word2vec

Tylerwang

2015.4.30

Word2vec是什么

- word2vec是一个将单词转换成向量形式的工具。通过转换，可以把对文本内容的处理简化为向量空间中的向量运算，计算出向量空间上的相似度，来表示文本语义上的相似度。

词向量

- 把自然语言中的一个词表示成一个向量
  - One-hot Representation

  例如：
    - “话筒”表示为 [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 ...]
    - “麦克”表示为 [0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 ...]
  - 存在两个问题
    - 维数灾难
    - 词汇鸿沟：任意两个词之间都是孤立的，不能体现词和词之间的关系

词向量

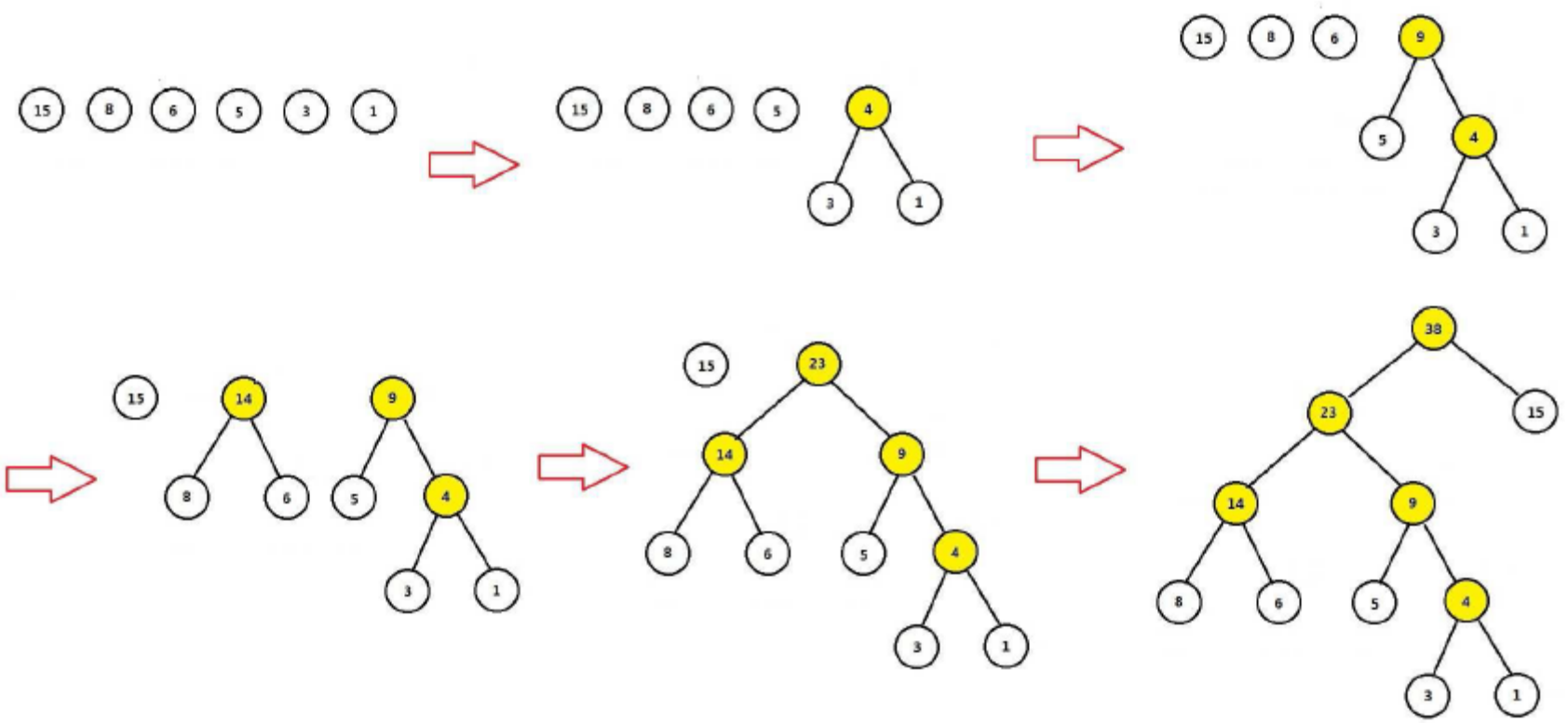
- Distributional Representation
  - 将词表示为：
    - 通过训练将每个词映射成K维实数向量，通过词之间的距离（比如cosine相似度、欧氏距离等）来判断它们之间的语义相似度。
    - 具体表示为如[0.792, −0.177, −0.107, 0.109, 0.542, ...]，常见维度几十到几百
  - 可解决“词汇鸿沟”问题
    - 可以通过计算向量之间的距离（欧式距离、余弦距离等）来体现词与词的相似性

    又叫Word Represention”或 Word Embedding

霍夫曼树

给定 n 个权值作为 n 个叶子结点，构造一棵二叉树，若它的带权路径长度达到最小，则称这样的二叉树为**最优二叉树**，也称为 **Huffman 树**.

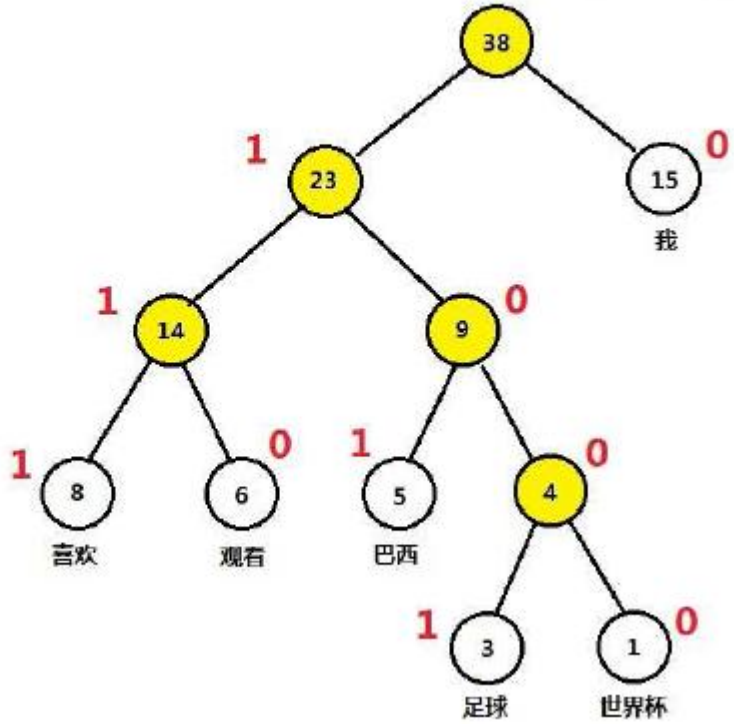
“我”、“喜欢”、“观看”、“巴西”、“足球”、“世界杯”这六个词出现的次数分别为 *15, 8, 6, 5, 3, 1*. 请以这 *6* 个词为叶子结点，以相应词频当权值，构造一棵 *Huffman* 树.



霍夫曼编码

利用 Huffman 树设计的二进制前缀编码, 称为 **Huffman 编码**

这六个词的 Huffman 编码分别为 0, 111, 110, 101, 1001 和 1000.



