Lecture2 Word Vectors and Word Senses

上一节中主要讲了更新参数的一些方式，如果用梯度下降法的话运算量太大，时间太长，因此引入了随机梯度下降法，只针对某一个参数进行优化，大大提升了效率。又因为skip-gram中需要使用softmax来计算语料库中各个词出现的概率。并且字典的大小决定了神经网络的权重也会很大，更新权重非常耗时。这是一个非常耗时的计算。一般语料中都有几千万个单词，计算量非常的大，所以才要引入负采样来降低计算量。

负采样

这里再对上周提到的负采样进行完善。对于一个text：

I want a glass of orange juice to go along with my cereal.

- 假设当前采样到的单词orange，则juice与其组成**正样本对**，在下表中标记为1
- **负样本对**可以从字典中随机选择
- 假设当前采样到单词orange，则king与其组成**负样本对**，在下表中标记为0
- 随机选择k个负样本

context	word	target
orange	juice	1
orange	king	0
orange	book	0
orange	the	0
orange	off	0

现在有一个**有监督学习任务**

- 输入两个单词，比如orange和juice
- 输出他们是否属于context-target

要解决这个任务，我们有很多种训练方式，比如:

1）用构建的所有语料来训练一个复杂神经网络，输入context，输出其与字典中各个词是否为context-target的概率 (也就是skip-gram)。但这样的做法，训练模型的代价很高。

2）训练很多个二分类器，每个二分类器只需要计算给定context与k+1个词是否属于context-target。这样做的优点在于，每个二分类器都很容易计算，所以这比计算一个很大的softmax代价要低。

所以我们选择（2）进行训练，每个二分类器，只需要输入k+1个词。比如对句子中的每一个词（选中某一个词作为context）

就是对orange训练一个二分类器，其输入为（juice,king,book,the,off），输出为target（1,0,0,0,0）。此时只更新网络中对应部分的权重，其他部分的权重不更新。

Window based co-occurrence matrix

上一节主要讲了word2vec模型，它是一种基于local context window的直接预测模型，对于学习word vector，还有另一类模型是count based global matrix factorization（基于计数的全局矩阵分解）。

Example corpus:

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

我们构建的co-occurrence matrix

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

Problems with simple co-occurrence vectors

- Increase in size with vocabulary
- Very high dimensional: requires a lot of storage
- Subsequent classification models have sparsity issues
- Models are less robust

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

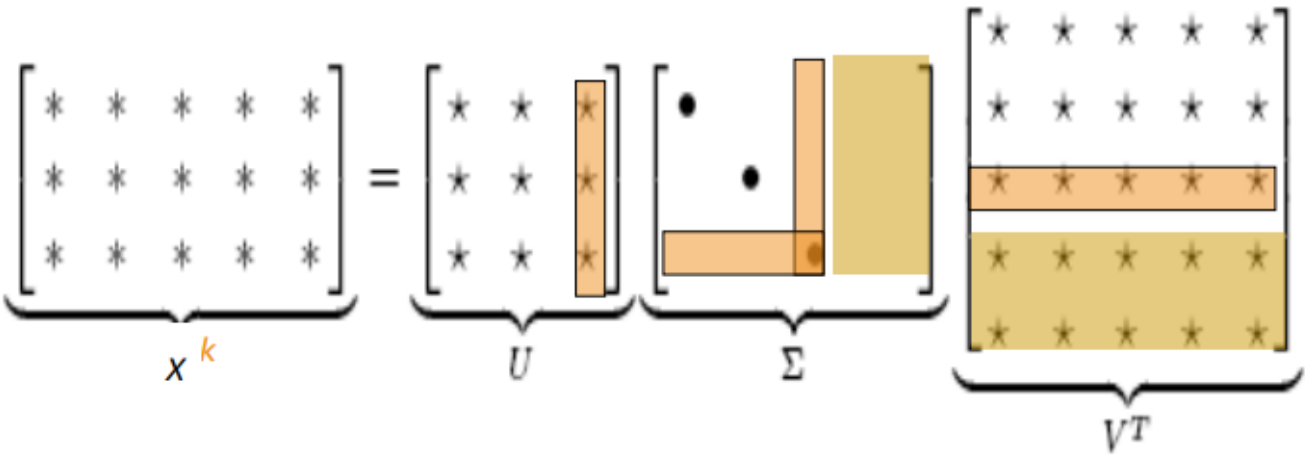
Method1: Dimensionality Reduction on matrix

这里用到了一个机器学习非常经典的算法：矩阵的奇异值分解又称为**SVD**，具体的求法这里不做详细展开，非常简单。需要知道的是它的用途是做降维，并且可以用三个小的矩阵来表示原始的大矩阵。PCA中就利用了SVD进行降维，这里需要知道两条SVD应用在PCA的性质：

