Homework #2 – Neural Network Overview – Activation Functions and Back Propagation
CAP 5619, Deep & Reinforcement Learning (Spring 2020), Department of Computer Science, Florida State University

---

Points: 75
Due: Beginning of the class (11:00am) on Thursday, February 6, 2020

Submission: You need to submit electronically via Canvas by uploading a) a pdf file (named "**hw2-Firstname-Lastname.pdf**") for your answers to the questions, and b) the program(s) you have created (for Problem 3 and other ones) (named as "**hw2-prog-Firstname-Lastname.???**"); if there are multiple program files, please zip them as a single archive. Here replace "Firstname" using your first name and replace "Lastname" using your last name in the file names

The main purpose of this assignment is to let you be familiar with neural network architecture elements including activation functions, and the basic back propagation algorithm.

Problem 1 (20 points) As neural networks are typically trained using (stochastic) gradient descent optimization algorithms, properties of the activation functions affect the learning. Here we divide the domain of an activation function into: 1) fast learning region if the magnitude of the gradient is larger than 0.9, 2) active learning region if the magnitude of the gradient is between 0.01 and 0.9 (inclusive), 3) slow learning region if the magnitude of the gradient is larger than 0 but smaller than 0.01, and 4) inactive learning region if the magnitude of the gradient is 0. For each of the following activation functions, plot its gradient in the range from -10 to 10 and then list the four types of regions. If the gradient is not well defined for an input value, any reasonable value is acceptable.

(1) Rectified linear unit $f(z) = \begin{cases} z & if\ z \geq 0 \\ 0 & otherwise \end{cases}$.
(2) Logistic sigmoid activation function $f(z) = \sigma(z) = \frac{1}{1+e^{-z}}$.
(3) Piece-wise linear unit $f(z) = \begin{cases} 0.05z + 0.95 & if\ z > 1 \\ z & if\ 1 \geq z \geq -1 \\ 0.05z - 0.95 & Otherwise \end{cases}$.
(4) Swish $f(z) = z\sigma(5z)$, where $\sigma(z) = \frac{1}{1+e^{-z}}$.
(5) Exponential Linear Unit (ELU) $f(z) = \begin{cases} z & if\ z \geq 0 \\ 0.5(e^z - 1) & otherwise \end{cases}$. (Note here is a special case of the general ELU function with α=0.5.)

Problem 2 (15 points) Answer the following questions regarding the back propagation algorithm.
(1) (**10 points**) Describe in your words Algorithm 6.1 and Algorithm 6.2 in the Deep textbook (on pages 202 and 204). For each algorithm, you need to describe what each line is doing and why the algorithm would work correctly.
(2) (**5 points**) Explain in your own words why the two algorithms require "the same order of the number of computations". (See page 204 of the textbook for more information.)

Problem 3 (20 points) Using the framework you have established, design and train a neural network to learn how to approximate

$$g(x) = \sin\left(\frac{3\pi}{4}x\right).$$

Note that your neural network should have as few as possible neurons in the hidden layer(s) and the largest error (i.e., the absolute difference between g(x) and the output of your neural network) should be no more than 0.05 in the range of inputs from -1 to 1. You can generate and use no more than 200 training samples.

Problem 4 (**20 points**) The weights and biases of a softmax layer (implemented using equations (6.28) and (6.29) in the Deep textbook) can be downloaded from http://www.cs.fsu.edu/~liux/courses/deepRL/assignments/hw2_softmax_weights.m. (Note that they are given as a Matlab program and you should be able to extract the numbers easily from the file.)

(1) (**2 points**) What is the architecture of the softmax layer? In other words, how many inputs, how many neurons, and how many connections are there in the layer?

(2) **(4 points)** Classify the following example, which can be downloaded from http://www.cs.fsu.edu/~liux/courses/deepRL/assignments/hw2_softmax_sample.txt .

(3) **(8 points)** Suppose that we use cross entropy loss, the learning rate is 0.1, and the correct label for the sample is the first class (i.e., class 0 if you labels the classes as 0, 1, …, 19), after we apply one step gradient decent optimization using the given sample in (2), how many of the weights and biases will be increased, decreased, or remain the same respectively? For numerical stability, we consider that any change smaller than 0.0005 is the same.

(4) **(6 points)** Compute a small vector so that the new sample created by adding the vector to the given sample will be classified as from the second class (i.e., class 1 if you labels the classes as 0, 1, …, 19). The length of the vector (in L_2 norm) should be as small as possible. Explain how you have obtained your solution and confirm that the resulting sample is indeed classified as from the second class.

(5) **(extra credit option 3 points)** As in (4), could you make a small change to the given sample so that it will be classified as from the second class but with the maximal changed value to be minimized? In other words, we like the vector to be as small as possible according to the L_∞ norm. Explain how you have obtained your solution and confirm that the resulting sample is indeed classified as from the second class.

Extra Credit Problem

Problem 5 (8 points) In the paper "Distributed Representations of Words and Phrases and their Compositionality" (which can be downloaded from https://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf), an objective function based on negative sampling is defined (see Equation (4) in the paper).

(1) (**2 points**) Suppose that we use k negative samples to approximate the expectation, write down the objective function for one sample. Make sure that you define/explain all the variables in your function.

(2) (**4 points**) Derive the learning rule using gradient descent optimization for the objective function given in (1).

(3) (**2 points**) Explain the key differences between the learning rule for part (2) and the one we derived in class for the skip gram model; if you think that the two learning rules are equivalent, justify your answer.