

Word2Vec (Continue.)

对于高频词如the/and/of，可能大多数次和它们的点积都会很高，甚至是最高的，这会影响找到有效的搭配。

一种解决方法是去掉高频词。

梯度下降

Stochastic Gradient Descent

- Problem: $J(\theta)$ is a function of **all** windows in the corpus (potentially billions!)
 - So $\nabla_{\theta} J(\theta)$ is very expensive to compute
- You would wait a very long time before making a single update!
- **Very** bad idea for pretty much all neural nets!
- Solution: Stochastic gradient descent (SGD)
 - Repeatedly sample windows, and update after each one
 - Algorithm:

```
while True:  
    window = sample_window(corpus)  
    theta_grad = evaluate_gradient(J, window, theta)  
    theta = theta - alpha * theta_grad
```

SG(Skip-grams)和CBOW(Continuous Bag of Words)

刚才的word2vec方法每个单词会得到两个向量， u 和 v ，最后取平均。这样做求导会方便。

可以改进成只用一个向量，流程大致一致，只是求导计算变复杂。

Skip-grams

给定center，预测context。就是刚才的word2vec方法。

CBOW

给定context, 预测center。

Negative Sampling

$$\bullet P(o|c) = \frac{\exp(u_o^T v_c)}{\sum_{w \in V} \exp(u_w^T v_c)}$$

上述公式在实现时, 分母要进行词汇量大小次数的点积运算, 耗时太长。

在word2vec论文中, 采用**negative sampling**。

核心思想是训练一个二分类器(sigmoid), true pair预测为正, false pair预测为负。一般训逻辑回归都不训练 $y = 0$ 的情况, 而只训练正例, 但这里也考虑了使负例尽可能概率为0。一组true对应K组false:

The skip-gram model with negative sampling (HW2)

- Notation more similar to class and HW2:

$$J_{neg-sample}(\mathbf{o}, \mathbf{v}_c, \mathbf{U}) = -\log(\sigma(\mathbf{u}_o^\top \mathbf{v}_c)) - \sum_{k=1}^K \log(\sigma(-\mathbf{u}_k^\top \mathbf{v}_c))$$

- We take k negative samples (using word probabilities)
- Maximize probability that real outside word appears, minimize prob. that random words appear around center word
- $P(w) = U(w)^{3/4}/Z$,
the unigram distribution $U(w)$ raised to the $3/4$ power
(We provide this function in the starter code).
- The power makes less frequent words be sampled more often

Counting

通过统计一个window里每个单词出现的数量来做word2vec, 而不是每次出现都放入训练。

Window based co-occurrence matrix

- Example corpus:

- I like deep learning.
- I like NLP.
- I enjoy flying.

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

但每个单词都这样统计(每行作为feature), 会构成一个1000000*1000000的矩阵, 空间消耗太大, 其中大多数都是0。

解决方法:

Solution: Low dimensional vectors

- Idea: store “most” of the important information in a fixed, small number of dimensions: a dense vector
- Usually 25–1000 dimensions, similar to word2vec
- How to reduce the dimensionality?

Method 1: Dimensionality Reduction on X (HW1)

Singular Value Decomposition of co-occurrence matrix X

Factorizes X into $U\Sigma V^T$, where U and V are orthonormal

$$\underbrace{\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix}}_{X^k} = \underbrace{\begin{bmatrix} * & * & * \\ * & * & * \\ * & * & * \end{bmatrix}}_U \underbrace{\begin{bmatrix} \bullet & & \\ & \bullet & \\ & & \bullet \end{bmatrix}}_{\Sigma} \underbrace{\begin{bmatrix} * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \\ * & * & * & * & * \end{bmatrix}}_{V^T}$$

Retain only k singular values, in order to generalize.

\hat{X} is the best rank k approximation to X , in terms of least squares.

Classic linear algebra result. Expensive to compute for large matrices.