- 左奇异矩阵可以用于行数的压缩（对数据进行压缩）
- 右奇异矩阵可以用于列数即特征维度的压缩

Singular Value Decomposition of co-occurrence matrix X

Factorizes X into $U\sum V^T$, where U and V are orthonomal

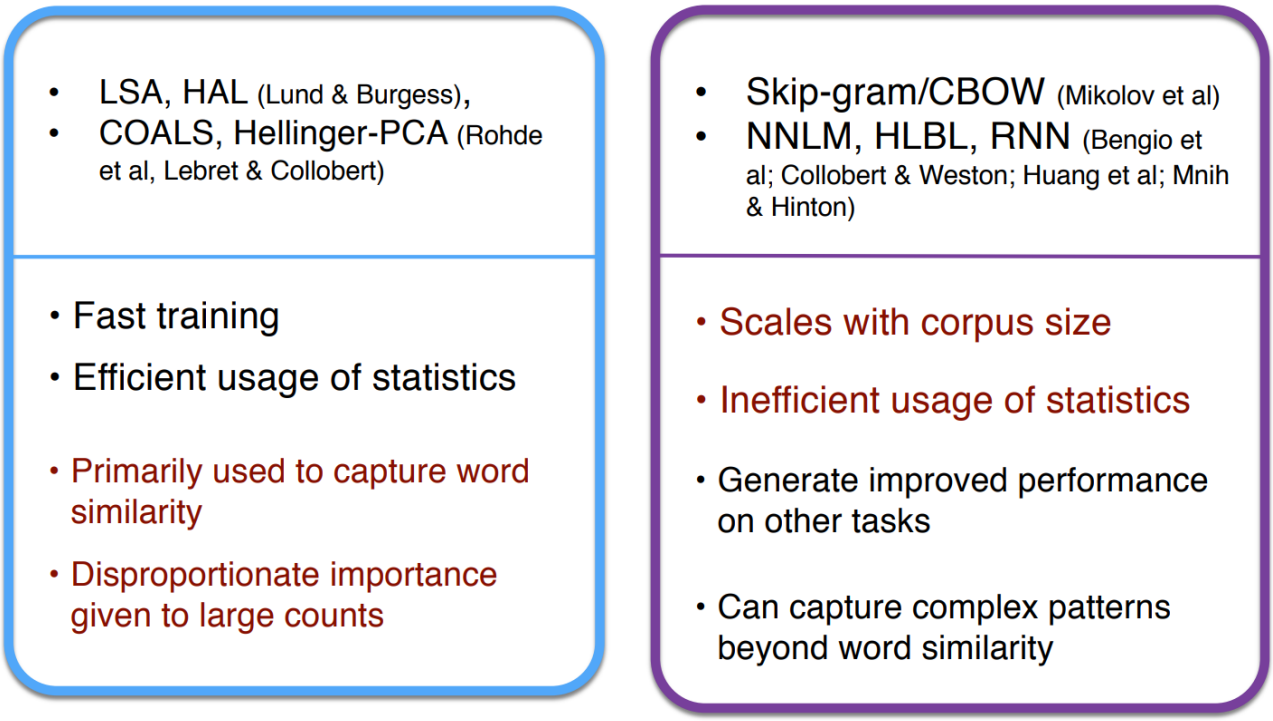


Reatain only k singular values, in order to generalize.

X is the best rank k approximation to X, in terms of least squares.

这里有一个细节，就是U和V是正交的，这个涉及到SVD计算过程，本质就是**实对称矩阵的特征向量是正交的。**

Count based vs. direct prediction



比较SVD这种count based模型与Word2vec这种direct prediction模型，它们各有优缺点：
Count based模型的有点是：

- 训练速度快，有效利用了统计信息

缺点是：

- 对于高频词汇较为偏向，并且仅能概括词组的相关性，而且有的时候产生的word vector对于解释词的含义如word analogy等任务效果不好

Direct Prediction优点是：

- 可以概括比相关性更为复杂的信息，进行word analogy等任务时效果较好

缺点是：

- 对统计信息利用的不够充分

因此就有了将两种优势结合的算法就是GloVe算法

问题

学到这里其实我不太理解这种基于计数的方式是如何变为word vector的? 只是知道有这种方法。

GloVe

为了更好理解如何有效利用window-based word-word co-occurrence count并能学习到词语背后的含义。我们先定义一些符号：

对于矩阵 X，X_{ij} 代表单词j出现在单词 i 上下文中的次数，则 $X_i = \sum_k X_{ik}$ 即代表所有出现在单词 i 的上下文中的单词次数。我们用 $P_{ij} = P(j|i) = X_{ij}/X_i$ 代表单词 j 出现在单词 i 上下文的概率。

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

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

例如我们想区分热力学上不同状态ice冰与蒸汽steam，它们之间的关系可通过与不同的单词 x 的co-occurrence probability的比值来描述，例如对于solid固态，虽然 $P(solid|ice)$ 与 $P(solid|steam)$ 本身很小，不能透露有效的信息，但是它们的比值 $P(solid|ice)/P(solid|steam)$ 却较大，因为solid更常用来描述 ice 的状态而不是steam的状态，所以在ice的上下文中出现几率较大，对于gas则恰恰相反，而对于water这种描述icw与steam均可或者fashion这种与两者都没什么联系的单词，则比值接近于1。所以相较于单纯的co-occurrence probability，实际上co-occurrence probability的**相对比值**更有意义。

