Assignment 3
Baktash Ansari
NLP

Question 1 :

a.i)

Based on hyper parameter β , which has a value between 0.9 and 0.999 our next time step m has most value of previous time step (between 0.9 * m and 0.999 * m) so the gradient function can't effects highly on the value of m and hyper parameter β prevents gradient to makes high variance. This low variance may helpful to learning because effect of high variance can leads to jump from local optimum values, but low variance helps to move slightly so we can find local optimum better and learning better.

a.ii)

Dividing updates by $\sqrt{v}$ in third formula shows us that the new value of theta will leads to larger update when the value of V becomes smaller, because the value of $\frac{m}{\sqrt{v}}$ becomes larger and theta value changes more. And it will leads to smaller update when the value of V becomes bigger. The reason of this approach is that parameters that have frequent and large gradients (i.e high variance in the gradient) will have a small learning rate to ensure that the updates are not too large and cause instability. On the other hand, parameters that have infrequent and small gradients (i.e low variance in the gradient) will have a large learning rate to speed up the convergence process.

b.i)

Based on formula we have :

$E_{pdrop}[h_{drop}] = h \rightarrow E_{pdrop}[yd \odot h] = h \rightarrow y.E_{pdrop}[d] \odot E_{pdrop}[h] = h$
Since $E[d] = (1 - p_{drop})$ and $E[h] = h \rightarrow y.(1-p_{drop}).h = h \rightarrow$
$y = 1/(1 - p_{drop})$

b.ii)

Dropout should applied during training to prevent overfitting in neural network.
It randomly drops out some of the units in the hidden layer which leads to learn more robust representations that are less dependent on any single unit.This prevents network from memorizing the trained data and improve abilities to generalize for new data.

On the other hand dropout should not be applied during evaluation because it introduces randomness into the network's predictions. During evaluation, we want the network to make deterministic predictions that are not affected by any random noise or dropout.

Question 2

a)

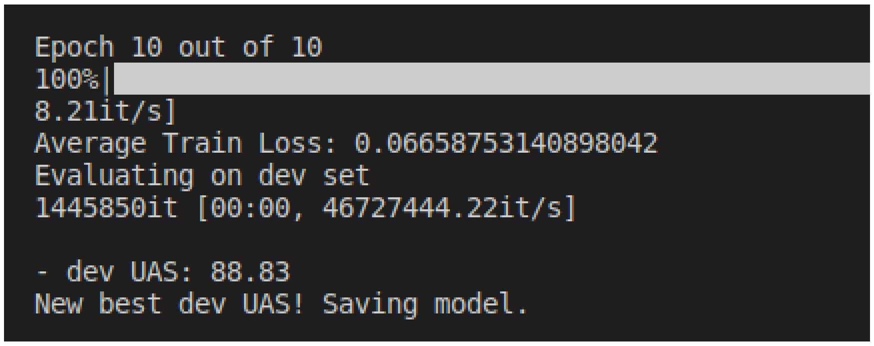
Stack	Buffer	New dependency	Transition
[ROOT]	[I, attended, lectures, in, the, NLP, class]	-	Initial Configuration
[ROOT, I]	[attended, lectures, in, the, NLP, class]	-	SHIFT
[ROOT, I, attended]	[lectures, in, the, NLP, class]	-	SHIFT
[ROOT, attended]	[lectures, in, the, NLP, class]	attended →I	LEFT-ARC
[ROOT, attended,lectures]	[in, the, NLP, class]	-	SHIFT
[ROOT, attended]	[in, the, NLP, class]	Attended → lectures	RIGHT-ARC
[ROOT, attended,in]	[the, NLP, class]	-	SHIFT
[ROOT, attended,in,the]	[NLP, class]	-	SHIFT
[ROOT, attended,in,the,NLP]	[class]	-	SHIFT
[ROOT, attended,in,the,NLP, class]	[ ]	-	SHIFT
[ROOT, attended,in,the,class]	[ ]	Class → NLP	LEFT-ARC
[ROOT, attended,in,class]	[ ]	Class → the	LEFT-ARC

[ROOT, attended,class]	[ ]	Class → in	LEFT-ARC
[ROOT, attended]	[ ]	Attended → class	RIGHT-ARC
[ROOT]	[ ]	ROOT →attended	RIGHT-ARC

b)

We have n steps for adding each of n words in Stack and n steps for removing each of n words from Stack and create a new dependency (the last word has dependency with ROOT token) so we should apply **2n steps.**

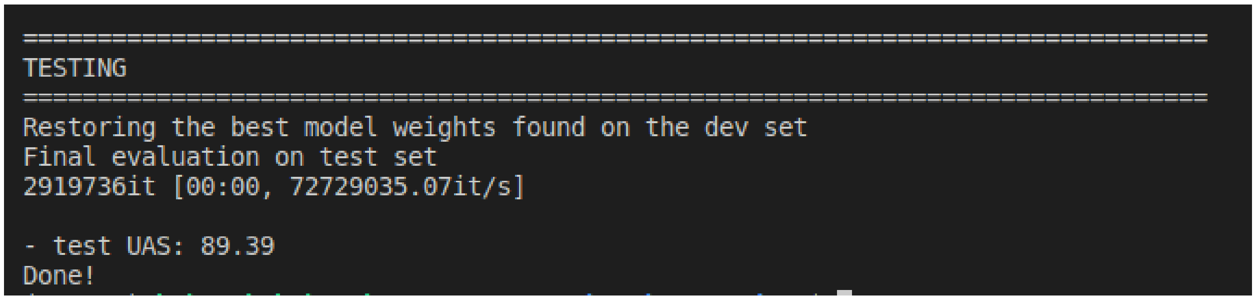
e) the best UAS that achieves on the dev set is : 88.83



```
Epoch 10 out of 10
100%|
8.21it/s]
Average Train Loss: 0.06658753140898042
Evaluating on dev set
1445850it [00:00, 46727444.22it/s]

- dev UAS: 88.83
New best dev UAS! Saving model.
```

UAS that achieves on the test set : 89.39



```
================================================================================
TESTING
================================================================================
Restoring the best model weights found on the dev set
Final evaluation on test set
2919736it [00:00, 72729035.07it/s]

- test UAS: 89.39
Done!
```

f.i)

Error type: Verb Phrase Attachment Error
Incorrect dependency : acquisition → citing
Correct dependency: blocked → citing

f.ii)

Error type : Modifier Attachment Error
incorrect dependency: left → early
Correct dependency: afternoon → early

f.iii)

Error type: Prepositional Phrase Attachment Error
Incorrect dependency: declined → decision
Correct dependency: reasons → decision

f.iv)

Error type: Coordination Attachment Error
incorrect dependency: affects → one
Correct dependency: plants → one

g)

The POS tagging can be used to guide the parser in generating more accurate dependency trees.
For example Many words in English can have multiple meanings depending on their context. POS tags can help disambiguate the meaning of a word by providing information about its grammatical function within a sentence.
By using POS tags as features, the parser can limit the set of possible head-dependent relationships, as certain relationships are not allowed based on syntactic constraints. This can help reduce the search space and improve the efficiency of the parser.
POS tags are language-dependent, meaning that they can be used to parse sentences in any language that has a POS tagging system. This makes it easier to develop parsers for multiple languages using the same underlying algorithm.