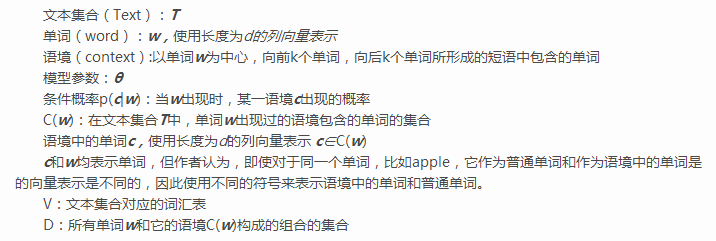
# Word2vec代价函数及优化问题

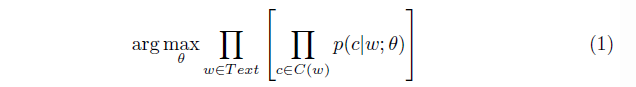
# 重点问题

定义及符号

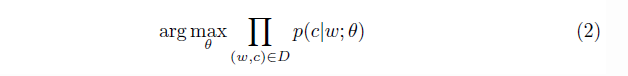


skip-gram

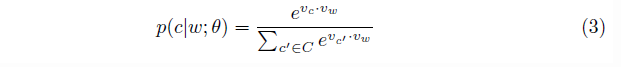
目标：寻找参数集合来最大化下面条件概率的乘积：



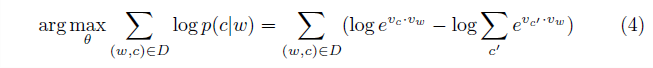
即：



这里我们利用softmax进行形式化处理，使得条件概率转化为：



其中vc和vw分别是c和w的列向量，长度为d。C是所有语境中的单词构成的集合，等同于词汇表V（集合有互斥性，而这里的V应该是包含重复单词的，因此用“集合”这个词是否不太恰当？）。参数就是vc和vw的每一维度的具体取值，参数的总数为|C|×|V|×d。将3带入2，两边取对数可得：

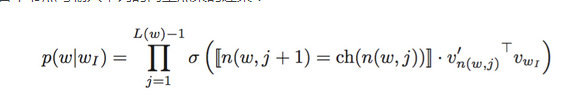


**作者认为，通过训练得到的单词的向量表示vw能够使得条件概率p(c|w)最大化，则vw是w的好的表示。这里潜在的基本假设是：相似的单词拥有相似的语境。换言之，特定的语境只有确定的语义才能够与之匹配。通过最大化条件概率，使得单词和语境之间的对应关系最大化，进而满足了基本假设：相似的单词拥有相似的语境。而满足条件概率最大化的单词矢量，也就成为了单词语义的合理表示。**

一般来说，使用上式计算过程较为复杂，作者使用了hierarchical softmax来近似softmax，并用霍夫曼树来构建hierarchical softmax

hierarchical softmax

主要思想是，构造一颗二叉树，使得单词表中的单词均为二叉树的叶子节点。该叶子节点的概率为：从二叉树的根节点到该叶子节点的路径上各个节点与输入单词的向量点乘的连乘：

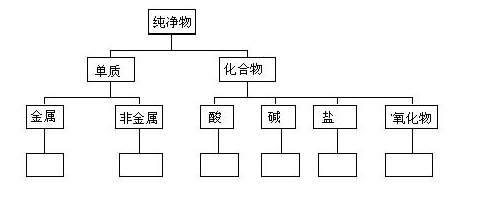


并选择任意方向，如向左为正方向，向右为负方向，利用sigmoid作为概率激活函数。

这样计算每个输入时softmax的计算复杂度由原来的O(n)变成了log(n)。

其实上述分层softmax利用了二分类的思想。每次为一个单词从两个抽象分类中选择其中一种，利用sigmoid函数为两个类分配概率。一直对单词分类直到标定清楚一个单词的具体表示。