Encoding meaning in vector differences

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

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

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

推导GloVe算法

基于对于以上概率比值的观察，我们假设模型的函数有如下形式：

$$F\left(w_i, w_j, \tilde{w_k}\right) = \frac{P_{ik}}{P_{jk}}$$

其中 $\tilde{w}$ 代表了context vector，如上例中的solid, gas, water, fashion等。w_i, w_j 则是我们要比较的两个词汇，如上例中的ice, steam。

- $\frac{P_{ik}}{P_{jk}}$ 考察了 i, j, k 三个word两两之间的相似关系，不妨单独考察 i, j 两个词和他们的词向量 w_i, w_j，线性空间中的相似关系自然想到的是两个向量的差 $(v_i - v_j)$。所以 F 的函数形式可

以是

$F\left(w_i-w_j,\tilde{w}_k\right)=\frac{P_{ik}}{P_{jk}}$

- $\frac{P_{ik}}{P_{jk}}$ 是一个标量，而 F 是作用在这两个向量上的，向量和标量之间的关系自然想到了使用内积。所以 F 的函数形式可以进一步确定为

  $F\left(\left(w_i-w_j\right)^T\tilde{w}_k\right)=F\left(w_i^Tw_k-w_j^Tw_k\right)=\frac{P_{ik}}{P_{jk}}$

- 对于上述公式来讲，左边是差，右边是商，那么模型将通过 F 取 exp 将差和商联系起来

  KaTeX parse error: Can't use function '$' in math mode at position 1: $\exp \left(w_{...

- 现在只要让分子与分子相等，分母与分母相等即可

  $\exp\left(w_i^Tw_k\right)=P_{ik},\exp\left(w_j^Tw_k\right)=P_{jk}$

- 因此有公式

  $w_i^Tw_k=\log X_{ik}-\log X_i$

- 作为向量，交换 i 和 k 的顺序是相等的，但是公式右侧交换 i 和 k 的顺序并不相等，为了解决对称性的问题，模型引入了两个bias项，从而模型变成了

  $w_i^Tw_k+b_i+b_k=\log X_{ik}$

- 其中 b_i 项包含了 $logX_k$ 项，为了保持模型的对称性，又加入了 b_k 项

- 上面的公式只是理想情况下相等，在实际实验中左右两边只能要求接近。从而就有了代价函数

  $J=\sum_{i,k=1}^V\left(w_i^T\tilde{w}_k+b_i+\tilde{b}_k-\log X_{ik}\right)^2$

- 根据经验，如果两个词共同出现的次数越多，那么这两个词在代价函数中的影响就应该越大，所以可以根据两个词共同出现的次数设计一个权重项来对代价函数中的每一项进行加权

  $J=\sum_{i,k=1}^Vf\left(X_{ik}\right)\left(w_i^T\tilde{w}_k+b_i+\tilde{b}_k-\log X_{ik}\right)^2$

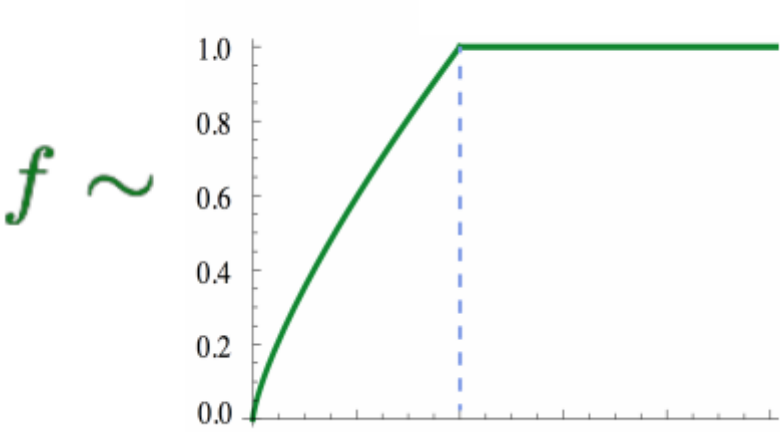
- 模型认为权重函数 f 应该符合以下三个特点

  - $f(0)=0$ 如果两个词没有共同出现过，权重就是0
  - $f(x)$ 必须是非减函数，因为两个词出现的次数多，反而权重变小了，违反了设置权重项的初衷
  - $f(x)$ 对于较大的 x 不能取太大的值（就像汉语中“的”这个字，在很多文章中出现过很多次，但是其在文中的重要程度非常小）

- 综合上述，$f(x)$ 定义为：

$$f\left(x\right)=\begin{cases}\left(\frac{x}{x_{\max}}\right)^{\alpha},if\ x<x_{\max}\\1,otherwise\end{cases}$$

根据经验，GloVe作者认为 $x_{max}=100,\alpha=3/4$ 是一个比较好的选择。



GloVe的优势

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