20

GloVe

Count based和Direct prediction分成两派:

Count based vs. direct prediction

- LSA, HAL (Lund & Burgess),
- COALS, Hellinger-PCA (Rohde et al, Lebret & Collobert)

- Fast training
- Efficient usage of statistics
- Primarily used to capture word similarity
- Disproportionate importance given to large counts

- Skip-gram/CBOW (Mikolov et al)
- NNLM, HLBL, RNN (Bengio et al; Collobert & Weston; Huang et al; Mnih & Hinton)

- Scales with corpus size
- Inefficient usage of statistics
- Generate improved performance on other tasks
- Can capture complex patterns beyond word similarity

两者结合就是GloVe

核心思想是Co-occurrence的概率比值有意义。



Encoding meaning in vector differences

[Pennington, Socher, and Manning, EMNLP 2014]

Crucial insight: Ratios of co-occurrence probabilities can encode meaning components

	$x = \text{solid}$	$x = \text{gas}$	$x = \text{water}$	$x = \text{fashion}$
$P(x \text{ice})$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(x \text{steam})$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$\frac{P(x \text{ice})}{P(x \text{steam})}$	8.9	8.5×10^{-2}	1.36	0.96

下图是GloVe的具体原理。让 $w_i \cdot w_j$ 接近 $\log P(i|j)$ 是目标函数的目标。 X_{ij} 是通过词频统计的 $P(i|j)$ 。
 $f()$ 是一个系数，限制高频单词的影响。



Encoding meaning in vector differences

Q: How can we capture ratios of co-occurrence probabilities as linear meaning components in a word vector space?

A: Log-bilinear model: $w_i \cdot w_j = \log P(i|j)$

with vector differences $w_x \cdot (w_a - w_b) = \log \frac{P(x|a)}{P(x|b)}$



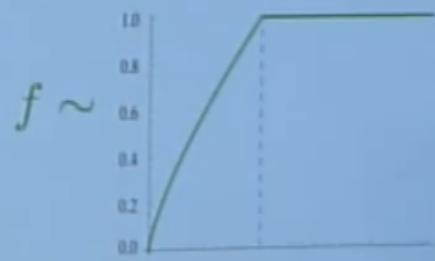
Combining the best of both worlds GloVe [Pennington et al., EMNLP 2014]



$$w_i \cdot w_j = \log P(i|j)$$

$$J = \sum_{i,j=1}^V f(X_{ij}) \left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log X_{ij} \right)^2$$

- Fast training
- Scalable to huge corpora
- Good performance even with small corpus and small vectors



Evaluate Word Vectors

How to evaluate word vectors?

- Related to general evaluation in NLP: Intrinsic vs extrinsic
- Intrinsic:
 - Evaluation on a specific/intermediate subtask
 - Fast to compute
 - Helps to understand that system
 - Not clear if really helpful unless correlation to real task is established
- Extrinsic:
 - Evaluation on a real task
 - Can take a long time to compute accuracy
 - Unclear if the subsystem is the problem or its interaction or other subsystems
 - If replacing exactly one subsystem with another improves accuracy → Winning!

一个Analogy数据集(intrinsic)

Details of intrinsic word vector evaluation

- Word Vector Analogies: **Syntactic** and Semantic examples from

: gram4-superlative
bad worst big biggest
bad worst bright brightest
bad worst cold coldest
bad worst cool coolest
bad worst dark darkest
bad worst easy easiest
bad worst fast fastest
bad worst good best
bad worst great greatest

一个Similarity数据集(intrinsic)

Another intrinsic word vector evaluation

- Word vector distances and their correlation with human judgments
- Example dataset: WordSim353
<http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/>

Word 1	Word 2	Human (mean)
tiger	cat	7.35
tiger	tiger	10
book	paper	7.46
computer	internet	7.58
plane	car	5.77
professor	doctor	6.62
stock	phone	1.62
stock	CD	1.31
stock	jaguar	0.92

AA
(extrinsic)

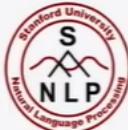
Extrinsic word vector evaluation

- Extrinsic evaluation of word vectors: All subsequent tasks in this class
- One example where good word vectors should help directly: named entity recognition: finding a person, organization or location

Model	Dev	Test	ACE	MUC7
Discrete	91.0	85.4	77.4	73.4
SVD	90.8	85.7	77.3	73.7
SVD-S	91.0	85.5	77.6	74.3
SVD-L	90.5	84.8	73.6	71.5
HPCA	92.6	88.7	81.7	80.7
HSMN	90.5	85.7	78.7	74.7
CW	92.2	87.4	81.7	80.2
CBOW	93.1	88.2	82.2	81.1
GloVe	93.2	88.3	82.9	82.2

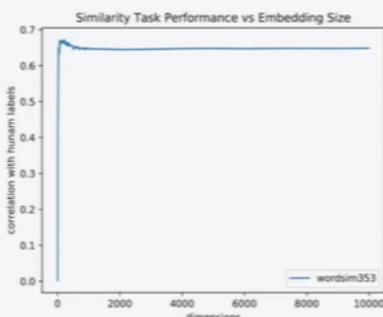
- Next: How to use word vectors in neural net models!

