box will not be evaluated. You are provided with a separate rough sheet for the scribbles, if you need to.
5. Please read the honour code given below carefully, and sign in the designated area to accept the same. Your answer sheet will not be evaluated in the absence of such signature.

Honour Code: As a student of IIT Delhi, I will not give or receive aid in examinations. I will do my share and take an active part in seeing to it that others as well as myself uphold the spirit and letter of the Honour Code.

Mittal
Your Signature

Question	1	2	3	4	5	6	7	8	9	10	11	12	Total
Max Marks	8	6	4	4	4	8	8	8	8	4	4	4	70
Earned Marks													

1. [8 marks] Which of the following propositions are true about a CONV layer? Give justification for each true/false (no marks without justification):
   1. The number of weights depends on the depth of the input volume. ✓
   2. The number of biases is equal to the number of filters.
   3. The total number of weights depends on the stride.
   4. The total number of weights depends on the padding.

1. TRUE. The # of weights depends on the input volume. if input is $H * W * C_{in}$. Then any kernel will have shape $K_x * K_y * C_{in}$. Hence the weights in the kernel and hence the layers are a function of the input depth.

2. TRUE. The number of biases is equal to the number of filters. There is one bias per output feature map / filter. Because of weight sharing every output neuron in a feature map is computed by adding the same bias. Hence 1 bias / filter or output feature map.

3. FALSE. The weights of one feature map are $K * K * C_{in} + 1$ which depend on the size of the kernel and the # of input channels and not the stride, stride determines the size of the output volume.

4. FALSE. The weights of one feature map are $K * K * C_{in} + 1$ which depend on the size of the kernel and not the padding. The padding only determines the size of the output volume.

2. [6 marks] You want to solve a classification task. You first train your neural network model on 20 samples. Training converges, but the training loss is very high. You then decide to train this network on 10,000 examples. Is your approach to fixing the problem correct? If yes, explain the most likely results of training with 10,000 examples. If not, give a solution to this problem.

If the neural network has a high but stable training loss on few images. ⇒ It is not able to fit this small data it is unlikely that it will be able to fit a larger dataset. The problem lies in the network and not the dataset. Possible fixes.
(i) Network Architecture, current network architecture might not be able to model the problem at hand. (eg Q4). Changing the neural network architecture to atleast ensuring that the training data can be fit.
(ii) Weight initialization. Some particular weight initializations can restrict the kind of functions learnt. eg Symmetric initial weights remain symmetric during training. There is a need to break the symmetry. Changing the initialization strategy can help.

3. [4 marks] Give two benefits of using convolutional layers instead of fully connected ones for visual classification task.

Shared weights and local connectivity of convolutional layers leads to the following parameters.
(i) Fewer network parameters, since the weights to each output neuron in a feature map are tied. This leads to fewer parameters. Making the optimization easier / reduces the memory requirements of the network
(ii) Local connectivity Allows for translational invariance. Since one kernel with the same weights is slided across the entire image. The feature can be present anywhere in the image and will be detected. (unlike fully connected layers).

4. [4 marks] You are solving the binary classification task of classifying images as cat vs. non-cat. You design a CNN with a single output neuron. Let the output of this neuron be z. The final output of your network, $\widehat{y}$ is given by: $\widehat{y} = \sigma(\text{ReLU}(z))$. You classify all inputs with a final value $\widehat{y} \geq 0.5$. What problem are you going to encounter?

sigmoid

$\hat{y} = \sigma(\text{RELU}(x))$

$\text{RELU}(x) \geq 0 \quad \forall x \in \mathbb{R}$

$\sigma(x) \geq 0.5 \quad \forall x \in \mathbb{R}, x \geq 0$

$\Rightarrow \hat{y} \geq 0.5 \quad \forall x \in \mathbb{R}.$

Hence the network will predict all inputs as cat. And such a neural network cannot model the problem at hand.

5. [4 marks] Why is it important to place non-linearities between the layers of neural networks?

Consider a network with no non-linearities. This network is hence a series of linear transformations to the input.

