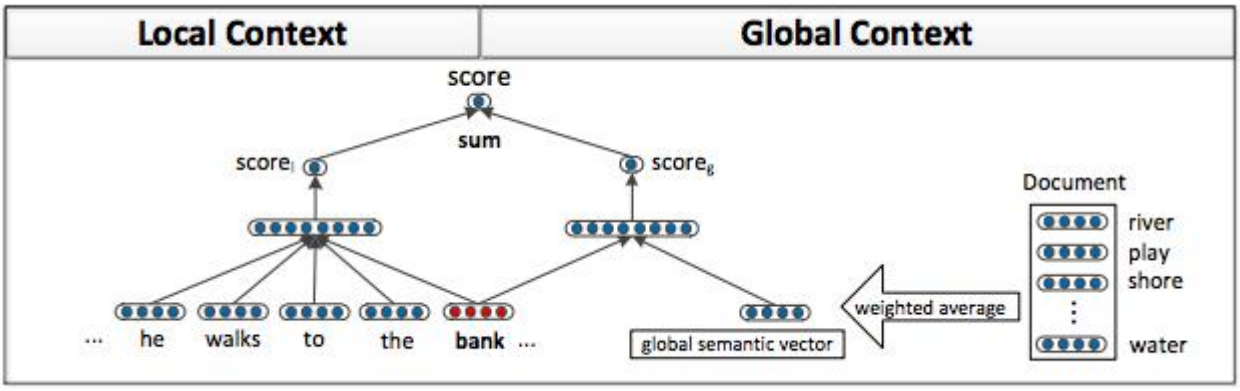
Glove

Global Vectors 的目的就是做到对 word 的表示，即 sementic 的表达效果好，syntactic 的表达效果也好。



GloVe 模型

GloVe: Global Vectors。

模型输入：语料库 corpus

模型输出：每个词的表示向量

先介绍两个其他方法：

一个是基于奇异值分解（SVD）的 LSA 算法，该方法对 term-document 矩阵（矩阵的每个元素为 tf-idf，词项-文档矩阵（term frequency times inverse document frequency）矩阵中的每个元素值代表了相应行上的词项对应于相应列上的文档的权重,即这个词对于这篇文章来说的重要程度。）进行奇异值分解，从而得到 term 的向量表示和 document 的向量表示。此处使用的 tf-idf 主要还是 term 的全局统计特征。

另一个方法是 word2vec 算法，该算法可以分为 skip-gram 和 continuous bag-of-words（CBOW）两类,但都是基于局部滑动窗口计算的。即，该方法利用了局部的上下文特征（local context）

LSA 和 word2vec 作为两大类方法的代表，一个是利用了全局特征的矩阵分解方法，一个是利用局部上下文的方法。

GloVe 模型就是将这两中特征合并到一起的，即使用了语料库的全局统计（overall statistics）特征，也使用了局部的上下文特征（即滑动窗口）。为了做到这一点 GloVe 模型引入了 Co-occurrence Probabilities Matrix。

由 Co-occurrence Probabilities Matrix 可以看出 Ratio=PikPjkRatio=PikPjk 的取值是有一定的规律的。

ratioi,j,k 的值	单词 j,k 相关	单词 j,k 不相关
单词 i,k 相关	趋近 1	很大
单词 i,k 不相关	很小	趋近 1

也就是说 Ratio 值能够反映 word 之间的相关性，**而 GloVe 模型就是利用了这个 Ratio 值**。

再明确一下，**GloVe 模型的目标就是获取每一个 word 的向量表示 v**。不妨假设现在已经得到了 word i,j,k 的词向量 w_i, w_j, w_k。**GloVe 认为，这三个向量通过某种函数的作用后所呈现出来的规律和 Ratio=P_{ik}/P_{jk} 具有一致性，即相等，也就可以认为词向量中包含了共现概率矩阵中的信息。**

假设这个未知的函数是 F,则:

$$F(w_i, w_j, w_k)=P_{ik}/P_{jk}$$

此处可以类比 word2vec 的基本思想（以基于哈弗曼树的 CBOW 为例），假设 word i，和其 context words 的词向量已知，通过一层神经网络作用于 context words 的向量得到的结果与 word i 在哈夫曼树中的位置具有一致性。

模型推导

公式

$$F(w_i, w_j, w_k)=P_{ik}/P_{jk}$$

右侧的 P_{ik}/P_{jk} 可以通过统计求的;

左侧的 wi,wj,wk 是我们模型要求的量;

