Word2vec

王妍

2021年10月19日

一、背景介绍

◆ 词的表示方式

自然语言处理（NLP）相关任务中，要将自然语言交给机器来处理，首先需要将语言数学化，转换为机器可以理解的表示。

（1）One-hot representation

“话筒”表示为[0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]^T

“麦克”表示为[0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]^T

优定：表示简单

缺点：① 词越多维数越高，容易受到维度诅咒的困扰；

② 向量中元素都是离散的，只能取0或1，且只有一位能起决定性作用，不能很好地表示词和词之间的关系，任意两个词之间都是孤立的。

一、背景介绍

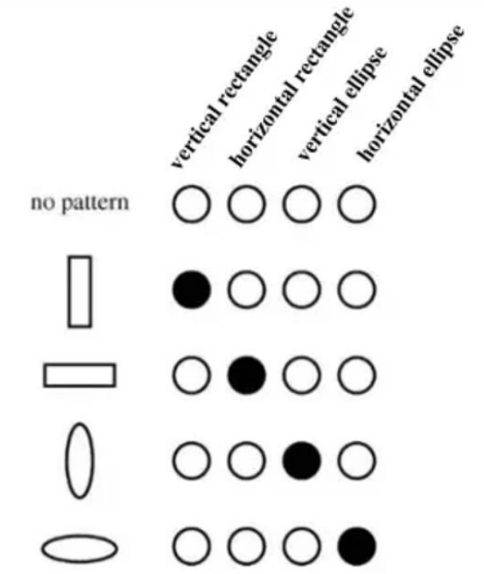
◆ 词的表示方式

（2）Distributed representation/ Word embedding

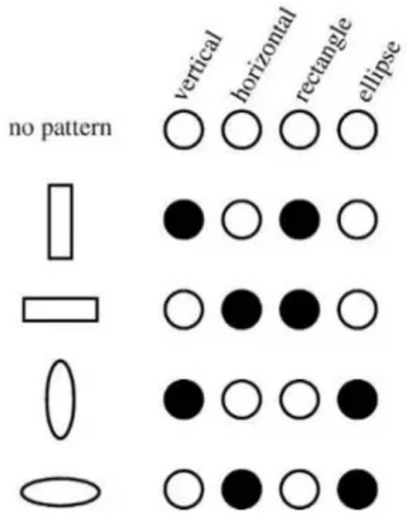
基本想法：直接用一个普通的向量表示一个词，以50维和100维比较常见，远小于一般情况下词库的大小。例如：[0.286, 0.792, -0.177, -0.108, -0.178, 0.156, 0.243, …]

特点：克服了one-hot representation的缺点，降低了单词表示的维数，向量的元素值是连续的，且向量的每个维度上的值都会影响单词的表达，用较好的训练算法得到的词向量一般是有空间上的意义的。

One-hot representation



Distributed representation



1986 由Hinton提出Distributed representation的概念

2003 Bengio首次使用神经网络训练词向量

2013 Tomas Mikolov团队word2vec

一、背景介绍

◆ 词向量的应用

（1）衡量词语之间的相似程度

Similarity(word1, word2) = Distance(wordvec1, wordvec2)

