CS224N: Assignment #1

Kangwei Ling

February 27, 2017

1 Tensorflow Softmax

(a) `q1_softmax.py`

(b) `q1_softmax.py`

(c)
- placeholder variables are nodes in the computation graph that need the data from outside to give its value. We use placeholder variables for input data and labels.
- feed dictionaries is used to map placeholder variables to data provided when we are about to do the computation.

(d) `q1_classifier.py`

(e) `q1_classifier.py`

2 Neural Transition-Based Dependency Parsing

(a)

stack	buffer	new dependency	transition
...	...	...	...
[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) Every word will be pushed onto the stack exactly 1 time. And each word will be associated with exactly 1 arc which itself is the pointee of the arc, so there are n `-ARC` step. Therefore, a sentence containing n words will be parsed in $2n$ steps.

(c) `q2_parser_transitions.py`

(d) `q2_parser_transitions.py`

(e) `q2_initialization.py`

(f) $\gamma = \frac{1}{p_{drop}}$. ($\boldsymbol{d}$ is independent of $\boldsymbol{h}$)

$$\mathbb{E}[\boldsymbol{h}_{drop}]_i = \gamma \mathbb{E}[\boldsymbol{d}]_i \cdot \mathbb{E}[\boldsymbol{h}]_i = \gamma p_{drop} \mathbb{E}[\boldsymbol{h}]_i$$

(g) (i) Using $\boldsymbol{m}$, the update of each time is a combination of current gradient and previous updates. This method helps damp the oscillation (reversing direction updates will be combined and damped), while their momentum towards the optima will be compounded, greatly boosting the converging speed.

(ii) The model parameters received smaller and infrequent gradients will get larger updates, while those with larger and frequent gradients will get lower learning rate. In this way, the learning rate is auto-adjusted and tuned for the training.

(h) UAS on dev set: 88.56
UAS on train set: 89.16

3 Recurrent Neural Networks: Language Modeling

(a) Suppose $y_c^t = 1$ (since y is one-hot), then $J^t(\theta) = -\log \hat{y}_c^{(t)}$. Thus,

$$\begin{aligned} PP^{(t)}(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}) &= \frac{1}{\hat{y}_j^{(t)}} \\ &= 2^J \end{aligned}$$

From the relation between perplexity and the cross-entropy loss, it's obvious that minimizing the (arithmetic) mean cross-entropy loss will also minimize the (geometric) mean perplexity.

For a totally random model, the perplexity would be $|V|$, the cross-entropy would be $J = -\log \frac{1}{|V|} = |V|$.

(b) Let $\boldsymbol{z}^{(t)} = \boldsymbol{h}^{(t-1)}\boldsymbol{H} + \boldsymbol{e}^{(t)}\boldsymbol{I} + \boldsymbol{b}_1, \boldsymbol{\theta} = \boldsymbol{h}^{(t)}\boldsymbol{U} + \boldsymbol{b}_2$

$$\begin{aligned} \frac{\partial J^{(t)}}{\partial \boldsymbol{b}_2} &= \hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)} \\ \boldsymbol{\delta} = \frac{\partial J^{(t)}}{\partial \boldsymbol{\theta}} \frac{\partial \boldsymbol{\theta}}{\partial \boldsymbol{h}^{(t)}} \frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{z}^{(t)}} &= (\hat{\boldsymbol{y}}^{(t)} - \boldsymbol{y}^{(t)})\boldsymbol{U}^T \circ \sigma'(\boldsymbol{z}^{(t)}) \\ \frac{\partial J^{(t)}}{\partial \boldsymbol{I}}\Big|_{(t)} &= \boldsymbol{\delta} \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{I}} = (\boldsymbol{e}^{(t)})^T \boldsymbol{\delta} \\ \frac{\partial J^{(t)}}{\partial \boldsymbol{H}}\Big|_{(t)} &= \boldsymbol{\delta} \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{H}} = (\boldsymbol{h}^{(t-1)})^T \boldsymbol{\delta} \\ \frac{\partial J^{(t)}}{\partial \boldsymbol{L}_{\boldsymbol{x}^{(t)}}} = \frac{\partial J^{(t)}}{\partial \boldsymbol{e}^{(t)}} &= \boldsymbol{\delta} \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{e}^{(t)}} = \boldsymbol{\delta} \boldsymbol{I}^T \\ \frac{\partial J^{(t)}}{\partial \boldsymbol{h}^{(t-1)}} &= \boldsymbol{\delta} \boldsymbol{H}^T \end{aligned}$$

(c) Use the notation in (c).

$$\left.\frac{\partial J^{(t)}}{\partial \boldsymbol{H}}\right|_{(t-1)} = (\boldsymbol{h}^{(t-2)})^T(\boldsymbol{\delta}^{(t-1)} \circ \sigma'(\boldsymbol{z}^{(t-1)}))$$

$$\left.\frac{\partial J^{(t)}}{\partial \boldsymbol{I}}\right|_{(t-1)} = (\boldsymbol{e}^{(t-1)})^T(\boldsymbol{\delta}^{(t-1)} \circ \sigma'(\boldsymbol{z}^{(t-1)}))$$

$$\frac{\partial J^{(t)}}{\partial \boldsymbol{L}_{\boldsymbol{x}^{(t-1)}}} = (\boldsymbol{\delta}^{(t-1)} \circ \sigma'(\boldsymbol{z}^{(t-1)}))\boldsymbol{I}^T$$

(d) forward: $O(D_h^2 + D_h(d + |V|))$
backward: $\tau \cdot O(D_h^2 + D_h(d + |V|))$

(e) bonus: hierarchical softmax?