CS 7650 : Midterm

1) WORD EMBEDDINGS

(1) $$\bar{v}_m = \frac{1}{2h} \sum_{n=1}^{h} \left(v_{w_{m+n}} + v_{w_{m-n}} \right)$$

$$\bar{v}_m = \frac{1}{2(3)} \cdot \sum_{n=1}^{3} \left(v_{w_{m+n}} + v_{w_{m-n}} \right)$$

$$\bar{v}_3 = \frac{1}{6} \cdot \sum_{n=1}^{3} \left(v_{w_{3+n}} + v_{w_{3-n}} \right)$$

$$= \frac{1}{6} \cdot \left[v_{w_4} + w_{w_2} + v_{w_5} + v_{w_1} + v_{w_6} + v_{w_0} \right]$$

$$= \frac{1}{6} \left[[5,-1] + [-2,0] + [2,-3] + [3,2] + [2,1] + [2,1] \right]$$

$$= \frac{1}{6} \cdot [12,0]$$

$$\boxed{\bar{v}_3 = [2,0]}$$

(2)

word	embedding	unnormalized score	normalized score
dog	[1, 2]	2	0.010
horse	[3,4]	6	0.165
motorcycle	[0, -1]	0	0.003
leopard	[3,0]	6	0.165
wolf	[4,0]	8	0.658

I have
Here, ~~I~~ used these formulae to find the scores:

- unnormalized score = $u_i \cdot v_j$
- normalized score = $\frac{2^{z_i}}{\sum_j 2^{z_j}}$

(3) The CBOW algorithm would predict "wolf" as the missing word, since it has the maximum normalized score.

SHOW WORK

$[2,0]\begin{bmatrix}1\\2\end{bmatrix} = 2$

$[2,0]\begin{bmatrix}3\\4\end{bmatrix} = 6$

$[2,0]\begin{bmatrix}0\\-1\end{bmatrix} = 0$

$[2,0]\begin{bmatrix}3\\0\end{bmatrix} = 6$

$[2,0]\begin{bmatrix}4\\0\end{bmatrix} = 8$

$\frac{2^2}{389} = 0.010$

$\frac{2^6}{389} = 0.165$

$\frac{1}{389} = 0.003$

$\frac{2^6}{389} = 0.165$

$\frac{2^8}{389} = 0.658$

2) LSTMs

(1) $f_{m+1} = \sigma(\theta^{h\to f} h_m + \theta^{x\to f} x_{m+1} + b_f)$
$i_{m+1} = \sigma(\theta^{h\to i} h_m + \theta^{x\to i} x_{m+1} + b_i)$
$\tilde{c}_{m+1} = \tanh(\theta^{h\to c} h_m + \theta^{w\to c} x_{m+1})$

LSTM equations

$c_{m+1} = f_{m+1} \odot c_m + i_{m+1} \odot \tilde{c}_{m+1}$

$o_{m+1} = \sigma(\theta^{h\to o} h_m + \theta^{x\to o} x_{m+1} + b_o)$
$h_{m+1} = o_{m+1} \odot \tanh(c_{m+1})$

Elman RNN

$h_t = \tanh(wx_t + Vh_{t-1} + b_t)$
$y_t = \tanh(uh_t + b_y)$

By setting $f_{m+1} = 0$, $i_{m+1} = 1$ and $o_{m+1} = 1$, the hidden state at the next time step will only depend on the previous hidden state and the ~~nex~~ current input. This effectively turns the LSTM into an Elman RNN.
h_{m+1} only depends on h_m and x_{m+1}, and it doesn't depend on c_m.

(2.) $\tilde{c}_{m+1}$ (the candidate memory update in an LSTM) most closely resembles the hidden state of a standard Elman RNN (h_t).
Both multiply the previous hidden state and the current input with parameters, sum them up ($\tilde{c}_{m+1}$ also adds a bias 'bi') and take the tanh(). of this.

(3) In order to create vector gates, the parameters $\theta^{(h \to f)}$, $\theta^{(x \to f)}$, b_f, $\theta^{(h \to i)}$, $\theta^{(x \to i)}$, b_i, $\theta^{(h \to o)}$, $\theta^{(x \to o)}$, b_o would have to be changed. The following table shows what the dimensions of these parameters would have to be.

