Homework 4 Solution

Fall 2019

Exercises

1. (The Convolution Operation)

   a) The definition of convolution, for the one-dimensional discrete-time case, between an input, x, and a filter, h, is given by:

   $$(x * h)(t) = \sum_{i=-\infty}^{\infty} x(i)h(t-i). \tag{1}$$

   The definition of cross-correlation, for the one-dimensional discrete-time case, between x and y, is given by:

   $$(x * y)(\tau) = \sum_{i=-\infty}^{\infty} x(i)y(i+\tau). \tag{2}$$

   The difference between convolution and cross-correlation is that the kernel is flipped for convolution whereas the kernel is not flipped in cross-correlation.
   Convolution is commutative (i.e., $(x * h) = (h * x)$). This can be shown by using the substitution $u = t - i$ in (1).

   b) DNNs use cross-correlation because the commutative property is not an important property for neural networks, as a neural network can learn to flip the kernel.

2. (Variants of the Basic Convolution Function)

   a) In convolutional layers, the same kernel is used for every region of the input into that layer. Locally connected layers, on the other hand, use different kernels for different locations of the input. In other words, several kernels are learned in each layer and each kernel is convolved with a different location on the image.
   A convolutional layer can be useful if we are interested in determining if a particular attribute is present in an input. For example, if we are trying to find whether a cat is present in an image, and we do not care about where in the image it is, we could use convolutional layers, which would find a cat regardless of its spatial location in the image.
   Locally connected layers can be useful if we know beforehand that we will want to weight different parts of the input differently. For example, if we are interested in recognizing a particular face, that we know will be in the center of an image, we may want to give less weights to the chin and mouth of the image (in the bottom half of the image) and more weights to the eyes (top half of the image). This can be achieved by using different kernels in different regions of the image.

   b) Convolution paired with pooling (over spatial regions) causes convolutional layers to have a property called equivaraince to translation. This means that if the input is translated (shifted), the output is shifted in the same way.

3. (Pooling)

   a) In a general sense, *pooling* summarizes the values in a particular region of an output. This can be done by taking the maximum value in a region of values and discarding the rest, taking the average of all values in a particular region, taking a weighted average in a particular region, etc.

   b) i. If we are summarizing a region of k samples during pooling, then the output of a pooling layer will contain approximately k times fewer inputs. As a result of this reduction, we will reduce the memory required to store the parameters (in comparison to using no pooling).

      ii. In general, pooling helps make the input invariant to small translations of the input. This means that if we translate an image, the output of the pooling layer will not change. This helps in identifying if particular features are present in an input rather than where the particular features are located. As a result, pooling is able to generalize certain features regardless of translations among samples at the input.

   c) i. The pooling width is the region of samples over which we are going to summarize. We can define a pooling width ($1 \times m$) or a pooling window ($n \times m$). Then, we would take a summary statistic over the specified width or window.

      ii. The stride is how many samples we are going to shift the pooling width (or window in each dimension) each time after we compute the summary statistic in the prior region.

   d) Pooling over spatial regions introduces translational invariance.

   e) By using multiple convolutional kernels, and applying pooling across the outputs of these kernels, the network can learn the type of transformation, over which pooling introduces invariance.

4. (Zero Padding)

   a) *Downsampled convolution* refers to regularly skipping positions of the convolutional kernel (which can result in savings of computational cost). If we are interested in sampling every k^{th} sample in each direction of the output, we set the stride to k.

   b) After applying the first convolutional layer, we would be left with an output that is 1×11. Then, after applying the second convolutional layer, we would have an output that is 1×6. Then, after applying another convolutional layer, we would have an output that is 1×1. Therefore, we can apply a maximum of **3** convolutional layers.

   c) The kernel is not moving in the last layer because the size of its input is 1×6. Therefore, the kernel is being applied to that specific window and cannot shift.

   d) The problem with using smaller kernels is that they may not capture the spatial feature of interest (for example it may be too small to capture the features representing a cat in an image) and, as a result, will limit the expressive power of the neural network.

   e) Zero padding adds additional '0s' to the output of the convolution to preserve the size of the input after convolution. For example, if we consider the outputs of each of the three convolutional layers discussed in (b), then the output of the first convolutional layer (which without padding would result in a size of 1×11) could add five '0s' to the end and hence we would have an output with a shape of the original 1×16. We could continue doing this for multiple convolutional layers and we would not be left with a 1×1, and hence would not be limited to a certain number of convolutional layers.

   f) In MATLAB, the operation described in (e) is referred to as **same** convolution.

5. (Tiled Convolution)

   a) Tiled convolution refers to using different kernels for a number of adjacent locations equal to the tile t, and then sharing these kernels for the next t locations, and so on. It uses aspects of locally connected layers and convolutional layers. Specifically, similar to locally connected layers, tiled convolution employs learning a set of kernels that are used in various regions of an input. Also, it is similar to convolutional layers in that the same kernel will be used in certain regions of the image (although these regions are not adjacent). The memory requirement is proportional to the number of parameters in each convolutional kernel multiplied by the tile t.

b) i. When using convolutional layers, we only have one kernel that is used for every region in the input. Therefore, $\boldsymbol{t = 1}$ would make tiled convolution equivalent to normal convolution.

ii. When using locally connected layers, each region is convolved with a different kernel. Therefore, if $\boldsymbol{t}$ **is equal to the output width**, then tiled convolution is the same as using a locally connected layer.

6. (Data Types)

a) i. An example of a 1-D single channel data type could a time-domain signal (such as an acoustic wave or the output voltage of an AC circuit).
An example of a 1-D multichannel data type could be multiple time-series signals that were collected simultaneously. For example, a single time sample contains multiple corresponding measurements from various sensors.

ii. An example of a 2-D single channel data type could be gray-scale images (that also have height and width). For example, the MNIST dataset of hand written digits corresponds to 2-D single channel images.
An example of a 2-D multichannel data type would be color images where different channels correspond to different colors (red, green, and blue).

iii. An example of a 3-D single channel data type could be a volumetric gray-scale image. For example, a CT scan (which has length, height, and width) can be grayscale and hence correspond to a single channel.
An example of a 3-D multichannel data type could be a color video where different channels correspond to height, width, time, etc.

b) Convolution is suitable for processing variably sized inputs, when the variability is due to having different amounts of the same type of information. For example, images belonging to the same category that have different sizes. On the other hand, it is not suitable when the variability is due to optional fields that contain information of different types. For example, if an image could be input with optional fields containing information about the signal-to-noise ratio, object description, ... etc.