Lab 4A: Image Classification

Juan Elenter, Ignacio Hounie and Alejandro Ribeiro[*]

April 1, 2023

1 Images

Images are mathematically modeled as functions of two variables. These variables represent the vertical coordinate m and the horizontal coordinate n. These variables can be discrete or continuous but we will consider them here to be discrete. In this case the coordinate pair (m, n) is called a pixel. We restrict each pixel coordinate to be between 0 and $N-1$. We use $\mathbf{X}$ to denote the image and and $x(m, n)$ to represent the value at pixel (m, n). Pixels of an image are arranged in a matrix,

$$\mathbf{X} = \begin{pmatrix} x(0,0) & x(0,1) & \cdots & x(0,n) & \cdots & x(0,N) \\ x(1,0) & x(1,1) & \cdots & x(1,n) & \cdots & x(1,N) \\ \vdots & \vdots & & \vdots & & \vdots \\ x(m,0) & x(m,1) & \cdots & x(m,n) & \cdots & x(m,N) \\ \vdots & \vdots & & \vdots & & \vdots \\ x(N,0) & x(N,1) & \cdots & x(N,n) & \cdots & x(N,N) \end{pmatrix}. \tag{1}$$

Thus, an image $\mathbf{X}$ is a matrix in which entries $x(m, n)$ represent pixel values.

The interpretation of this matrix is that pixel values $x(m, n)$ represent the luminance of the pixel. The luminance is how much light is reflected by the pixel. This determines how bright the pixel appears in a black and

[*]In alphabetical order.

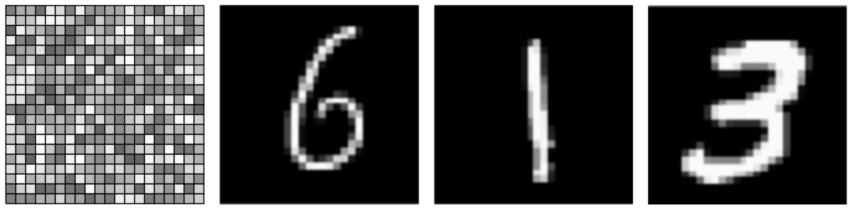

Figure 1. Images. An image is a function of two discrete indexes in which function values represent luiminance. This information is arranged in a matrix $\mathbf{X}$ in which the entry $x(m,n)$ is the luminance at coordinates (m,n) [cf. (1)]. In this lab we work with images that represent handwritten digits [cf. (3)].

white image. For that reason we do not represent images as functions. We represent them as luminance colormaps; see Figure 1.

To represent color images we just consider several matrices with different color contents. For instance, in a red, blue, and green (RGB) representation we have matrices $\mathbf{X}_\text{R}$, $\mathbf{X}_\text{G}$, and $\mathbf{X}_\text{B}$ that represent the luminance restricted to the red, blue, and green colors. This information can be represented with a tensor,

$$\mathbf{X} = [\mathbf{X}_\text{R}, \mathbf{X}_\text{G}, \mathbf{X}_\text{B}] \tag{2}$$

In this tensor, each of the color matrices is called a channel.

1.1 Handwritten Digits

In this lab we work with black and white images that represent handwritten digits. In Figure 1 we show an image of a handwritten number 1 and a handwritten number 3. The dataset we are given contains pairs $(\mathbf{X}_q, y_q)$ of images $\mathbf{X}_q$ and human annotations y_q that identify the correct digits. For the images in Figure 1, the dataset entries are

$(\mathbf{X}_q, y_q) = \Big($ $, 6\Big),\quad (\mathbf{X}_q, y_q) = \Big($ $, 3\Big),\quad (\mathbf{X}_q, y_q) = \Big($ $, 1\Big).$ (3)

The dataset contains $Q =??$ images.

Our mission, which we have accepted, is to train a machine learning system that maps images to the corresponding digit. As it should be obvious by now, this requires that we introduce a proper kind of convolution. We do that in the next section.