Parameter	Dimensions
$\theta^{(h \to f)}$	(3, 3)
$\theta^{(x \to f)}$	(3, 3)
b_f	(3, 1)
$\theta^{(h \to i)}$	(3, 3)
$\theta^{(x \to i)}$	(3, 3)
b_i	(3, 1)
$\theta^{(h \to o)}$	(3, 3)
$\theta^{(x \to o)}$	(3, 3)
b_o	(3, 1)

The benefit of having vector gates instead of scalars is that each "unit" in c_m, $\tilde{c}_{m+1}$ and $\tanh(c_{m+1})$ can be assigned a separate "weight". Using scalar gates imposes a constraint that the "weight" applied to each "unit" must be the same.

As a result, having vector gates gives us the ability to forget certain units, while remembering other units in the same vector. This will allow the LSTM to learn better.

(4) The inclusion of the memory cell c_{t+1} improves the following problems in RNNs:

(i) Vanishing gradients: gradients won't decay to 0 during backprop as fast.

(ii) Exploding gradients: gradients won't explode to ∞ during backprop as fast

(iii) ~~longer~~ short "memory": information can propagate in the hidden state across several time steps, without attenuating a lot

The vanishing gradients problem is alleviated since a recurrence function isn't repeatly applied while computing c_{t+1}.

The short memory problem is alleviated since we use forget and input gates to determine how much ~~of the~~ previous cell state should contribute to the next memory cell state.

3) Beam search Decoding.

(1) Prefix string : " <s> I know"
p(prefix [redacted]) = 1 , k=2

↳ 1st decoding step.
only 2 options : ',' and 'the'

- p(<s> I know, [redacted]) = p(<s> I know). p(, | know)
= 1 × 0.4
= 0.4
- p(<s> I know the) = p(<s> I know). p(the | know)
= 1 × 0.6
= 0.6

∴ beam = [<s> I know , [redacted] , "<s> I know the"]
0.4 0.6

↳ 2nd decoding step.

✓ : p(<s> I know, I) = p(<s> I know,) · p(I|,)
= 0.4 × 0.8 = 0.32
- p(<s> I know, it) = p(<s> I know,) · p(it|,)
= 0.4 × 0.2 = 0.08
✓ · p(<s> I know the correct) = p(<s> I know the). p(correct | the)
= 0.6 × 0.5 = 0.3
- p(<s> I know the important) = p(<s> I know the). p(important | the)
= 0.6 × 0.3 = 0.18

P(<s> I know the exact) = P(<s> I know the) · P(exact | the)
= 0.6 × 0.2 = 0.12

∴ beam = ["<s> I know, I" ; "<s> I know the correct"]
↓ 0.32 ↓ 0.30

↳ 3rd decoding step.

· P(<s> I know, I will) = P(<s> I know, I) · P(will | I)

· ~~P(<s> I know~~ = 0.32 × 0.4 = 0.128

· P(<s> I know, I know) = P(<s> I know, I) · P(know | I)
= 0.32 × 0.6 = 0.192

· P(<s> I know the correct response) = P(<s> I know the correct) · P(response | correct)
= 0.3 × 0.6 = 0.18

· P(<s> I know the correct answer) = P(<s> I know the correct) · P(answer | correct)
= 0.3 × 0.25 = 0.075

· P(<s> I know the correct problems) = P(<s> I know the correct) · P(problem | correct)
= 0.3 × 0.15 = 0.045

∴ beam = ["<s> I know, I know", "<s> I know the correct response"]
↓ 0.192 ↓ 0.18

Most likely sequence : <s> I know, I know

(a) number of output sequences that can be generated = 7

(b) The sequence with the maximum probability in S is "<s> I know, I know" with probability 0.192

(c.) The sequence with the minimum probability in S is "<s> I know the with probability 0.075 correct answer"

(2) (a) prefix string: "<s> I know"

