AI2100/AI5100 Deep Learning, Spring 2022

Indian Institute of Technology Hyderabad
Homework 2, Mathematical Preliminaries, Assigned 20.02.2022, Due **11:59 pm on 02.03.2022**

Do. Or do not. There is no try. – Jedi Master Yoda

Instructions:

- It is **strongly recommended** that you work on your homework on an *individual* basis. If you have any questions or concerns, feel free to talk to the instructor or the TAs.
- **The functions implemented for HW1 can be reused here.**
- Use `matplotlib` where specified - `https://matplotlib.org/tutorials/introductory/images.html`.
- Do not use other built-in functions that directly solve a problem - *especially for convolution and correlation*.
- Use images from University of Southern California's image database at `http://sipi.usc.edu/database/database.php?volume=misc`.
- Please turn in Python Notebooks with the following notation for the file name: `your-roll-number-hw2.ipynb`.
- Do not turn in images. Please use the same names for images in your code as in the database. The TAs will use these images to test your code.

Problem Set:

1. **Distance between PDFs:** In this question you will explore the other "distances" between PDFs discussed in class. To verify the implementation of these distances, use the normalized histogram of the stereo image pair (`left.png, right.png`) used in the previous assignment.

   (a) **Cross Entropy (CE):** The cross entropy between two PDFs (PMFs) p and q is given by: $H(p,q) = H(p) + D(p||q)$ where $H(p)$ is the entropy of p and $D(p||q)$ is the KL divergence between p and q. Write a function that accepts two PDFs (PMFs) p, q and outputs the CE between them.

      i. Verify your function using the stereo image normalized histogram pair. (1)

      ii. As with the KL divergence problem, choose a fixed PMF $p \sim \text{Bern}(r)$. Choose another PMF $q \sim \text{Bern}(s)$ where s can be varied. Plot $H(p,q)$ as a function of s. From the plot, does minimizing $H(p,q)$ give us matched PMFs? (1)

   (b) **Jensen Shannon (JS) Divergence:** The definition of JS divergence between two PDFs p and q is given by: $J(p,q) = D(p||m) + D(q||m)$ where $m = \frac{p+q}{2}$ and $D(p||q)$ is the KL divergence between p and q. Write a function that accepts two PDFs (PMFs) p, q and outputs the JS divergence between them. Verify that the $JS(p,q)$ is symmetric indeed while $D(p||q)$ is not. Again, use the normalized histograms of the stereo image pair. (1)

   (c) **Wasserstein Distance:** The Wasserstein-1 distance between two PDFs r and s is given by: $W_1(r,s) = \inf_{\pi \in \Pi(r,s)} \mathbb{E}_{(x,y)\sim\pi}|x-y|$. The set $\Pi(r,s)$ is composed of all bivariate joint PDFs whose marginals equal r and s. Given a tuple $(p_{(X,Y)}, r_X, s_Y)$ of a joint histogram $p_{(X,Y)}$, and marginals r_X, s_Y, write a function that accepts this tuple and checks if $p_{X,Y} \in \Pi(r,s)$. Verify your function with a positive example and a negative example. (2)

2. **Visualizing Data Using t-SNE:**

   (a) Read the t-SNE paper and answer the following questions. *Do not reproduce text from the paper verbatim in your answers*.

      i. What is the crowding problem? (1)

ii. How does the choice of the Student t-distribution in the low dimensional embedding space help address the crowding problem? (1)

iii. What other important changes have been made in t-SNE relative to SNE? (1)

(b) In this problem, implement Algorithm 1 from the paper, albeit in a simplified setting as described in the following. (5)

- Generate two clusters of points from a ten-dimensional multivariate Gaussian (MVG) distribution $\mathcal{N}(\mu, 0.01 \cdot I)$ where I is the ten-dimensional identity matrix.
- Use $\mu_1 = \mathbf{1}$ for one cluster and $\mu_2 = 10 \cdot \mathbf{1}$ for the other (where $\mathbf{1}$ is the ten-dimensional vector of ones).
- Generate 10 points from each cluster for a total of 20 points to form the set $\mathcal{X}$.
- Choose the dimension of the embedding to be two.
- Choose $T = 50$.
- Experiment with different choices for η and $\alpha(t)$. For simplicity, let $\alpha(t)$ not change with iterations.
- Use your knowledge of how $\mathcal{X}$ was generated for choices of σ_i (as opposed to finding them using the user-defined *Perplexity*).
- Plot the points in $\mathcal{Y}$ at the beginning and at the end of 50 iterations. Print your observations from the plots.
- Find and print $D(P||Q)$ at the beginning and at the end of 50 iterations. Print your observations from these values.
- The YouTube video by the first author Laurens van der Maaten can be found here.

(c) Now, experiment with the built-in t-SNE utility in `matplotlib`. Choose four different perplexity values (between 5 and 50) and generate t-SNE plots for these choices. How does perplexity affect the plots? (2)

3. **Convolution and Correlation:** In this question, you will compare convolution and correlation by implementing each function. This will build on the example we did in class. Use a color image from the USC database as the input to your functions.

(a) **Convolution:** Write a function that accepts an image I of size $W \times H \times C$ and a convolution kernel h of size $k \times k \times C$ as inputs, and generates the channel-wise convolved image $J = I * h$ as the output. Note that the per the definition of convolution, the size of J is going to be $(W + k - 1) \times (H + k - 1)$. Your output must be of the same size as that of one input channel $W \times H$. To do so, extract the central patch of size $W \times H$ from the result of the convolution. Reflect the input image about the edges to handle overlap and add at image boundaries. The amount of the reflection is dictated by the size of the convolution kernel. Ensure that the you flip either the image or the kernel about both axes in your function. (3)

(b) **Correlation:** Write a function that accepts an image I of size $W \times H \times C$ and a correlation template h of size $k \times k \times C$ as inputs, and generates the channel-wise correlation image $J = \texttt{corr}(I, h)$ as the output. Your output must be of the same size as of one input channel. As with convolution, extract the central patch of size $W \times H$ from the result of correlation. Reflect the input image about the edges to handle overlap and add at image boundaries. The amount of the reflection is dictated by the size of the correlation template. (2)

To understand and visualize the differences easily, choose a kernel that is asymmetric and input the same to both functions. One kernel/template can be a $k \times k \times C$ patch extracted from the image. Also illustrate the difference with another choice of kernel/template. For easy "centering", choose odd values for k. Display the output of each function for each choice of of your kernel/template. You may have to rescale your output before display.