1.Frog

2.Frogs

3.Toad

4.Litoria

5.Leptodactylidae

6.Rana

7.Lizard

8.Eleutherodactylus

3. litoria

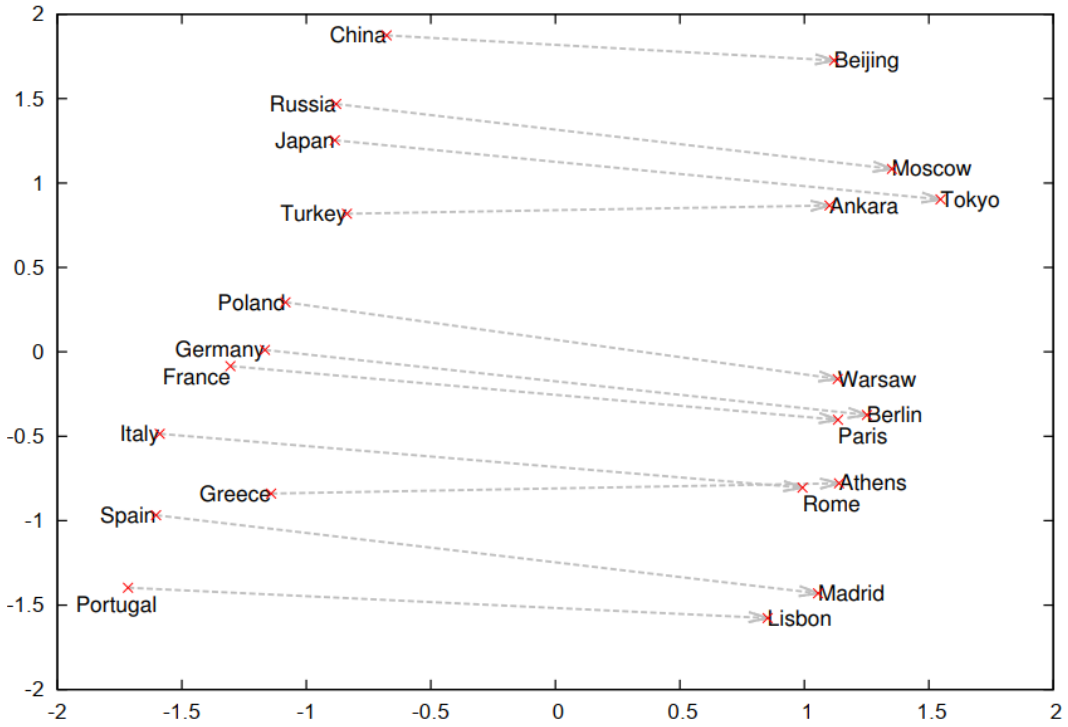
4. leptodactylidae

5. rana

7. eleutherodactylus

（2）词类比

Distance(wordvec1-wordvec2+ wordvec3， wordvec4)



wordvec(China) - wordvec(Beijing) = wordvec(Russia) – wordvec(Moscow)

（3）作为预训练模型提升NLP任务，如命名实体识别、文本分类等。

一、背景介绍

◆ 语言模型

语言模型是计算一个句子是句子的概率的模型，在NLP任务中应用广泛，比如机器翻译、语音识别中得到若干候选解之后，可以利用语言模型选择一个较好的结果。

目前较常用的是统计语言模型，通过概率计算来刻画语言模型：

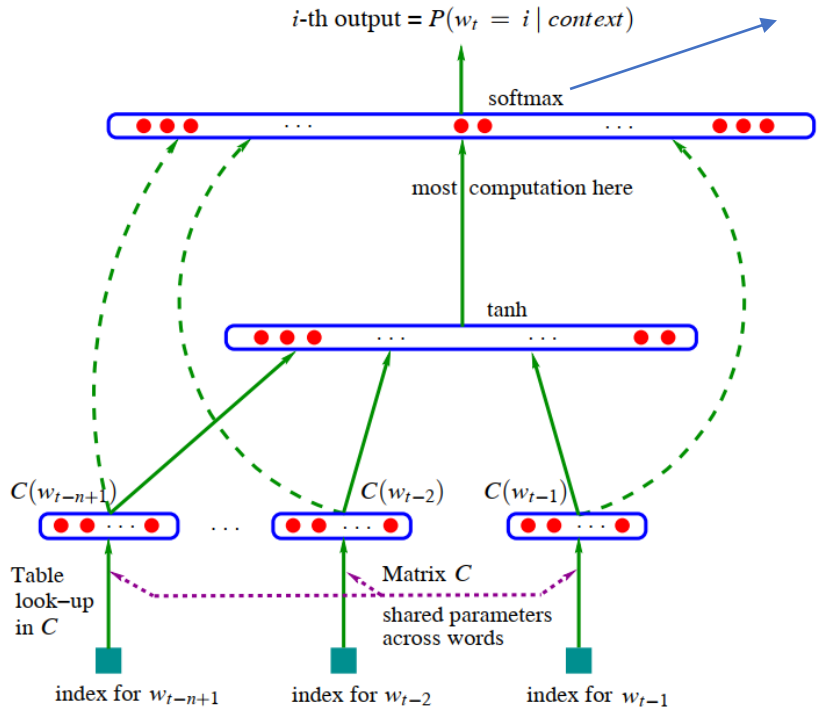
$$P(s) = P(w_1, w_2, \ldots, w_n)$$

$$\begin{aligned} & P(w_1, w_2, \ldots, w_n) \\ = & P(w_1)P(w_2|w_1)P(w_3|w_1w_2)\ldots P(w_n|w_1w_2\ldots w_{n-1}) \\ = & \prod_{t=1}^{T} P\left(w_t|context(w_t)\right) \end{aligned}$$

