Question 1, Assignment 3, CS224n

(a) (i) m tends to accumulate gradients from past steps and let the learner moves faster and faster, which leads to quicker convergence.

(ii) None of model parameters will get larger updates, since this is used to normalize gradients and hence the learner won't go too large steps(overshooting).

(b) (i) $\frac{1}{p_{drop}}$, so that $\mathbb{E}_{p_{drop}}[h_{drop}] = \mathbb{E}_{p_{drop}}[\frac{1}{p_{drop}} p_{drop} h] = \mathbb{E}_{p_{drop}}[h]$.

(ii) During training, dropout is used to reduce the variance of the learner by 'removing' some neurons, in order to prevent overfitting. Evaluation is used to verify the learner where we need the learner to behave stable.

(a) The whole transitions are shown as below:

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]	I ← parsed	`LEFT-ARC`
[ROOT, parsed, this]	[sentence, correctly]		`SHIFT`
[ROOT, parsed, this, sentence]	[correctly]		`SHIFT`
[ROOT, parsed, sentence]	[correctly]	this ← sentence	`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) $2n$ times. Because for each word, it will *only* be pop into and out of the stack *once*.

(c)

(d)

(e)

(f) The final average training loss is 0.57927 and the UAS is 88.06 and 88.10 on devset and testset respectively.

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

(ii) **Error type**: Coordination Attachment Error
Incorrect dependency: makes → rescue
Correct dependency: rush → rescue

(iii) **Error type**: Prepositional Phrase Attachment Error
Incorrect dependency: named → Midland
Correct dependency: guy → Midland

(iv) **Error type**: Modifier Attachment Error
Incorrect dependency: elements → most
Correct dependency: crucial → most