Name: _____

We Recur!

7. (12 points) We have seen in class recurrent neural networks (RNNs) that are structured as:

$$z_t^1 = W^{ss} s_{t-1} + W^{sx} x_t$$
$$s_t = f_1(z_t^1)$$
$$z_t^2 = W^o s_t$$
$$p_t = f_2(z_t^2)$$

where we have set biases to zero. Here x_t is the input and y_t the actual output for (x_t, y_t) sequences used for training, with p_t as the RNN output (during or after training).

Assume our first RNN, call it RNN-A, has s_t, x_t, p_t all being vectors of shape 2×1. In addition, the activation functions are simply $f_1(z) = z$ and $f_2(z) = z$.

(a) For RNN-A, give dimensions of the weights:

W^{ss}: ___2×2___ W^{sx}: ___2×2___ W^0: ___2×2___

(b) We have finished training RNN-A, using some overall loss $J = \sum_t Loss(y_t, p_t)$ given the per-element loss function $Loss(y_t, p_t)$. We are now interested in the derivative of the overall loss with respect to x_t; for example, we might want to know how sensitive the loss is to a particular input (perhaps to identify an outlier input). What is the derivative of overall loss at time t with respect to x_t, $\partial J / \partial x_t$, with dimensions 2×1, in terms of the weights W^{ss}, W^{sx}, W^0 and the input x_t? Assume we have $\partial Loss / \partial z_t^2$, with dimensions 2×1. Use $*$ to indicate element-wise multiplication.

Solution:

$$\frac{\partial J}{\partial x_t} = \frac{\partial Loss}{\partial x_t} = \frac{\partial z_t^2}{\partial x_t} \frac{\partial Loss}{\partial z_t^2}$$
$$= \frac{\partial z_t^1}{\partial x_t} \frac{\partial z_t^2}{\partial z_t^1} \frac{\partial Loss}{\partial z_t^2}$$
$$= W^{sxT} W^{oT} \frac{\partial Loss}{\partial z_t^2}$$

Check: $(2 \times 1) = (2 \times 2)(2 \times 2)(2 \times 1)$ matrix dimensions.

Or more generally, for x_t being $(d \times 1)$, s_t and z_t^1 being $(m \times 1)$, and p_t and z_t^2 being $(n \times 1)$, then W^{sx} has dimensions $(m \times d)$ and W^o has dimensions $(n \times m)$. Then the above derivative dimension check is $(d \times 1) = (d \times m)(m \times n)(n \times 1)$ dimensions.

Now consider a modified RNN, call it RNN-B, that does the following:

$$z_t^1 = W^{ssx} \begin{bmatrix} s_{t-1} \\ x_t \end{bmatrix}$$
$$s_t = z_t^1$$
$$z_t^2 = W^{ox} \begin{bmatrix} s_t \\ x_t \end{bmatrix}$$
$$p_t = f_2(z_t^2)$$

where s_t, x_t, p_t are all vectors of shape 2×1, $\begin{bmatrix} s_{t-1} \\ x_t \end{bmatrix}$ and $\begin{bmatrix} s_t \\ x_t \end{bmatrix}$ are vectors of shape 4×1.

(c) For RNN-B, give dimensions of the weights:

W^{ssx}: 2×4 W^{ox}: 2×4

(d) Imagine we are using RNN-B to generate a description sentence given an input word, as in language modeling. The input is a single 2×1 vector embedding, x_1, that encodes the input word. The output will be a sequence of words $p_1, p_2, ..., p_n$ that provide a description of that word. In this setting, what would be an appropriate activation function f_2?

Solution: Softmax to select a best next word.

(e) Continuing with RNN-B for one-to-many description generation using our language modeling approach, we calculate p_1 in a forward pass. How do we calculate x_2 (what is x_2 equal to)?

Solution: $x_2 = p_1$

(f) For RNN-B, we are also interested in the derivative of loss at time t with respect to x_t, $\partial Loss/\partial x_t$. Indicate all of the following that are true about RNN-B, and the derivative of loss with respect to x_t :

√ $\partial Loss/\partial x_t$ **depends on** W^{ox}
√ $\partial Loss/\partial x_t$ **depends on all elements of** W^{ox}
√ $\partial Loss/\partial x_t$ **depends on** W^{ssx}
◯ $\partial Loss/\partial x_t$ depends on all elements of W^{ssx}

Solution: The stacking of s_t and s_{t-1} with x means we need to carry through the differentiation carefully.

$$\frac{\partial Loss}{\partial x_t} = \frac{\partial Loss}{\partial z_t^2} \frac{dz_t^2}{dx_t}$$

Name: _____

Since $z_t^2 = W^{ox} \begin{bmatrix} s_t \\ x_t \end{bmatrix}$ we might think that only the third and fourth columns of W^{ox} multiply by the two elements of x_t and so only a subset of the elements of W^{ox} come into play when we take dz_t^2/dx_t. However, s_t *also* depends on x_t through W^{ssx}, and so all elements of W^{ox} matter when we take dz_t^2/dx_t. However, it actually is the case that only the third and fourth columns of W^{ssx} multiply x_t, so when we carry through the derivative there, only some elements of W^{ssx} matter in this overall gradient.