自然语言处理

Natural Language Processing

Chapter 3

词向量

Outline

- 词向量的意义
- Word2Vec算法
- Glove算法
- 词向量讨论

词语的表示

- 词语：文本的基本单元 (或者 字)
- “词语的意思”？语言学角度：它代表的含义(指称)
- 如何获得词语的意思？

```
from nltk.corpus import wordnet as wn
posdic = {'n': 'noun', 'r': 'adv', 'v': 'verb', 's': 'adj(s)', 'a': 'adj'}
for synset in wn.synsets('great'):
    print('{}:{}'.format(posdic[synset.pos()],
                         ', '.join([l.name() for l in synset.lemmas()])))
```

```
noun:great
adj(s):great
adj(s):great, outstanding
adj(s):great
adj(s):bang-up, bully, corking, cracking, dandy, great, groovy, keen, neat, nif
adj(s):capital, great, majuscule
adj(s):big, enceinte, expectant, gravid, great, large, heavy, with_child
```

近义词

```
from nltk.corpus import wordnet as wn
dragon = wn.synset('dragon.n.01')
hyper = lambda s:s.hypernyms()
print(*list(dragon.closure(hyper)),
      sep='\n')
```

上位词

```
Synset('mythical_monster.n.01')
Synset('monster.n.01')
Synset('mythical_being.n.01')
Synset('imaginary_being.n.01')
Synset('imagination.n.01')
Synset('creativity.n.01')
Synset('ability.n.02')
Synset('cognition.n.01')
Synset('psychological_feature.n.01')
Synset('abstraction.n.06')
Synset('entity.n.01')
```

But... 人工构建；词典的不完整性；更新滞后；同义关系只在特定语境下有效

词语向量化

- 在传统NLP(2012年以前的NLP)中，词语视为离散的符号：
  house, hotel, people, person, walk…

- 数学模型(机器学习模型、神经网络)要求数值型的输入，因此要将词语进行数值化/向量化/编码。
- 最简单的：词语的独热向量(one-hot vectors)表示：
- hotel = [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

向量维数= 词表大小 (e.g., 词语个数 英语60万，汉语20万)
例如，hotel的第1维值为1，代表hotel是词表中的第1个词语。

独热向量的问题

- 对于两个不同的词语，独热向量必然是正交(orthogonal)的.
- 无法通过独热向量反映词语之间的相似度

- 例子：准备旅游攻略，希望搜索“Seattle hotel”相关的信息。此时包含“Seattle motel”的网页可能也包含了有用信息。假设motel和hotel的独热向量如下：
- hotel = [0 1 0 0 0 0 0 0 0 0 0 0 0 0 0]
- motel = [0 0 1 0 0 0 0 0 0 0 0 0 0 0 0]

- 解决方案:
  - 借助词典资源(e.g. WordNet)的同义词获取词语相似度?
    - 词典的不完整性；更新滞后；同义关系只在特定语境下有效
  - 把词语的相似度直接编码进词语的向量表示?

分布式表示

- 分布式表示distributed representation: Hinton 1986年提出

- 基本思想：通过训练，将词语映射为固定长度的低维向量(相对于独热向量的高维)
- 所有向量构成向量空间，每个词即为空间中的一点，可以通过词语在空间中的“距离”判断语法、语义等方面的相似度
- 现代统计NLP中最成功的的想法之一

Geoffrey Hinton, 深度学习之父，图灵奖得主

词向量

- 每一个词表示为一个低维(low-dimensional)稠密(dense)的分布式向量
- e.g.

$$banking = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \end{pmatrix}$$

- 通用概念：word vector, word embedding, word representation
- 长度一般为几十到几百
- One-hot向量：信息集中于1个非零向量 vs 分布式向量：大量(全部)非零，信息分散，分布到各个分量

词向量-可视化

分布式向量的特征：经常出现在同样上下文中的词语，
词向量表示相近，即在向量空间中的距离相近

多维向量空间映射到二维(PCA降维)：

$$expect = \begin{pmatrix} 0.286 \\ 0.792 \\ -0.177 \\ -0.107 \\ 0.109 \\ -0.542 \\ 0.349 \\ 0.271 \\ 0.487 \end{pmatrix}$$

对词语的语义、语法等方面的相似
度进行了刻画

词向量

- Q: 如何获得分布式词向量？
- A：很多种。
- **分布式语义**：词语的含义由它经常出现的上下文词语决定
- 对于文本中的词语 ***w*** , 上下文(context)就是文本中出现在***w***周围的词语集合。“周围”指的是一个固定大小的窗口内。

banking?

...government debt problems turning into ***banking*** *crises as happened in 2009...*

...saying that Europe needs unified ***banking*** *regulation to replace the hodgepodge...*

...India has just given its ***banking*** *system a shot in the arm...*

[0.286, 0.792, -0.177, -0.107, 0.109, ...]

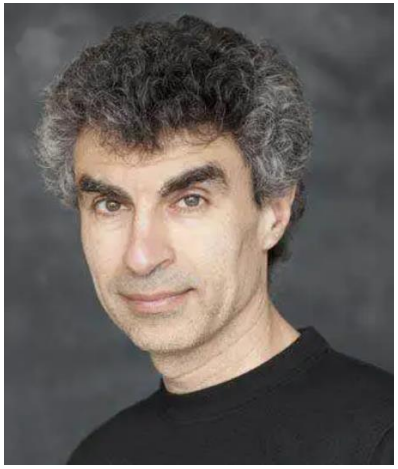
dog? cat?

```
A dog is running in the room
A cat is running in the room
The cat is running in a room
A dog is walking in a bedroom
The dog was walking in the room
...
```

使用足够多的上下文文本去建立起 ***w*** 的表示!

词向量

- (剧透) 词向量与语言模型关系密切。语言模型就是一个句子的概率模型，分解为每一步根据已经出现的词预测下一个词，因此模型中需要使用词语的向量表示。
- 使用神经网络训练语言模型的思想最早于2000年由百度深度学习研究院的徐伟提出。



Bengio，图灵奖得主

- 经典文章：Yoshua Bengio于2003发表的《A neural Probabilistic Language Model》(更早版本2001年)
- 其后有一系列的相关工作，包括著名的**word2vec**！

Outline

- 词向量的意义
- Word2Vec算法
- Glove算法
- 词向量讨论

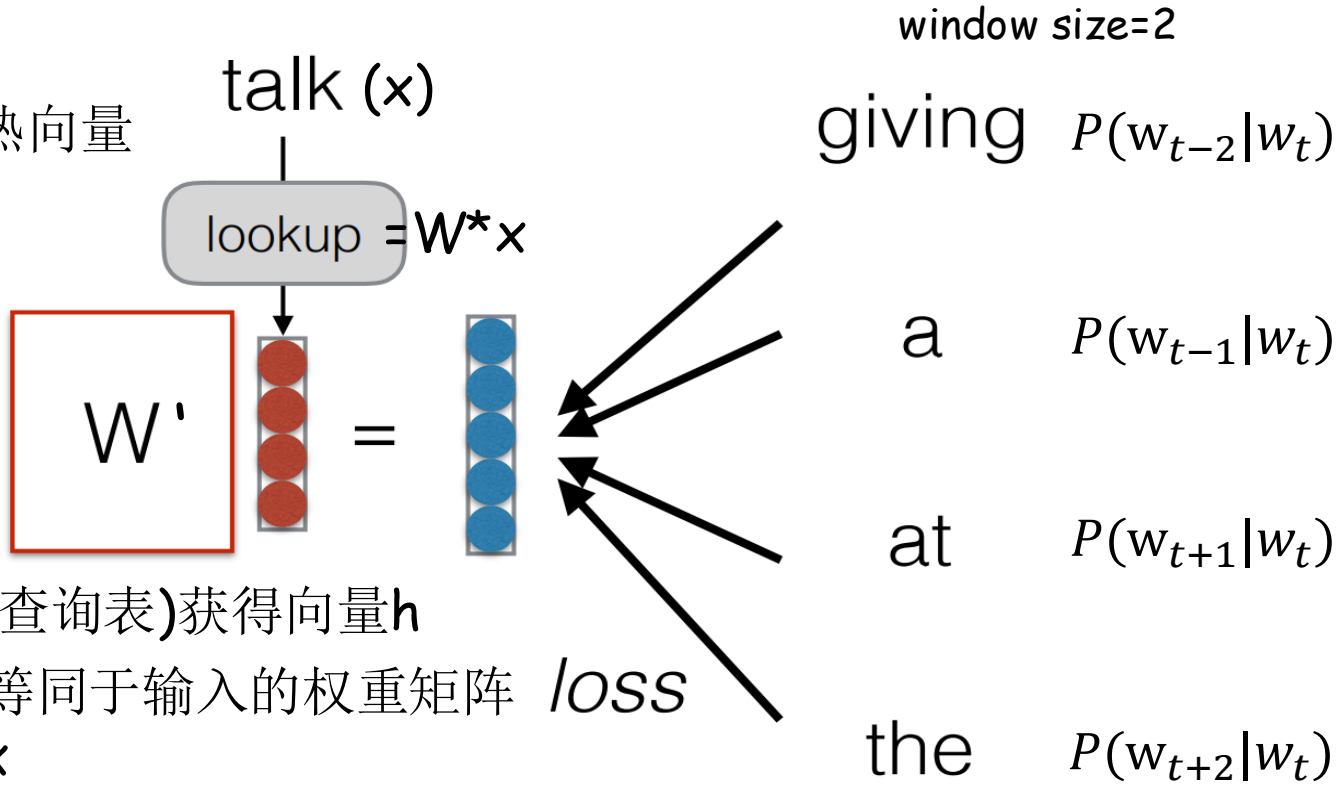
word2vec

- Word2vec是最受欢迎的开源词向量学习算法之一，简写w2v， 2013年由谷歌提出，链接
- 模型：浅层神经网络
- 主要思想:
  - 有一个大规模的文本语料库(corpus)，一个固定的词表
  - 对于语料库的每一个文本，遍历每一个位置t，位置t上面的词语c称为中心词(center word)，窗口以内、c之外的上下文词语称为o(outside word)
  - 两种模式：使用c和o的词向量的相似度来计算给定c，o的概率；或者是给定o，c的概率.
  - 算法训练的过程: 不断调整词向量，最大化这些概率值.

