Homework #1

Deep Learning for Computer Vision

Due: 108/3/20 (Wed.) 03:00 AM

Total Score: 120 points

Contact TAs by ntudlcvta2019@gmail.com

- **Academic Honesty**

  Plagiarism/cheating is strictly prohibited and against school regulations. If any form of plagiarism/cheating is discovered, all students involved would receive an *F* score for the course (which is **NOT** negotiable).

- **Collaboration Policy**

  Self-learning by researching on the Internet or discussing with fellow classmates is highly encouraged. However, you must obtain and write the final solution by yourself. Please specify, if any, the references for each of your answers (e.g. the name and student ID of your collaborators and/or the Internet URL you consult with) in your report. If you complete the assignment all by yourself, you must also specify "*no collaborators*".

- **Late HW Submission Policy**

  No late submission is allowed for this homework.

Problem 1: Bayes Decision Rule (20%)

For a 2-class problem based on a single feature $\mathbf{x} \in [\mathbf{0}, \mathbf{9}]$, the class PDFs are defined as below:

$$P(\mathbf{x}|\omega_1) = \text{uniform over } (0, 5)$$
$$P(\mathbf{x}|\omega_2) = \text{uniform over } (2, 9)$$

Determine the minimum P_e decision scheme with $P(\omega_2) = 7/9$. Please state clearly what the decision regions R_1 and R_2 are. What is the resulting probability of error P_e?

Problem 2: Principal Component Analysis (40%)

Principal component analysis (PCA) is a technique for dimensionality reduction which performs a linear mapping of the data to a lower-dimensional space in such a way that the variance of the data in the low-dimensional representation is maximized. In this problem, you will perform PCA on a dataset of face images.

The folder `p2_data` contains face images of 40 different subjects (classes) and 10 grayscale images for each subject, all of size $(56, 46)$ pixels. Note that `i_j.png` is the j-th image of the i-th person, which is denoted as **person**$_i$**image**$_j$ for simplicity.

First, split the dataset into two subsets (i.e., training and testing sets). The first subset contains the first 6 images of each subject, while the second subset contains the remaining images. Thus, a total of $6 \times 40 = 240$ images are in the training set, and $4 \times 40 = 160$ images in the testing set.

In this problem, you will compute the eigenfaces of the training set, and project face images from both the training and testing sets onto the same feature space with reduced dimension.

1. (10%) Perform PCA on the training set. Plot the mean face and the first four eigenfaces.
2. (8%) Take $\textbf{person}_1\textbf{image}_1$, and project it onto the PCA eigenspace you obtained above. Reconstruct this image using the first $n = 3, 45, 140, 229$ eigenfaces. Plot the four reconstructed images.
3. (4%) For each of the four images you obtained in 2., compute the mean squared error (MSE) between the reconstructed image and the original image. Record the corresponding MSE values in your report.
4. (10%) Now, apply the k-nearest neighbors algorithm to classify the testing set images. First, you will need to determine the best k and n values by 3-fold cross-validation. For simplicity, the choices for such hyperparameters are $k = \{1, 3, 5\}$ and $n = \{3, 45, 140\}$. Show the cross-validation results and explain your choice for (k, n).
5. (8%) Use your hyperparameter choice in 4. and report the recognition rate of the testing set.

✓ ***Hint***

- When plotting eigenfaces, be sure to start from the most dominant one to the least dominant one. Note that the calculated eigenvalues may be sorted in either ascending or descending order depending on the programming language/packages you use.
- Display your output faces in grayscale colormap instead of other ones.
- When calculating MSE, your pixel values should be in the range of $[0, 255]$.

Problem 3: Visual Bag-of-Words (40%)

A **bag-of-words model** (BoW) can be applied to image classification, by treating image features as words. In computer vision, a bag of visual words is a vector of occurrence counts of a vocabulary of local image *features*. In this problem, you will implement a basic image-based BoW model for an image dataset with 4 categories, where we use small image patches as features.

The folder `p3_data` contains images of 4 categories (classes) and 500 RGB images for each category, all of size $(64, 64, 3)$ pixels.

First, split the dataset into two subsets (i.e., training and testing sets). The first subset contains the first 375 images of each category, while the second subset contains the remaining images. Thus, a total of $375 \times 4 = 1500$ images are in the training set, and $125 \times 4 = 500$ images in the testing set. We will denote them as `X_train` and `X_test`, respectively.

1. (5%) Divide up each image in both `X_train` and `X_test` into a grid of $(16, 16, 3)$ image patches. Since each image is of size $(64, 64, 3)$, this will result in 16 different patches of size $(16, 16, 3)$ for each image. You can imagine the patches as puzzle pieces which together would reconstitute the whole image. Pick 5 images (one from each category) randomly and plot 3 such patches from each image you choose. Describe whether you are able to classify an image by seeing just a few patches and write why.