词频越高，编码的长度越短，从根节点到相应词的路径越短

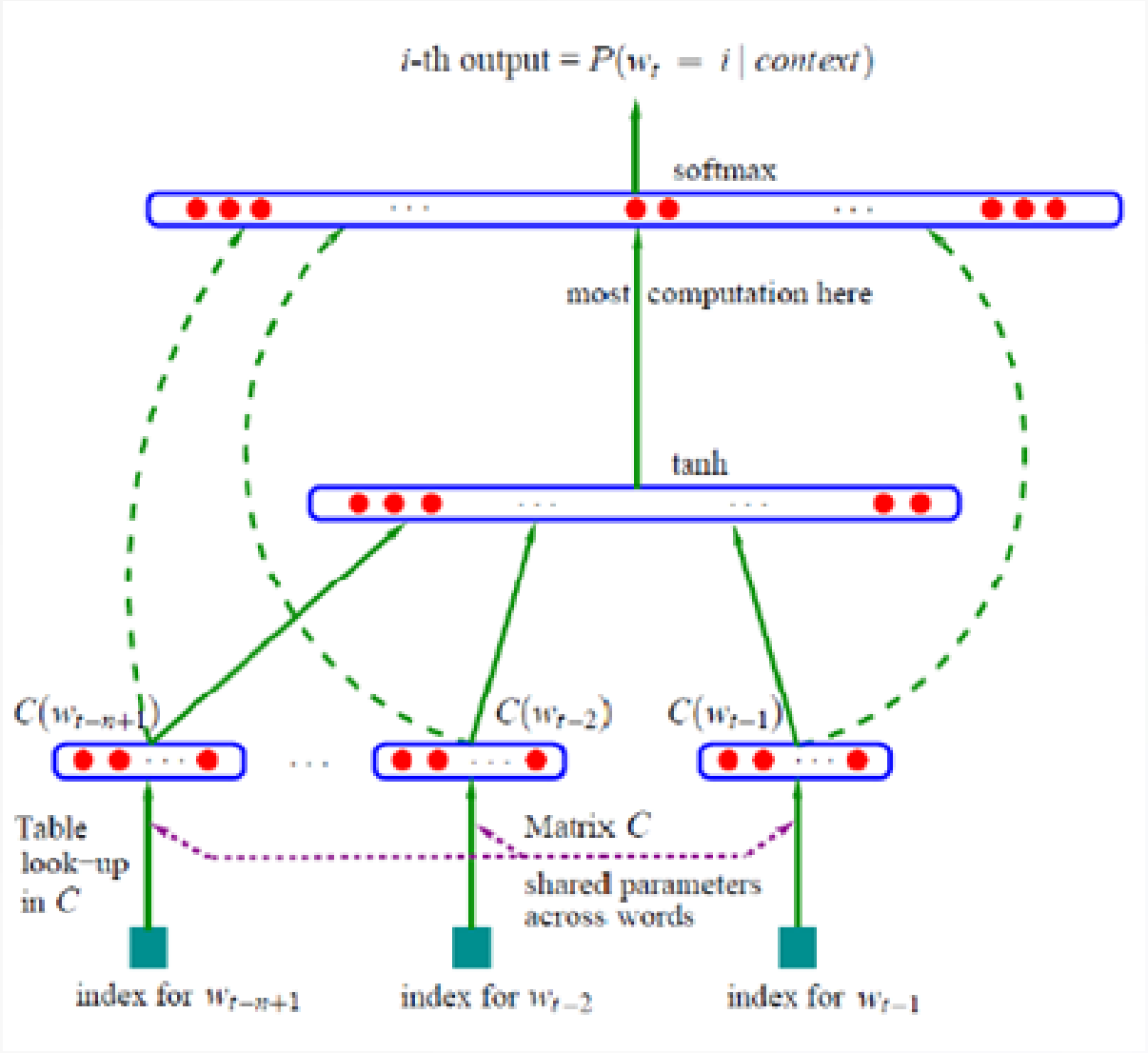
语言模型

- 判断一句话是不是正常人说出来的，用数学符号描述为
  - 给定一个字符串"w1,w2,...,wt",计算它是自然语言的概率 $p(w_1, w_2, ..., w_t)$ ，一个很简单的推论是

    $$p(w_1, w_2, ..., w_t) = p(w_1) \cdot p(w_2 \mid w_1) \cdot p(w_3 \mid w_1, w_2) \cdot ... \cdot p(w_t \mid w_1, w_2, ..., w_{t-1})$$

  - 例如，有个句子"大家,喜欢,吃,苹果"
    - P(大家，喜欢，吃，苹果)=p(大家)p(喜欢|大家)p(吃|大家,喜欢)p(苹果|大家,喜欢,吃)
  - 简单表示为 $p(s) = p(w_1, w_2, ..., w_T) = \prod_{i=1}^{T} p(w_i \mid Context_i)$
- 计算 $p(w_i \mid Context_i)$ 问题

神经网络语言模型


i-th output = $P(w_t = i \mid context)$
softmax
most computation here
tanh
$C(w_{t-n+1})$
$C(w_{t-2})$
$C(w_{t-1})$
Table look-up in C
Matrix C
shared parameters across words
index for w_{t-n+1}
index for w_{t-2}
index for w_{t-1}

神经网络语言模型

- 输入是向量，也会被更新，每次查询C矩阵
- 论文中输入向量间用catenation, word2vec用sum
- 输入层和输出层之间构建一层传递关系，可更快传导输入word的变化
- 可以选择双曲正切或者sigmoid

神经网络语言模型

- N: 词语window大小，当前词前后多少个词
- D: 词向量维度
- H: hidden layer节点数
- V: 词汇表vocabulary大小
- 复杂度：N * D + N * D * H + N *D * V + H * V
- 第三第四项复杂度最高，word2vec去掉第三项，优化第四项(用hs或者neg)

word2vec原理

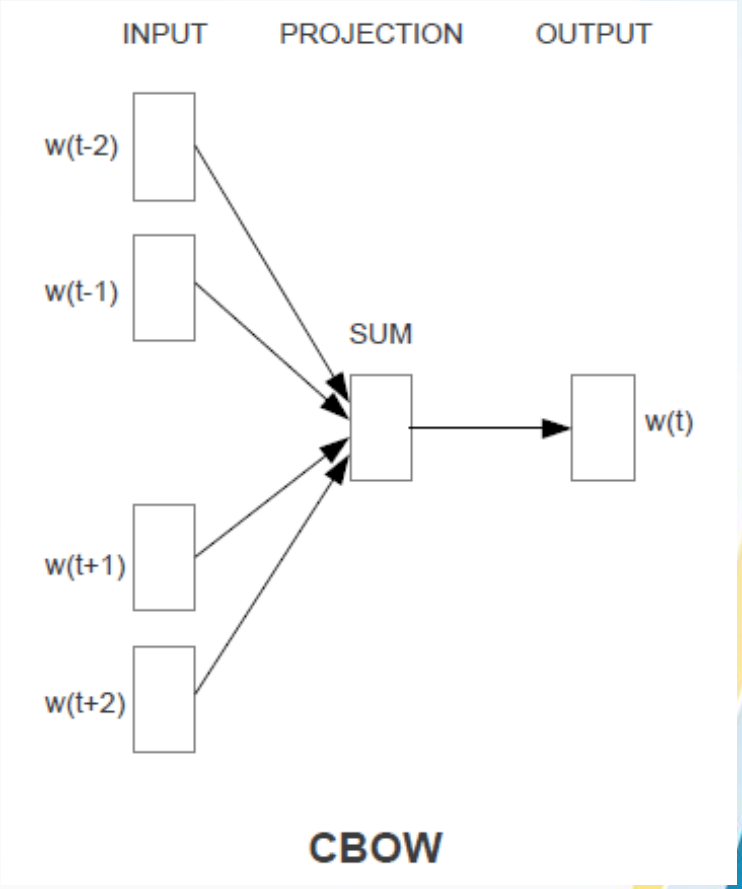
- 两种模型，两种加速策略

模型	CBOW		Skip-Gram	
方法	Hierarchical Softmax	Negative Sampling	Hierarchical Softmax	Negative Sampling

