CS 224n: Assignment 3 Written Part

April 22, 2022

Question 1: Machine Learning & Neural Networks (8 points)

(a) Adam Optimizer

i. Briefly explain in 2-4 sentences (you don't need to prove mathematically, just give an intuition) how using m (momentum) stops the updates from varying as much and why this low variance may be helpful to learning, overall.

Solution:
When updating gradients, momentum helps keep the direction of previous gradients, and use the current gradient to slightly adjust the direction. This makes the gradient descent process more stable and help the algorithm to converge faster. It can also prevent the problem of local minima, plateau, and saddle points.

ii. Since Adam divides the update by $\sqrt{v}$, which of the model parameters will get larger updates? Why might this help with learning?

Solution:
Model parameters that receive small or infrequent updates will get larger updates. For parameters that receive large updates, the learning rate will decay faster. This further improves the stability of gradient descent.

(b) Dropout

i. What must γ equal in terms of p_{drop}? Briefly justify your answer or show your math derivation using the equations given above.

Solution:

$$\mathbb{E}_{p_{drop}}[h_{drop}]_i = (1 - p_{drop}) * \gamma * h_i$$

Therefore, γ must equal $\frac{1}{1-p_{drop}}$ so that $\mathbb{E}_{p_{drop}}[h_{drop}]_i = h_i$.

ii. Why should dropout be applied during training? Why should dropout NOT be applied during evaluation?

Solution:
Applying dropout during training can prevent the overfitting problem and improve model generalization ability. During evaluation, we should use the model that is well trained and doesn't have any overfitting problem. Therefore, dropout should not be applied during evaluation.

Question 2: Neural Transition-Based Dependency Parsing (44 points)

(a) Go through the sequence of transitions needed for parsing the sentence *"I parsed this sentence correctly"*. At each step, give the configuration of the stack and buffer, as well as what transition was applied this step and what new dependency was added (if any).

Solution:

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 how many steps (in terms of n)? Briefly explain in 1-2 sentences why.

Solution:
A sentence containing n words will be parsed in 2n steps. Because for each words, the first step is SHIFT, so that it will be moved to the Stack. And then the second step is to decide its dependency and move it out of the Stack.