2. (15%) Flatten the patches into 1D vectors and store them as variables `X_train_patches` and `X_test_patches`. Since each patch is of size $16 \times 16 \times 3 = 768$ where each image contains 16 such patches, `X_train_patches` and `X_test_patches` should be of size $(24000, 768)$ and $(8000, 768)$, respectively.

   Use the k-means algorithm to divide the training patches (features) into C clusters. You may choose $C = 15$ and maximum number of iterations $= 5000$ for simplicity. The centroid of each cluster then indicates a visual word.

   Construct the 3-dimensional PCA subspace from the training features. Randomly select 6 clusters from the above results. Plot the visual words (i.e., centroids) and their associated features (i.e., patches) in this PCA subspace. Use the same color for features belonging to the same cluster in your visualization.

3. (10%) With the derived dictionary of visual words, you can now represent each image as BoW features. We will adopt the **soft-max** strategy when encoding image patches as detailed below.

   Take an image with 4 patches and a learned dictionary with 3 visual words ($C = 3$) for example. Table 1 lists the Euclidean distance between the patches f_i (with $i = 1, 2, 3, 4$) and the centroids c_j (with $j = 1, 2, 3$). For each patch, the reciprocal of its distance to each centroid would be normalized (i.e., the reciprocal of each entry in a row in Table 1 sums to unity, as shown in Table 2). Each attribute in the BoW is then determined by the maximum value of the soft-encoded features in that dimension (i.e., **max-pooling**). For example, the BoW of the patches in Table 1 would be $[0.55\, 0.27\, 0.55]$ by taking the maximum value of each column.

   Now, compute the BoW of training images in `X_train`, resulting in a matrix of size $(1500, C)$. Choose one image from each category and visualize its BoW using histogram plot.

4. (10%) Adopt the k-nearest neighbors classifier (k-NN) to perform classification on `X_test` using the above BoW features (you may choose $k = 5$ for simplicity). Report the classification accuracy.

	c_1	c_2	c_3
f_1	1	2	3
f_2	2	3	1
f_3	2	3	1
f_4	3	2	1

Table 1: Euclidean distances

	c_1	c_2	c_3
f_1	0.55	0.27	0.18
f_2	0.27	0.18	0.55
f_3	0.27	0.18	0.55
f_4	0.18	0.27	0.55

Table 2: Normalized reciprocals of distances

Problem 4: Image Filtering (20%)

In the lectures, we introduced the concept of image filtering and its applications. In this problem, you are asked to implement basic **Gaussian filters** and apply them to images for evaluation. Below are the 1D and 2D kernels of a Gaussian filter:

$$\text{1D kernel :} \quad G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

$$\text{2D kernel :} \quad G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

1. (2%) Given a variance σ^2, the convolution of a 2D Gaussian kernel can be reduced to two sequential convolutions of a 1D Gaussian kernel. Show that convolving with a 2D Gaussian filter is equivalent to sequentially convolving with a 1D Gaussian filter in both vertical and horizontal directions.

2. (6%) Implement a discrete 2D Gaussian filter using a 3×3 kernel with $\sigma \approx \frac{1}{2\ln 2}$. Use the provided `lena.png` as input, and plot the output image in your report. Briefly describe the effect of the filter.

3. (8%) Consider the image $I(x, y)$ as a function $I : \mathbb{R}^2 \to \mathbb{R}$. When detecting edges in an image, it is often important to extract information from the derivatives of pixel values. Denote the derivatives as follows:

$$I_x(x,y) = \frac{\partial I}{\partial x} \approx \frac{1}{2}(I(x+1,y) - I(x-1,y))$$
$$I_y(x,y) = \frac{\partial I}{\partial y} \approx \frac{1}{2}(I(x,y+1) - I(x,y-1)).$$

   Implement the 1D convolution kernels $k_x \in \mathbb{R}^{1\times 3}$ and $k_y \in \mathbb{R}^{3\times 1}$ such that

$$I_x = I * k_x$$
$$I_y = I * k_y.$$

   Write down your answers of k_x and k_y. Also, plot the resulting images I_x and I_y using the provided `lena.png` as input.

4. (4%) Define the **gradient magnitude** image I_m as

$$I_m(x,y) = \sqrt{I_x(x,y)^2 + I_y(x,y)^2}.$$

   Use **both** the provided `lena.png` **and** the Gaussian-filtered image you obtained in 2. as input images. Plot the two output gradient magnitude images in your report. Briefly explain the differences in the results.

✓ ***Hint***

- When using `lena.png` for this problem, be sure to load it as a *grayscale* image.

Remarks

- For this homework, we will **not** grade your source code. Thus, you may use **any** programming language you desire. You are also allowed to use any related packages, libraries, and functions for your implementation. However, you must provide figure outputs with detailed discussions or explanations.

- Please convert your report into a single `.pdf` file (with arbitrary file name) and upload it to CEIBA before the deadline. Yo do **NOT** need to upload anything other than your report for this homework.