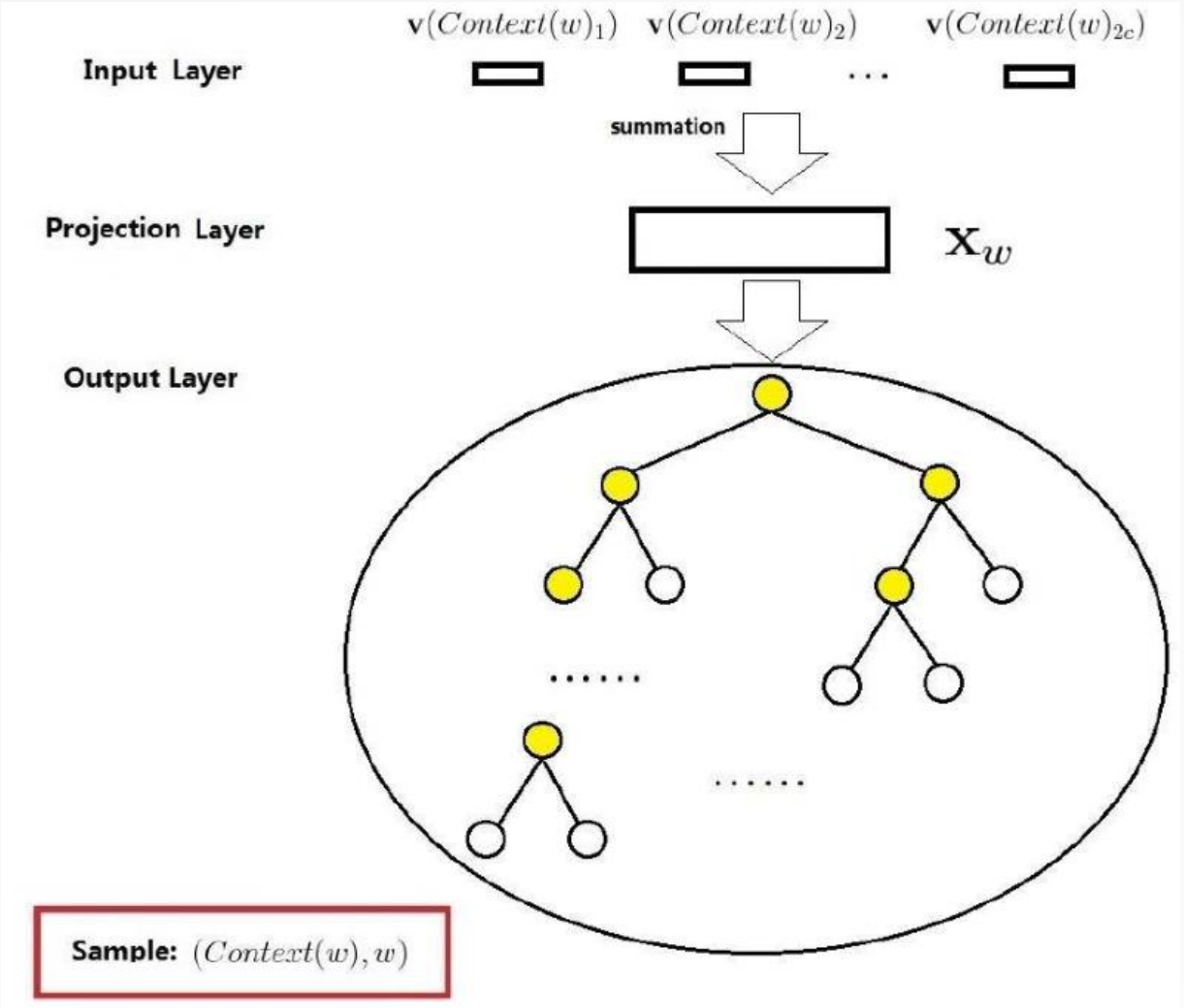
CBOW模型+Hierarchical Softmax方法

- CBOW模型
  - INPUT:输入层
  - PROJECTION:投影层
  - OUTPUT:输出层
  - w(t):当前词语（向量）
  - w(t-2),w(t-1),w(t+1),w(t+2):当前词语的上下文
  - SUM:context的累加和

$$\mathcal{L} = \sum_{w \in \mathcal{C}} \log p(Context(w)|w)$$



CBOW模型+Hierarchical Softmax方法（续）



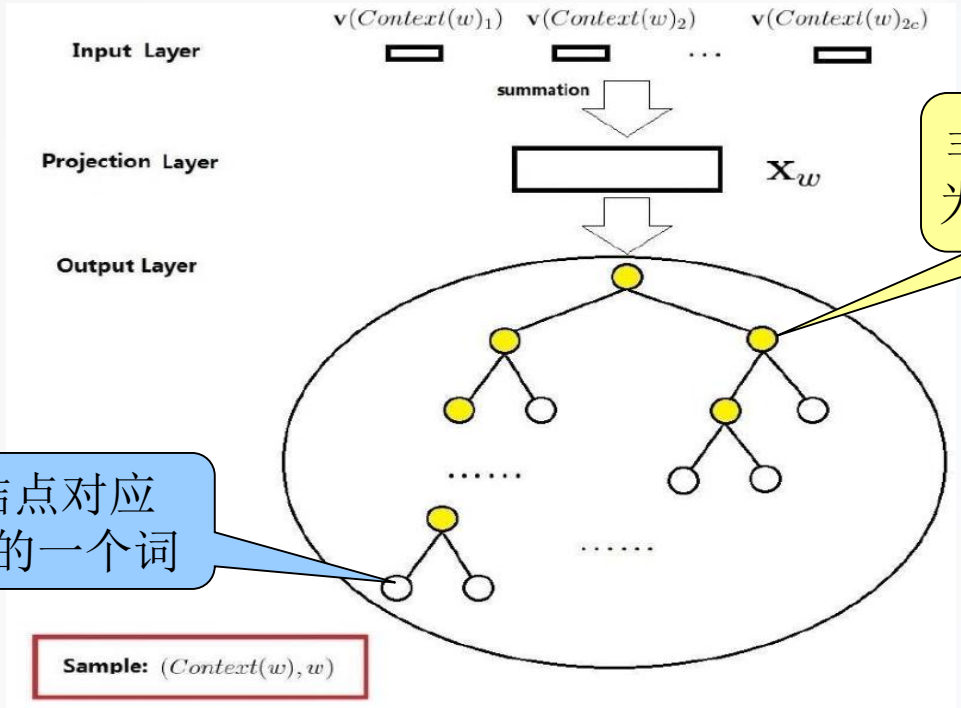
CBOW模型+Hierarchical Softmax方法（续）

1. **输入层**: 包含 $Context(w)$ 中 $2c$ 个词的词向量 $\mathbf{v}(Context(w)_1), \mathbf{v}(Context(w)_2), \cdots, \mathbf{v}(Context(w)_{2c}) \in \mathbb{R}^m$. 这里, m 的含义同上表示词向量的长度.

