CSC485 A1

Haolin Fan

October 11, 2022

1.

(a)

Step	Stack	Buffer	New dep	Transition
0	[ROOT]	[To, ask, those, questions, is, to, answer, them]		
1	[ROOT, To]	[ask, those, questions, is, to, answer, them]		SHIFT
2	[ROOT, To, ask]	[those, questions, is, to, answer, them]		SHIFT
3	[ROOT, ask]	[those, questions, is, to, answer, them]	ask $\xrightarrow{\text{mark}}$ to	LEFT-ARC
4	[ROOT, ask, those]	[questions, is, to, answer, them]		SHIFT
5	[ROOT, ask, those, questions]	[is, to, answer, them]		SHIFT
6	[ROOT, ask, questions]	[is, to, answer, them]	questions $\xrightarrow{\text{det}}$ those	LEFT-ARC
7	[ROOT, ask]	[is, to, answer, them]	ask $\xrightarrow{\text{dobj}}$ questions	RIGHT-ARC
8	[ROOT, ask, is]	[to, answer, them]		SHIFT
9	[ROOT, is]	[to, answer, them]	is $\xrightarrow{\text{csubj}}$ ask	LEFT-ARC
10	[ROOT, is, to]	[answer, them]		SHIFT
11	[ROOT, is, to, answer]	[them]		SHIFT
12	[ROOT, is, answer]	[them]	answer $\xrightarrow{\text{mark}}$ to	LEFT-ARC
13	[ROOT, is, answer, them]	[]		SHIFT
14	[ROOT, is, answer]	[]	answer $\xrightarrow{\text{dobj}}$ them	RIGHT-ARC
15	[ROOT, is]	[]	is $\xrightarrow{\text{xcomp}}$ answer	RIGHT-ARC
16	[ROOT]	[]	ROOT $\xrightarrow{\text{root}}$ is	RIGHT-ARC

(b)

In 2n steps. Over the entire parse, n steps would be spent shifting each word from the buffer to the stack at some point and another n steps would be spent arc'ing each word out of the stack at some point.

2.

(a)

The prior algorithm is insufficient because it can only ever apply arc transitions between words in the top two positions of the stack.

Non-projective dependency trees will have same word i with a gap degree > 0. That means there exists a word j that is a descendant of i and word a that is not a descendant of i where $min(i, j) < a < max(i, j)$, and a has an arc to another word b where $b < min(i, j)$ or $b > max(i, j)$.

If $min(i, j) < a < max(i, j) < b$, we won't be able to apply the arc transition connecting i and j until b enters the stack for us to apply the arc transition connecting a and b but that can't be done because $max(i, j)$ would exist above a and below b in the stack. There algorithm is therefore stuck.

If $b < min(i, j) < a < max(i, j)$, we won't be able to apply the arc transition connecting a and b and until $max(i, j)$ enters the stack for us to apply the arc transition connecting i and j but that can't be done because a would exist above $min(i, j)$ and below $max(i, j)$ in the stack. There algorithm is therefore stuck.

(c)

To ask those questions is to answer them. Max number of gaps among all words is 0 so gap degree of this tree is 0.

Word	Index	{Dependents}	{Dependents $\cup$ Index} (Ordered)	{Gaps}
ROOT	0	5, 2, 7, 1, 4, 6, 8, 3	0, 1, 2, 3, 4, 5, 6, 7, 8	
To	1		1	
ask	2	1, 4, 3	1, 2, 3, 4	
those	3		3	
questions	4	5	4, 5	
is	5	2, 7, 1, 4, 6, 8, 3	1, 2, 3, 4, 5, 6, 7, 8	
to	6		1	
answer	7	6, 8	6, 7, 8	
them	8		8	

John saw a dog yesterday which was a Yorkshire Terrier. Max number of gaps among all words is 1 so gap degree of this tree is 1.

Word	Index	{Dependents}	{Dependents $\cup$ Index} (Ordered)	{Gaps}
ROOT	0	2, 1, 3, 4, 6, 5, 7	0, 1, 2, 3, 4, 5, 6, 7	
John	1		1	
saw	2	1, 3, 4, 6, 5, 7	1, 2, 3, 4, 5, 6, 7	
a dog	3	6, 5, 7	3, 5, 6, 7	4
yesterday	4		4	
which	5		5	
was	6	5, 7	5, 6, 7	
a Yorkshire Terrier	7		7	

(e)

A uniform distribution $U_{[a,b]}$ has $\mu = \frac{1}{2}(a+b)$ and $\sigma^2 = \frac{1}{12}(b-a)^2$

If $\mu = 0$ we have:

$$\begin{aligned} \frac{1}{2}(a+b) &= 0 \\ a+b &= 0 \\ a &= -b \end{aligned}$$

If $\sigma^2 = \frac{2}{m}$ we have:

$$\begin{aligned} \frac{1}{12}(b-a)^2 &= \frac{2}{m} \\ (b-a)^2 &= \frac{24}{m} \\ b-a = &= \sqrt{\frac{24}{m}} \\ b-(-b) &= \sqrt{\frac{24}{m}} \\ 2b &= \sqrt{\frac{24}{m}} \\ b &= \frac{1}{2}\sqrt{\frac{24}{m}} \end{aligned}$$

$$a = -b = -\frac{1}{2}\sqrt{\frac{24}{m}}$$

(j)

While making the final arc prediction, the set of arguments includes all possible edges (arcs) between words (nodes) in the sentence graph. In this context, our desired return is a tree to represent the sentence's dependency tree so the final prediction must include all the edges required to make a connected tree that includes all the nodes without any cycles. Using argmax simply returns the edges with the greatest score but does not guarantee that these edges form a single connected tree nor a tree with all the nodes nor a tree without any cycles. As such, if argmax was used we might get one or more separate dependency trees, where each tree might be a proper subset of all the words from the sentence, and where each tree might have some word that is a descendant of itself.

(g)

Because in the arc scorer each arc score is a single number whereas in the label scorer each score is a vector.

(h)

Because we have to account for the bias twice, once for when the input is treated as head and once for when the input is treated as dependent.