就像这样：只不过，下图每个决策点不全是二分类。

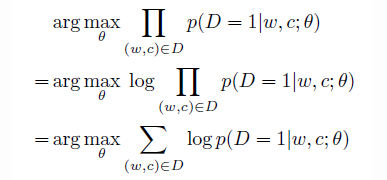


因此训练结果中的二叉树的非叶子节点都表示某一抽象类。

构造二叉树的方法有很多，不同构造方法会对计算效率和准确率有略微的影响。   
一种常见方法是根据单词的词频构建huffman编码树。可以利用编码01标记每次的正负概率。更重要的思想是，是高频词汇路径短，提高上式连乘速度。

Negative Sampling

对于一个单词、语境组合（w,c），使用p(D=1|w,c)来表示这个组合存在于T中的概率，对应的p(D=0|w,c) = 1-p(D=1|w,c)，表示（w,c）不在T中的概率。与前文类似，假设集合θ是控制p(D=1|w,c)分布的参数，那么此时的目标是寻找参数集合θ来最大化（w,c）存在于T中的概率：



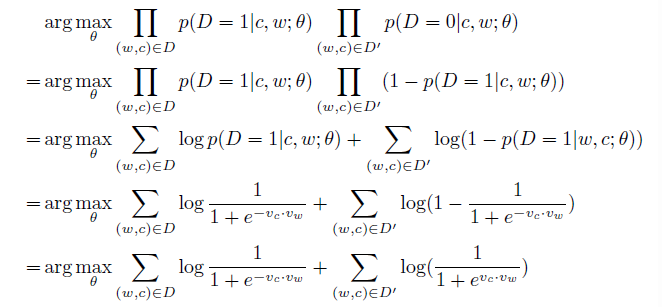
同样，使用softmax来量化p(D=1|w,c;θ)：



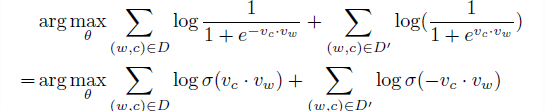
因此，最终的目标函数为：



为了使目标函数最大化，有一个很简单的方法，即使得vc=vw,且vc·vw=K。当K足够大时，可以得到p(D=1|w,c)=1，从而达到目标函数最大化。因此，为了所有的矢量有相同的值（？？？），作者生成了一个数据集D'，D'中的所有单词、语境组合都不存在于T中，这些样例被称之为反例（negative examples）,而获得反例的采样方法被称之为反例采集（negative-sampling）。引进了反例之后的目标函数演变为：



假设σ(x)=1/(1+e-x)，则：



和式4进行比较，可以明显的看到累加嵌套的消除。两者最大的区别是式4根据条件概率p(c|w)进行建模而此处根据联合概率p(D=1|w,c)进行建模

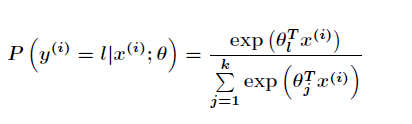
参考文献：<http://blog.csdn.net/fangpinlei/article/details/52200832>

<http://blog.sina.com.cn/s/blog_66a6172c0102v1k9.html>

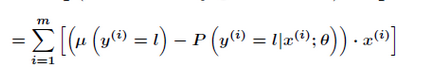
# 补充问题

softmax回归

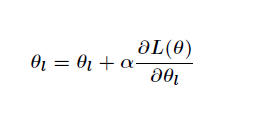
总的来说，softmax的核心公式：



跟Logistic回归一样，softmax也可以用梯度下降法或者牛顿迭代法求解，对对数似然函数求偏导数，得到



然后我们可以通过梯度上升法来更新参数



在softmax回归中直接用上述对数似然函数是不能更新参数的，因为它存在冗余的参数，通常用牛顿方法中的**Hessian**

矩阵也不可逆，是一个非凸函数，那么可以通过添加一个权重衰减项来修改代价函数，使得代价函数是凸函数，并且得到的Hessian矩阵可逆。

参考文献：<http://blog.csdn.net/acdreamers/article/details/44663305>