$y = B f_n \cdot f_{n-1} \cdots f_1(x)$. Any composition of linear transform remains a linear transformation. Hence the neural network can only learn linear functions of the input. Allowing for non-linearities lets the network model more complicated functions, with an increased hypothesis set which can lead to better models with lower E_{in} and E_{out}. One can show that with appropriate activation functions and network architecture, a neural network can model any continuous function.

6. [8 marks] Recall that softmax function is defined as $S_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$, where z is the logit vector. Derive the expression for $\frac{\partial S_i}{\partial z_i}$ and $\frac{\partial S_i}{\partial z_j}$

Define $\Sigma = \sum_j \exp(z_j) \Rightarrow S_i = \frac{\exp(z_i)}{\Sigma}$

if $z = \frac{f}{g} \Rightarrow z' = \frac{f'g - g'f}{g^2}$

$$\Rightarrow \frac{\partial S_i}{\partial z_j} = \frac{\frac{\partial \exp(z_i)}{\partial z_j}\Sigma - \frac{\partial \Sigma}{\partial z_j}\exp(z_i)}{\Sigma^2}$$

if $i = j \Rightarrow \frac{\partial \exp(z_i)}{\partial z_j} = \frac{\partial \exp(z_i)}{\partial z_i} = \exp(z_i)$

$$\frac{\partial \Sigma}{\partial z_j} = \frac{\partial\left(\sum_j \exp(z_j)\right)}{\partial z_i} = \frac{\partial \exp(z_i)}{\partial z_i} = \exp(z_i)$$

$$\Rightarrow \frac{\partial S_i}{\partial z_i} = \frac{\exp(z_i)\Sigma - \exp(z_i)\exp(z_i)}{\Sigma^2}$$

$$= \frac{\exp(z_i)}{\Sigma} - \frac{\exp(z_i)}{\Sigma}\frac{\exp(z_i)}{\Sigma}$$

$$= S_i - S_i^2$$

if $i \neq j$

$$\Rightarrow \frac{\partial \exp(z_i)}{\partial z_j} = 0$$

$$\frac{\partial \Sigma}{\partial z_j} = \frac{\partial\left(\sum_j \exp(z_j)\right)}{\partial z_j} = \frac{\partial \exp(z_j)}{\partial z_j} = \exp(z_j)$$

$$\Rightarrow \frac{\partial S_i}{\partial z_j} = \frac{0 - \exp(z_j)\exp(z_i)}{\Sigma^2}$$

$$= -\frac{\exp(z_j)}{\Sigma} * \frac{\exp(z_i)}{\Sigma}$$

$$= -S_i S_j$$

7. [8 marks] We are drawing sample points from a distribution with the probability density function $f(x) = \frac{1}{2}\exp(|x-\mu|)$, but we do not know the mean $\mu \in \mathbb{R}$. We decide to estimate

$\frac{1}{2}\exp(-|x-\mu|)$

μ with maximum likelihood estimation (MLE). Unfortunately, we have only two sample points $X_1, X_2 \in \mathbb{R}$. Derive the likelihood and the log-likelihood for this problem. Then show that every value of μ between X_1 and X_2 is a maximum likelihood estimate.

likelihood $= P(D) = \pi P(x_i) \rightarrow$ independence $= P(x_1)P(x_2) = \frac{1}{2}\exp(-|x_1-u|)$ $\frac{1}{2}\exp(-|x_2-u|)$

log likelihood $\Rightarrow \log P(D) = \sum \log P(x_i)$

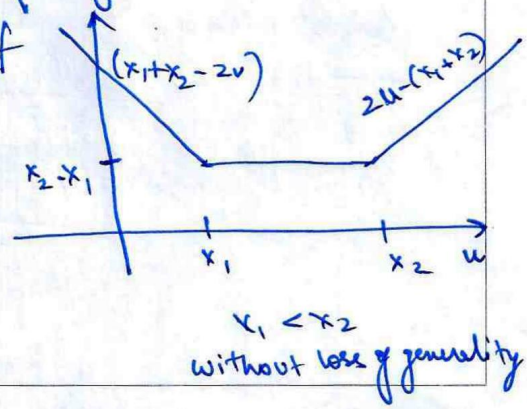
$= \log\left(\frac{1}{2}\exp(-|x_1-u|)\right) + \log\left(\frac{1}{2}\exp(-|x_2-u|)\right)$

$= 2 * \log \frac{1}{2} - \underbrace{(|x_1-u| + |x_2-u|)}_{f}$

plot of this function vs u

To maximize the log likelihood we need to minimize f

and value of u where x_1 and x_2 minimizes f from the graph.



