Final Exam - Moed B

Total time for the exam: 3 hours!

Please copy the following sentence and sign it: " I am respecting the rules of the exam: Signature:_____ "

Important: For **True / False** questions, copy the statement to your notebook and write clearly true or false. You should prove the statement if true, or disprove it, e.g. by providing a counter-example, otherwise.

1) **Imbalanced Data and Backpropagation (32 Points):** You're trying to classify RGB images if giraffe present (1) and giraffe absent (0) using a deep neural network. Unfortunately, your data set is imbalanced, and consist of:
   - 2000 images with a giraffe
   - 200 images with no giraffe

   a) (4 points) To address the imbalance problem, we would like to oversample our data by using augmentations, while avoiding having the same example twice in our dataset. Suggest two data augmentation techniques you could use to help address the class imbalance problem.
   Solution:
   Adding white noise, image reflection, cropping, rotation.

   b) (4 points) Instead of data augmentation, you want to experiment with other techniques. Here's the architecture of your network:

$$\begin{aligned} z_1 &= W_1 x^{(i)} + b_1, \\ a_1 &= ReLU(z_1), \\ z_2 &= W_2 a_1 + b_2, \\ \hat{y}^{(i)} &= \sigma(z_2), \\ L^{(i)} &= \alpha \cdot y^{(i)} \cdot \log(\hat{y}^{(i)}) + \beta \cdot (1 - y^{(i)}) \cdot \log(1 - \hat{y}^{(i)}), \\ J &= -\frac{1}{m} \sum_{i=1}^{m} L^{(i)}, \end{aligned}$$

   where $\hat{y}^{(i)} \in \mathbb{R}$, $y^{(i)} \in \mathbb{R}$, $x^{(i)} \in \mathbb{R}^{D_x \times 1}$, $W_1 \in \mathbb{R}^{D_{a1} \times D_x}$, $W_2 \in \mathbb{R}^{1 \times D_{a1}}$. Note that m is the size of the dataset and that the RGB images are flattened into vectors of length D_x before being fed into the network. What are the dimensions of b_1 and b_2?
   Solution:
   $b_1 \in \mathbb{R}^{D_{a1} \times 1}, b_2 \in \mathbb{R}^{1 \times 1}$.

   c) (4 points) Explain why α and β are useful for the imbalance data problem?
   Solution:
   Weighting how much each class contributes to the loss function can help gradient descent because the network will take larger steps when learning from instances of the underrepresented class

   d) (6 points) What are a reasonable values for the pair (α, β)? Provide specific values for these weightings.
   Hint: if $\alpha = 1, \beta = 1$ we get the original Binary Cross-Entropy loss function. Think why this function isn't right for the question's scenario.
   Solution:
   $\alpha = 0.1, \beta = 1.0$. Roughly, the ratio should be somewhere near $\beta = 10 \cdot \alpha$ but not ridiculously large or small.

   e) (4 points) You decide to add $L2$ regularization to this model. Write your new cost function.
   Solution:

$$J = -\sum_i \left(\beta \cdot (1 - y^{(i)}) \cdot log(1 - \hat{y}^{(i)}) + \alpha \cdot y^{(i)} \cdot log(\hat{y}^{(i)}) \right) + ||W_2||_2^2 + ||W_1||_2^2 + b_1^2 + b_2^2.$$

   f) (4 points) Using this new cost function, write down the update rule for W_1 as a function of $\frac{\partial J}{\partial W_1}$ and W_1.
   Hint: assume you are using gradient descent. Use η as your learning rate.
   Solution:

$$W_1' = W_1 - \eta \cdot \left(\frac{\partial J}{\partial W_1} + 2 \cdot W_1 \right).$$

   Note: can be with or without factor 2, according to the denominator in the regularization formula.

   g) (6 points) Suppose you use $L1$ regularization instead. How would you expect the weights learned using $L1$ regularization to differ from these learned using $L2$ regularization?
   Solution:
   The weights obtained by $L1$ regularization are expected to be more sparse (more zeros than $L2$).

2) **Probability mass function estimation (34 Points):** In this question, we will develop an algorithm for probability mass function (PMF) estimation based on a given sample set. Let $X \sim P_X$, $Y \sim P_Y$ and denote the joint PMF of (X, Y) by P_{XY}. Further, let U_X be the PMF of the uniform discrete probability measure over the alphabet of X, i.e. $U_X(x) = \frac{1}{\mathcal{X}}$ for any $x \in \mathcal{X}$.

a) (5 points) Prove the following equality:

$$H(X,Y) = H(P_{XY}, U_{XY}) - D_{KL}(P_{XY}||U_{XY}),$$

where $H(P_{XY}, U_{XY})$ denote the cross-entropy between P_{XY} and U_{XY}.

