CS 224N: Assignment 2

PETER888@STANFORD.EDU SATURDAY 3RD FEBRUARY, 2018

Problem 1: Tensorflow Softmax (25 points)

1.1 (a) Implement Softmax use Tensorflow (5 points, coding)

Answer:
See code: `~/q1_softmax.py`.

1.2 (b) Implement Cross-Entropy use Tensorflow (5 points, coding)

Answer:

See code: `~/q1_softmax.py`.

1.3 (c) Tensorflow Placeholder and Feed Dictionary (5 points, coding/written)

Answer:

See code: `~/q1_classifier.py`.

Explanation:

1.4 (d) Implement Classifier (5 points, coding)

Answer:
See code: ∼/`q1_classifier.py`.

1.5 (e) Implement Model (5 points, coding/written)

Answer:
See code: ∼/`q1_classifier.py`.

Explanation:

Problem 2: Neural Transition-Based Dependency Parsing (50 points + 2 bonus points)

2.1 (a) Dependency Parsing (6 points, written)

Answer:

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`

2.2 (b) How many steps (2 points, written)

Answer:

$2n$ *parse steps. Because each word take exactly one shift transition from buffer to stack, take exactly one *-ARC (either LEFT-ARC or RIGHT-ARC) transition move out from stack, and every transition either add or remove word in the stack.*

2.3 (c) Parser Step (6 points, coding)

See code: ~/q2_parser_transitions.py.

2.4 (d) Parser Transitions (6 points, coding)

See code: ~/q2_parser_transitions.py.

2.5 (e) Xavier Initialization (4 points, coding)

See code: ~/q2_initialization.py.

2.6 (f) Dropout (2 points, written)

Answer:

$$\gamma = \frac{1}{1 - p_{drop}}$$

The mask vector $\boldsymbol{d}$ *set entries in* $\boldsymbol{h}$ *to zero at probability* p_{drop}*, let's say* $\boldsymbol{h}$ *has full expectation value* $\mathbb{E}[\boldsymbol{h}]$*=1, so* $\mathbb{E}[\boldsymbol{d} \circ \boldsymbol{h}]$*=(1-*$p_{drop}$*), this because* p_{drop} *of* $\boldsymbol{h}$ *values are become zero.*
So, $1 = \frac{\mathbb{E}[\boldsymbol{d} \circ \boldsymbol{h}]}{(1-p_{drop})} \Rightarrow 1 = \mathbb{E}[\boldsymbol{h}_{drop}] = \frac{1}{(1-p_{drop})}\mathbb{E}[\boldsymbol{d} \circ \boldsymbol{h}] \Rightarrow \gamma = \frac{1}{1-p_{drop}}$

2.7 (g) Adam Optimizer (4 points, written)

2.7.1 i) Momentum

Answer:

2.7.2 ii) Adaptive Learning Rates

Answer:

2.8 (h) Parser Model (20 points, coding/written)

Answer:
See code: ~/q2_parser_model.py. *Report the best UAS*
The dev UAS: 87.83, the test UAS: 88.09
List of predicted labels, see file $q2_test.predicted.pkl$

2.9 (i) Bonus (2 points, coding/written)

Answer: Implemented L2 regularization. And attempt

Problem 3: Recurrent Neural Networks (25 points + 1 bonus point)

3.1 (a) Perplexity (4 points, written)

3.1.1 i) Derive Perplexity (2 points)

Answer:

As $\boldsymbol{y^{(t)}}$ *is an one-hot vector, suppose the k-th element* $y_k^{(t)}$ *is 1*

so,

$\boldsymbol{J}^{(t)}(\boldsymbol{\theta}) = -log(\hat{y}_k^{(t)}) = log(\frac{1}{\hat{y}_k^{(t)}})$

and we also have

$PP^{(t)}(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}) = \frac{1}{\hat{y}_k^{(t)}}$

thus, we have

$\boldsymbol{J}^{(t)}(\boldsymbol{\theta}) = log(PP^{(t)}(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}))$

3.1.2 ii) Equivalent (1 point)

Answer:
We know $min\{log(f(x)\} = min\{f(x)\}, f(x) > 0$

then we have,

$min\{\boldsymbol{J}^{(t)}(\boldsymbol{\theta}) = log(PP(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)})) : \boldsymbol{\theta} > 0\} = min\{PP^{(t)}(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)})\}\ \forall(t \in [1..T])$

from convex theory not hard to know minimizing the geometric mean equivalent to minimizing the arithmetic mean if the related function have same minimizing equivalent.

$min\{(\prod_{j=1}^{T} f_j(x))^{\frac{1}{T}}\} = min\{\frac{1}{T}\sum_{i=1}^{T}(g_i(x))\}$*, when* $min\{\boldsymbol{f}(x)\} = min\{\boldsymbol{g}(x)\}$*, for* $\boldsymbol{f}$*,*$\boldsymbol{g}$ *are positive functions*

Finally, we can get the minimizing geometric mean perplexity equivalent to minimizing the arithmetic mean cross-entropy loss

$min\{(\prod_{t=1}^{T} PP(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}))^{\frac{1}{T}}\} = min\{\frac{1}{T}\sum_{t=1}^{T} CE(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)})\}$

3.1.3 iii) Perplexity for a single word (1 point)