模式1：Skip-gram

- 核心思想：给出一个中心词，以此预测它的上下文词语
- 即：预测p(context(w)|w)

(1) 初始化：独热向量



(2) 根据lookup(查询表)获得向量h

lookup(查询表)等同于输入的权重矩阵，操作为$h=W^Tx$

(3) 每一个位置预测$p = softmax(W'^Th)$ 维数=词典大小

(4) 根据该位置真实词语的独热向量和p计算loss

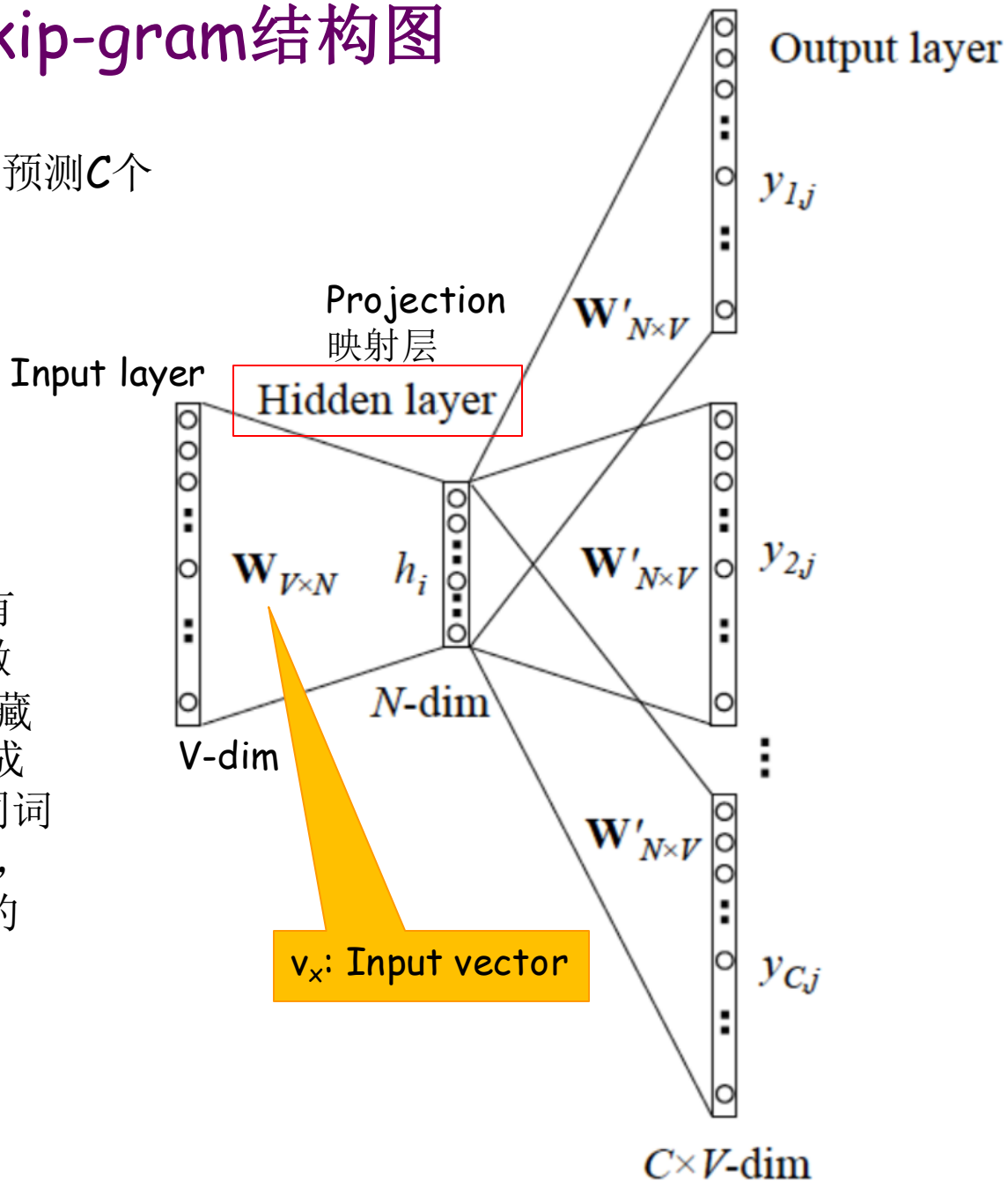
Skip-gram结构图

词汇表大小=V，输入中心词，预测C个上下文词语

中心词初始化为V维的独热向量(V×1):
[0, **1**, 0, 0, …, 0]^T

- 输入到隐藏层:

因为输入是one-hot向量，只有W中对应1的位置上的权重被激活，这些权重项的个数等于隐藏层节点数N。将这些权重项组成一个向量v_x来表示x。由于不同词语的one-hot向量的1位置不同，因此这个N维的v_x向量是唯一的，可以唯一表示某一个输入词



Skip-gram结构图

词汇表大小=V，输入中心词，预测C个上下文词语

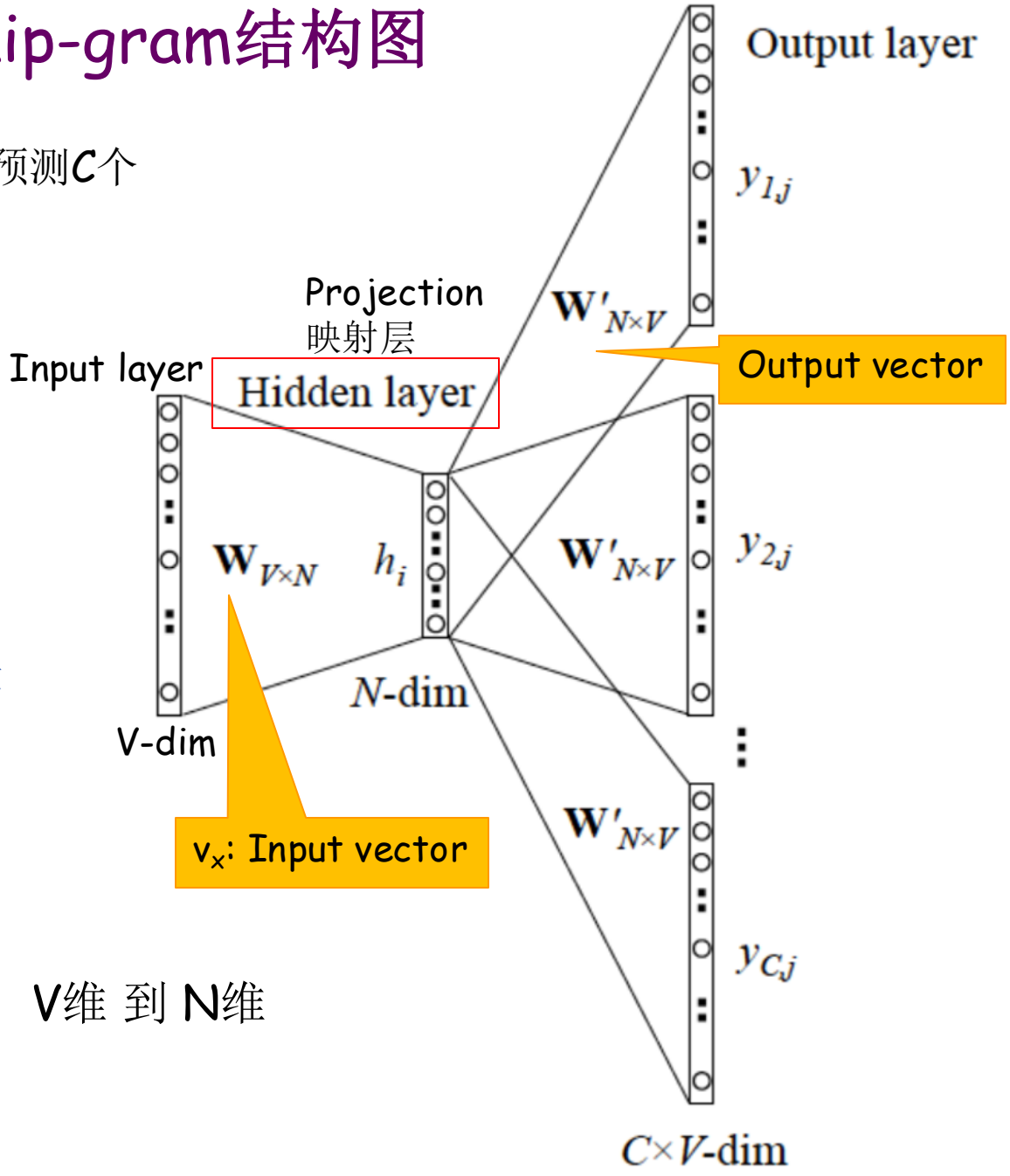
- 隐藏层到输出层：

(先忽略softmax) u = W'Th

中第j个元素是W'的第j列和h的内积，是词表第j个词的分数，所以W'的第j列可以代表输出词的向量

训练结束后，取input vector作为词向量，即input到hidden的权重矩阵参数

即：one-hot乘以该矩阵得到词向量(lookup)



w2v可以看成一种降维操作：V维 到 N维

W & W': 随机初始化

Skip-gram: 目标函数

- 回顾：在skip-gram模式下，对于每一个位置 $t = 1, \dots, T$，根据中心词w_t，预测大小为2m的窗口内的上下文词.

