1. Probability

Part 1: (c), 0.20

To find the unigram probability of dog, we first take counts of the unigrams in the training sentences. As per the instructions, we can ignore the start symbol <s> for the unigrams:

Unigram	Count
dog	4
chases	3
cat	6
bites	2
</s> 5	5

We have 4 instances of dog out of 20 unigrams, so p(dog) = 4/20 = **0.20**

Part 2: (e), 0.33 (Give ½ point if they circled (e) but didn't fill in any value. 0 points for circling (e) and filling in the wrong value.)

The bigram probability of chases dog can be expressed as the conditional probability:

$$p(dog \mid chases) = \frac{p(chases, dog)}{p(chases)}$$

We have 3 instances of chases in the training data, only 1 of which is followed by dog = 1/3 = **0.33**

Part 3:
We first find the unigram and bigram probabilities of the sentence. Note that for the unigrams, we are ignoring the start symbol but not the stop symbol:

Unigrams
:

p(dog) ∗ p(chases) ∗ p(dog) ∗ p(</s>)
= 4/20 ∗ 3/20 ∗ 4/20 ∗ 5/20

= 3/2000 = **0.0015**

Bigrams
:

p(dog | <s>) ∗ p(chases | dog) ∗ p(dog | chases) ∗ p(</s>) | dog)
= 2/5 ∗ 1/4 ∗ 1/3 ∗ 2/4

= 1/60 = **0.0166**

Next, we apply linear interpolation smoothing:

0.0015 ∗ 0.2 = **0.0003**
0.0166 ∗ 0.8 =
0.0133

Finally, we add the interpolated values:
0.0003 + 0.0133 = **0.0136**

Alternate method: do MLE/smoothing for each word
0.8*2/5 + 0.2*4/20 + 0.8*1/4 + 0.2*3/20 + 0.8*1/3 + 0.2*4/20 + 0.8*2/4 + 0.2*5/20 = **0.0114**

2. Hidden Markov Model

P (at) = P(at|S1,S1) + P (at|S1,S2)
*= 1.0 * 0.2 * 0.8 * 0.1*
*+ 1.0 * 0.2 * 0.2 * 0.2*
= 0.024

P(act) = P(at | S1, S1, S1) + P(at | S1, S1, S2) + P(at | S1, S2, S1) + P(at | S1, S2, S2) = 0.01512

Alternatively, still count it correct if they start at 'S1' rather than 'Start':

P (at) = P(at|S1,S1) + P (at|S1,S2) + P(at|S2,S1) + P(at|S2,S2)
*= 0.8 * 0.2 * 0.8 * 0.1*
*+ 0.8 * 0.2 * 0.2 * 0.2*
*+ 0.2 * 0.5 * 0.6 * 0.1*
*+ 0.2 * 0.5 * 0.4 * 0.2*
= 0.0332

P(act) = 0.018816

3. N-Gram Models

	k = 1	k = 2	k = 3	k = 4
v	Bats (V/N)	are (V)	cool (V/N/Adj)	. (STOP)
V	-0.1 -6 = -6.1	-6.1-0.05-4 = -10.15	-10.15 – 2 – 7 = -19.15	
N	-0.1 – 6 = -6.1		-10.15 - .3 – 8 = -18.45	
Adj			-10.15 - .7 – 4 = -14.85	
STOP				-14.85-6=-20.85

N V Adj STOP

-20.85

b) Trigram contains more dependencies/might incorporate more information.

4. Training a Neural Network

Loss Function:

$$L = \frac{1}{n}\sum_{i=1}^{n} - log\, p(Y = Y_i|X_i) + \frac{\lambda}{2}(\|W_1\|^2 + \|W_2\|^2 + \|W_3\|^2)$$

$$Letp(Y = Y_i|X_i) = P_k = \frac{e^{V_k}}{\sum_j e^{V_j}} where V_k = (W_3 h_2 + b_3)_k, v = W_3 h_2 + b_3$$

$$\frac{\partial L}{\partial V_k} = \frac{\partial L}{\partial P_k} * \frac{\partial P_k}{\partial V_k} = \left(\frac{1}{n}\Sigma - \frac{1}{P_k}\right) * \left(\frac{\sum_j e^{V_j} * e^{V_k} - (e^{V_k})^2}{\left(\sum_j e^{V_j}\right)^2}\right)$$
$$= \left(\frac{1}{n}\Sigma - \frac{1}{P_k}\right) * (P_k - (P_k)^2) = \frac{1}{n}\Sigma(P_k - 1)$$

$$\frac{\partial L}{\partial W_3} = \frac{\partial L}{\partial V_k} * \frac{\partial V_k}{\partial W_3} + \lambda\|W_3\| = \frac{1}{n}\Sigma(P_k - 1){h_2}^T + \lambda\|W_3\|$$
$$\frac{\partial L}{\partial b_3} = \frac{\partial L}{\partial V_k} * \frac{\partial V_k}{\partial b_3} = \frac{1}{n}\Sigma(P_k - 1)$$
$$\frac{\partial L}{\partial h_2} = \frac{\partial L}{\partial V_k} * \frac{\partial V_k}{\partial h_2} = \frac{1}{n}\Sigma {W_3}^T (P_k - 1), \frac{\partial L}{\partial h_2} = 0 if h_2 < 0$$