Task 1 Load the data from https://dsd.seas.upenn.edu and visualize three images. ■

2 Shifts in Space

Before defining convolutions to process images we have to introduce vertical and horizontal shift operators. We name these operators $\mathcal{S}_{\text{V}}$ and $\mathcal{S}_{\text{H}}$ and define them as the operators whose action on an image $\mathbf{X}$ results in a shifting of vertical and horizontal coordinates, respectively. Thus, if applying the vertical shift $\mathcal{S}_{\text{V}}$ to the image $\mathbf{X}$ yields the image $\mathbf{U}_{10} = \mathcal{S}_{\text{V}}\mathbf{X}$, the entries $u_{10}(m,n)$ of the vertically shifted image $\mathbf{U}_{10}$ are

$$u_{10}(m,n) = x(m-1,n), \quad \text{when } \mathbf{U}_{10} = \mathcal{S}_{\text{V}}\mathbf{X}. \tag{4}$$

Likewise if applying the vertical shift $\mathcal{S}_{\text{H}}$ to the image $\mathbf{X}$ yields the image $\mathbf{U}_{01} = \mathcal{S}_{\text{V}}\mathbf{X}$, the entries $u_{01}(m,n)$ of the horizontally shifted image $\mathbf{U}_{01}$ are

$$u_{01}(m,n) = x(m,n-1), \quad \text{when } \mathbf{U}_{01} = \mathcal{S}_{\text{H}}\mathbf{X}. \tag{5}$$

In (4) and (5) we adopt the convention that $x(m,n) = 0$ when either of the arguments m or n are negative.

To understand (4) and (5) it is convenient to represent them in matrix form. Applying the vertical shift to the image in (1) yields the matrix

$$\mathcal{S}_{\text{V}}\mathbf{X} = \begin{pmatrix} 0 & 0 & \cdots & 0 & \cdots & 0 \\ x(0,0) & x(0,1) & \cdots & x(0,n) & \cdots & x(0,N) \\ \vdots & \vdots & & \vdots & & \vdots \\ x(m,0) & x(m,1) & \cdots & x(m,n) & \cdots & x(m,N) \\ \vdots & \vdots & & \vdots & & \vdots \\ x(N-1,0) & x(N-1,1) & \cdots & x(N-1,n) & \cdots & x(N-1,N) \end{pmatrix}. \tag{6}$$

In the vertically shifted image $\mathbf{U}_{10} = \mathcal{S}_{\text{V}}\mathbf{X}$ all of the rows of the matrix $\mathbf{X}$ are shifted down. We fill the first row with zeros because there are no pixels that can be shifted into the first row.

When we apply the horizontal shift $\mathcal{S}_{\text{H}}$ to the image $\mathbf{X}$ in (1) we obtain the image

$$\mathcal{S}_{\text{H}}\mathbf{X} = \begin{pmatrix} 0 & x(0,0) & \cdots & x(0,n) & \cdots & x(0,N-1) \\ 0 & x(1,0) & \cdots & x(1,n) & \cdots & x(1,N-1) \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & x(m,0) & \cdots & x(m,n) & \cdots & x(m,N-1) \\ \vdots & \vdots & & \vdots & & \vdots \\ 0 & x(N,0) & \cdots & x(N,n) & \cdots & x(N,N-1) \end{pmatrix}. \tag{7}$$

In the horizontally shifted image $\mathbf{U}_{01} = \mathcal{S}_{\text{H}}\mathbf{X}$ all of the columns of the matrix $\mathbf{X}$ are shifted right. We fill the first column with zeros because there are no pixels that can be shifted into the first column.

2.1 Shift Compositions and Spatial Shift Sequences

As in the case of time signals, image shifts can be composed. For instance, applying the vertical shift twice yields the signal

$$\mathbf{U}_{20} = \mathcal{S}_{\text{V}}\,\mathbf{U}_{10} = \mathcal{S}_{\text{V}}^2\,\mathbf{X}. \tag{8}$$

This is a signal in which the entries are $u_{20}(m,n) = x(m-2,n)$. That is, the image $\mathbf{U}_{20}$ is one in which the pixels are shifted two pixels down.

As another example consider the application of l shifts in the horizontal direction. We can define define this recursively as

$$\mathbf{U}_{0l} = \mathcal{S}_{\text{H}}\,\mathbf{U}_{0(l-1)} = \mathcal{S}_{\text{H}}^2\,\mathbf{U}_{0(l-2)} = \ldots = \mathcal{S}_{\text{H}}^{l-1}\,\mathbf{U}_{01} = \mathcal{S}_{\text{H}}^{l}\,\mathbf{X}. \tag{9}$$

This is a signal in which the entries are $u_{0l}(m,n) = x(m,n-l)$. That is, the image $\mathbf{U}_{0l}$ is one in which the pixels are shifted l pixels to the right.

In general, we can compose any number of vertical shifts with any number of horizontal shifts. This results in the definition of the spatial shift sequence composed of images

$$\mathbf{U}_{kl} = \mathcal{S}_{\text{V}}^{k}\mathcal{S}_{\text{H}}^{l}\mathbf{X}. \tag{10}$$

This is an image in which the entries are $u_{kl}(m,n) = x(m-k,n-l)$. That is, the image $\mathbf{U}_{kl}$ is one in which the pixels are shifted k pixels down and l pixels to the right.

