1(a):

|  |  |  |  |
| --- | --- | --- | --- |
| **stack** | **buffer** | **new dependency** | **transitions** |
| [ROOT] | [Nadia, rode, the, old, donkey, with, dexterity] |  | Initial Config |
| [ROOT, Nadia] | [rode, the, old, donkey, with, dexterity] |  | SHIFT |
| [ROOT, Nadia, rode] | [the, old, donkey, with, dexterity] |  | SHIFT |
| [ROOT, rode] | [the old, donkey, with, dexterity] | rode Nadia | LEFT-ARC |
| [ROOT, rode, the] | [old, donkey, with, dexterity] |  | SHIFT |
| [ROOT, rode, the, old] | [donkey, with, dexterity] |  | SHIFT |
| [ROOT, rode, the, old, donkey] | [with, dexterity] |  | SHIFT |
| [ROOT, rode, the, donkey] | [with, dexterity] | donkey old | LEFT-ARC |
| [ROOT, rode, donkey] | [with, dexterity] | donkey the | LEFT-ARC |
| [ROOT, rode] | [with, dexterity] | rode donkey | RIGHT-ARC |
| [ROOT, rode, with] | [dexterity] |  | SHIFT |
| [ROOT, rode, with dexterity] |  |  | SHIFT |
| [ROOT, rode, with] |  | with dexterity | RIGHT-ARC |
| [ROOT, rode] |  | rode with | RIGHT-ARC |
| [ROOT] |  | ROOT rode | RIGHT-ARC |

**1(b):**

A sentence containing n words will be parsed in 2n+1 steps. Adding and removing the word on the stack will make it 2 steps for each word in the sentence. Therefore, n words will take 2n steps and plus one initializing step.

**1(c):**

Parsing mechanism described in question is insufficient to generate non-projective dependency trees. Since LEFT-ARC and RIGHT-ARC can only be done on the last 2 words on the stack. In the example of non-projective tree, ‘saw’ to ‘yesterday’ and ‘a dog’ to ‘was’ cannot be done with RIGHT-ARC because they are not adjacent to each other.

**2(b):**

The best LAS and UAS that my model achieved on the dev and test sets are below:

best LAS on validation sets: 0.884 best LAS on test sets: 0.884

best UAS on validation sets: 0.899 best UAS on test sets: 0.900

I’ve also tried to change some of the hyperparameters to achieve a better performance.

When I changed the Epoch Size from 10 to 20 and remaining stay unchanged:

best LAS on validation sets: 0.887 best LAS on test sets: 0.902

best UAS on validation sets: 0.887 best UAS on test sets: 0.902

You can easily tell that using a lager Epoch Size can slightly increase the performance.

When I changed the learning rate from 0.001 to 0.007 and remaining stay unchanged:

best LAS on validation sets: 0.861 best LAS on test sets: 0.879

best UAS on validation sets: 0.865 best UAS on test sets: 0.883

When you increase the learning rate, the performance is not as good as before.

Finally, I tried to change the hidden layer size from 200 to 500:

best LAS on validation sets: 0.891 best LAS on test sets: 0.907

best UAS on validation sets: 0.895 best UAS on test sets: 0.909

As I expected, increase the hidden layer size can bring out a better performance with the model.