CS224 Winter 2019 Assignment 3

Name: Dat Nguyen

By turning in this assignment, I agree by the Stanford honor code and declare that all of this is my own work.

1. Machine Learning & Neural Networks (8 points)

(a) i Because β_1 is set close to 1, the updated $\mathbf{m}$ has the majority of the weight by the old, accumulated $\mathbf{m}$ and only incorporates a little bit of the new value of $\mathbf{m}$. This makes the values of $\mathbf{m}$ in successive updates not vary too much. This low variance may be helpful to learning because suppose the majority of recent gradients points to the same direction, then it is quite likely that it might be a good direction to follow and adjusting only a bit by the new direction is a good thing to do. In addition for the dimensions that fluctuate very much then by taking the rolling average of gradients those fluctuation will be canceled out. That will help speed up convergence because consistent gradients will be valued more.

ii Because Adam divides the learning rate by $\sqrt{\mathbf{v}}$. The learning rate corresponding to larger gradient will be smaller and the learning rate for smaller gradient will be larger. It helps stabilize training because it avoids large gradient in one direction having big impact on other dimensions.

(b) i We have

$$\begin{aligned} \mathbb{E}_{p_{\text{drop}}}[\mathbf{h}_{\text{drop}}] &= h_i \\ \iff p_{\text{drop}} 0 + (1 - p_{\text{drop}})\gamma h_i &= h_i \\ \iff \gamma &= \frac{1}{1 - p_{\text{drop}}} \end{aligned}$$

ii We should apply dropout during training to add more randomness to the training so that the weight will not be too focused on a certain number of parameters but every parameter will be given roughly equal weights. This makes the set of parameters more robust and help avoid overfitting.

On the other hand if we apply dropout during evaluation then some contribution of weights that learned during training will not be used so that makes the prediction less accurate.

2. Neural Transition-Based Dependency Parsing (42 points)

(a) The sequence of transition

Stack	Buffer	New dependency	Transition
[ROOT]	[I, parsed, this sentence, correctly]		Initial Configuration
[ROOT, I]	[parsed, this, sentence, correctly]		SHIFT
[ROOT, I, parsed]	[this, sentence, correctly]		SHIFT
[ROOT, parsed]	[this, sentence, correctly]	parsed→ I	LEFT-ARC
[ROOT, parsed, this]	[sentence, correctly]		SHIFT
[ROOT, parsed, this, sentence]	[correctly]		SHIFT
[ROOT, parsed, sentence]	[correctly]	sentence → this	LEFT-ARC
[ROOT, parsed]	[correctly]	parsed → sentence	RIGHT-ARC
[ROOT, parsed, correctly]	[]		SHIFT
[ROOT, parsed]	[]	parsed → correctly	RIGHT-ARC
[ROOT]	[]	ROOT → parsed	RIGHT-ARC

(b) A sentence containing n words will be parsed in 2n steps, because it takes n steps to do the SHIFT of all words from buffer to stack and an addition of n steps to remove words from stack by either RIGHT-ARC or LEFT-ARC transition.

(e) The best UAS that my model achieved on dev set is 88.82 and the best UAS achieved on test set is 89.17.

(f) i
- **Error type:** Verb Phrase Attachment Error
- **Incorrect dependency:** wedding → fearing
- **Correct dependency:** heading → fearing

ii
- **Error type:** Coordination Attachment Error
- **Incorrect dependency:** make → rescue
- **Correct dependency:** want → rescue

iii
- **Error type:** Prepositional Phrase Attachment Error
- **Incorrect dependency:** named → Midland
- **Correct dependency:** O'Neill → Midland

iv
- **Error type:** Modifier Attachment Error

- **Incorrect dependency:** elements → most
- **Correct dependency:** crucial → most