2. **投影层**: 将输入层的 $2c$ 个向量做求和累加, 即 $\mathbf{x}_w = \sum_{i=1}^{2c} \mathbf{v}(Context(w)_i) \in \mathbb{R}^m$.

3. **输出层**: 输出层对应一棵二叉树, 它是以语料中出现过的词当叶子结点, 以各词在语料中出现的次数当权值构造出来的 Huffman 树. 在这棵 Huffman 树中, 叶子结点共 N $(= |\mathcal{D}|)$ 个, 分别对应词典 $\mathcal{D}$ 中的词, 非叶子结点 $N-1$ 个 (图中标成黄色的那些结点).

目标：$p(w_i \mid Context_i)$



CBOW模型+Hierarchical Softmax方法（续）

- 句子：我,喜欢,观看,巴西,足球,世界杯
- w=足球

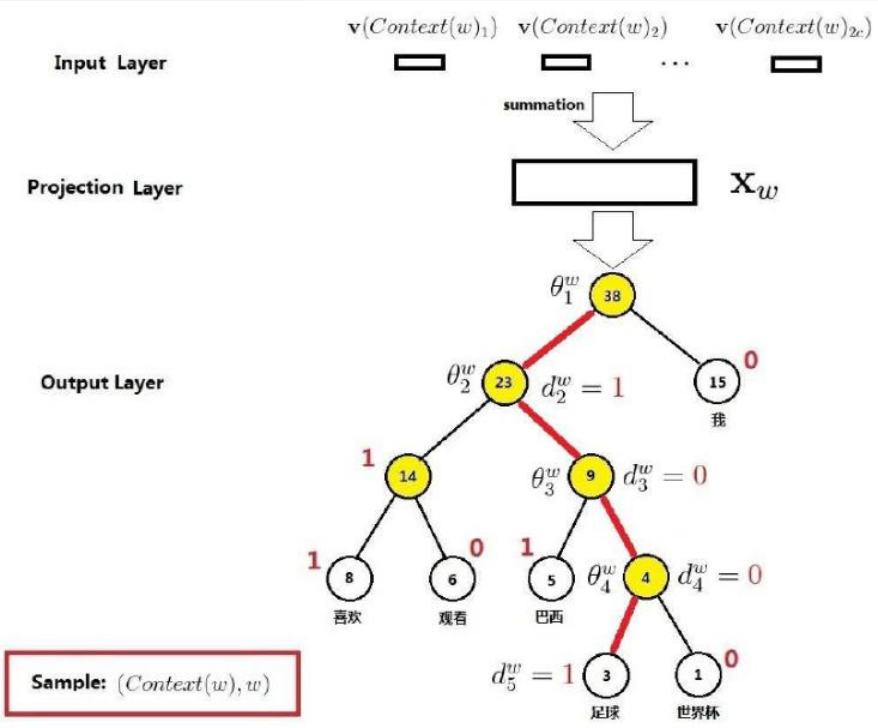
d_j^w ：编码（1或0）

θ_j^w ：非叶子节点向量

二分类

d_j^w ：正负类（1：负类,0：正类）

θ_j^w ：类别向量



CBOW模型+Hierarchical Softmax方法（续）

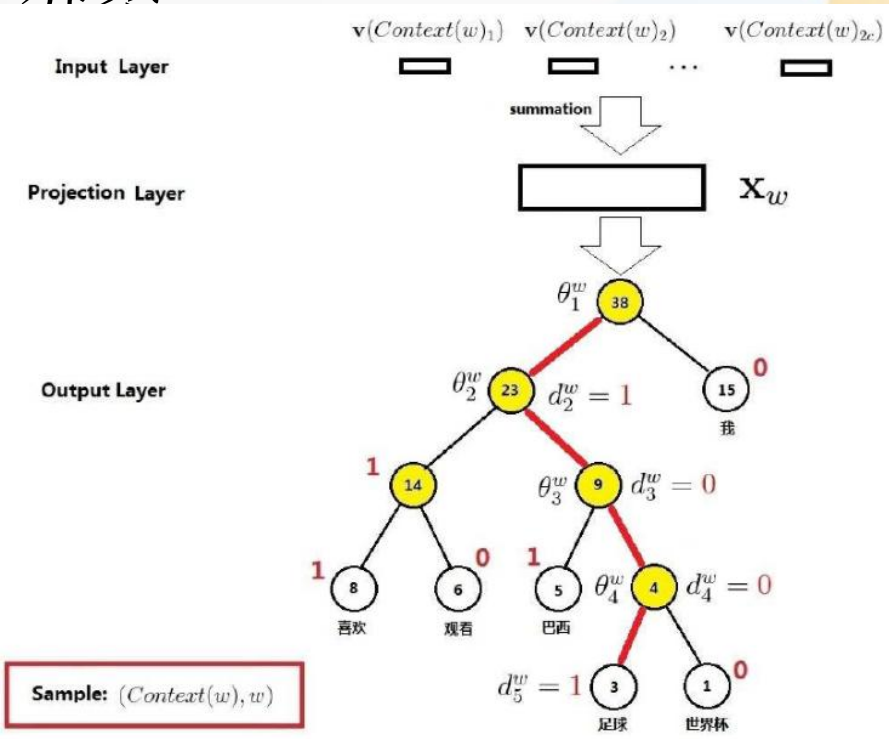
- 正类概率: $\sigma(\mathbf{x}_w^\top\theta) = \dfrac{1}{1+e^{-\mathbf{x}_w^\top\theta}}$,
- 负类概率: $1-\sigma(\mathbf{x}_w^\top\theta)$,
- "足球" 叶子节点经过4次二分类
- 每次分类结果对应的概率为

$$\text{第 1 次:}\quad p(d_2^w|\mathbf{x}_w,\theta_1^w) = 1-\sigma(\mathbf{x}_w^\top\theta_1^w);$$

$$\text{第 2 次:}\quad p(d_3^w|\mathbf{x}_w,\theta_2^w) = \sigma(\mathbf{x}_w^\top\theta_2^w);$$

$$\text{第 3 次:}\quad p(d_4^w|\mathbf{x}_w,\theta_3^w) = \sigma(\mathbf{x}_w^\top\theta_3^w));$$

$$\text{第 4 次:}\quad p(d_5^w|\mathbf{x}_w,\theta_4^w) = 1-\sigma(\mathbf{x}_w^\top\theta_4^w),$$