It is important to note that the spatial shift sequence in (10) can be computed recursively. To do that we need to define the vertical and horizontal recursions

$$\mathbf{U}_{kl} = \mathcal{S}_{\text{V}}\mathbf{U}_{(k-1)l}, \qquad \mathbf{U}_{kl} = \mathcal{S}_{\text{H}}\mathbf{U}_{k(l-1)}, \tag{11}$$

with the initial condition $\mathbf{U}_{00} = \mathbf{X}$. Recursive computation of the spatial diffusion sequence is important in practical implementations.

2.2 Negative Shifts

A negative vertical shift is a shift that moves the pixels up. We denote this shift by $\mathcal{S}_{\text{V}}^{-1}$ and define it as the shift that when acting on the image $\mathbf{X}$ yields the image $\mathbf{U}_{(-1)0} = \mathcal{S}_{\text{V}}^{-1}\mathbf{X}$ whose entries are given by

$$u_{(-1)0}(m,n) = x(m+1,n), \quad \text{when } \mathbf{U}_{(-1)0} = \mathcal{S}_{\text{V}}^{-1}\mathbf{X}. \tag{12}$$

In this definition we adopt the convention that $x(N+1,n) = 0$. This means that when applying the negative shift $\mathcal{S}_{\text{V}}^{-1}$ we fill the last row of $\mathbf{U}_{(-1)0}$ with zeros. This is because there are no entries of $\mathbf{X}$ that can be shifted into this row.

Likewise a negative horizontal shift is a shift that moves the pixels left. We denote this shift by $\mathcal{S}_{\text{H}}^{-1}$ and define it as the shift that when acting on the image $\mathbf{X}$ yields the image $\mathbf{U}_{0(-1)} = \mathcal{S}_{\text{H}}^{-1}\mathbf{X}$ whose entries are given by

$$u_{0(-1)}(m,n) = x(m,n+1), \quad \text{when } \mathbf{U}_{0(-1)0} = \mathcal{S}_{\text{H}}^{-1}\mathbf{X}. \tag{13}$$

In this definition we adopt the convention that $x(m,N+1) = 0$. This means that when applying the negative shift $\mathcal{S}_{\text{H}}^{-1}$ we fill the last column of $\mathbf{U}_{(-1)0}$ with zeros. This is because there are no entries of $\mathbf{X}$ that can be shifted into this column.

As is the case of the positive shifts $\mathcal{S}_{\text{V}}$ and $\mathcal{S}_{\text{H}}$, the negative shifts $\mathcal{S}_{\text{V}}^{-1}$ and $\mathcal{S}_{\text{H}}^{-1}$ can be composed. It is also possible to compose the positive vertical shift $\mathcal{S}_{\text{V}}$ with the negative horizontal shift $\mathcal{S}_{\text{H}}^{-1}$. The converse composition of the positive horizontal shift $\mathcal{S}_{\text{H}}$ with the negative vertical shift $\mathcal{S}_{\text{V}}^{-1}$ is also possible. We can therefore generalize (10) into

$$\mathbf{U}_{kl} = \mathcal{S}_{\text{V}}^{k}\mathcal{S}_{\text{H}}^{l}\mathbf{X}. \tag{14}$$

This is, in fact, the exact same equation as (10). The difference is that we now allow for k and l to be positive or negative numbers. The entries of

$\mathbf{U}_{kl}$ are $u_{kl}(m,n) = x(m-k, n-l)$. When k is positive this entails shifting pixels down and when k is negative this entails shifting pixels up. When l is positive this entails shifting pixels to the right and when k is negative this entails shifting pixels to the left.

3 Convolutions in Space

A spatial convolution is a linear combination of the components of the spatial diffusion sequence in (14). For a formal definition we consider a filter range K and define filter coefficients h_{kl} for k and l ranging from $-K$ to K. The outcome of applying the filter with coefficients h_{kl} to the image $\mathbf{X}$ is the image

$$\mathbf{Y} = \sum_{k=-K}^{K} \sum_{l=-K}^{K} h_{kl}\, \mathcal{S}_{\mathrm{V}}^{k}\, \mathcal{S}_{\mathrm{H}}^{l}\, \mathbf{X}. \tag{15}$$

Observe that in this definition we allow for positive and negative shifts. For simplicity we assume that the maximum number of shifts in either direction is the same. They can be made different, but it is standard practice to keep them equal.