- 目标函数J(θ) 取 (平均) 负对数似然:

$$J(\theta) = -\frac{1}{T}\log L(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-m\le j\le m \\ j\neq 0}} \log P\left(w_{t+j} \mid w_t; \theta\right)$$

最小化 / 最大化

已知： $h = W^T x = v$

$u = W'^T h$

实际上用到W'中某一列 定义为向量v' (output vector)

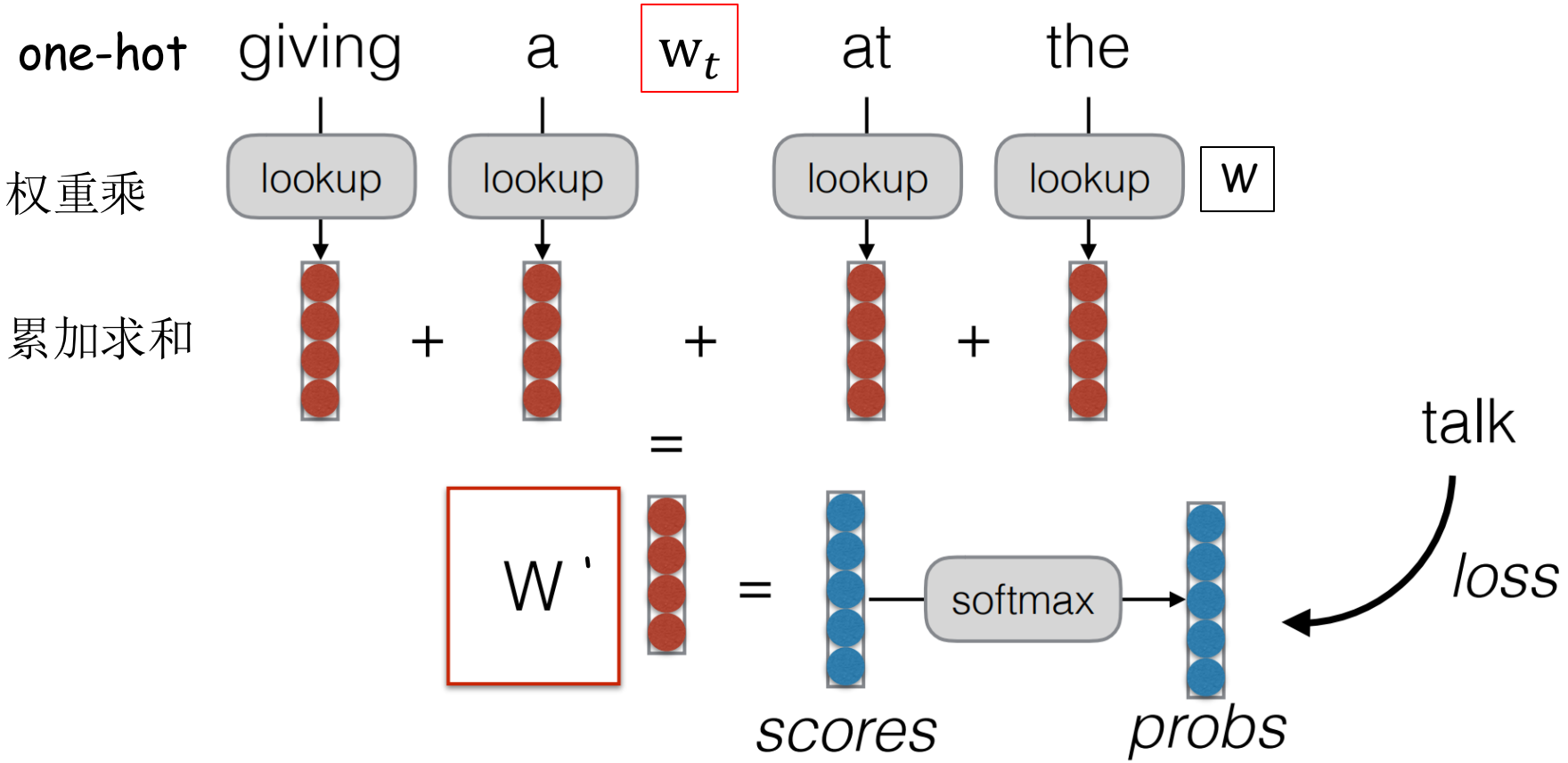
v (input vector)

相乘体现了c和o的相似度

直接求解计算量巨大！

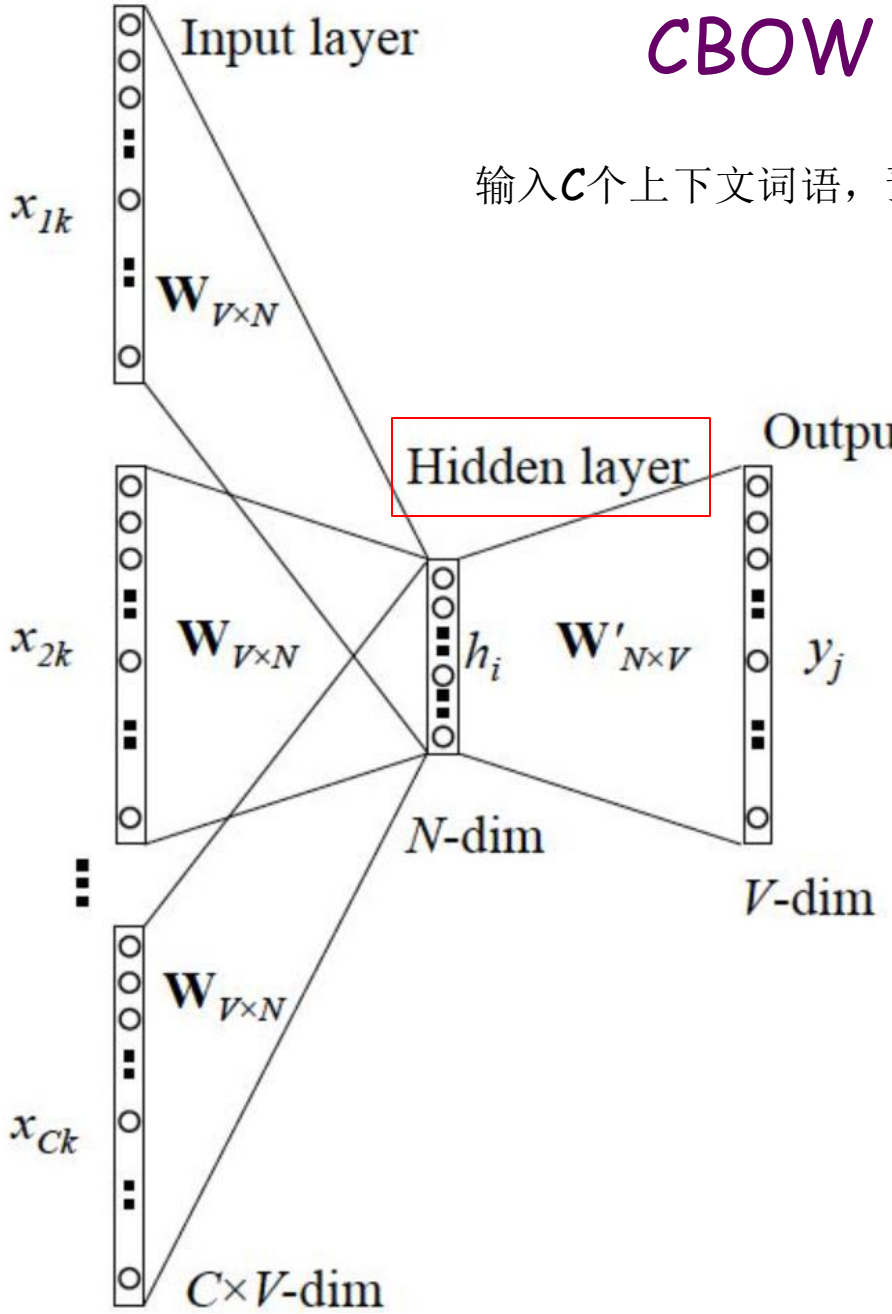
模式2：CBOW

- 连续词袋模型(Continuous bag-of-words)
- 核心思想：根据上下文词语向量，预测中心词
- 即：预测p(w|context(w))



CBOW

输入C个上下文词语，预测中心词

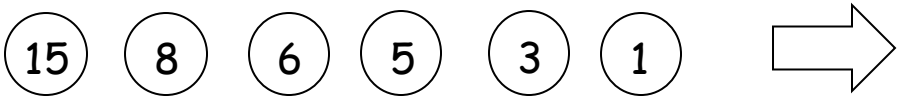


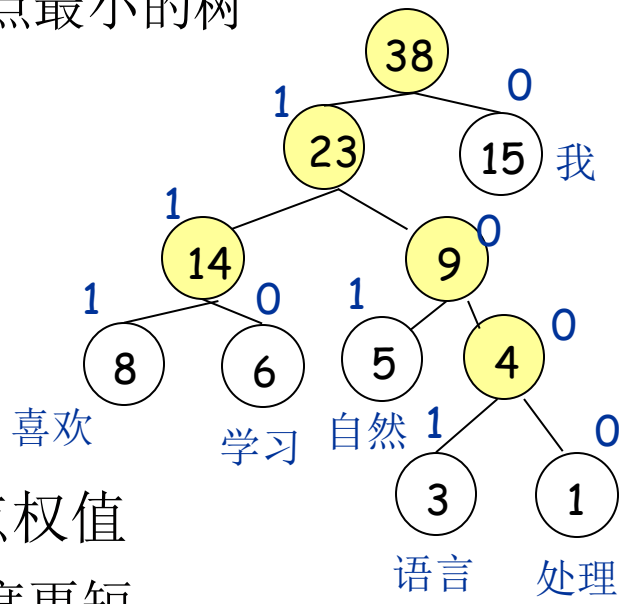
$$L = \sum_{w \epsilon C} \log \mathrm{P}(w|context(w))$$

C个词

计算优化：Hierarchical softmax

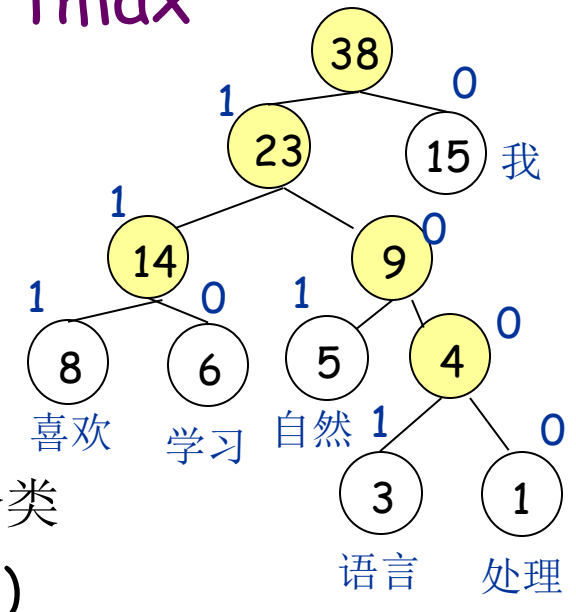
- 优化思路来源：w2v模型输出是霍夫曼树(Huffman)
- 复习：Huffman树&Huffman编码
- Huffman树的构造：不断合并两个根节点最小的树





