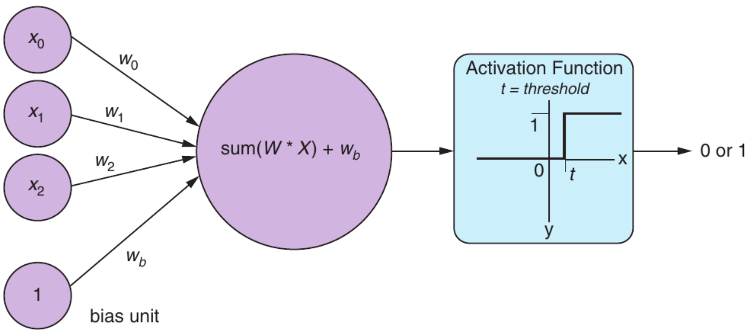
COMP 474/6741 Intelligent Systems (Winter 2024)

Worksheet #8: Neural Networks & Word Embeddings

Task 1. Word analogy questions often appear on standardized tests, like the SSAT, to test language aptitude and reasoning. Here's a simple one (fill in the blank): *Japan is to Sushi what Germany is to* .... Can we solve this type of question with an AI? Stay tuned for the answer!

Task 2. Calculate your first neuron activation for the *Perceptron* (only 100 billion−1 more to go!):



Your input vector $\vec{x} = [0, 1, 1]$ and your weights are $\vec{w} = [0.25, 0.5, 0.75]$.
Activation function:

$$f(\vec{x}) = \begin{cases} 1, & \text{if } \vec{x} \cdot \vec{w} \geq \text{threshold} \\ 0, & \text{otherwise} \end{cases}$$

(use a threshold of 0.5):
$f(\vec{x}) =$ ....

Task 3. Let's train our Perceptron to learn the logical *and* function. Here, we have a two-dimensional input vector and four labeled training examples $l_0, \ldots, l_3$:

	x_0	x_1	$x_0 \wedge x_1$
l_0	1	1	1
l_1	1	0	0
l_2	0	1	0
l_3	0	0	0

Epoch	Input	w_0	w_1	w_2	$f(\vec{x})$	ok?
0	l_0	0	0.2	0.2		
	l_1					
	l_2					
	l_3					
1	l_0					
	l_1					
	l_2					
	l_3					

Note that x_2 is our *bias* (input always 1). Use a threshold for the activation function of 0.5 and set the learning rate $\eta = 0.1$. Train the Perceptron by checking the output for each training sample. Update the weights if there is an error: $w_i' = w_i + \eta \cdot (\textit{label} - \textit{predicted}) \cdot x_i$.

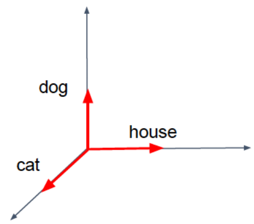
Task 4. Let's compute the *loss* of the Perceptron above at each epoch using mean squared error (MSE) as the cost function:

$$\text{MSE}(x) = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2$$

Loss at end of Epoch 0: .... Epoch 1: ....

Task 5. Here are three words in one-hot vector representation (three words, so three dimensions):

'cat'	=	[0, 0, 1]
'dog'	=	[0, 1, 0]
'house'	=	[1, 0, 0]



What is the *distance* between the one-hot word vectors for (cat, dog) and (cat, house):

....

Using the Euclidian distance, $d(\vec{p}, \vec{q}) = \sqrt{\sum_{i=1}^{n}(p_i - q_i)^2}$

Task 6. Ok, now re-write the question from Task 1 in form of a word vector calculation:

....

Verify with https://www.cs.cmu.edu/~dst/WordEmbeddingDemo/

Task 7. Consider the following sentence: *"the cat drinks the milk"*. We will use this sentence to train a Word2Vec model using the skip-gram approach. Assume that you use a context window of size 1 (1 word before and 1 word after the input word), and your vocabulary only contains the words in the sentence above.

Using only the sentence above, create the training instances using the skip-gram method:

Instance	Input Word	To Predict	Instance	Input Word	To Predict
1			5		
2			6		
3			7		
4			8		

Task 8. Now, (a) encode the vocabulary using one-hot vectors, assuming alphabetical ordering, no stop-word filtering (left) and (b) using these vectors, encode the first three training instances above as input vectors for the network:

Word	One-Hot Vector			
cat	1			
drinks				
milk				
the				

Instance	Context	Word	One-Hot Vector			
1	Input					
	Output					
2	Input					
	Output					
3	Input					
	Output					

Task 9. Compute the *softmax* function σ on the vector v below:

$$\sigma(v)_j = \frac{e^{v_j}}{\sum_{k=1}^{K} e^{v_k}} \qquad v = \begin{bmatrix} 0.5 \\ 0.9 \\ 0.2 \end{bmatrix}, \sigma(v) = \begin{bmatrix} \dots\dots\dots \\ \dots\dots\dots \\ \dots\dots\dots \end{bmatrix}$$

(e = *Euler's* number ≈ 2.71828)