Carolyn Kao (chkao831@stanford.edu)
February 2, 2021 **CS224n Assignment 3**

1) Machine Learning & Neural Networks

a)

i) By keeping track of $\mathbf{m}$, which is a rolling average of the gradients, we basically capture the previous update m_{i-1}, integrate that with other gradient components to some degree $\beta_1 \in [0, 1]$, and then smooth transitions from one state to another by controlling the hyperparameter β_1 on the previous state. This allows us to place different weights on different gradient points at position θ over time such that the dimensions whose gradients point in the same directions is increased while decreasing the updates for dimensions whose gradients change directions. With such smoothing transitions, we are able to lower the variance (mainly because only a (1-β_1) proportion of $\mathbf{m}$ gets to be updated at each iteration, preventing from varying too much) with reduced oscillation and faster convergence , which is optimal to the overall learning; this results from the fact that decreasing the oscillations straightens out the trail that descends to the local optimum, yielding more efficiency.

ii) As stated, now we have the rolling average of the magnitudes of the gradients $\mathbf{v}$, so the model parameters with corresponding smaller gradients (i.e. smaller $\mathbf{v}$) would get bigger updates. Dividing the term by $\sqrt{v}$, for parameters with smaller gradients, we basically cut off more (with a smaller divisor) during an update. This allows us to get effective update for stagnant parameters at flat areas by having them move faster along axes and accelerate the learning and convergence.

b)

i) Mathematically speaking, as $d_i \sim Bernoulli(1 - p_{drop})$, we could rewrite the given equation as,

$$\mathbb{E}_{p_{drop}}[\mathbf{h}_{drop}]_i = \gamma \mathbb{E}_{p_{drop}}[d_i h_i] \quad = \gamma h(1 - p_{drop}) = h$$

This gives us $\gamma = \frac{1}{1-p_{drop}}$ during the test phase.

ii) On the one hand, We should apply dropout during training because we'd like to reduce overfitting with a more generalizable network. With its randomness, dropout basically reduces the capacity and "thin" the network during training, allowing our model to avoid breaks-up situations where network layers co-adapt to correct mistakes from prior layers, in turn making the model more robust.
On the other hand, we want to evaluate the integrated representational ability of all hidden units during evaluation time. If using "thinned" network, we would not be able to utilize the capability of each learned neurons by skipping some of them randomly, as we haven't been sure which neurons should be laid off with such underlying randomness. Hence, we need to ensemble those hidden units at this stage for examining the generalization power of the model.

2) Neural Transition-Based Dependency Parsing

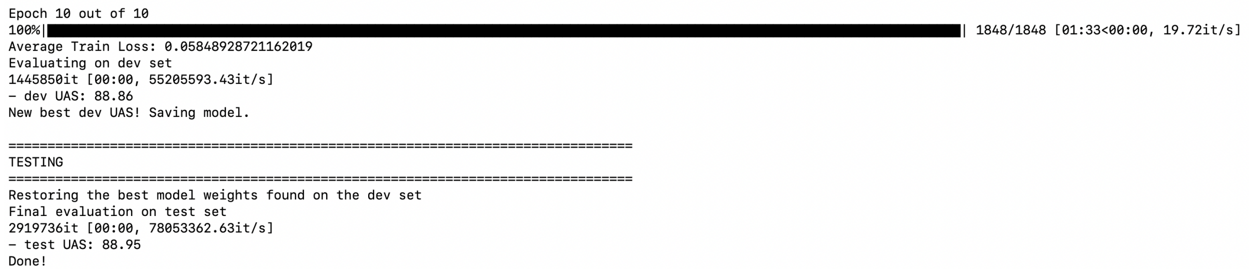
(a)

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
$[ROOT, parsed]$	$[this, sentence, correctly]$	$parsed \rightarrow I$	LEFT-ARC
$[ROOT, parsed, this]$	$[sentence, correctly]$		SHIFT
$[ROOT, parsed, this, sentence]$	$[correctly]$		SHIFT
$[ROOT, parsed, sentence]$	$[correctly]$	$sentence \rightarrow this$	LEFT-ARC
$[ROOT, parsed]$	$[correctly]$	$parsed \rightarrow sentence$	RIGHT-ARC
$[ROOT, parsed, correctly]$	$[]$		SHIFT
$[ROOT, parsed]$	$[]$	$parsed \rightarrow correctly$	RIGHT-ARC
$[ROOT]$	$[]$	$ROOT \rightarrow parsed$	RIGHT-ARC

(b) For a sentence containing n words to be parsed, it would roughly take linear time $O(n)$. At each step, we would have two possible state transitions, either to shift word from buffer to stack or to remove a dependent from the stack. Hence, for each word to be shifted, it takes a single step; for a word to be "arc"-ed over from the stack as a dependent, it's also a single step. Specifically speaking, we need $2n$ steps to do so.

(e)

The best UAS (Unlabeled Attachment Score) my model achieves on the dev set is 88.86; the UAS my model achieves on the test set is 88.95.



```
Epoch 10 out of 10
100%|████████████████████████████████████████████████████████████████████████████| 1848/1848 [01:33<00:00, 19.72it/s]
Average Train Loss: 0.05848928721162019
Evaluating on dev set
1445850it [00:00, 55205593.43it/s]
- dev UAS: 88.86
New best dev UAS! Saving model.

================================================================================
TESTING
================================================================================
Restoring the best model weights found on the dev set
Final evaluation on test set
2919736it [00:00, 78053362.63it/s]
- test UAS: 88.95
Done!
```

(f)

(i) **I disembarked and was heading to a wedding fearing my death.**
Error type: Verb Phrase Attachment Error
Incorrect dependency: $wedding \rightarrow fearing$
Correct dependency: $heading \rightarrow fearing$
(ii) **It makes me want to rush out and rescue people from dilemmas of their own making.**
Error type: Coordination Attachment Error
Incorrect dependency: $makes \rightarrow rescue$
Correct dependency: $rush \rightarrow rescue$
(iii) **It is on loan from a guy named Joe O'Neill in Midland, Texas.**
Error type: Prepositional Phrase Attachment Error
Incorrect dependency: $named \rightarrow Midland$

Correct dependency: $guy \rightarrow Midland$
(iv) **Brian has been one of the most crucial elements to the success of Mozilla software.**
Error type: Modifier Attachment Error
Incorrect dependency: $elements \rightarrow most$
Correct dependency: $crucial \rightarrow most$