- 由Context("足球")预测"足球"出现的概率

$$p(\text{足球}|Contex(\text{足球})) = \prod_{j=2}^{5} p(d_j^w|\mathbf{x}_w,\theta_{j-1}^w).$$

CBOW模型+Hierarchical Softmax方法（续）

- 对于词典中的每个词w有，l^w 结点个数

$$p(w|Context(w))=\prod_{j=2}^{l^w} p(d_j^w|X_w,\theta_{j-1}^w)$$

- 其中，$$p(d_j^w|X_w,\theta_{j-1}^w)=\begin{cases}\sigma(X_x^T\cdot\theta_{j-1}^w), d_j^w=0;\\ 1-\sigma(X_x^T\cdot\theta_{j-1}^w), d_j^w=1.\end{cases}$$
- 或者表示为 $p(d_j^w|X_w,\theta_{j-1}^w)=[\sigma(X_w^T\cdot\theta_{j-1}^w)]^{1-d_j^w}\cdot[1-\sigma(X_w^T\cdot\theta_{j-1}^w)]^{d_j^w}$
- 对于由S个句子组成的语料库C有

$$L(X,\theta)=\prod_{s\in C}\prod_{w\in s} p(w|Context(w))=\prod_{s\in C}\prod_{w\in s}\prod_{j=2}^{l^w} p(d_j^w|X_w,\theta_{j-1}^w)$$

- 取对数似然函数

参数1

$$\log L(X,\theta)=\sum_{s\in C}\sum_{w\in s}\sum_{j=2}^{l^w}\log p(d_j^w|X_w,\theta_{j-1}^w)$$

参数2

$$=\sum_{s\in C}\sum_{w\in s}\sum_{j=2}^{l^w}[(1-d_j^w)\cdot\log\sigma(X_w^T\cdot\theta_{j-1}^w)+d_j^w\cdot\log(1-\sigma(X_w^T\cdot\theta_{j-1}^w))]$$

CBOW模型+Hierarchical Softmax方法（续）

- 梯度下降法进行求解
  - 令 $f(w,j)=-(1-d_j^w)\cdot\log\sigma(X_w^T\cdot\theta_{j-1}^w)-d_j^w\cdot\log(1-\sigma(X_w^T\cdot\theta_{j-1}^w))$
  - f(w,j)关于 θ_{j-1}^w 和 X_w 的梯度分别为

$$\frac{\partial f(w,j)}{\partial\theta_{j-1}^w}=-[1-d_j^w-\sigma(X_w^T\cdot\theta_{j-1}^w)]\cdot X_w$$

$$\frac{\partial f(w,j)}{\partial X_w}=-[1-d_j^w-\sigma(X_w^T\cdot\theta_{j-1}^w)]\cdot\theta_{j-1}^w$$

  - 更新公式

$$\theta_{j-1}^w:=\theta_{j-1}^w-\eta\cdot\frac{\partial f(w,j)}{\partial\theta_{j-1}^w}\qquad V(\tilde{w}):=V(\tilde{w})-\eta\cdot\sum_{j=2}^{l^w}\frac{\partial f(w,j)}{\partial X_w},\ \tilde{w}\in Context(w)$$

- Xw是sum得来的，有人提出应均分梯度到context，即

$$\mathbf{v}(\tilde{w}):=\mathbf{v}(\tilde{w})+\frac{\eta}{|Context(w)|}\sum_{j=2}^{l^w}\frac{\partial\mathcal{L}(w,j)}{\partial\mathbf{x}_w},\quad\tilde{w}\in Context(w)$$

为什么建huffman树

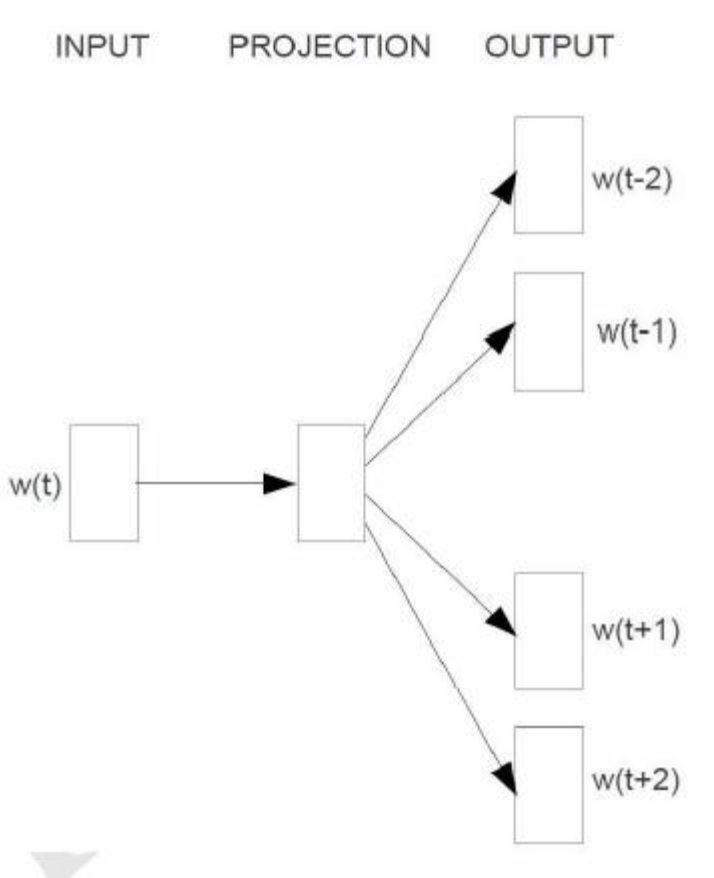
- 遇到词频高的词语时更快到达叶子节点，提高训练速度
- 复杂度从H*V变为H*log（V）
- 输出即概率，不需归一化，神经网络语言模型的输出需要softmax归一化