$$w_{t-n+1} \quad w_{t-n} \quad \ldots \quad w_{t-1} \boxed{w_t}$$

用神经网络拟合分布函数

$$softmax(V_i) = \frac{e^{V_i}}{\sum_j e^{V_j}}$$



Feedforward Neural Net Language Model(NNLM)，2003

二、Word2vec

◆ 基本原理

语言模型的基本思想：句子中下一个词的出现和前面的词是有关系的，所以可以使用前面的词预测下一个词。

Word2vec的基本思想：句子中相近的词之间是有关系的，所以word2vec的基本思想就是用词来预测附近的词，skip-gram使用中心词预测周围词，cbow使用周围词预测中心词。

◆ Word2vec-Skip-gram

假设1：window size：[-c, c]

$$P(w_1, w_2, \ldots, w_T) = P(w_t)P(context(w_t)|w_t) \triangleq P(w_t)P(w_{t-c}, \ldots w_{t-1}, w_{t+1}, \ldots, w_{t+c}|w_t)$$

Conditional likelihood：

$$\prod_{t=1}^{T} P\left(w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}|w_t\right)$$

假设2：P(context(w_t)|w_t)独立同分布

Average conditional log likelihood：

$$\frac{1}{T}\sum_{t=1}^{T} \log P(w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}|w_t)$$

$$\triangleq \frac{1}{T}\sum_{t=1}^{T} \log \prod_{i\in[-c,c], i\neq 0} P\left(w_{t+i}|w_t\right)$$

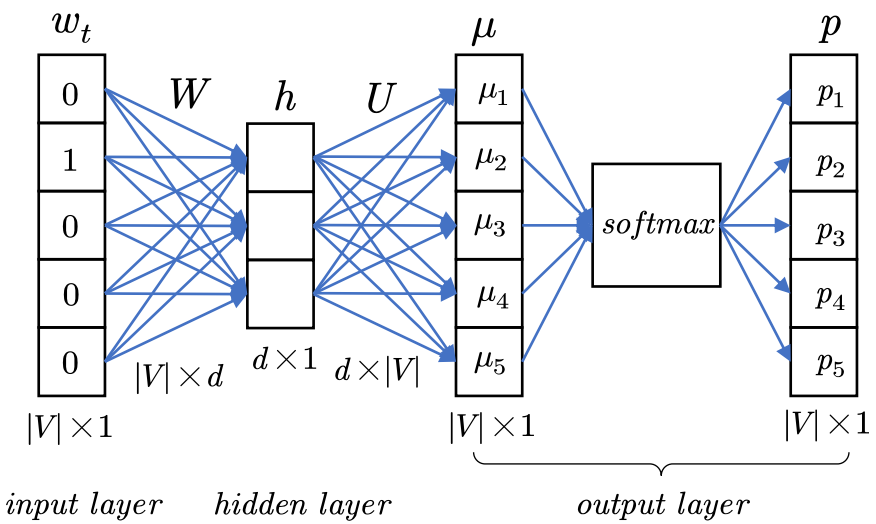
假设3：P(W_t+i|Wt) 独立同分布

$$= \frac{1}{T}\sum_{t=1}^{T} \sum_{i\in[-c,c], i\neq 0} \log P(w_{t+i}|w_t)$$

二、Word2vec

◆ Word2vec-Skip-gram

$$Loss = -\frac{1}{T}\sum_{t=1}^{T}\sum_{i\in[-c,c],i\neq 0}\log P(w_{t+i}|w_t)$$



神经网络正向传播过程

$$w_t^T \cdot W \cdot U = \mu^T$$

$$W = \begin{pmatrix} v_1^T \\ v_2^T \\ \cdots \\ v_{|V|}^T \end{pmatrix} \qquad w_t^T \cdot W = v_t^T$$

$$U = (u_1, u_2, \ldots, u_{|V|}) \qquad \mu_{t+i} = v_t^T \cdot u_{t+i}$$

$$\begin{aligned} p_{t+i} &= P(w_O = w_{t+i}|w_I = w_t) \\ &= \text{softmax}(\mu_{t+i}) \\ &= \frac{\exp(v_t^T \cdot u_{t+i})}{\sum_{k=1}^{|V|}\exp(v_t^T \cdot u_k)} \end{aligned}$$

反向传播过程推导

Rong X . word2vec Parameter Learning Explained[J]. Computer Science, 2014.