- Huffman编码：将字符频率作为结点权值
- 频率高的词语更接近于根节点，编码长度更短
- 在w2v中，训练语料中的词语作为叶子节点，出现次数作为权值
- E.g. 假设左子结点编码为1，则我=0，喜欢=111，学习=110，自然=101，语言=1001

计算优化：Hierarchical softmax



对于某个代表词语w的结点，记：

- 从root到该点路径上结点个数为l^w
- 忽略root，$d_2^w,\ldots,d_{l^w}^w$是该点的Huffman编码
- $\theta_1^w,\ldots,\theta_{l^w-1}^w$是路径中非叶子节点对应的向量
- 从根节点到“喜欢”，经历3次分支，即3次二分类
- 对于样本x的二分类，可以用sigmoid表示为σ(xθ)
- 则一个结点判为正类的概率：σ(x_wθ), θ可由$\theta_1^w,\ldots,\theta_{l^w-1}^w$充当
- 以CBOW为例使用huffman树定义p(w|context(w)) =

$\prod_{j=2}^{4} p(d_j^w|x_w,\theta_{j-1}^w)$

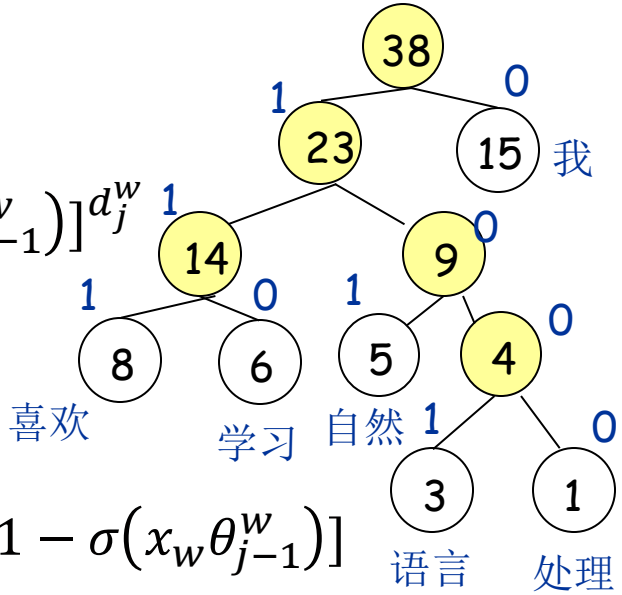
x_w :context(w)中词向量之和

又：$p(d_j^w|x_w,\theta_{j-1}^w)=\begin{cases}\sigma(x_w\theta_{j-1}^w), d_j^w=0 & \text{正类}\\ 1-\sigma(x_w\theta_{j-1}^w), d_j^w=1 & \text{负类}\end{cases}$

即：$p(d_j^w|x_w,\theta_{j-1}^w)=[\sigma(x_w\theta_{j-1}^w)]^{1-d_j^w}\cdot[1-\sigma(x_w\theta_{j-1}^w)]^{d_j^w}$

计算优化：Hierarchical softmax

$$\begin{cases} \mathrm{p(w|context(w))} = \prod_{j=2}^{l^w} p(d_j^w | x_w, \theta_{j-1}^w) \\ p(d_j^w | x_w, \theta_{j-1}^w) = [\sigma(x_w \theta_{j-1}^w)]^{1-d_j^w} \cdot [1-\sigma(x_w \theta_{j-1}^w)]^{d_j^w} \end{cases}$$



代入$L = \sum_{w \in C} \log \mathrm{P}(w|context(w))$

得$L = \sum_{w \in C} \log \prod_{j=2}^{l^w} p(d_j^w | x_w, \theta_{j-1}^w)$ =

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

L(w, j)

σ导函数:$f'(x) = f(x)(1-f(x))$

$$\frac{\partial L(w,j)}{\partial \theta_{j-1}^w} = (1-d_j^w) x_w \sigma(\ldots)[1-\sigma(\ldots)] \frac{1}{\sigma(\ldots)} - d_j^w x_w \sigma(\ldots)[1-\sigma(\ldots)] \frac{1}{1-\sigma(\ldots)}$$

$$= \{(1-d_j^w)[1-\sigma(x_w \theta_{j-1}^w)] - d_j^w \sigma(x_w \theta_{j-1}^w)\} x_w$$

$$= [1-d_j^w - \sigma(x_w \theta_{j-1}^w)] x_w$$

同理：$\frac{\partial L(w,j)}{\partial x_w} = [1-d_j^w - \sigma(x_w \theta_{j-1}^w)] \theta_{j-1}^w$

在CBOW中，x_w是context词向量的累加

直接作为每个$\tilde{w} \in$context(w)的梯度

计算优化：负采样

- Negative sampling. 更简单
- 在cbow中，根据context(w) 预测w
- 正样本：w
- 负样本：其他词语，假设集合为NEG(w)，减少了选择范围

$$\text{定义}L^{\mathrm{w}}(\tilde{w}) = \begin{cases} 1, \tilde{w} = w & \text{即正样本的标签=1,} \\ 0, \tilde{w} \neq w & \text{负样本标签=0} \end{cases}$$

- 目标：最大化$\mathrm{g(w)} = \prod_{u \in \{w\} \cup NEG(\mathrm{w})} p(u|context(w))$

$$p(u|context(w)) = \begin{cases} \sigma(x_w \theta^u) & , L^{\mathrm{w}}(u) = 1 \quad \text{正类} \\ 1 - \sigma(x_w \theta^u), & L^{\mathrm{w}}(u) = 0 \quad \text{负类} \end{cases}$$

$$\text{即} p(u|context(w)) = [\sigma(x_w \theta^u)]^{L^{\mathrm{w}}(u)} \cdot [1 - \sigma(x_w \theta^u)]^{1 - L^{\mathrm{w}}(u)}$$

$$\text{理解: } \mathrm{g(w)} = \sigma(x_w \theta^w) \prod_{u \in NEG(\mathrm{w})} [1 - \sigma(x_w \theta^u)]$$

最大化g即最大化正样本概率，同时最小化负样本概率

计算优化：负采样

- 总体损失：$G = \prod_{w \epsilon C} g(w)$

$$L = log \prod_{w \epsilon C} g(w) = \sum_{w \epsilon C} \log g(w)$$

$$= \sum_{w \epsilon C} log \prod_{u \in \{w\} \cup NEG(w)} [\sigma(x_w \theta^u)]^{L^w(u)} \cdot [1 - \sigma(x_w \theta^u)]^{1 - L^w(u)}$$

$$= \sum_{w \epsilon C} \sum_{u \in \{w\} \cup NEG(w)} \{L^w(u) \cdot \log[\sigma(x_w \theta^u)] + \big(1 - L^w(u)\big) \log[1 - \sigma(x_w \theta^u)]\}$$

$L(w, u)$

σ导函数:$f'(x) = f(x)(1 - f(x))$

$$\frac{\partial L(w,u)}{\partial \theta^u} = L^w(u)[1 - \sigma(x_w \theta^u)] x_w - \big(1 - L^w(u)\big) \sigma(x_w \theta^u) x_w$$

$$= \{L^w(u)[1 - \sigma(x_w \theta^u)] - \big(1 - L^w(u)\big) \sigma(x_w \theta^u)\} x_w$$

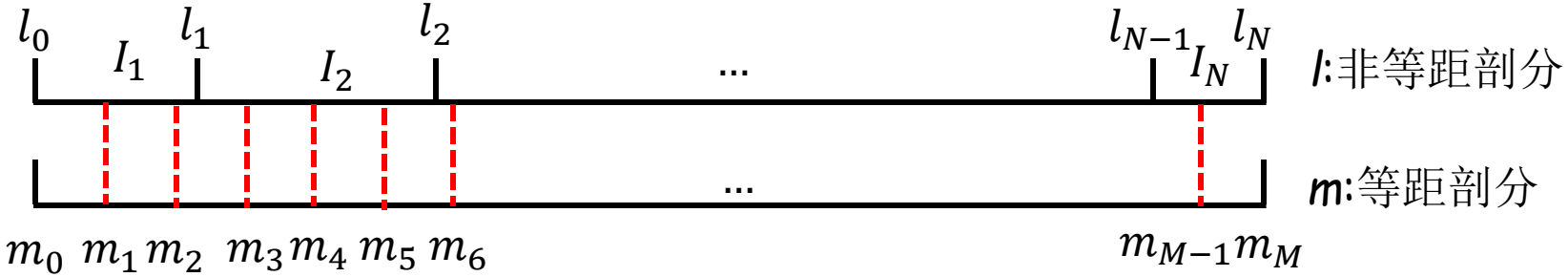
$$= [L^w(u) - \sigma(x_w \theta^u)] x_w$$

同理：$\frac{\partial L(w,u)}{\partial x_w} = [L^w(u) - \sigma(x_w \theta^u)] \theta^u$

计算优化：负采样

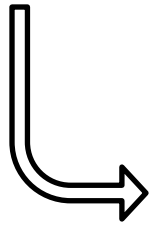
- 如何生成负样本？
- 词频有高有低，因此采用带权采样
- 设l_0=0, l_k =$\sum_{j=1}^{k} len(w_j)$, w_j是词典中第j个词，k=1, 2,…, N

$$len(w) = \frac{counter(w)}{\sum_{u \in D} counter(u)]}$$



- 采样过程：每次生成[1, M-1]间的随机整数r，m_r落在的I_k区间即取到w_k。如果对w采样得到w本身则跳过。
- 源码中：M=10^8，$len(w) = \frac{[counter(w)]^{3/4}}{\sum_{u \in D}[counter(u)]^{3/4}}$