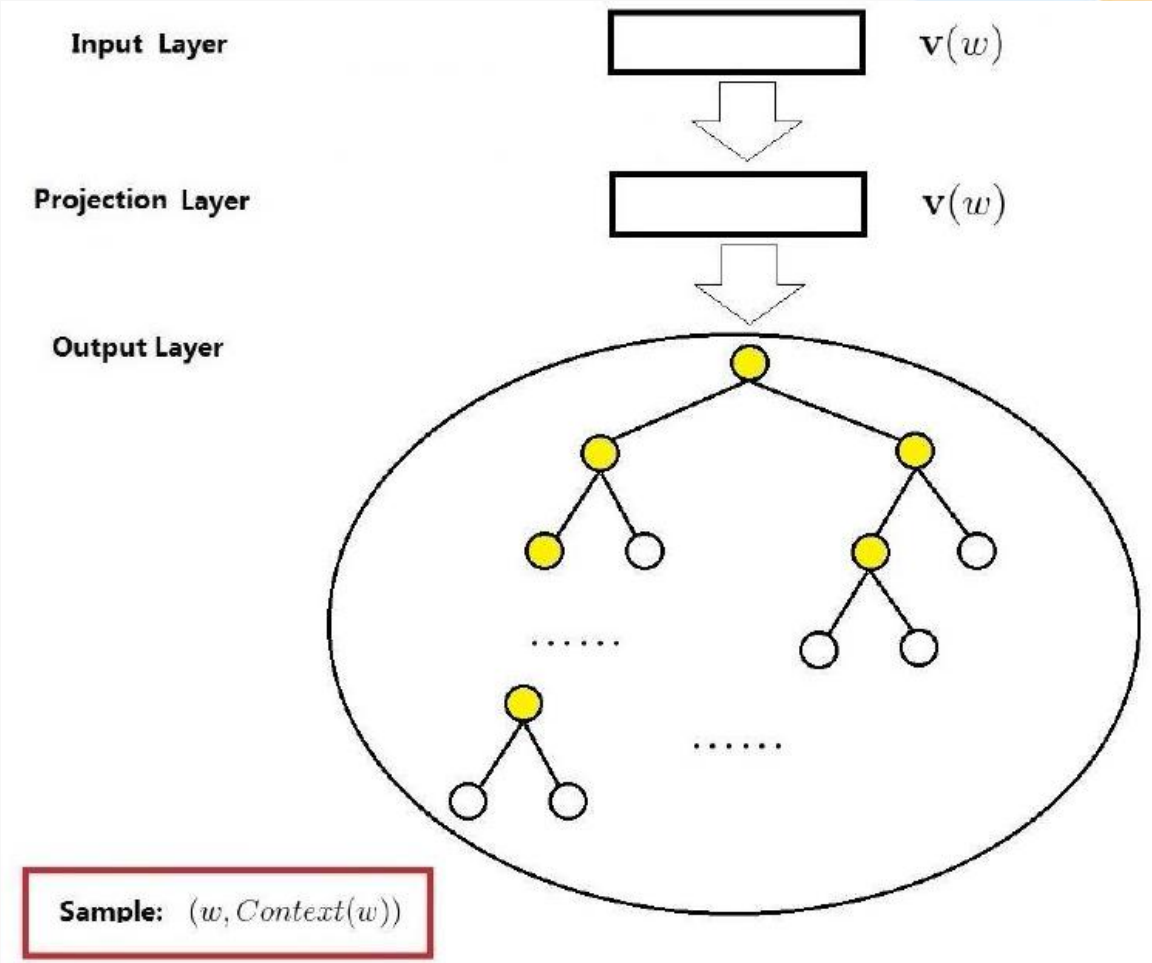
经过上述两步计算得到的 $\mathbf{y}_w = (y_{w,1}, y_{w,2}, \cdots, y_{w,N})^\top$ 只是一个长度为 N 的向量, 其分量不能表示概率. 如果想要 $\mathbf{y}_w$ 的分量 $y_{w,i}$ 表示当上下文为 $Context(w)$ 时下一个词恰为词典 $\mathcal{D}$ 中第 i 个词的概率, 则还需要做一个 **softmax** 归一化, 归一化后, $p(w|Context(w))$ 就可以表示为

$$p(w|Context(w)) = \frac{e^{y_{w,i_w}}}{\sum_{i=1}^{N} e^{y_{w,i}}}, \tag{3.6}$$

其中 i_w 表示词 w 在词典 $\mathcal{D}$ 中的索引.

Skip-gram + HS





$$p(Context(w)|w) = \prod_{u \in Context(w)} p(u|w)$$

U和w可调换位置

$$p(u|w) = \prod_{j=2}^{l^u} p(d_j^u|\mathbf{v}(w), \theta_{j-1}^u)$$

$$p(d_j^u|\mathbf{v}(w), \theta_{j-1}^u) = [\sigma(\mathbf{v}(w)^\top \theta_{j-1}^u)]^{1-d_j^u} \cdot [1 - \sigma(\mathbf{v}(w)^\top \theta_{j-1}^u)]^{d_j^u}$$

Negative sampling + CBOW

- 无严格理论证明，目的：提高训练速度并且提高训练所得词向量的质量
- 做法：按词频随机负采样

在 CBOW 模型中, 已知词 w 的上下文 $Context(w)$, 需要预测 w, 因此, 对于给定的 $Context(w)$, 词 w 就是一个**正样本**, 其它词就是**负样本**了

假定现在已经选好了一个关于 w 的负样本子集 $NEG(w) \neq \emptyset$. 且对 $\forall\, \tilde{w} \in \mathcal{D}$, 定义

$$L^w(\tilde{w}) = \begin{cases} 1, & \tilde{w} = w; \\ 0, & \tilde{w} \neq w, \end{cases}$$

表示词 $\tilde{w}$ 的标签, 即正样本的标签为 1, 负样本的标签为 0.

对于一个给定的正样本 $(Context(w), w)$, 我们希望最大化

$$g(w) = \prod_{u \in \{w\} \cup NEG(w)} p(u|Context(w)),$$

Negative sampling + CBOW

$$p(u|Context(w)) = \begin{cases} \sigma(\mathbf{x}_w^\top \theta^u), & L^w(u) = 1; \\ 1 - \sigma(\mathbf{x}_w^\top \theta^u), & L^w(u) = 0, \end{cases}$$

$$p(u|Context(w)) = \left[\sigma(\mathbf{x}_w^\top \theta^u)\right]^{L^w(u)} \cdot \left[1 - \sigma(\mathbf{x}_w^\top \theta^u)\right]^{1-L^w(u)}$$

$$g(w) = \sigma(\mathbf{x}_w^\top \theta^w) \prod_{u \in NEG(w)} \left[1 - \sigma(\mathbf{x}_w^\top \theta^u)\right]$$

最大化 $g(w)$, 相当于最大化 $\sigma(\mathbf{x}_w^\top \theta^w)$, 同时最小化所有的 $\sigma(\mathbf{x}_w^\top \theta^u)$, $u \in NEG(w)$.

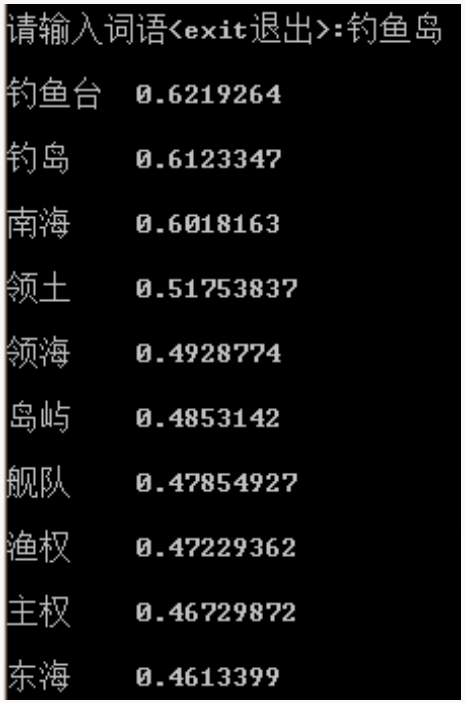
增大正样本的概率同时降低负样本的概率.