二、Word2vec

◆ Word2vec-CBOW

$$P(w_1, w_2, \ldots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1w_2)\ldots P(w_n|w_1w_2\ldots w_{n-1})$$

$$= \prod_{t=1}^{T} P\left(w_t|context(w_t)\right)$$

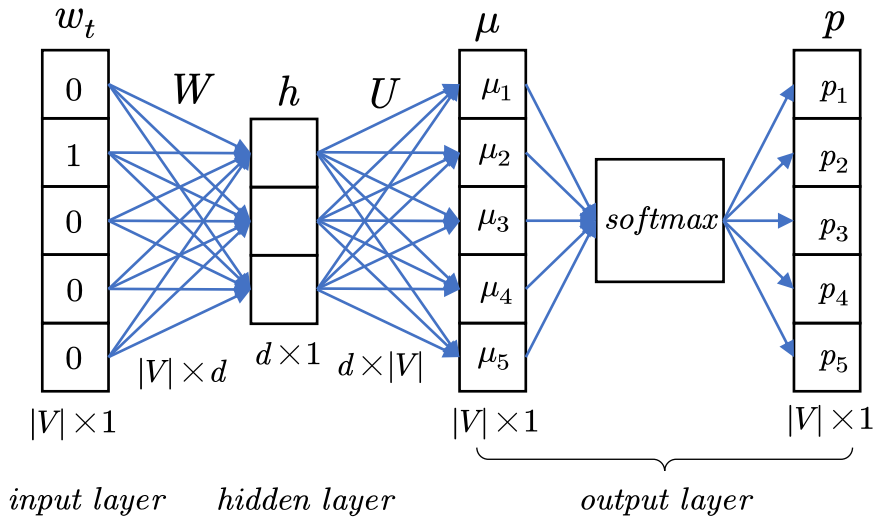
$$\triangleq \prod_{t=1}^{T} P\left(w_t|w_{t-c}, \ldots, w_{t-1}, w_{t+1}\ldots, w_{t+c}\right)$$

likelihood $$\prod_{t=1}^{T} P\left(w_t|w_{t-c}, \ldots, w_{t-1}, w_{t+1}\ldots, w_{t+c}\right)$$

$$\triangleq \prod_{t=1}^{T} \prod_{i\in[-c,c], i\neq 0} P\left(w_t|w_{t+i}\right)$$

average log likelihood $$\frac{1}{T}\sum_{i=1}^{T} \sum_{i\in[-c,c], i\neq 0} logP\left(w_t, w_{t+i}\right)$$

$$Loss = -\frac{1}{T}\sum_{t=1}^{T} \sum_{i\in[-c,c], i\neq 0} logP(w_t|w_{t+i})$$



二、Word2vec

假设1：window size：[-c, c]