$$\frac{\partial L}{\partial W_2} = \frac{\partial L}{\partial h_2} * \frac{\partial h_2}{\partial W_2} + \lambda\|W_2\| = \frac{1}{n}\Sigma {W_3}^T (P_k - 1){h_1}^T + \lambda\|W_2\|$$
$$\frac{\partial L}{\partial b_2} = \frac{\partial L}{\partial h_2} * \frac{\partial h_2}{\partial b_2} + \lambda\|W_2\| = \frac{1}{n}\Sigma {W_3}^T (P_k - 1)$$
$$\frac{\partial L}{\partial h_1} = \frac{\partial L}{\partial h_2} * \frac{\partial h_2}{\partial h_1} = \frac{1}{n}\Sigma {W_2}^T {W_3}^T (P_k - 1), \frac{\partial L}{\partial h_1} = 0 if h_1 < 0$$

$$\frac{\partial L}{\partial W_1} = \frac{\partial L}{\partial h_1} * \frac{\partial h_1}{\partial W_1} + \lambda\|W_1\| = \frac{1}{n}\Sigma {W_2}^T {W_3}^T (P_k - 1)x^T + \lambda\|W_1\|$$
$$\frac{\partial L}{\partial b_1} = \frac{\partial L}{\partial h_1} * \frac{\partial h_1}{\partial b_1} = \frac{1}{n}\Sigma {W_2}^T {W_3}^T (P_k - 1)$$

5. Word Embeddings

A. b = w - c + a OR w-a+c //depending on how you think about it

B. [[w2]] \subseteq [[w1]]

C. No. *Mammal* is a hypernym of *cat*, but if you are in a pet store, you are likely to refer to cats, but not likely to refer to mammals.

D. Word2vec embeddings assume that a word's semantics can be entirely inferred from its distribution, so its notion of semantic similarity is really distributional similarity. But, by C, we have seen how the distribution of a word and its hypernym are not necessarily similar. Thus, we should not expect to necessarily see semantic similarity between words and their hypernyms.

6. Sentence Similarity

(1) What is the Jaccard similarity of D1 and D2 if considering documents as sets of unigrams? You can leave your answer as fraction form.

Jaccard(D1,D2) = __2 / 6__

intersection: dogs, cats
union: dogs, chase, cats, I, love, and
Jaccard similarity = intersection / union

(2) What is the Jaccard similarity of D2 and D3 if considering documents as sets of bigrams?

Jaccard(D2,D3) = ___1 / 6_____

intersection: I love
union: I love, love cats, cats and, and dogs, love computer, computer science

(3) Fill out the term-document matrix below, considering the vocabulary = {dogs, chase, cats, love, computer, science}.

	dogs	chase	cats	love	computer	science
D1	1	1	1	0	0	0
D2	1	0	1	1	0	0
D3	0	0	0	1	1	1

(4) What is the cosine similarity of D1 and D2 based on term-document matrix above?

cosine(D1,D2) = ___2/3_____

2 / (sqrt(3)* sqrt(3))

(5) What is the euclidean distance of D2 and D3 based on term-document matrix above?

euclidean(D2,D3) = ___2_____

sqrt(1+1+1+1)

(6) Fill out the term-term matrix below, considering target words = {chase,love} and context words = {dogs,cats,computer,science}. Each cell represents the number of times the two words appear in the same document.

	dogs	cats	computer	science
chase	1	1	0	0
love	1	1	1	1

(7) Calculate the PPMI(w = chase, c = cats) and PPMI(w = chase, c = computer). You can leave your answer in log form.

PPMI(w = chase, c = cats) = __log_2(1.5)___
PPMI(w = chase, c = computer) = ___0___

log_2 ((1 / 6) / (2/6 * 2/6)) = log_2(1.5)

log_2 ((0 / 6) / (2/6 * 1/6)) = -infinity -> PPMI = 0

7. Noisy Channel Model

a.

P(*cat*) = 6 / 42
P(*car*) = 4 / 42
P(*as)* = 3 / 42

(if they counted <s>, their denominator will be 47)

b.

Substitution:
sub[s, t]: 6/14
sub[s, r]: 4/14

Insertion:
ins[c, a]: 4/14 (also count 3/14)

c.

argmax { P(cat | cas) P(cat) , P(car | cas) P(car) , P(cas | cas) P(cas) }
argmax { 6/42*6/14 , 4/42*4/14 , 3/42*4/14 }
argmax { 0.06 , 0.027 , 0.02}

0.06 is the max, so **cat**

8. Activation Functions

Rewrite tanh(x) as sigmoid as per: https://sebastianraschka.com/faq/docs/tanh-sigmoid-relationship.html

Use this to rewrite f(x) as f'(x)

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

$$= \frac{e^x + e^{-x} - 2e^{-x}}{e^x + e^{-x}}$$

$$= 1 + \frac{-2e^{-x}}{e^x + e^{-x}}$$

$$= 1 - \frac{2}{e^{2x} + 1}$$

Now, from the logistic sigmoid's perspective, we have:

$$\tanh(x) = 1 - \frac{2}{e^{2x} + 1} = 1 - 2\sigma(-2x)$$
$$= 1 - 2(1 - \sigma(2x))$$
$$= 1 - 2 + 2\sigma(2x)$$
$$= 2\sigma(2x) - 1$$