Homework 3

Deadline: Saturday, Oct. 26, at 11:59pm.

Submission: You need to submit your answers to all 3 questions through MarkUs[1] as a PDF file titled `hw3_writeup.pdf`. You can produce the file however you like (e.g. LaTeX, Microsoft Word, scanner), as long as it is readable.

Late Submission: 10% of the marks will be deducted for each day late, up to a maximum of 3 days. After that, no submissions will be accepted.

Collaboration: Homeworks are individual work. See the course web page for detailed policies.

1. **[2pts] Multilayer Perceptron.** Give the weights and biases of a multilayer perceptron which takes as input two scalar values (x_1, x_2) and outputs the values in sorted order, i.e. (y_1, y_2) with $y_1 = \min(x_1, x_2)$ and $y_2 = \max(x_1, x_2)$. The hidden units should all use the ReLU activation function, and the output units should be linear. You should explain why your solution works, but you don't need to provide a formal proof.

2. **[4pts] Backprop.** The deep residual network, or ResNet, is the state-of-the-art architecture for image classification. It's based on a kind of layer called a *residual block*; in this question, you'll figure out how to backprop through a residual block. While the actual ResNet is a convolutional architecture, we'll consider a toy version that's fully connected.

   Consider the following architecture, which takes as input a vector $\mathbf{x}$ and outputs a vector $\mathbf{y}$ of the same size. Its hidden representation $\mathbf{h}$ also has the same size (i.e. number of units). The computations are as follows:

   $$\mathbf{h} = \phi(\mathbf{W}\mathbf{x} + \mathbf{b}) + \mathbf{x}$$
   $$\mathbf{y} = \phi(\mathbf{V}\mathbf{h} + \mathbf{c}) + \mathbf{h}$$

   The parameters are the weight matrices $\mathbf{W}$ and $\mathbf{V}$ and the bias vectors $\mathbf{b}$ and $\mathbf{c}$. Here, ϕ is the activation function, and you can write its elementwise derivatives as $\phi'(\cdots)$.

   To help with the backprop derivations, it's useful to decompose out these computations in a way that introduces variables to hold some intermediate results:

   $$\mathbf{z} = \mathbf{W}\mathbf{x} + \mathbf{b}$$
   $$\mathbf{h} = \phi(\mathbf{z}) + \mathbf{x}$$
   $$\mathbf{r} = \mathbf{V}\mathbf{h} + \mathbf{c}$$
   $$\mathbf{y} = \phi(\mathbf{r}) + \mathbf{h}$$

   (a) **[1pt]** Draw the computation graph for all the variables ($\mathbf{x}$, $\mathbf{z}$, $\mathbf{h}$, $\mathbf{r}$, $\mathbf{y}$, $\mathbf{W}$, $\mathbf{b}$, $\mathbf{V}$, and $\mathbf{c}$).

   (b) **[3pts]** Determine the backprop rules (in vector form) for computing the gradients with respect to all the parameters ($\mathbf{W}$, $\mathbf{b}$, $\mathbf{V}$, and $\mathbf{c}$).

[1] `https://markus.teach.cs.toronto.edu/csc2515-2019-09`

3. **[4pts] AlexNet.** For this question, you will first read the following paper:

   > A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems (NIPS), 2012.
   >
   > `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks`

   This is a highly influential paper (over 45,000 citations on Google Scholar!) because it was one of the first papers to demonstrate impressive performance for a neural network on a modern computer vision benchmark. It generated lots of excitement both in academia and in the tech industry. The architecture presented in this paper widely used today, and is known as "AlexNet", after the first author. Reading this paper will also help you review a lot of the important concepts from this class.

   (a) **[3pts]** They use a conv net architecture which has five convolution layers and three fully connected layers (one of which is the output layer). Your job is to count the number of units, the number of weights, and the number of connections in each layer. I.e., you should complete the following table:

	# Units	# Weights	# Connections
Convolution Layer 1			
Convolution Layer 2			
Convolution Layer 3			
Convolution Layer 4			
Convolution Layer 5			
Fully Connected Layer 1			
Fully Connected Layer 2			
Output Layer			

   You can ignore the pooling layers when doing these calculations, i.e. you don't need to consider the units in the pooling layers or the connections between convolution and pooling layers. You can also ignore the biases. Note that the paper gives you the answers for the numbers of units in the caption to Figure 2. Therefore, we won't mark the column for units, though you would benefit from trying to work it out yourself.

   When counting the number of connections, we'll adopt the convention that when the input to a convolution layer is zero-padded, the connections to the dummy zero values count towards the total. (This is the most convenient way to do it, since it means the number of incoming connections is the same for each unit in a given layer.)

   (b) **[1pt]** Now suppose you're working at a software company and want to use an architecture similar to AlexNet in a product. Your project manager gives you some additional instructions; for each of the following scenarios, based on your answers to Part 1, suggest a change to the architecture which will help achieve the desired objective. I.e., modify the sizes of one or more layers. (These scenarios are independent.)

      i. You want to reduce the memory usage at test time so that the network can be run on a cell phone; this requires reducing the number of parameters for the network.

      ii. Your network will need to make very rapid predictions at test time. You want to reduce the number of connections, since there is approximately one add-multiply operation per connection.