$x_1 < x_2$ without loss of generality

8. [8 marks] Which of the following are true about principal components analysis (PCA)? Assume that no two eigenvectors of the sample covariance matrix have the same eigenvalue. Give justification for each true/false (no marks without justification)

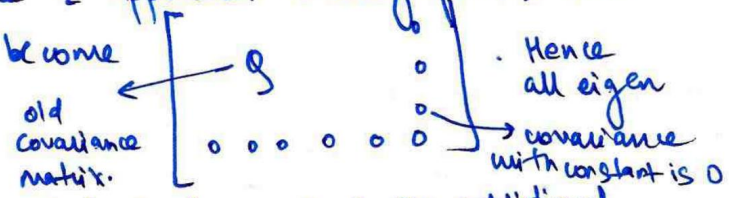
   1. Appending a 1 to the end of every sample point doesnt change the results of performing PCA (except that the useful principal component vectors have an extra 0 at the end, and theres one extra useless component with eigenvalue zero).

   2. If you use PCA to project d-dimensional points down to j principal coordinates, and then you run PCA again to project those j-dimensional coordinates down to k principal coordinates, with $d > j > k$, you always get the same result as if you had just used PCA to project the d-dimensional points directly down to k principle coordinates.

   3. If you perform an arbitrary rigid rotation of the sample points as a group in feature space before performing PCA, the principal component directions do not change.

   4. If you perform an arbitrary rigid rotation of the sample points as a group in feature space before performing PCA, the largest eigenvalue of the sample covariance matrix does not change.

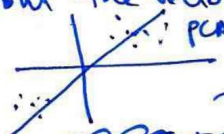
1. With a new feature 1 appended to every point, new covariance matrix will become



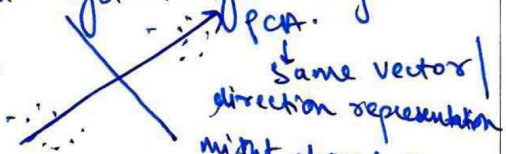
vectors have a zero appended in them and an additional eigen vectors with zero eigen value.
Hence results of performing do not change
TRUE.

2. The k principal components lie in the subspace of the j principal component. (and they remain the top components in that space.) Hence the results remain the same TRUE.

3. TRUE → the directions of maximum variance depend on the datapoints and their interspacing. This does not change when we rotate all the points. (The values representing the eigen vectors in the new co-ordinate system might change but the vector itself remains same).



TRUE



4. → the direction of the eigen vector remains the same and the matrix is also rotated by the same amount

$Ax = \lambda Ax$

rotation matrix → remains same.

9. [8 marks] Consider running AdaBoost with Multinomial Naive Bayes as the weak learner for two classes and k binary features. After t iterations, of AdaBoost, how many parameters

do you need to remember? In other words, how many numbers do you need to keep around to predict the label of a new example? Assume that the weak-learner training error is non-zero at iteration t. Dont forget to mention where the parameters come from

Consider one weak learner, the parameters needed are $P(Y)$ for $P(Y)$, $P(X_1|Y), P(X_2|Y) \dots P(X_K|Y)$ and $P(Y)$
$\Rightarrow$ the # of parameters are $1 + 2K$ (2 parameters each for $P(X_i|Y)$ for $Y=+1$ and $Y=-1$)
$\Rightarrow$ in t iterations we learn t seperate. weak learners
$\Rightarrow$ total # of parameters are $t(2K+1)$
Also we need to remember the weights of each weak learner ($\alpha_m = \ln\left(\frac{1-\epsilon}{\epsilon}\right)$) there will be t such weights $\Rightarrow$ total numbers to remember are $t + t(2K+1) = t(2K+2)$

10. [4 marks] For linearly separable data, can a small slack penalty (C) hurt the training accuracy when using a linear SVM (no kernel)? If so, explain how. If not, why not?