Negative sampling + skip-gram

$$
\begin{aligned}
\mathcal{L} &= \log G = \log \prod_{w \in \mathcal{C}} \prod_{u \in Context(w)} g(u) = \sum_{w \in \mathcal{C}} \sum_{u \in Context(w)} \log g(u) \\
&= \sum_{w \in \mathcal{C}} \sum_{u \in Context(w)} \log \prod_{z \in \{u\} \cup NEG(u)} p(z|w) \\
&= \sum_{w \in \mathcal{C}} \sum_{u \in Context(w)} \sum_{z \in \{u\} \cup NEG(u)} \log p(z|w) \\
&= \sum_{w \in \mathcal{C}} \sum_{u \in Context(w)} \sum_{z \in \{u\} \cup NEG(u)} \log \left\{ \left[\sigma(\mathbf{v}(w)^\top \theta^z)\right]^{L^u(z)} \cdot \left[1 - \sigma(\mathbf{v}(w)^\top \theta^z)\right]^{1 - L^u(z)} \right\} \\
&= \sum_{w \in \mathcal{C}} \sum_{u \in Context(w)} \sum_{z \in \{u\} \cup NEG(u)} \\
&\quad \left\{ L^u(z) \cdot \log \left[\sigma(\mathbf{v}(w)^\top \theta^z)\right] + \left[1 - L^u(z)\right] \cdot \log \left[1 - \sigma(\mathbf{v}(w)^\top \theta^z)\right] \right\}.
\end{aligned}
$$

word2vec实战（一）

- 训练数据集：经过分词后的新闻数据，大小184MB
  - 查看"中国"，"钓鱼岛"，"旅游"，"苹果"几个词语的相似词语如下所示

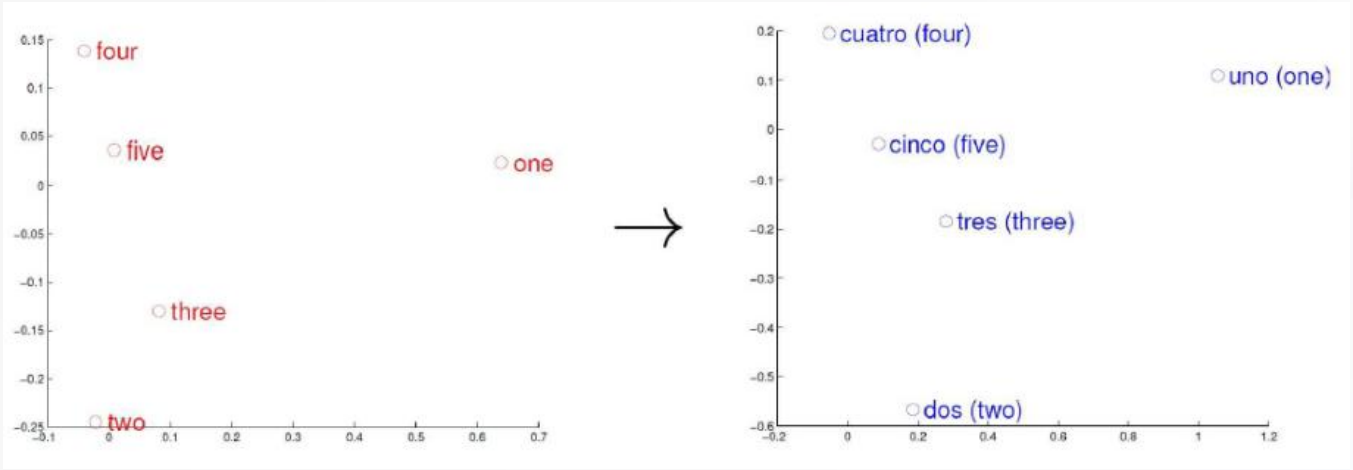


word2vec实战（二）

- 向量加减法
  - "中国+北京-日本"，"中国+北京-法国"，"家庭+孩子-学校"

word2vec应用（三）

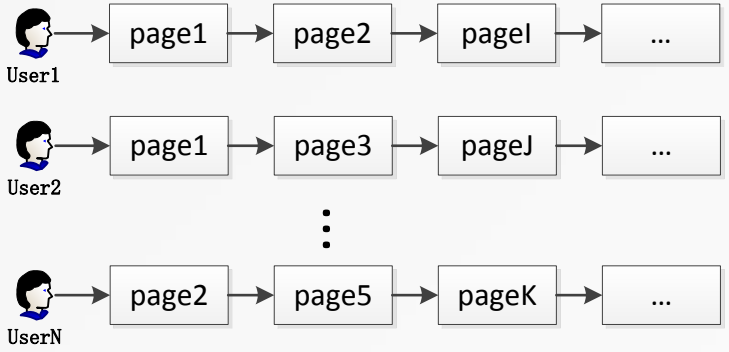
- 机器翻译
  - 语言词语的关系集合被表征为向量集合
  - 向量空间内，不同语言享有许多共性
  - 实现一个向量空间到另一个向量空间的映射和转换