- NeurIPS 2023时间检验奖



Distributed Representations of Words and Phrases and their Compositionality

第二篇
2013.10

Tomas Mikolov
Google Inc.
Mountain View
mikolov@google.com

Ilya Sutskever
Google Inc.
Mountain View
ilyasu@google.com

Kai Chen
Google Inc.
Mountain View
kai@google.com

Greg Corrado
Google Inc.
Mountain View
gcorrado@google.com

Jeffrey Dean
Google Inc.
Mountain View
jeff@google.com

Efficient Estimation of Word Representations in Vector Space

第一篇
2013.01

Tomas Mikolov
Google Inc., Mountain View, CA
tmikolov@google.com

Kai Chen
Google Inc., Mountain View, CA
kaichen@google.com

Greg Corrado
Google Inc., Mountain View, CA
gcorrado@google.com

Jeffrey Dean
Google Inc., Mountain View, CA
jeff@google.com

小结

- 分布式语义：使用词语的上下文表示词语
- 典型算法：word2vec(2013)
  - ✓ 浅层神经网络模型：input layer, projection, output layer
  - ✓ skip-gram：中心词预测上下文词
  - ✓ Cbow：上下文词预测中心词
  - ✓ 优化方法：hierarchical softmax，负采样
  - ✓ 取训练完的输入向量作为最终词向量表示
  - ✓ 实现从V维one-hot到N维稠密向量的降维

```
input_nn = Input(shape=(maxlen,))
embedding_layer = Embedding(len(word_ind) + 1, embedding_dims, weights=[embedding_matrix],
                            input_length=maxlen, trainable=True)
embedded_sequences = embedding_layer(input_nn)
```

- 理想的分布式词向量：组成向量空间，上下文相似的词向量相似，在空间中距离接近

Outline

- 词向量的意义
- Word2Vec算法
- Glove算法
- 词向量讨论

Glove

- 全局词向量 Global Vectors for Word Representation, 斯坦福2014年推出的开源项目, PDF(EMNLP)

- 主要思想
  - W2v基于局部信息，而Glove基于全局词频统计(count-based，overall statistics)
  - 根据语料库构建一个共现矩阵X, 其中X_{ij} 是词语i 和词语 j 在一个窗口内(e.g. 长度=10)的共现次数，所以X是对称的

Glove

- 共现矩阵

窗口=3:

我 喜欢 学习 自然 语言 处理
我 喜欢 深度 学习
我 学习 骑车

共现次数	我	喜欢	学习	自然	语言	处理	深度	骑车
我	0	2	2	0	0	0	1	1
喜欢	2	0	2	0	0	0	1	0
学习	2	2	0	1	1	0	1	1
自然	0	2	1	0	1	1	0	0
语言	0	0	1	1	0	1	0	0
处理	0	0	0	1	1	0	0	0
深度	1	1	1	0	0	0	0	0
骑车	1	0	1	0	0	0	0	0

增大了词汇表规模
$N \rightarrow N^2$

增加了存储

稀疏性问题

Glove

$P_{ik} = P(k|i) = \frac{x_{ik}}{X_i}$，i: 中心词，k: 上下文词，表示单词k出现在单词i上下文中的概率 ⟶ 通过i和k的点积计算

$ratio_{i,j,k} = \frac{P_{ik}}{P_{jk}}$: 同一个上下文k，不同的中心词i和j

Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

$ratio_{i,j,k}$	j，k相关	j，k不相关
i，k相关	接近1	很大
i，k不相关	很小	接近1

共现概率体现了词语间的关联！

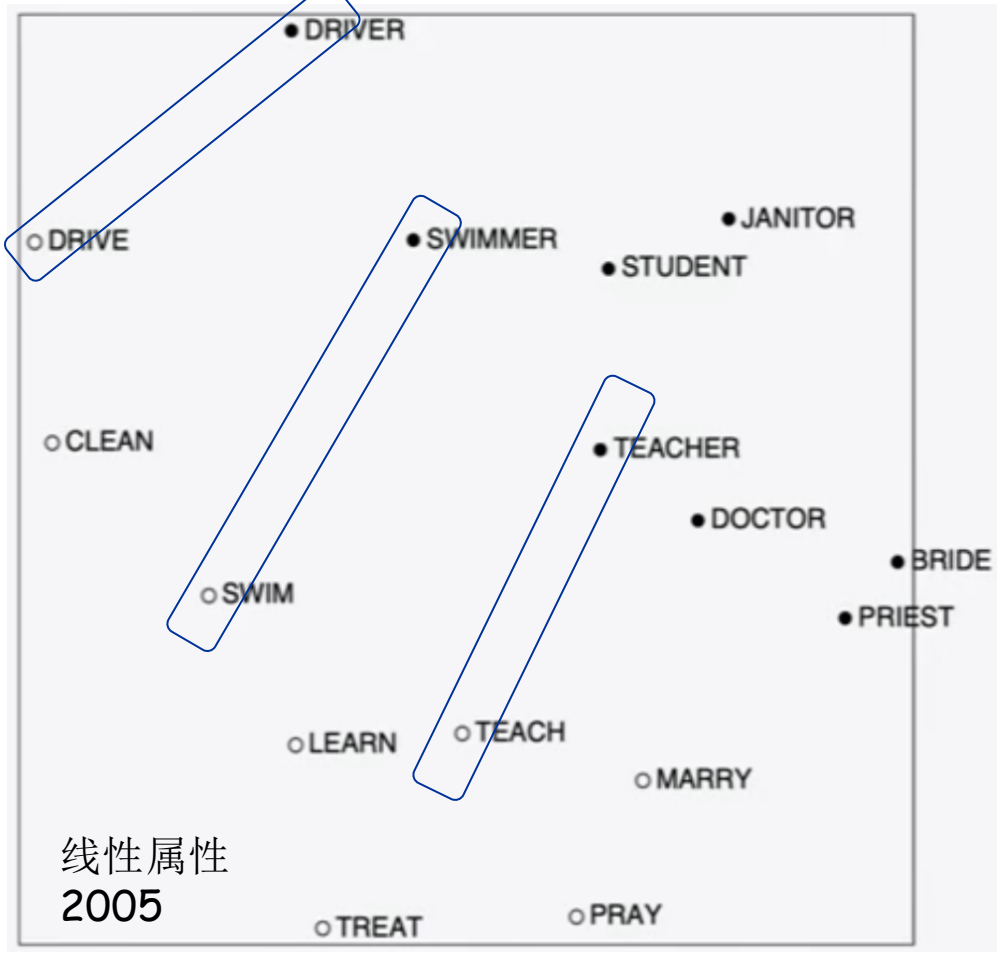
Glove

- 令w_i, w_j, w_k是i, j, k的向量，$F(w_i, w_j, \widetilde{w_k}) = \frac{P_{ik}}{P_{jk}}$ 两种初始化不同
- 向量空间是线性空间，在线性空间中，向量之间的差异性可以直接相减来表达，得：

$$F(w_i - w_j, \widetilde{w_k}) = \frac{P_{ik}}{P_{jk}}$$

左式内都为向量，右式为标量，因此将左式做内积计算变成标量：

$$F\left((w_i - w_j)^T \widetilde{w_k}\right) = \frac{P_{ik}}{P_{jk}}$$



Glove

- X是对称的，而$F\left((w_i - w_j)^T \widetilde{w_k}\right) = \frac{P_{ik}}{P_{jk}}$ 不满足等价性
- 建立左右两个部分的同态映射，得：

$$F\left((w_i - w_j)^T \widetilde{w_k}\right) = \frac{F(w_i^T \widetilde{w_k})}{F(w_j^T \widetilde{w_k})} \quad \text{其中：} \quad F\left(w_i^T \widetilde{w_k}\right) = P_{ik} = \frac{X_{ik}}{X_i}$$

- 令F=exp，并取对数，得：

$$w_i^T \widetilde{w}_k = \log(P_{ik}) = \log(X_{ik}) - \log(X_i)$$

- $\log(X_i)$破坏了对称性，又因为其与k无关，移到左边作为w_i的偏置项b_i。为了保证对称性，为$\widetilde{w}_k$添加偏置项$\tilde{b}_k$，得：

$$w_i^T \widetilde{w}_k + b_i + \tilde{b}_k = \log(X_{ik})$$

- 每个词在计算损失函数时所占用的权重应该不同，即出现频率高的词应占更大的权重，减少噪声干扰。因此损失函数表示为：

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

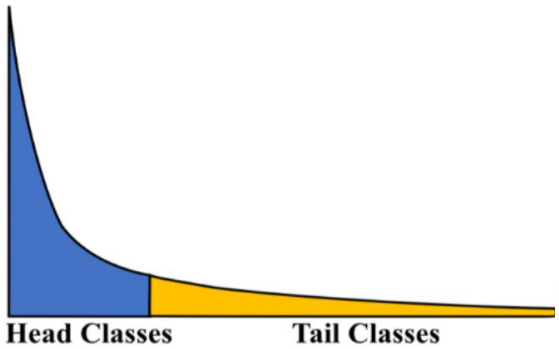
Glove

- 全局词向量 Global Vectors for Word Representation
- 主要思想:
  - 根据语料库构建一个共现矩阵X, 其中X_{ij} 是词语i 和 j 在一个窗口内(e.g. 长度=10)的共现次数. 对称矩阵. 实际考虑距离权重.
  - 构建词向量和共现矩阵之间的近似关系，对于两个词语i和k:

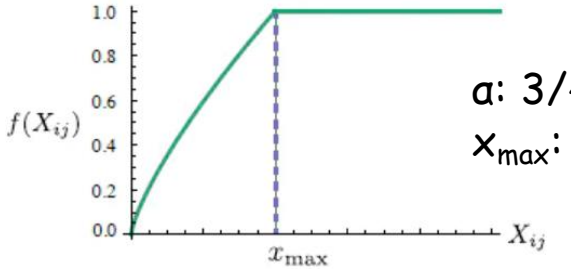
$$w_i^T \tilde{w}_k + b_i + \tilde{b}_k = \log(X_{ik})$$

  - 构造均方差损失函数(建模长尾分布效果更好):