Solution:

Consider the following chain of equalities:

$$\begin{aligned} H(X,Y) &= \mathbb{E}_{P_{XY}}\left[\log \frac{1}{P_{XY}}\right] \\ &= \mathbb{E}_{P_{XY}}\left[\log \frac{U_{XY}}{P_{XY}U_{XY}}\right] \\ &= \mathbb{E}_{P_{XY}}\left[\log \frac{1}{U_{XY}}\right] - \mathbb{E}_{P_{XY}}\left[\log \frac{P_{XY}}{U_{XY}}\right] \\ &= H(P_{XY}, U_{XY}) - D_{KL}(P_{XY}||U_{XY}). \end{aligned}$$

b) (5 points) Express $H(P_{XY}, U_{XY})$ as function of the alphabets $\mathcal{X}$ and $\mathcal{Y}$.

Solution:

$$\begin{aligned} H(P_{XY}, U_{XY}) &= \mathbb{E}_{P_{XY}}\left[\log \frac{1}{U_{XY}}\right] \\ &= \sum_{x,y \in \mathcal{X}\times Y} P_{XY}(x,y) \log \frac{1}{U_{XY}(x,y)} \\ &= \sum_{x,y \in \mathcal{X}\times Y} P_{XY}(x,y) \log |\mathcal{X}||\mathcal{Y}| \\ &= \log |\mathcal{X}||\mathcal{Y}| \sum_{x,y \in \mathcal{X}\times Y} P_{XY}(x,y) \\ &= \log |\mathcal{X}||\mathcal{Y}|. \end{aligned}$$

c) (8 points) Propose a neural network based algorithm to estimate $H(X,Y)$ from a sample set $\{(x_i, y_i)\}_{i=1}^n \sim P_{XY}$. Denote the estimator by $\hat{H}_n(X,Y)$, and provide a block diagram of your proposed algorithm.

Solution:

Following the previous item, we only need to estimate the KL divergence. In class we studied about the Donsker-Vardhan representation. We'll replace the expectations with empirical means. The objective of the KL divergence is of the form:

$$\hat{D}_{KL}^{(n)}(X,Y) = \sup_{\theta_{XY} \in \Theta_{XY}} \frac{1}{n}\sum_{i=1}^n T_{\theta_{XY}}(x_i, y_i) - \log\left(\frac{1}{n}\sum_{i=1}^n e^{T_{\theta_{XY}}(\tilde{x}_i, \tilde{y}_i)}\right),$$

where $(\tilde{x}_i, \tilde{y}_i)$ are drawn uniformly over $(\mathcal{X}, \mathcal{Y})$. After achieving the supermization objective via a neural network $T_{\theta_{XY}}$ with parameters θ_{XY}, we can achieve the entropy,

$$\hat{H}_n(X,Y) = \log |\mathcal{X}||\mathcal{Y}| - \hat{D}_{KL}^{(n)}(X,Y).$$

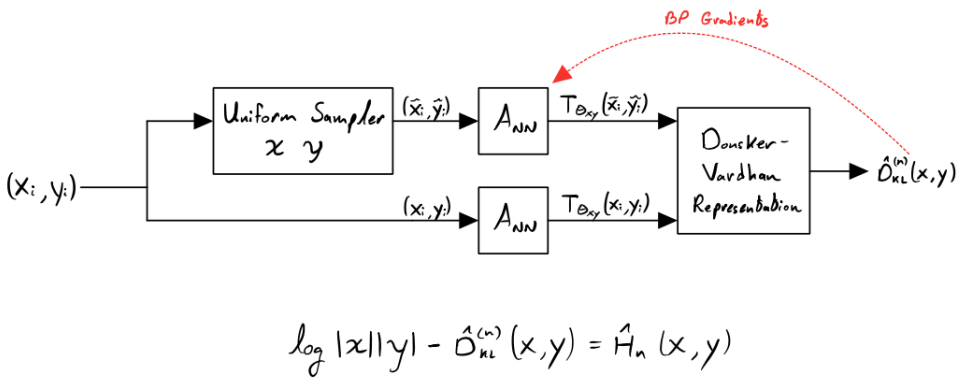

Fig. 1: Donsker-Vardhan Block Diagram.

