CS7643: Deep Learning
Fall 2017
Problem Set 3

Instructor: Dhruv Batra
TAs: Michael Cogswell, Abhishek Das, Zhaoyang Lv
Discussions: http://piazza.com/gatech/fall2017/cs7643

Due: Saturday October 28, 2017

1 Convolution Basics

The convolution layer inside of neural network is intrinsically an *affine transformation*: A vector is received as input and is multiplied with a matrix to produce an output (to which a bias vector is usually added before passing the result through a nonlinearity). This operation can be represented as $y = Ax$, in which A describes the affine transformation.

1. **[2 points]** We will first revisit the convolution layer similarly discussed in the class. Now considering a convolution layer with a simple 2x2 kernel W, operated on a single input channel X, represented as:

$$W = \begin{bmatrix} w_{(0,0)}, w_{(0,1)} \\ w_{(1,0)}, w_{(1,1)} \end{bmatrix}, X = \begin{bmatrix} x_{(0,0)}, x_{(0,1)}, x_{(0,2)} \\ x_{(1,0)}, x_{(1,1)}, x_{(1,2)} \\ x_{(2,0)}, x_{(2,1)}, x_{(2,2)} \end{bmatrix} \tag{1}$$

   Now let us work out a **stride-3** convolution layer, with **zero padding size 1**. In our implementation, we have flattened the input tensor X in row-major order as:

$$X = \left[x_{(0,0)}, x_{(0,1)}, x_{(0,2)}, ..., x_{(2,0)}, x_{(2,1)}, x_{(2,2)} \right]^\top \tag{2}$$

   Write down the convolution as a matrix operation A that: $Y = AX$. Output Y is also flattened in row-major order.

2. **[2 points]** With transposed convolution, we can upsample the feature size spatially. Now similarly considering a transposed convolution layer with a simple 2x2 kernel W, operated on a single input channel X, represented as:

$$W = \begin{bmatrix} w_{(0,0)}, w_{(0,1)} \\ w_{(1,0)}, w_{(1,1)} \end{bmatrix}, X = \begin{bmatrix} x_{(0,0)}, x_{(0,1)} \\ x_{(1,0)}, x_{(1,1)} \end{bmatrix} \tag{3}$$

We also flattened the input tensor in row-major order as $X = [x_{(0,0)}, x_{(0,1)}, x_{(1,0)}, x_{(1,1)}]$. Write down the affine transformation A with **stride 2, padding 0**, for a transposed convolution layer. Output Y is also flattened in row-major order.

3. **[4 points]** The implementation of convolution layers in most CNN frameworks is usually performed over multiple input and output feature maps. The collection of kernels form a 4D array (output channels o, input channels i, filter rows k, filter columns k), in short as (o, i, k, k). For each output channel, each input channel is convolved with a distinct part of the kernel and the resulting set of feature maps is summed element-wise to produce the corresponding output feature map. The result of this procedure is a set of output feature maps, one for each output channel, that is the output of the convolution operation.

   There is an interesting property that a convolutional layer with kernel size $(o \times r^2, i, k, k)$ is identical to a transposed convolution layer with kernel size $(o, i, k \times r, k \times r)$. Here the word 'identical' means with the same input feature X, both operations will give the same output Y with only a difference in the ordering of linear equations.

   Now prove the property with this simple case: $o = 1, r = 2, i = 1, k = 1$. Given the same input feature X in equation 3, write down the affine transformation for a convolutional layer with kernel size (4, 1, 1, 1), and show it is an operation identical to a transposed convolution layer with kernel size $(1, 1, 2, 2)$.

2 Directed Acyclic Graphs (DAG)

One important property for feed-forward network that we have discussed in class is that it must be a directed acyclic graph (DAG). Recall that a *DAG is a directed graph that contains no directed cycles*. We will discuss some of its properties in this question.
Define $G = (V, E)$ in which V is the set of all nodes as $\{v_1, v_2, ..., v_i, ...v_n\}$ and E is the set of all possible edges $e_{i,j} = (v_i, v_j)$.
A *topological order of a directed graph* $G = (V, E)$ is an ordering of its nodes as $\{v_1, v_2, ..., v_i, ...v_n\}$ so that for every edge (v_i, v_j) we have $i < j$.
There are several lemmas can be inferred from the definition of DAG. One lemma is: if G is a DAG, then G has a node with no incoming edges.
Given the above lemma, prove the following two lemmas:

1. **[2 points]** If the graph G is a DAG, then G has a topological ordering.
2. **[2 points]** If the graph G has a topological order, then G is a DAG.

3 Recurrent Neural Network

1. **[Vanilla RNN for parity function : 4 points]**

   A parity function is a function that takes in a sequence of binary inputs, and returns 1 if the number of 1's is odd, and 0 if it is even. For example, given input sequence $[0, 1, 0, 1, 1, 0]$, the parity bits is $[0, 1, 1, 0, 1, 1]$. We will use this function to represent a temporal sequence. x_t is the boolean digit at index t in the sequence, and y_t is the boolean output of this parity function.

Design a vanilla recurrent neural network to achieve the parity function. Your implementation should include the equations of your hidden units and the details about activations with different input at x_t (0 or 1).

Hint: recall that we have implemented AND and OR functions in problem set 2, which may be useful here.

2. **[Exploding Gradients : 4 points]**

There is a mathematical challenge of learning long-term dependencies in recurrent networks: gradients propagated over many stages tend to either vanish (most of the time) or explode (rarely, but with much damage to the optimization). To discuss this problem for simplicity, consider the following recurrence relation without a nonlinear activation function and input x:

$$h_t = W^\top h_{t-1} \tag{4}$$

Where W is a weight sharing matrix for recurrent relation at any time t. Let $\lambda_1, ..., \lambda_n$ be the eigenvalues of the weight matrix $W \in \mathbb{C}^{n \times n}$. Its spectral radius $\rho(W)$ is defined as:

$$\rho(W) = \max\{|\lambda_1|, ..., |\lambda_n|\} \tag{5}$$

Refering to the long-range relationship between the hidden states, discuss the convergence behavior of this RNN in equation 4 with respect to weight matrix W.