Answer:
for given word $\boldsymbol{\omega}_j$*,*

$\bar{P}\left(\boldsymbol{x}^{(t+1)} = \boldsymbol{\omega}_j | \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)}\right) = \frac{1}{|V|}$

so the perplexity for that single word $\boldsymbol{\omega}_j$, is $|V|$

$PP^{(t)}(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)} = 1/\frac{1}{|V|} = |V|$

because, $\boldsymbol{J}^{(t)}(\boldsymbol{\theta}) = log(PP^{(t)}(\boldsymbol{y}^{(t)}, \hat{\boldsymbol{y}}^{(t)}))$, when $|V|$=10000

$\boldsymbol{J}^{(t)}(\boldsymbol{\theta}) = log(|V|) \approx 9.213$

3.2 (b) Gradients on Single Point (7 points, written)

Answer:

$$\boldsymbol{\delta_1^{(t)}} = \frac{\partial \boldsymbol{J}}{\partial \boldsymbol{\theta}^{(t)}} = -\mathbf{y} + \hat{\mathbf{y}} \tag{3.1}$$

Write sigmoid(x) as $\sigma(x)$, and it's derivative as $\sigma^{'}(x)$

$$\begin{aligned}\boldsymbol{\delta_2^{(t)}} &= \frac{\partial \boldsymbol{J}}{\partial \boldsymbol{z}^{(t)}} = \boldsymbol{\delta_1^{(t)}} \frac{\partial \boldsymbol{\theta}^{(t)}}{\partial \boldsymbol{h}^{(t)}} \frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{z}^{(t)}} \\ &= \boldsymbol{U}^T \cdot \boldsymbol{\delta_1^{(t)}} \circ \sigma^{'}(\boldsymbol{z})\end{aligned} \tag{3.2}$$

$$\frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{U}} = \frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{\theta}^{(t)}} \frac{\partial \boldsymbol{\theta}^{(t)}}{\partial \boldsymbol{U}} = \boldsymbol{\delta_1^{(t)}} (\boldsymbol{h}^{(t)})^T \tag{3.3}$$

$$\frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{e}^{(t)}} = \frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{z}^{(t)}} \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{e}^{(t)}} = (\boldsymbol{W}_e|_{(t)})^T \boldsymbol{\delta_2^{(t)}} \tag{3.4}$$

$$\left.\frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{W_e}}\right|_{(t)} = \left.\frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{z}^{(t)}} \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{W_e}}\right|_{(t)} = \boldsymbol{\delta_2^{(t)}} (\boldsymbol{e}^{(t)})^T \tag{3.5}$$

$$\left.\frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{W}_h}\right|_{(t)} = \left.\frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{z}^{(t)}} \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{W}_h}\right|_{(t)} = \boldsymbol{\delta_2^{(t)}} (\boldsymbol{h}^{(t-1)})^T \tag{3.6}$$

$$\frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{h}^{(t-1)}} = \frac{\partial \boldsymbol{J}^{(t)}}{\partial \boldsymbol{z}^{(t)}} \frac{\partial \boldsymbol{z}^{(t)}}{\partial \boldsymbol{h}^{(t-1)}} = (\boldsymbol{W}_h|_{(t)})^T \boldsymbol{\delta_2^{(t)}} \tag{3.7}$$

3.3 (c) Gradients (7 points, written)

Answer:

3.4 (d) How Many Operations for Single Timestep (3 points, written)

Answer:

Derivatives for the skip-gram model

$$
\begin{aligned}
\frac{J_{skip-gram}(word_{c-m\ldots c+m})}{\partial \boldsymbol{v}_c} &= \sum_{-m \leqslant j \leqslant m, j \neq 0} \frac{\partial F(\boldsymbol{w}_{c+j}, \boldsymbol{v}_c)}{\partial \boldsymbol{v}_c} \\
\frac{J_{skip-gram}(word_{c-m\ldots c+m})}{\partial \boldsymbol{v}_j} &= 0, \forall j \neq c \\
\frac{J_{skip-gram}(word_{c-m\ldots c+m})}{\partial \boldsymbol{U}} &= \sum_{-m \leqslant j \leqslant m, j \neq 0} \frac{\partial F(\boldsymbol{w}_{c+j}, \boldsymbol{v}_c)}{\partial \boldsymbol{U}}
\end{aligned}
$$

Derivatives for the CBOW model

$$
\begin{aligned}
\frac{J_{CBOW}(word_{c-m\ldots c+m})}{\partial \boldsymbol{v}_j} &= \frac{\partial F(\boldsymbol{w}_c, \hat{\boldsymbol{v}})}{\partial \hat{\boldsymbol{v}}}, \forall (j \neq c) \in \{c-m\ldots c+m\} \\
\frac{J_{CBOW}(word_{c-m\ldots c+m})}{\partial \boldsymbol{v}_j} &= 0, \forall (j \neq c) \notin \{c-m\ldots c+m\} \\
\frac{J_{CBOW}(word_{c-m\ldots c+m})}{\partial \boldsymbol{U}} &= \frac{\partial F(\boldsymbol{w}_c, \hat{\boldsymbol{v}})}{\partial \boldsymbol{U}}
\end{aligned}
$$

3.5 (e) How Many Operations for Entire Sequence (3 points, written)

Answer: See code: ~/q3_word2vec.py.

3.6 (f) Which largest? Term RNN? (1 point, written)

Answer: See code: ~/q3_sgd.py.

3.7 (g) Bonus (1 point, written)

Answer:

Explain: In the Word Vectors image, words clustered at similarity, such as the emotion words ”amazing”, ”wonderful” and ”great” are very close to each other. The word ”well” a little further but still close to ”amazing” , the connection characters and words ”the” ”a” ”,” etc are spread around alone.