一篇最近很火的word2vec论文



On the Dimensionality of Word Embedding

[Zi Yin and Yuanyuan Shen, NeurIPS 2018]



(b) WordSim353 Test

<https://papers.nips.cc/paper/7368-on-the-dimensionality-of-word-embedding.pdf>

Using matrix perturbation theory, reveal a fundamental bias-variance trade-off in dimensionality selection for word embeddings

Table 3: PIP loss minimizing dimensionalities and intervals for GloVe on Text8 corpus

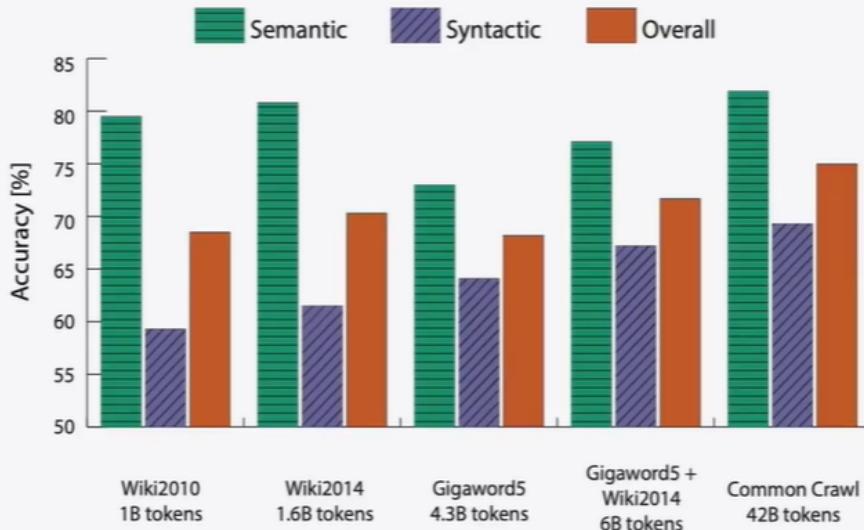
Surrogate Matrix	arg min	+5% interval	+10% interval	+20% interval	+50% interval	WS353	MT771	Analogy
GloVe (log-count)	719	[290,1286]	[160,1663]	[55,2426]	[5,2426]	220	860	560

~~So we're going way higher than we talked about before.~~

Wiki的text数据源更好

Analogy evaluation and hyperparameters

- More data helps, Wikipedia is better than news text!



Word senses and Word sense ambiguity

一个word可能对应了多种语义，word2vec是否包含了各种意义呢？

一种传统方法是对所有词window做聚类，如果一个词属于不同簇，那么这个词就有不同意义，即把不同意义看成不同词。

如果不看成不同的词：

Linear Algebraic Structure of Word Senses, with Applications to Polysemy (Arora, ..., Ma, ..., TACL 2018)

- Different senses of a word reside in a linear superposition (weighted sum) in standard word embeddings like word2vec
- $v_{\text{pike}} = \alpha_1 v_{\text{pike}_1} + \alpha_2 v_{\text{pike}_2} + \alpha_3 v_{\text{pike}_3}$
- Where $\alpha_1 = \frac{f_1}{f_1+f_2+f_3}$, etc., for frequency f
- Surprising result:
 - Because of ideas from *sparse coding* you can actually separate out the senses (providing they are relatively common)

50

tie				
trousers	season	scoreline	wires	operatic
blouse	teams	goalless	cables	soprano
waistcoat	winning	equaliser	wiring	mezzo
skirt	league	clinching	electrical	contralto
sleeved	finished	scoreless	wire	baritone
pants	championship	replay	cable	coloratura

上图是一种对不同意义的向量进行加权求和。