d) (8 points) For sufficient large n, is your proposed algorithm provides a lower / upper bound on $H(X,Y)$? Theoretically, when will the algorithm achieve equality?
Solution:
The proposed algorithm provides an upper bound of $H(X,Y)$. Specifically, the Donsker-Vardhan representation provides a lower bound on $\hat{D}_{KL}^{(n)}(X,Y)$ and the cross entropy between P_{XY} and U_{XY} is a positive constant. From the lecture, the algorithm will achieve equality when,

$$\begin{aligned} T^*_{\theta_{XY}}(x,y) &= \log\left(\frac{P_{XY}(x,y)}{U_{XY}(x,y)}\right) \\ &= \log\left(|\mathcal{X}||\mathcal{Y}| \cdot P_{XY}(x,y)\right). \end{aligned} \tag{1}$$

e) (8 points) Assume that, for a sufficient large n, your suggested algorithm is converged. Suggest how to estimate P_{XY} based on the previous items.
Solution:
From the last section, after training our neural network $T_{\theta_{XY}}$, we can estimate the probability $P_{XY}(x,y)$ as follows,

$$\hat{P}_{XY}(x,y) = \frac{1}{|\mathcal{X}||\mathcal{Y}|} e^{T^*_{\theta_{XY}}(x,y)},$$

where $T^*_{\theta_{XY}}(x,y)$ is given in Eq. (1).

3) **Huffman codes (34 Points):** Consider a random variable X which takes 6 values $\{A,B,C,D,E,F\}$ with probabilities $(0.5, 0.25, 0.1, 0.05, 0.05, 0.05)$ respectively.

a) (5 points) Construct a binary Huffman code for this random variable. What is the average length of the code?
Solution:
The mapping constructed by Huffman is $A \to 0$, $B \to 10$, $C \to 1100$, $D \to 1101$, $E \to 1110$, and $F \to 1111$.
The average length of this code is

$$\begin{aligned} L_H &= 1 \cdot 0.5 + 2 \cdot 0.25 + 4 \cdot (0.1 + 0.05 + 0.05 + 0.05) \\ &= 2. \end{aligned}$$

The entropy $H(X)$ in this case is 1.98 bits.

b) (5 points) Construct a quaternary Huffman code for this random variable, i.e., a code over the alphabet of four symbols (call them a, b, c, and d). What is the average length of this code?
Solution:
The mapping in this case is $A \to a$, $B \to b$, $C \to c$, $D \to da$, $E \to db$, and $F \to dc$.
The average length of this code is

$$\begin{aligned} L_Q &= 1 \cdot 0.85 + 2 \cdot 0.15 \\ &= 1.15. \end{aligned}$$

c) (4 points) One way to construct a binary code for a random variable is to start with a quaternary code, and convert the symbols into binary using the mapping $a \to 00$, $b \to 01$, $c \to 10$, and $d \to 11$. What is the average length of the binary code for the above random variable constructed by this process?
Solution:
The code constructed by the above process is $A \to 00$, $B \to 01$, $C \to 11$, $D \to 1000$, $E \to 1001$, and $F \to 1010$. The average length of the code is

$$L_{QB} = 2 \cdot 0.85 + 4 \cdot 0.15 = 2.3 \textit{ bits}.$$

For any variable X, let L_H be the average length of the binary Huffman code for the random variable, and let L_{QB} be the average length code constructed by first building a quaternary Huffman code and converting it to binary.

d) (6 points) **True/False:** The inequality $L_H \leq L_{QB}$ always holds.
Solution:
True. Note that the binary code constructed from the quaternary code is also instantaneous. Therefore, its average length cannot be better than the average length of the best instantaneous code, i.e., the Huffman code. Hence, the lower bound holds.

e) (7 points) Show that $L_{QB} < L_H + 2$.
Hint: Consider to use the fact that the average length of a quaternary Huffman code satisfies $L_Q < \frac{H_2(X)}{2} + 1$.
Solution:

It is easy to see that $L_{QB} = 2L_Q$, since each symbol in the quaternary code is converted into two bits. Accordingly,

$$\begin{aligned} L_{QB} &= 2L_Q \\ &\overset{(a)}{<} 2 \cdot \left(\frac{H_2(X)}{2} + 1 \right) \\ &= H_2(X) + 2, \end{aligned}$$

where (a) follows from the hint. Combining this with the fact that $H_2(X) \leq L_H$, we obtain $L_{QB} < L_H + 2$.

f) (7 points) Give an example where the code constructed by converting an optimal quaternary code is also the optimal binary code, i.e., example for which $L_H = L_{QB}$.

Solution:

Consider a random variable that takes on four equiprobable values. Then, the quaternary Huffman code for this is 1 quaternary symbol for each source symbol, with average length 1 quaternary symbol. The average length L_{QB} for this code is 2 bits. The Huffman code for this case is also eaisly seen to assign 2 bit codewords to each symbol, and therefore for this case, $L_H = L_{QB}$.

Good Luck!