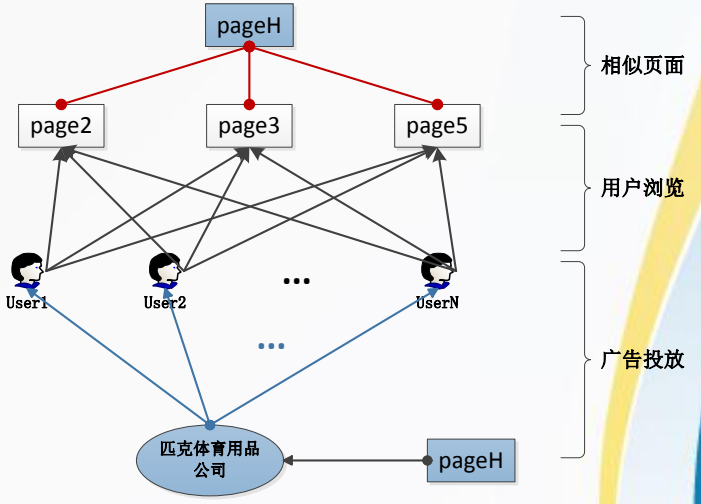


  - 图为英语和西班语的五个词在向量空间中的位置（已经过降维）
  - 对英语和西班语之间的翻译准确率高达90%

word2vec应用（四）

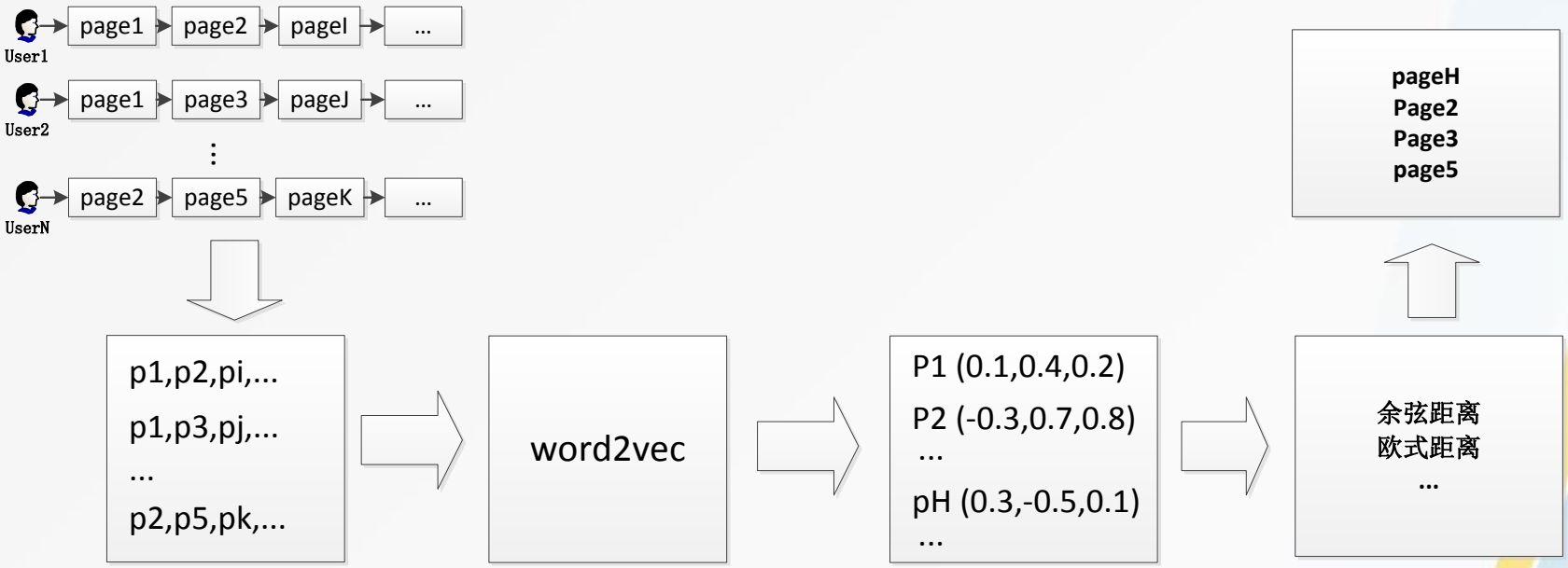
- 给广告主推荐用户
  - T媒体网站保存了用户浏览网页的记录
  - pageH是匹克体育用品公司在T上的官网
  - page2,page3,page5和pageH是比较相似的页面
  - 可给匹克体育用品公司推荐经常浏览page2,3,5这个几个页面的用户进行广告投放





word2vec应用（四）

- 相似的页面计算过程



也可用于挖掘同好用户：将一簇簇相似的用户作为doc（譬如QQ群），将单个用户作为word，我们则可以训练user distributed representation，可以借此挖掘相似用户。

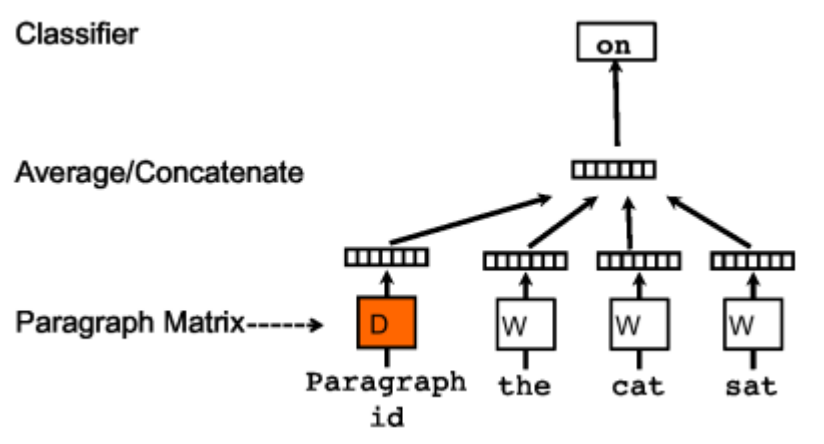
Paragraph2vec

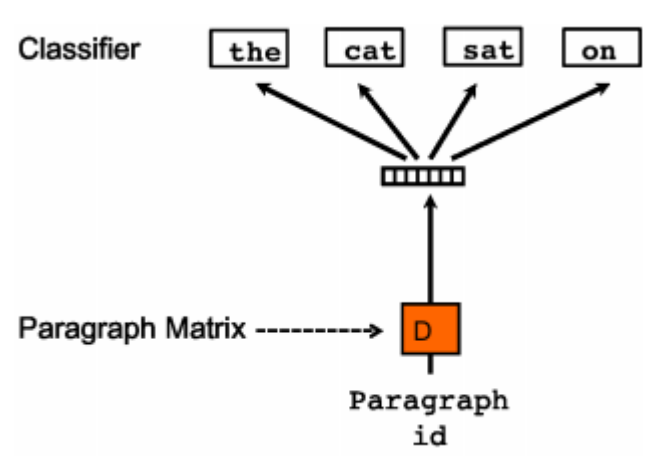
- 训练过程中新增了paragraph id，即训练语料中每个句子都有一个唯一的id。paragraph id和普通的word一样，也是先映射成一个向量。paragraph vector与word vector的维数虽一样，但是来自于两个不同的向量空间。
- 对应于cbow, 在计算里，paragraph vector和word vector累加或者连接起来，作为输出层softmax的输入。在一个句子或者文档的训练过程中，paragraph id保持不变，共享着同一个paragraph vector，相当于每次在预测单词的概率时，都利用了整个句子的语义。

Paragraph2vec

- 对应于skip-gram,

at each iteration of stochastic gradient descent, we sample a text window, then sample a random word from the text window and form a classification task given the Paragraph Vector.





Paragraph2vec

- 在预测阶段，给待预测的句子新分配一个paragraph id，词向量和输出层softmax的参数保持训练阶段得到的参数不变，重新利用梯度下降训练待预测的句子。待收敛后，即得到待预测句子的paragraph vector。

Paragraph2vec

参考文献

- [1] http://blog.csdn.net/mytestmy/article/details/26969149 深度学习word2vec笔记之算法篇
- [2] http://blog.csdn.net/itplus/article/details/37969979 word2vec 中的数学原理详解（四）基于 Hierarchical Softmax 的模型
- [3] http://www.zhihu.com/question/21661274/answer/19331979 @杨超在知乎上的问答《Word2Vec的一些理解》
- [4] http://xiaoquanzi.net/?p=156 hisen博客的博文
- [5] http://blog.csdn.net/mytestmy/article/details/38612907 深度学习word2vec笔记之应用篇
- [6] http://techblog.youdao.com/?p=915 Deep Learning实战之word2vec，网易有道的pdf
- [7] http://blog.csdn.net/lingerlanlan/article/details/38232755 word2vec源码解析之word2vec.c
- [8] Hierarchical probabilistic neural network language model. Frederic Morin and Yoshua Bengio.
- [9] Distributed Representations of Words and Phrases and their Compositionality T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean.
- [10] A neural probabilistic language model Y. Bengio, R. Ducharme, P. Vincent.
- [11] Linguistic Regularities in Continuous Space Word Representations. Tomas Mikolov,Wen-tau Yih,Geoffrey Zweig.
- [12] Efficient Estimation of Word Representations in Vector Space. Tomas Mikolov,Kai Chen,Greg Corrado,Jeffrey Dean.
- [13]http://licstar.net/archives/328 Deep Learning in NLP （一）词向量和语言模型

thank you !

Q&A