假设2：P(context(w_t)|w_t)独立同分布

假设3：P(W_t+i|Wt) 独立同分布

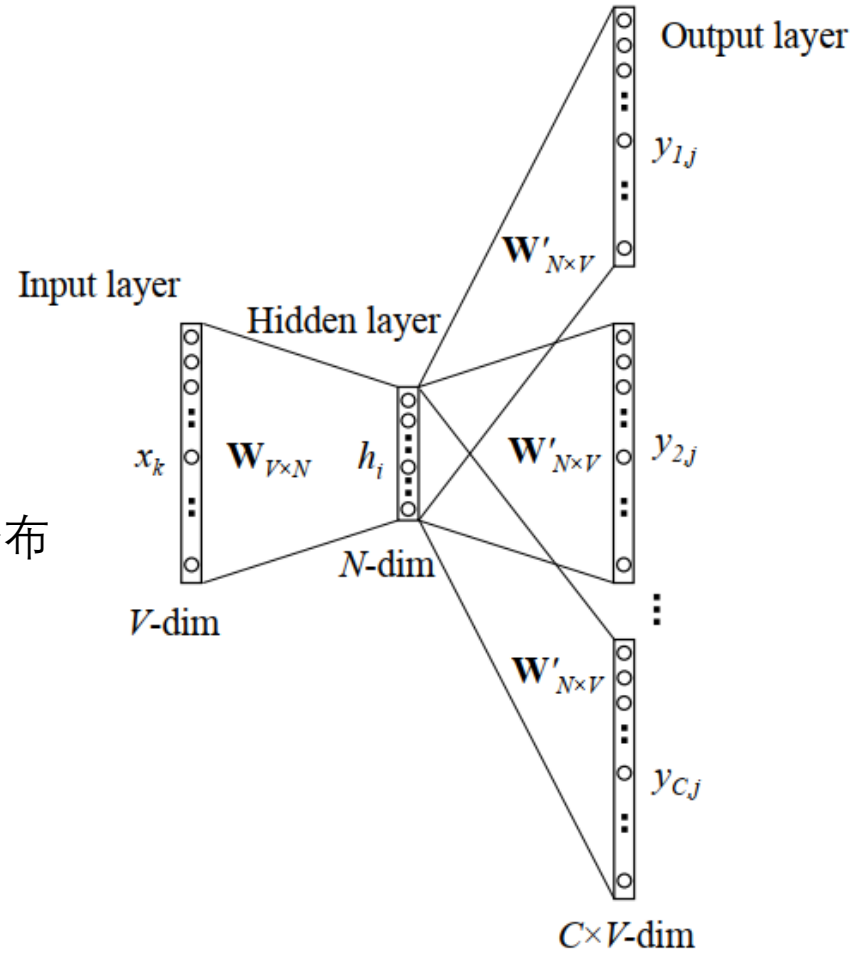


Figure 3: The skip-gram model.