We note that the total number of filter coefficients of a two dimensional convolution is $(2K+1)^2$. These coefficients are arranged in a matrix $\mathbf{H}$.

Task 2 Write a filter object and endow it with a function that takes an image $\mathbf{X}$ as an input and returns as an output the result of convolving the image with the filter $\mathbf{H}$ that is stored as an attribute of the class. Make sure that this function can operate on a batch of input images ■

Task 3 Define filters $\mathbf{H}$ with entries $h_{kl} = 1/((2K+1)^2)$. Use the class in Task 2 to instantiate these filters with $K = 2$, $K = 4$, and $K = 8$. Apply these filters to the images you visualized in Task 1. What is the effect of applying these filters? ■

Task 4 Pytorch comes with its own function for computing two dimensional convolutions. Consider the same filters of Task 3 with $K = 8$. Execute your own convolution code and the code that comes built in in Pytorch to compute convolutions for a batch of $B = 100$ images. Compare the execution times. ■

In solving Task 4 you must have noticed that the Pytorch code is much faster. This is because the Pytorch convolution function is just a wrapper that calls a compiled subroutine written in C. If this were the 1990's we would say that "true men code in C," which was indeed a quip that was common in the 1990's. Lucky for us all we live in a time when we do not require men to be tough, nor women to stay away from engineering, nor persons to define themselves as either men or women. So let us just remember that industry level data sciences still relies on low level languages. Python and Pytorch are prototyping languages.

We will use the Pytorch convolution function in the remainder of Lab 4.

4 Filterbanks

To use convolutional filters in classification tasks we will build convolutional neural networks (CNNs). An intermediate step is to use the energy content of the outputs of a convolutional filterbank.

Begin then by defining the energy of an image as the sum of the squares of the values of each pixel,

$$\|\mathbf{X}\|^2 = \sum_{m=0}^{M-1}\sum_{n=0}^{M-1} x^2(m,n). \tag{16}$$

Continue by defining a collection of G filters $\mathbf{H}^g$ each of which contains filter coefficients h_{kl}^g. Processing the input image $\mathbf{X}$ with each of these filters produces the output feature,

$$\mathbf{Z}^g = \sum_{k=-K}^{K}\sum_{l=-K}^{K} h_{kl}^g\, \mathcal{S}_{\text{V}}^k\, \mathcal{S}_{\text{H}}^l\, \mathbf{X}. \tag{17}$$

Each of the $\mathbf{Z}^g$ features is an image. This set of images has to be converted into a set of class scores as we did in Lab 2C. We will do that in two steps. The first step is to create a vector that groups the energies of the outputs of each filterbank

$$\mathbf{x}_1 = \Big[\ \|\mathbf{Z}^1\|^2;\ \|\mathbf{Z}^2\|^2;\ \ldots;\ \|\mathbf{Z}^G\|^2\ \Big]. \tag{18}$$

The second step is to process this vector with a readout layer. This is just a multiplication with a matrix $\mathbf{A}$ that matches the dimension G of the filterbank with the number of classes C,

$$\mathbf{x}_2 = \mathbf{A}\mathbf{x}_1. \tag{19}$$

Each of the entries of $\mathbf{x}_2$ is a class score. We will train this filterbank to correctly predict class labels.

Task 5 Create a filterbank class. In this class the filter coefficients $\mathbf{H}^g$ the matrix $\mathbf{A}$ are attributes. The class has a method that takes an image $\mathbf{X}$ as an input and implements the filterbank equations in (17) - (19). The method returns the vector of scores $\mathbf{x}_2$ as an output.

Task 6 Split the handwritten digit dataset into train and test sets. Instantiate the class of Task 5 to train a filterbank that classifies images to the corresponding digit class. Use the crossentropy loss as the optimization objective.

Evaluate the test loss and the percentage of images that are correctly classified.

5 Report

Do not take much time to prepare a lab report. We do not want you to report your code and we don't want you to report your work. Just give us answers to questions we ask. Specifically give us the following:

Question	**Report deliverable**
Task 1	Do not report
Task 2	Do not report
Task 3	One paragraph with a qualitative description of the filters' outputs
Task 4	Do not report
Task 5	Do not report
Task 6	Number of images in the train and test set. Train and test loss. Percentage of images correctly classified.

Number of images in the train and test set. Training loss Test loss

We will check that your answers are correct. If they are not, we will get back to you and ask you to correct them. As long as you submit responses, you get an A for the assignment. It counts for 10% of your lab grade.