$P(\text{<s> I know}) = 1$

$k = 2$

$t=1$

$P(\text{<s> I know}) \cdot P(\text{word} \mid \text{know})$

, → 0.4

the → 0.6

$t=2$

$P(\text{<s> I know,}) \cdot P(\text{word} \mid ,)$

I → 0.32

it → 0.08

$t=3$

$P(\text{<s> I know, I}) \cdot P(\text{word} \mid I)$

will → 0.128

know → 0.192

$P(\text{<s> I know, it}) \cdot P(\text{word} \mid \text{it})$

was → 0.08

DON'T CONSIDER SINCE NOT TOP K

$P(\text{<s> I know the}) \cdot P(\text{word} \mid \text{the})$

exact → 0.12

correct → 0.3

important → 0.17

$P(\text{<s> I know the exact}) \cdot P(\text{word} \mid \text{exact})$

answer → 0.12

$P(\text{<s> I know the important}) \cdot P(\text{word} \mid \text{imp})$

$P(\text{<s> I know the correct}) \cdot P(\text{word} \mid \text{correct})$

response → 0.18

answer → 0.075

problem → 0.045

DON'T CONSIDER SINCE NOT TOP-K

response → 0.09

answer → 0.09

4) Evaluation

(1) Barack and Michele Obama attend the WHCD

Predicted NER tags	O	O	B-PER	I-PER	O	O	B-LOC
Gold NER tags	B-PER	O	B-PER	I-PER	O	O	O

event at the Hilton Hotel in Washington.

	O	O	O	B-LOC	O	O	B-LOC
	O	O	O	B-LOC	I-LOC	O	B-LOC

True Positives : 2 (Michele Obama, Washington)
False Positives : 2 (WHCD, Hilton)
False Negatives : 2 (Barack, Hilton Hotel)
True Negatives : 1

$$\text{Precision} = \frac{TP}{TP + FP} = \frac{2}{4} = 0.5$$

$$\text{Recall} = \frac{TP}{TP + FN} = \frac{2}{4} = 0.5$$

$$\text{F1-score} = \frac{2 \cdot p \cdot r}{(p + r)} = 0.5$$

(2)

n-gram precision

$$p_n = \frac{\sum_{ngrams \in C} \min\left(\max_{i=1 \to K} count_{r_i}(ngram), count_C(ngram)\right)}{\sum_{ngrams \in C} count_C(ngram)}$$

$$BP = \begin{cases} 1 & \text{if } len(c) \geq \min_{i=1 \to K} len(r_i) \\ \exp\left(1 - \frac{len(r)}{len(c)}\right) & \text{otherwise} \end{cases}$$

$$BLEU = BP \exp\left[\sum_{n=1}^{N} (w_n \cdot \ln p_n)\right]$$

where $\sum_n w_n = 1$

c_1 : fruits are good for health
c_2 : vegetables are very important for health
&
r_1 : eating fruits is good for health
r_2 : fruits and vegetables are essential for good health.

$w_1, w_2, w_3 = 1/3$, $w_4 = 0$

~~P1~~

- computing the BLEU score for c_1

↳ $p_1 = \dfrac{\min(1,1) + \min(1,1) + \min(1,1) + \min(1,1) + \min(1,1)}{5}$

$= 1$

↳ $p_2 = \dfrac{\min(0,1) + \min(0,1) + \min(1,1) + \min(1,1)}{4}$

$= 1/2$

↳ $p_3 = \dfrac{\min(0,1) + \min(0,1) + \min(1,1)}{3}$

$= 1/3$

↳ $len(c) = 5,\ len(r) = 6$

$BP = \exp\left(1 - \frac{6}{5}\right) = \exp(-0.2)$

$= 0.818$

↳ $BLEU = 0.818 \cdot \exp\left[\frac{1}{3}\cdot\ln(1) + \frac{1}{3}\cdot\ln(0.5) + \frac{1}{3}\cdot\ln(1/3)\right]$

$= 0.818 \cdot e^{-0.597}$

$= 0.818 \times 0.55$

$BLEU = \boxed{0.45016}$

- computing the BLEU score for C_2

↳ $p_1 = \dfrac{\min(1,1) + \min(1,1) + \min(0,1) + \min(0,1) + \min(1,1) + \min(1,1) + \min(1,1)}{7}$