同时函数 F 是未知的。

推导过程

1. 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 两个词和他们词向量 wi,wj，线性空间中的相似关系自然想到的是两个向量的差(vi-vj)。 所以 F 函数的形式可以是
$$F(w_i-w_j, w_k)=P_{ik}/P_{jk}$$
2. P_{ik}/P_{jk} 是一个标量，而 F 是作用在两个向量上的，向量和标量之间的关系自然想到了使用内积。所以 F 函数的形式可以进一步确定为
$$F((w_i - w_j)^T w_k) = F(w_i^T w_k - w_j^T w_k) = \frac{P_{ik}}{P_{jk}}$$
3. 左边是差，右边是商，模型通过将 F 取作 exp 来将差和商关联起来
$$exp(w_i^T w_k - w_j^T w_k) = \frac{exp(w_i^T w_k)}{exp(w_j^T w_k)} = \frac{P_{ik}}{P_{jk}}$$

4. 现在只需要让分子分母分别相等上式就能够成立，所以

$$exp(w_i^T w_k) = P_{ik}, exp(w_j^T w_k) = P_{jk}$$

5. 所以只需要在整个文本库中考察，即

$$w_i^T w_k = log(\frac{X_{ik}}{X_i}) = logX_{ik} - logX_i$$

6. 作为向量，交换 i 和 k 的顺序 $w_i^T w_k$ 和 $w_k^T w_i$ 是相等的，即公式左边对于 i 和 k 的顺序是不敏感的，但是公式右边交换 i 和 k 的顺序 $logX_{ik}-logX_i \neq logX_{ki}-logX_k$。为了解决这个对称性问题，模型引入了两个偏执项 b_i, b_k，从而将模型变成了

$$logX_{ik} = w_i^T w_k + b_i + b_k$$

7. 上面的公式只是理想情况下，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（cost function）

$$J = \sum_{ik} (w_i^T w_k + b_i + b_k - logX_{ik})^2$$

8.

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约大，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：

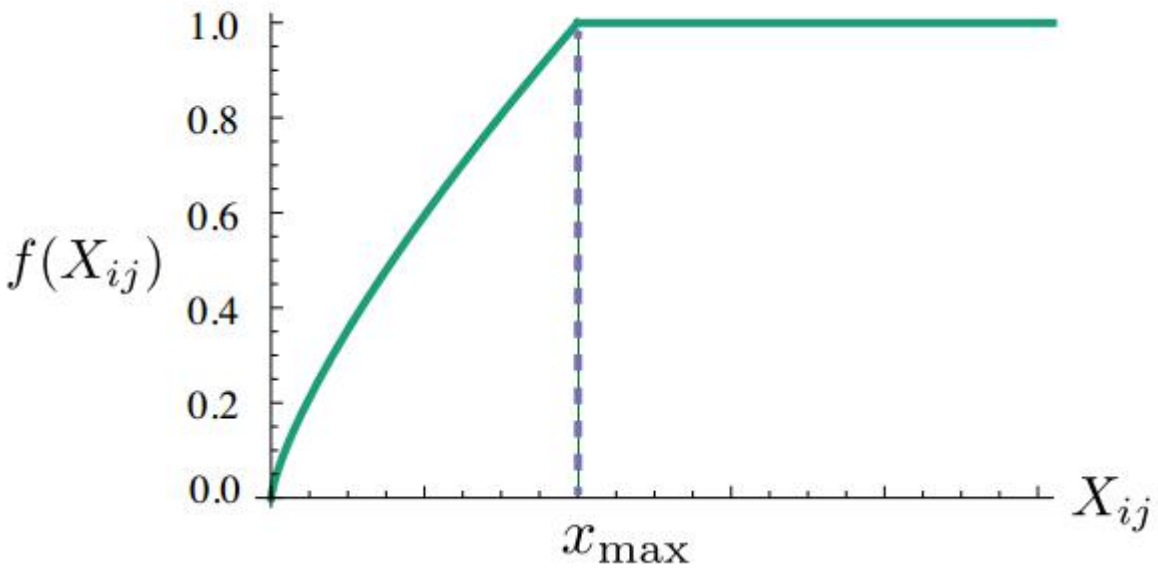
$$J = \sum_{ik} f(X_{ik})(w_i^T w_k + b_i + b_k - logX_{ik})^2$$

9.

模型认为权重函数 ff 应该符合以下三个特点，1. f(0)=0 （如果两个词没有共同出现过，权重就是 0）；2. f(x)必须是非减函数（两个词共同出现的次数多，反而权重变小了，违反了设置权重项的初衷）；3. f(x)对于较大的 x 不能取太大的值（就像是汉语中“的”这个字，在很多文章中都会出现很多次，但是其在文中的重要程度非常小）。综合这三条特点的 f(x)定义为：

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

10.



根据经验，xmax=100，α=34 是一个比较好的选择。