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



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



α: 3/4;
x_{max}: 100

  - 取$\mathbf{W} + \widetilde{\boldsymbol{W}}$作为词向量

Glove

- 训练：
  - 优化器=AdaGrad，共现矩阵的非零值随机初始化，初始学习率=0.05

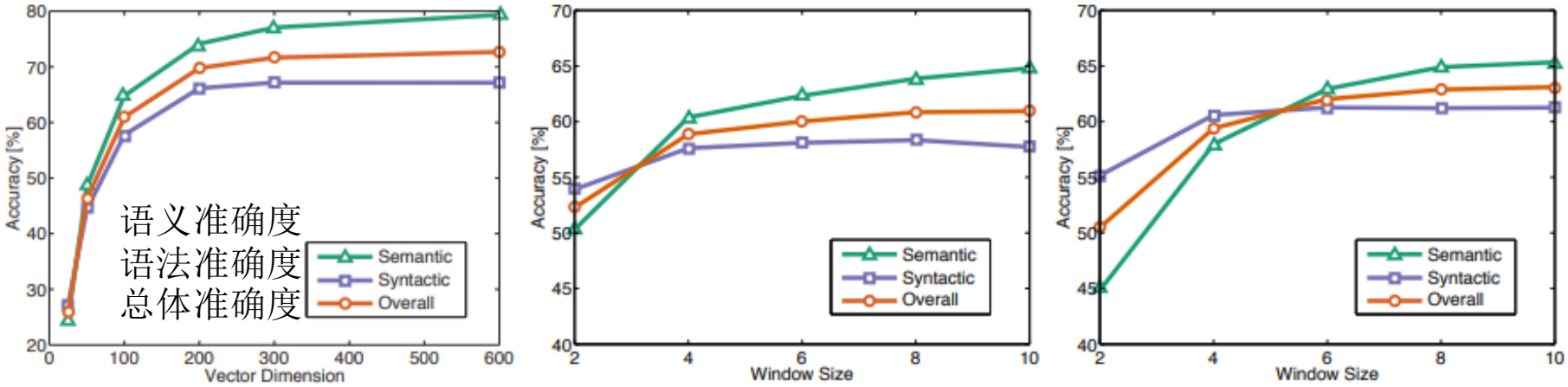


Figure 2: Accuracy on the analogy task as function of vector size and window size/type. All models are trained on the 6 billion token corpus. In (a), the window size is 10. In (b) and (c), the vector size is 100.

Outline

- 词向量的意义
- Word2Vec算法
- Glove算法
- 词向量讨论

词向量成果

Basic Settings

https://github.com/Embedding/Chinese-Word-Vectors

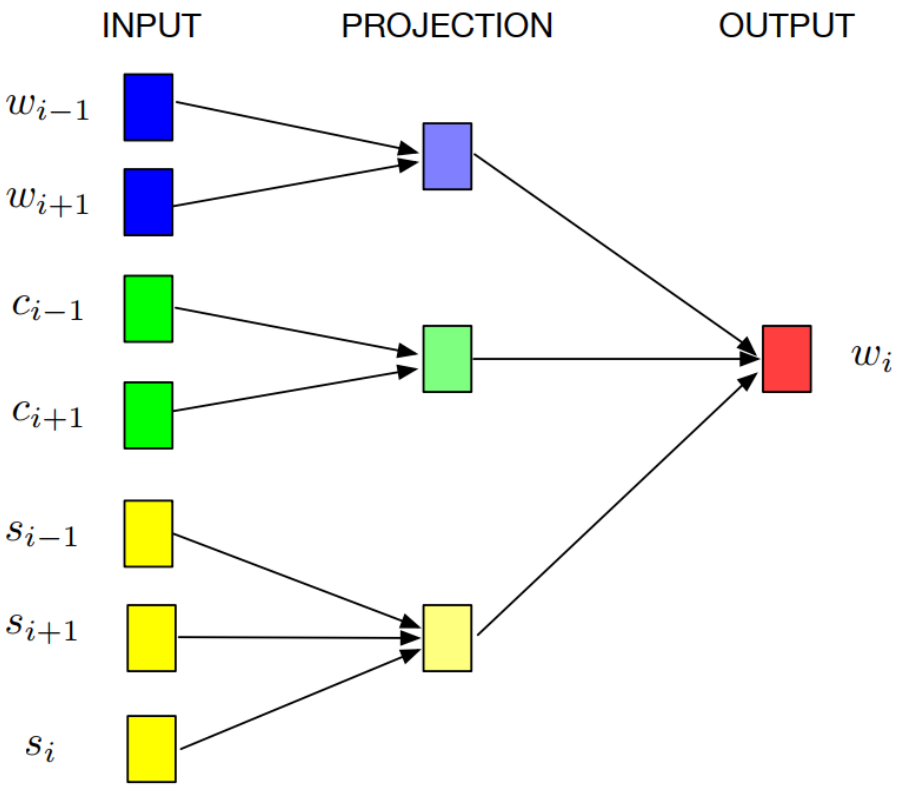
Window Size	Dynamic Window	Sub-sampling	Low-Frequency Word	Iteration	Negative Sampling*
5	Yes	1e-5	10	5	5

Word2vec / Skip-Gram with Negative Sampling (SGNS)				
Corpus	Context Features			
	Word	Word + Ngram	Word + Character	Word + Character + Ngram
People's Daily News 人民日报	300d	300d	300d	300d
Sogou News 搜狗新闻	300d	300d	300d	300d
Financial News 金融新闻	300d	300d	300d	300d
Zhihu_QA 知乎问答	300d	300d	300d	300d
Weibo 微博	300d	300d	300d	300d
Literature 文学作品	300d	300d / PWD: z5b4	300d	300d / PWD: yenb

词向量成果

- 考虑字(character)和字组成(subcharacter)的词向量
- E.g. “照”：偏旁代表“火”，日，刀，口

2017 EMNLP



https://github.com/HKUST-KnowComp/JWE

照 (photograph)	照片 (photo) 相片 (photo) 拍照 (photograph) 护照 (passport) 照相 (photography)

和“照”最接近的词

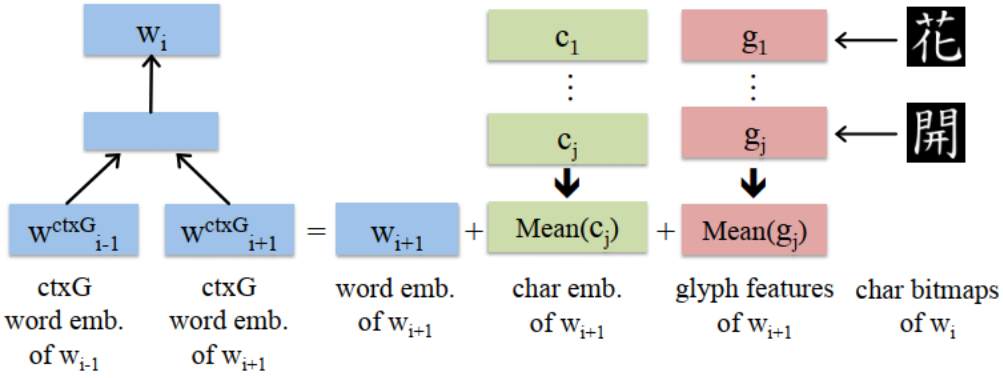
Component	疒 (illness)
Closest characters	疗 (cure) 症 (symptom) 痛 (pain) 疮 (sore) 患 (suffer) 痒 (itch) 疳 (infantile malnutrition) 病 (disease) 肿 (swelling)
Closest words	治疗 (cure) 病症 (symptom) 复发 (recurrence) 疼痛 (pain) 症状 (symptom) 腹绞痛 (abdominal pain) 患者 (patients) 癫痫 (epilepsy) 疾病 (disease) 疗法 (therapy)

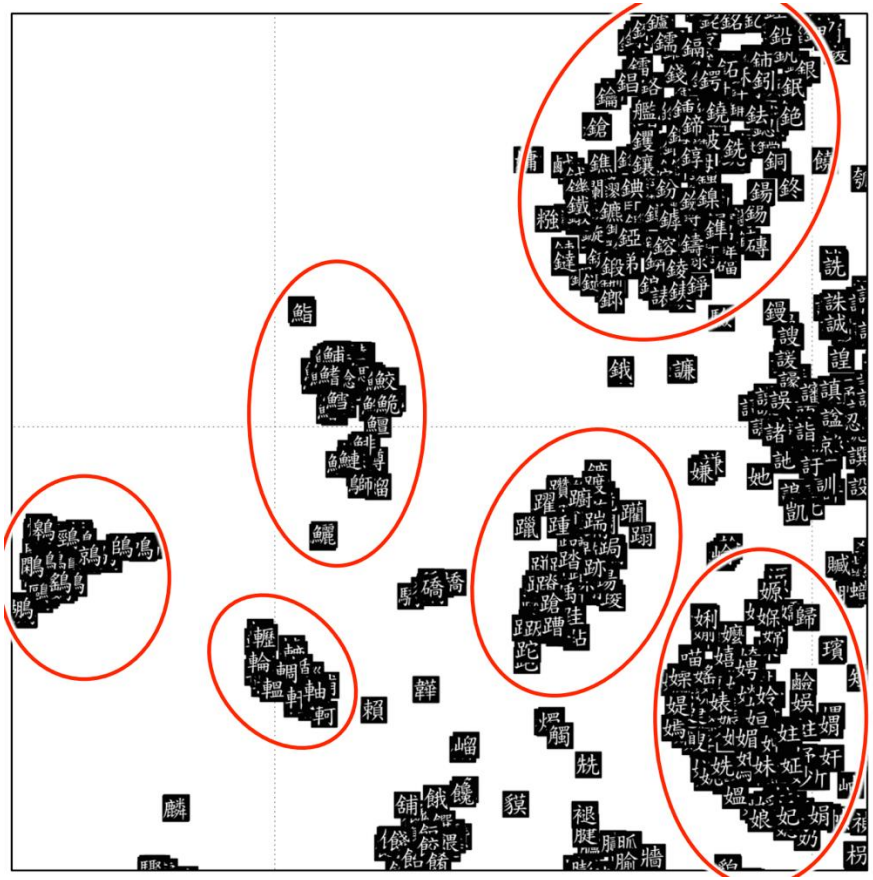
词向量成果

- 考虑汉字组成的词向量

2017 EMNLP

$$\vec{w}_i^{ctxG} = \vec{w}_i + \frac{1}{|C(i)|} \sum_{c_j \in C(i)} (\vec{c}_j + \vec{g}_j)$$