$\min \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i$ subject to $y_n(w^T x_n + b) \geq 1 - \xi_n$
for linearly seperable data $\xi_n \geq 0$
with no C (hard margin) $\Rightarrow$ training accuracy $= 100\%$
Small C allows to keep high values of ξ
if any $\xi_n > 1$ then some point might be misclassified.
it can so happen that w can be changed by further decreasing $\|w\|$ if some point is allowed to be misclassified
$\Rightarrow$ small C might lead to a decrease in training accuracy

11. [4 marks] Given n linearly independent feature vectors in n dimensions, show that for any assignment to the binary labels you can always construct a linear classifier with weight

vector w which separates the points. Assume that the classifier has the form $\text{sign}(w \cdot x)$. Note that a square matrix composed of linearly independent rows is invertible.

Assume any arbitrary assignment to the points $x_1\ x_2 \ldots x_n$ be $t_1\ t_2 \ldots t_n$ where $t_i \in \{-1, 1\}$.

We want $w \in \mathbb{R}^n$ such that $\text{sign}(x_i \cdot w_i) = \text{sign}(t_i)$

Consider the following equation

- $Xw = T$ — (i)

$X = \begin{bmatrix} — x_1^T — \\ — x_2^T — \\ \vdots \\ — x_n^T — \end{bmatrix}$

if any w satisfies (i)

Then $\text{sign}(x_i^T w) = \text{sign}(t_i)$ as required

since X has linearly independent feature vector rows

X is invertible $\rightarrow w = X^{-1}T$

for any T a. w exists which predicts the correct sign and hence seperates the points.

12. [4 marks] Let a configuration of the k means algorithm correspond to the k way partition (on the set of instances to be clustered) generated by the clustering at the end of each iteration. Is it possible for the k-means algorithm to revisit a configuration? Justify how your answer proves that the k-means algorithm converges in a finite number of steps.

Consider any step in the k-means algorithm.

(i) Re assignment step. Every point is assigned to a new cluster which is closer than the previous assignment same clusters.
or at the same distance.

$J_{old} = \sum_{i=1}^{N} \| x^i - z^i \|_2^2$ $\quad J_{new} = \sum_{i=1}^{N} \| x^i - \tilde{z}^i \|_2^2$

z^i → cluster to which z^i belongs.

Since clusters assignment ensures $\| x^i - \tilde{z}^i \| \leq \| x^i - z^i \|$

J decreases

(equality holds only when clusters assignment does not change for all points) → termination.

(ii) Similarly recomputing the cluster means only decreases the loss function. (Unless the cluster's don't change at termination)

$$\sum_{\text{cluster } C_i} \sum_{x \in C_i} \|x - \text{old cluster centroid}\|_2^2$$

The new $\hat{z}$ ensures that

$$\sum_{x \in C_i} \|x - \hat{z}\|_2^2 \geq \sum_{x \in C_i} \|x - \hat{z}\|_2^2$$

because the mean ($\hat{z}$) minimizes this function.

Hence every step of K-means can only decrease the cost (J).

if K means revisits a configuration $\Rightarrow$ cost is same (Assignment uniquely determines the cost) but this is not possible. Since we have shown that cost keeps decreasing. $\Rightarrow$ no assignment is repeated.

Since there is an upper bound on the number of assignments (from combinatorially solving for ways to divide n points in K groups) $\rightarrow T$

K-means has to converge in a finite # of steps if T is the # of assignments possible.

if K-means takes more than T steps $\Rightarrow$ some assignment has been revisited which is not possible.

$\Rightarrow$ K means will terminate in $\leq T$ steps.