$= 5/7$

↳ $p_2 = \dfrac{\min(1,1) + \min(0,1) + \min(0,1) + \min(0,1) + \min(1,1) + \min(1,1)}{6}$

$= 1/2$

↳ $p_3 = \dfrac{\min(0,1) + \min(0,1) + \min(0,1) + \min(0,1) + \min(1,1)}{5}$

$= 1/5$

↳ $len(c) = 7, \quad len(r) = 6$

$BP = 1$

↳ $BLEU = 1 \cdot \exp\left[\frac{1}{3} \cdot \ln\frac{5}{7} + \frac{1}{3} \cdot \ln\frac{1}{2} + \frac{1}{3} \cdot \ln 1/5 + 0\right]$

$= e^{-0.879}$

$= \boxed{0.4149}$

5) TRANSFORMER SELF ATTENTION

(1) $q_i = W^Q x_i$
$k_i = W^K x_i$
$v_i = W^V x_i$

$$score(x_i, x_j) = \frac{q_i \cdot k_j}{\sqrt{d_k}}$$

$$\alpha_{i,j} = softmax(score(x_i, x_j)) \quad \forall j \leq i$$

$$y_i = \sum_{j \leq i} \alpha_{i,j} \cdot v_j$$

	Attention	is	all	you	need.
Attention	-2.25	-1.916	1.541	6.0416	4.8125
is	-0.625	-0.625	0.625	2.0833	1.5625
all	-2.9375	-1.77	0.458	4.6875	4.5
you	-2.5	-0.833	-1.04	1.0416	2.1875
need	-0.9375	0.229	-1.541	-1.9791	-0.5

NORMALIZED SCORES

WORK

X → (N, D) matrix of word embeddings

$Q = XW^Q$
$K = XW^K$
$V = XW^V$

normalized scores = $\frac{QK^T}{\sqrt{d_k}}$

(2)

	Attention	is	all	you	need.
Attention	1.92×10^{-4}	2.68×10^{-4}	8.51×10^{-3}	7.66×10^{-1}	2.24×10^{-1}
is	3.4×10^{-2}	3.4×10^{-2}	1.18×10^{-1}	5.1×10^{-1}	3.63×10^{-1}
all	2.64×10^{-4}	8.49×10^{-4}	7.89×10^{-3}	5.41×10^{-1}	4.49×10^{-1}
you	6.5×10^{-3}	3.44×10^{-2}	2.79×10^{-2}	2.24×10^{-1}	7.06×10^{-1}
need	1.50×10^{-1}	4.82×10^{-1}	8.20×10^{-2}	6.29×10^{-2}	2.32×10^{-1}

Attention values = $\text{softmax}\left(\frac{Q \cdot K^T}{48}\right)$

(3)

Attention	11.254	1.759
is	10.729	1.678
all	7.421	3.560
you	2.922	5.488
need	8.082	0.331

attention head outputs = attention values . V

(4)

Attention	-60.952	28.447	-75.822
is	-47.750	53.749	-49.640
all	13.516	58.943	10.976
you	-9.153	24.379	-16.573
need	-27.528	51.277	-46.528

output of self attention layer = (concat output of each head) . W^O

6) BYTE PAIR ENCODING

Training corpus: fresh_, french_, fries_

step 1:
corpus:
1 fresh_
1 french_

step 1:
corpus
1 f r e n c

- step 1:
  - corpus:
    1 f r e s h _
    1 f r e n c h _
    1 f r i e s _
    merge f, r
  - vocabulary:
    _, f, r, e, s, h, n, c, i

- step 2
  - corpus:
    1 fr e s h _
    1 fr e n c h _
    1 fr i e s _
    merge fr, e (tie with e, s and h, _)
  - vocabulary:
    _, f, r, e, s, h, n, c, i, fr

- step 3
  - corpus
    1 fre s h _
    1 fre n c h _
    1 fr i e s _
    merge h, _
  - vocab
    _, f, r, e, s, h, n, c, i, fr, fre

· Step 4

- corpus:

1 fre s h-
1 fre n c h-
1 fr i e s -

- vocab:

-, f, r, e, s, h, n, c, i,
fr, fre, h-

No bigram occurs more than once