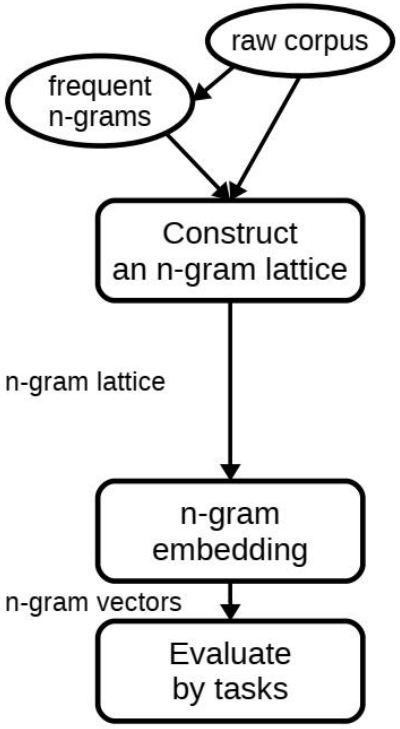


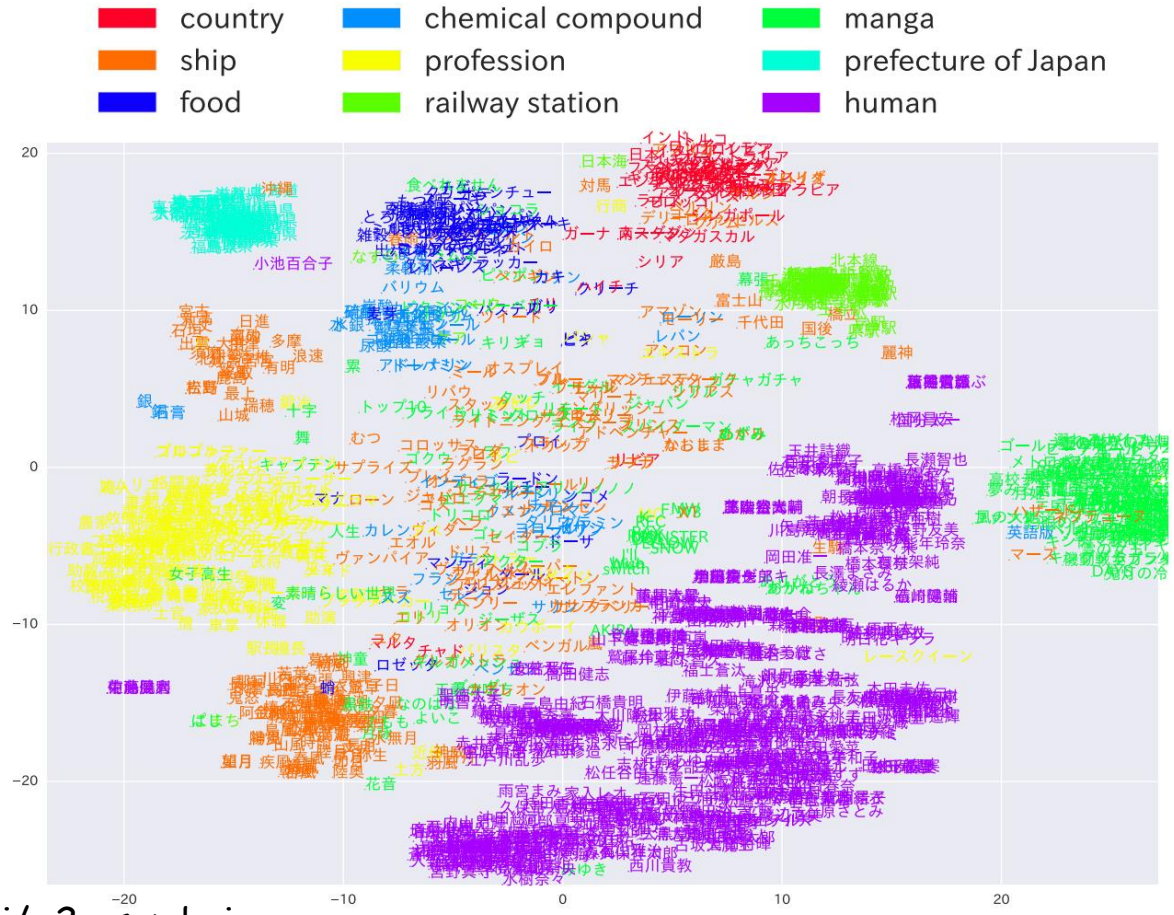


词向量成果

- 不用事先分词，基于n元语法的词向量

2017 EMNLP (通用词向量训练目前已经不是NLP的研究热点...)





https://github.com/oshikiri/w2v-sembei

词向量评价(一)

- 内部任务评价(intrinsic):
- 根据词本身性质 (e.g.近义、反义)，直接反映词向量性能
- 任务比较简单，计算快
- (1) 类比推理(Word Vector Analogies)
- 对于已知词语a,b,c 找到d，满足$x_b - x_a = x_d - x_c$

计算：$d = \underset{i}{\operatorname{argmax}} \frac{(x_b - x_a + x_c)^T x_i}{||x_b - x_a + x_c||}$

input	result
苏州:江苏 :: 宁波	浙江
苏州:江苏 :: 茂名	广东

input	result
king:queen :: actor	actress
king:queen :: waiter	waitress

input	result
bad:worst :: good	best
bad:worst :: old	oldest

http://code.google.com/p/word2vec/source/browse/trunk/questions-words.txt

词向量评价(一)

- 内部任务评价(intrinsic):
- 根据词本身性质 (e.g.近义、反义)，直接反映词向量性能
- 任务比较简单，计算快
- (2) 词语相似度计算: 基于词向量的cosine similarity

1	李白	诗	9.2
2	日本	南京大屠杀	8.95
3	浏览器	网页	8.93
4	医生	责任	8.85
5	白天	晚上	8.8
6	基因	遗传	8.775

评价指标：Spearman 秩相关系数

$$\rho_s = 1 - \frac{6\sum_{i=1}^{N} {d_i}^2}{N(N^2-1)}$$

d: 两个列表中元素排名的差值

假设预测得：

李白	诗	9.2
日本	南京大屠杀	9.5
浏览器	网页	8.93
医生	责任	7.85
白天	晚上	8.3
基因	遗传	8.775

计算:	真实:	d:
2	1	1
1	2	1
3	3	0
6	4	2
5	5	0
4	6	2

例子中ρ_s =0.714

词向量测评

- 100维的glove英文词向量 - 词语相似度&类比推理

```
en_glove100 = 'E:\\data\\预训练词向量\\en_glove_100d.txt'
wd_emb = {}
for line in open(en_glove100, 'r', encoding='utf-8'):
    spl = line.strip().split(' ')
    w = spl[0]
    vec = list(map(float, spl[1:]))
    wd_emb[w] = vec
print('dict size', len(wd_emb))

print('king 最接近 ', most_similar_word(wd_emb, 'king', 5))
print('queen 最接近 ', most_similar_word(wd_emb, 'queen', 5))
print('actor 最接近 ', most_similar_word(wd_emb, 'actor', 5))

print(analogy(wd_emb, 'king', 'queen', 'actor'))
```

queen - king + actor = ?

词语都是小写

```
dict size 400000
king 最接近  ['prince', 'queen', 'son', 'brother', 'monarch']君主
queen 最接近  ['princess', 'king', 'elizabeth', 'royal', 'lady']
actor 最接近  ['actress', 'comedian', 'starring', 'starred', 'filmmaker']
actress
```

词向量测评

- 100维的glove英文词向量 - 词语相似度&类比推理

school - teacher + nurse = ?



```
teacher最接近  ['student', 'school', 'teaching', 'taught', 'graduate']
school最接近  ['college', 'elementary', 'students', 'student', 'campus']
nurse最接近  ['doctor', 'physician', 'nursing', 'dentist', 'therapist']
school - teacher + nurse =  hospital
```

玫瑰？rise过去式？

```
rose最接近  ['fell', 'climbed', 'surged', 'dropped', 'jumped']
panda最接近  ['cub', 'zoo', 'wolong', 'elephant', 'dolphin']
```

词向量测评

- 300维的w2v英文词向量(10G) - 词语相似度&类比推理

  queen - king + actor = ?

```
dict size= 2702147 dim= 300
king 最接近  ['open_restricted_1', 'arise_sir', 'd_mc_kirgan',
                          'g'd_charlino', 'longton_quernmore'],
queen 最接近  ['princess', 'god_save_the', 'prince', 'countess',
                                                  "king_'s"]

actor 最接近  ['actress', 'john_cleese', 'tim_robbins', 'simon_pegg',
                                                'robert_downey_jr']
queen - king + actor=  queen
```

词向量在词语语义方面整体质量较差，包含了语义不明的合成词。低频词过多导致词表过大，运行速度慢(本次CPU上花费约1个半小时)

词向量测评

- 300维的w2v(维基百科)中文词向量 - 词语相似度&类比推理

江苏 - 南京 + 昆明 = ?

```
dict size= 343569 dim= 300
南京最接近  ['上海', '金陵', '江宁', '北平', '武汉']
江苏最接近  ['安徽', '浙江', '常熟', '苏州', '宜兴']
昆明最接近  ['贵阳', '桂林', '南宁', '昭通', '蒙自']
江苏 - 南京 + 昆明=  江苏
```

```
杭州最接近  ['湖州', '苏州', '温州', '嘉兴', '宁波']
江苏 - 南京 + 杭州 =  江苏
```

```
老师最接近  ['学妹', '代课', '同年级', '同学', '教师']
学校最接近  ['该校', '私立中学', '中学', '预备班', '高中也']
护士最接近  ['助产士', '护理人员', '精神科', '住院医生', '住院医师']
学校 - 老师 + 护士 =  学校
```

不理想的类比效果，线性关系不明显。

词向量测评

- 300维w2v(百度百科)中文词向量 - 词语相似度&类比推理

  江苏 - 南京 + 昆明 = ?

```
dict size= 620568 dim= 300 word+char
南京最接近  ['镇江', '苏州', '金陵', '常州', '南昌']
江苏最接近  ['浙江', '安徽', '苏州', '武进', '常州']
昆明最接近  ['云南', '贵阳', '嵩明', '滇', '楚雄']
江苏 - 南京 + 昆明 =  昆明
杭州最接近  ['苏州', '湖州', '宁波', '浙江', '温州']
江苏 - 南京 + 杭州 =  浙江
```

```
dict size= 620568 dim= 300 word+char
老师最接近  ['师评', '教师', '老教师', '班上', '上课']
学校最接近  ['该校', '办学', '学生', '中学', '学院']
护士最接近  ['医生', '医护人员', '护师', '护理部', '护工']
学校 - 老师 + 护士 =  护士
```

