CS 224d Midterm Review (word vectors)

Peng Qi

May 5, 2015

Outline

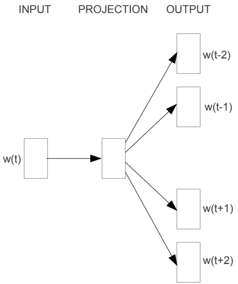
- `word2vec` and GloVe revisited
- `word2vec` with backpropagation

Outline

- `word2vec` and GloVe revisited
  - Skip-gram revisited
  - (Optional) CBOW and its connection to Skip-gram
  - (Optional) `word2vec` as matrix factorization (conceptually)
  - GloVe v.s. `word2vec`
- `word2vec` with backpropagation

Skip-gram

- *Task:* given a center word, predict its context words
- For each word, we have an "input vector" v_w and an "output vector" v'_w



Skip-gram

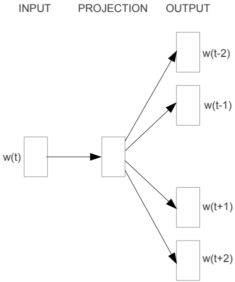
- We have seen two types of costs for an expected word given a vector prediction r

$$CE(w_i|r) = -\log\left(\frac{\exp(r^\top v'_{w_i})}{\sum_{j=1}^{|V|}\exp(r^\top v'_{w_j})}\right)$$

$$NEG(w_i|r) = -\log(\sigma(r^\top v'_{w_i})) - \sum_{k=1}^{K}\log(\sigma(-r^\top v'_{w_k}))$$

In the case of skip-gram, the vector prediction r is just the "input vector" of the center word, v_{w_i}.

$\sigma(\cdot)$ is the sigmoid (logistic) function.

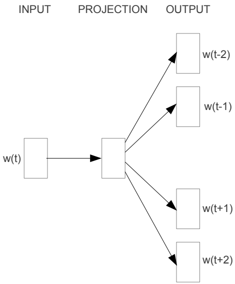


Skip-gram

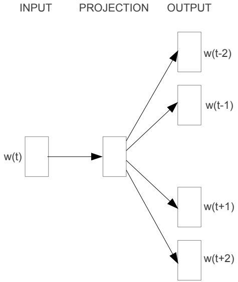
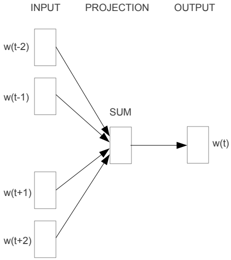
- Now we have all the pieces of skip-gram, the cost for a context window $[w_{i-C}, \cdots, w_{i-1}, w_i, w_{i+1}, \cdots, w_{i+C}]$ is (w_i is the center word)

$$J_{\text{skip-gram}}([w_{i-C}, \cdots, w_{i+C}]) = \sum_{i-C \leq j \leq i+C, i \neq j} F(w_j | v_{w_i})$$

  where F is one of the cost functions we defined in the previous slide.
- You might ask: but why are we introducing so many notations?



Skip-gram v.s. CBOW

	Skip-gram	**CBOW**
		
Task	Center word $\rightarrow$ Context	Context $\rightarrow$ Center word
r	v_{w_i}	$f(v_{w_{i-C}}, \cdots, v_{w_{i-1}}, v_{w_{i+1}}, \cdots, v_{w_{i+C}})$

All `word2vec` figures are from http://arxiv.org/pdf/1301.3781.pdf

word2vec as matrix factorization (conceptually)

- Matrix factorization

$$\begin{bmatrix} & & \\ & M & \\ & & \end{bmatrix}_{n\times n} \approx \begin{bmatrix} \\ A^\top \\ \\ \end{bmatrix}_{n\times k} \begin{bmatrix} & B & \end{bmatrix}_{k\times n}$$

$$M_{ij} \approx a_i^\top b_j$$

- Imagine M is a matrix of counts for events co-occurring, but we only get to observe the co-occurrences one at a time. E.g.

$$M = \begin{bmatrix} 1 & 0 & 4 \\ 0 & 0 & 2 \\ 1 & 3 & 0 \end{bmatrix}$$

but we only see

(1,1), (2,3), (3,2), (2,3), (1,3), . . .

`word2vec` as matrix factorization (conceptually)

$$M_{ij} \approx a_i^\top b_j$$

- Whenever we see a pair (i, j) co-occur, we try to increasing $a_i^\top b_j$
- We also try to make all the other inner-products smaller to account for pairs never observed (or unobserved yet), by decreasing $a_{\neg i}^\top b_j$ and $a_i^\top b_{\neg j}$
- Remember from the lecture that the word co-occurrence matrix usually captures the semantic meaning of a word? For `word2vec` models, roughly speaking, M is the windowed word co-occurrence matrix, A is the output vector matrix, and B is the input vector matrix.
- Why not just use one set of vectors? It's equivalent to $A = B$ in our formulation here, but less constraints is usually easier for optimization.

GloVe v.s. `word2vec`

	Fast training	Efficient usage of statistics	Quality affected by size of corpora	Captures complex patterns
Direct prediction (`word2vec`)	Scales with size of corpus	No	No*	Yes
GloVe	Yes	Yes	No	Yes

* Skip-gram and CBOW are qualitatively different when it comes to smaller corpora

Outline

- `word2vec` and GloVe revisited
- `word2vec` with backpropagation

word2vec with backpropagation

- $$CE(w_i|r) = -\log\left(\frac{\exp(r^\top v'_{w_i})}{\sum_{j=1}^{|V|}\exp(r^\top v'_{w_j})}\right)$$
- $$\begin{aligned} CE(w_i|r) &= CE(\hat{y}, y_i) \\ \hat{y} &= \mathrm{softmax}(\theta) \\ \theta &= (V')^\top r \end{aligned}$$
- $$\delta = \frac{\partial CE}{\partial \theta} = \hat{y} - y_i$$
- $$\begin{aligned} \frac{\partial CE}{\partial V'} &= r\delta^\top \\ \frac{\partial CE}{\partial r} &= V'\delta \end{aligned}$$

Thanks for your attention
and best of luck with the mid-term!

Any questions?