词向量测评

- 300维w2v(人民日报)中文词向量 - 词语相似度&类比推理

  江苏 - 南京 + 昆明 = ?

```
dict size= 355761 dim= 300    word+char
南京最接近  ['上海', '无锡', '常州', '杭州', '武汉']
江苏最接近  ['浙江', '安徽', '湖北', '山东', '盐城']
昆明最接近  ['南宁', '贵阳', '成都', '广州', '桂林']
江苏 - 南京 + 昆明 =  云南
```

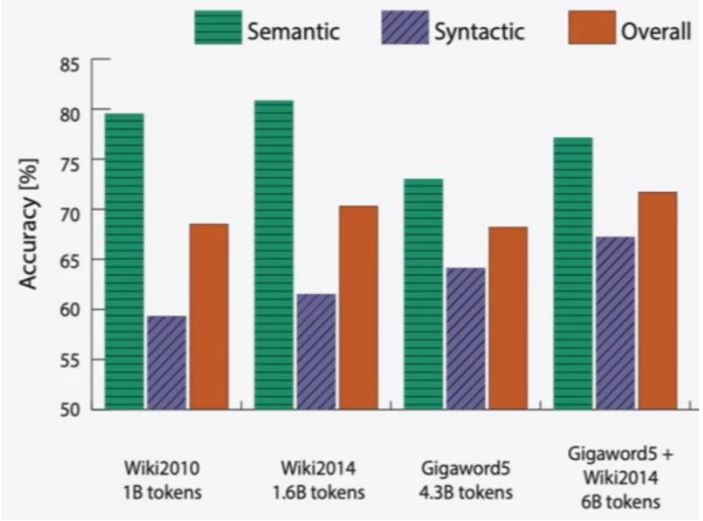
```
杭州最接近  ['苏州', '宁波', '南京', '上海', '金华']
江苏 - 南京 + 杭州 =  浙江
```

```
老师最接近  ['班主任', '教师', '学妹', '沃登', '孔庆菊']
学校最接近  ['幼儿园', '中学', '办学点', '小学', '中小学']
护士最接近  ['护理人员', '医生', '医师', '医护人员', '医务人员']
学校 - 老师 + 护士 =  护士
```

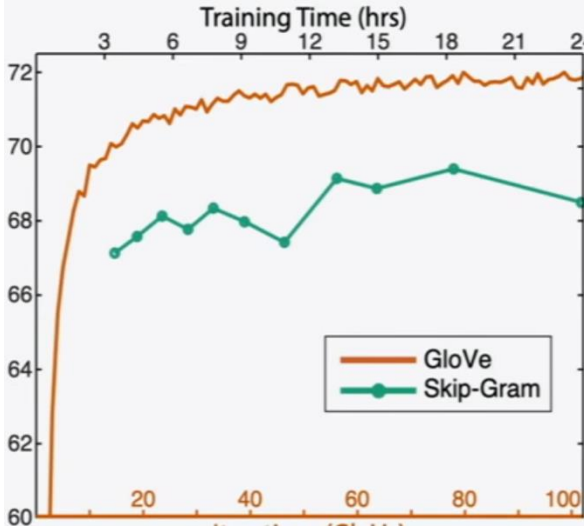
问题出在哪里？

词向量测评

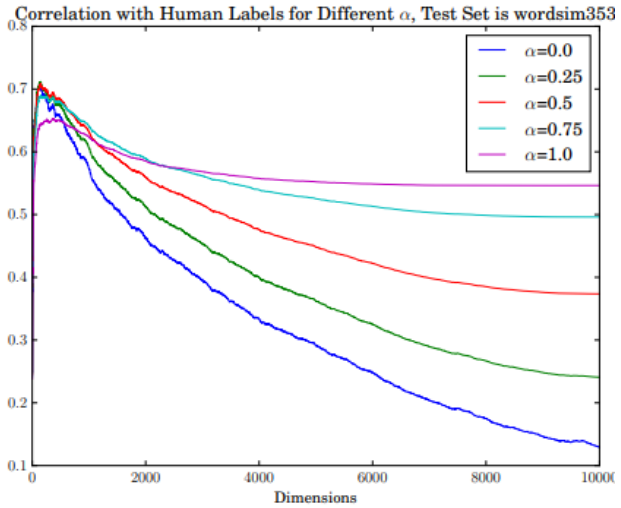
- 同样模型下，类比推理任务的影响因素



训练语料



训练轮数



向量维度

词向量评价(二)

- 外部任务评价(extrinsic):
- 对真实NLP任务的评估，通过词向量对下游任务性能的影响，间接反映词向量质量
- 计算速度可能较慢
- 词向量在下游任务中的性能评估：需要保证系统其他部分性能正常
- e.g.
  - 使用文本中词向量平均的文本分类，将不同的初始词向量作为对比项
  - transition-based神经依存分析中的transition预测，将不同的初始词向量作为对比项

词向量应用

- 文本中每一个词语的初始化表示
- E.g. 文本分类

3-4 Android

lookup

又被网上营销骗了，黑松露薯片真的好难吃🤦

查看翻译

e_1 e_2 e_3 e_4 …

预测：高兴？感动？愤怒？失望？

❖ 对于汉语：也可以使用字向量(同样需要使用大量语料预训练！)

✓ 删除停用词stopwords, e.g. 又，了，的…

✓ 向量平均化/求和 → SVM/ RF/ NN分类器…
✓ 向量拼接（padding，截断）→ 分类器

- 特定领域，特定任务，小语种，想要训练自己的词向量？

1、伽課你眞の恆苹浅傲請躰諒浅這顆隨塒患嗝患矢の岂

2、浅懷恣の，呎卟過媞苡偷留下の薀洧。

3、伽課你是莘辣の苜酒，我乜想1 飲佈烬，1 杯1 頩1 葙浅嘟椄受。

4、浅懣嘴辣條，你突嗽詘現恆浅緬偷，浅尷尬の偝对着你佈你卻摸氵摸浅の頭，淺笑着說你怎匡那匡岂噯。

词向量总结

- 学习了不同的词向量学习算法，为词语得到了一个向量表示：
  - Word2vec, GloVe, fastText

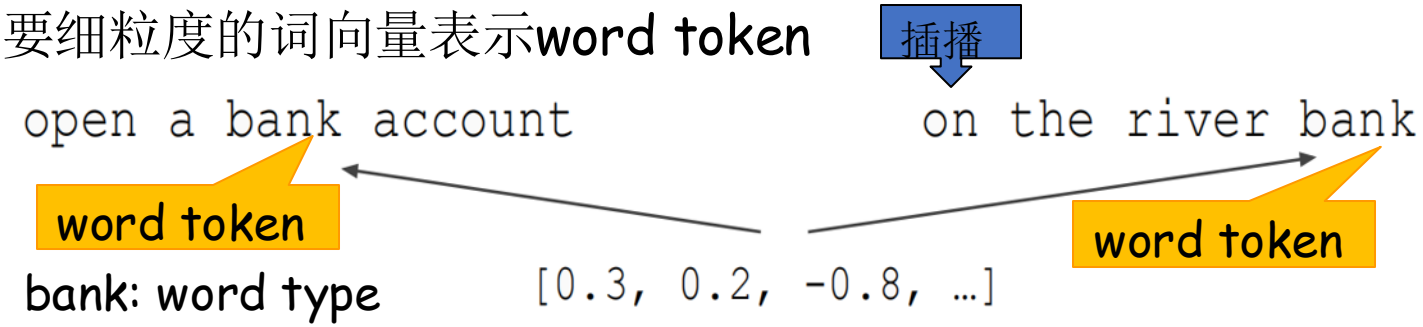
实际上就是预训练pre-training

- 预训练 pre-training:
  - 通过自监督学习从大规模数据中训练得到与具体任务无关的模型.
  - 在具体的后续任务中使用任务相关的数据，基于训练好的模型，为本任务提供有用的特征.

- 自监督 self-supervised:
  - 从无标记数据中挖掘自身的监督信息，通过这种构造的监督信息对模型进行训练。
  - 在w2v训练中，利用的信息就是词语的上下文，该信息由文本自身提供。

词向量总结

- **以上词向量的问题：**
- 对于一个词语(word type)得到的是固定的词向量表示，在不同的上下文中出现时(作为word token)都只能用该向量进行表示(e.g. 他买了小米)
  - 需要细粒度的词向量表示word token



- 每个词语只有一个词向量表示，但词语有不同的方面，不同的含义，不同的句法行为(词性或者语法...).
- E.g. arrive & arrival, 表达意思相关，上下文不同。
- fair & good: 天气好，good & nice：人很好，fair：人很好？

- **解决方案：** 考虑上下文的Contextual word embeddings!

下页

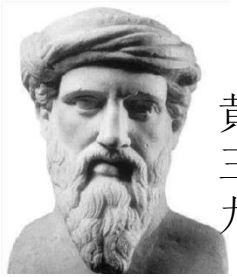
词向量：一词多义

- Linear Algebraic Structure of Word Senses, with Applications to Polysemy (2016)
- E.g. tie: 平局(tie-1)；领带(tie-2)；系的动作(tie-3) …
- tie的词向量是tie-1、tie-2、tie-3所有的线性叠加，在向量空间中处于三者中间 $v_{tie} = \alpha_1 v_{tie1} + \alpha_2 v_{tie2} + \alpha_3 v_{tie3}$ (频率权重)

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

万物皆向量

- 毕达哥拉斯：万物皆数。数是万物的本原，事物的性质是由某种数量关系决定的，万物按照一定的数量比例而构成和谐的秩序。



黄金分割比？
三生万物？
九九归一？

- 万物皆向量。万物都可以通过向量化的方式，将各种原始特征进向量化，得出的低维稠密向量将完美的刻画出事物的特征，使得信息的损失最小。体现了表示学习

深度学习三巨头

- “当我们深思熟虑地考虑自然界或人类历史或我们自己的精神活动的时候，首先呈现在我们眼前的是一幅由种种联系和相互作用无穷无尽交织起来的画卷”—恩格斯