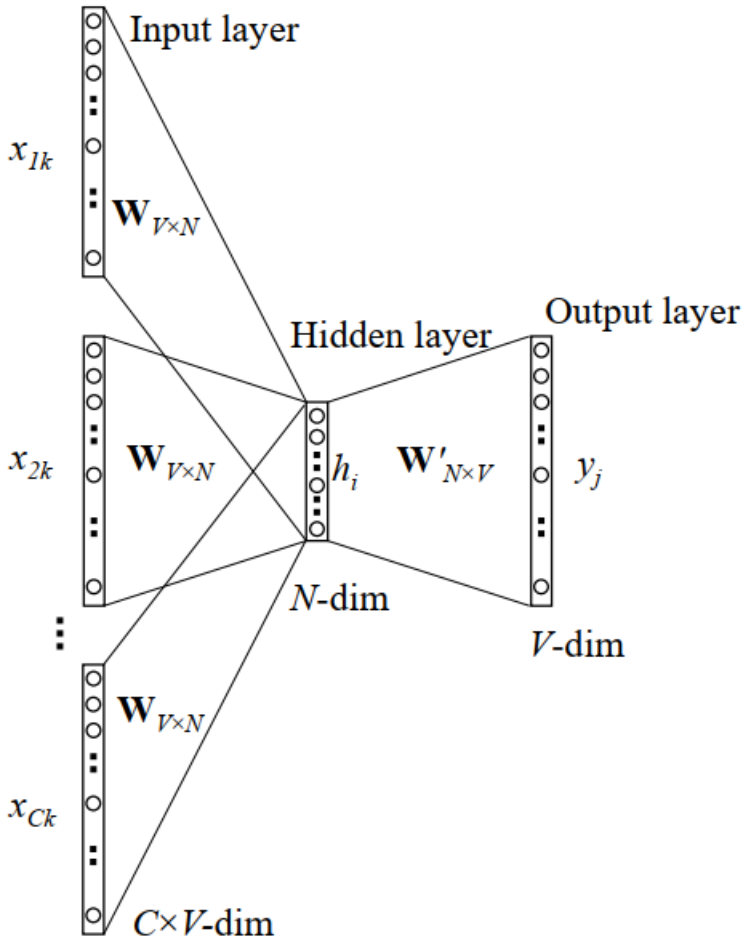


Figure 2: Continuous bag-of-word model

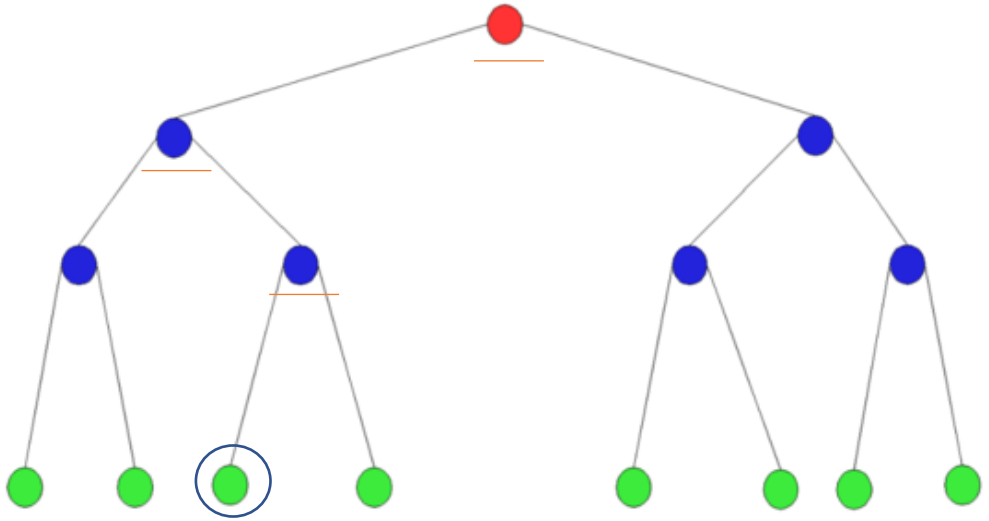
三、效率优化

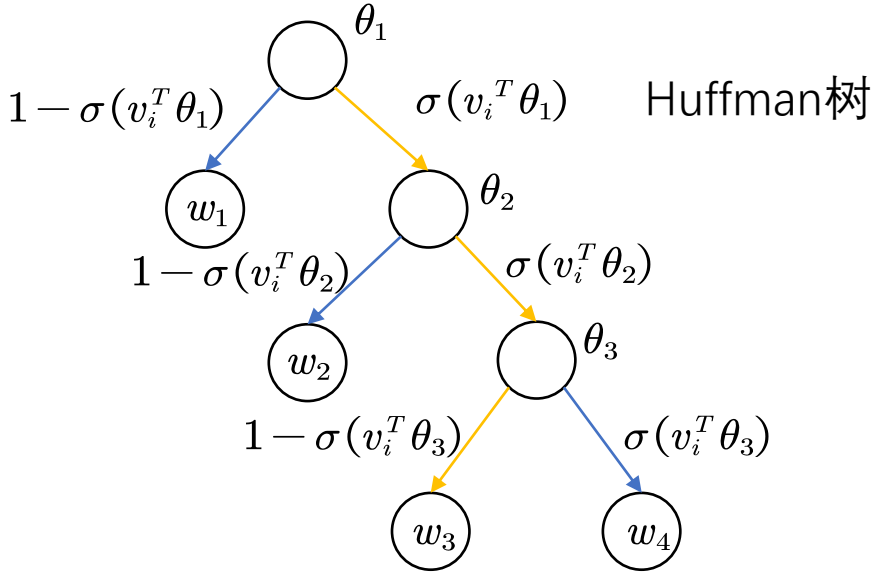
◆ Hierarchical Softmax

基本思想：把N分类问题转化成log(N)次二分类

$$p_j = P(w_O = w_j | w_I = w_i)$$
$$= \text{softmax}(\mu_j)$$
$$= \frac{\exp(v_i^T \cdot u_j)}{\sum_{k=1}^{N} \exp(v_i^T \cdot u_k)}$$

$$\text{Sigmoid}(\mu_j) = \frac{1}{1+\exp(-\mu_j)}$$





Huffman树

$$p(w_{t+i}|w_t) = \prod_{j=1}^{L(w_{t+i})-1} \sigma\left([[n(w_{t+i},j+1) = ch(n(w_{t+i},j))]] \cdot v_t^T \theta_j\right)$$

$L(w)$：树的高度

$n(w,j)$：词w在树上的第j个节点

$ch(n(w,j))$：n(w, j)节点的左孩子

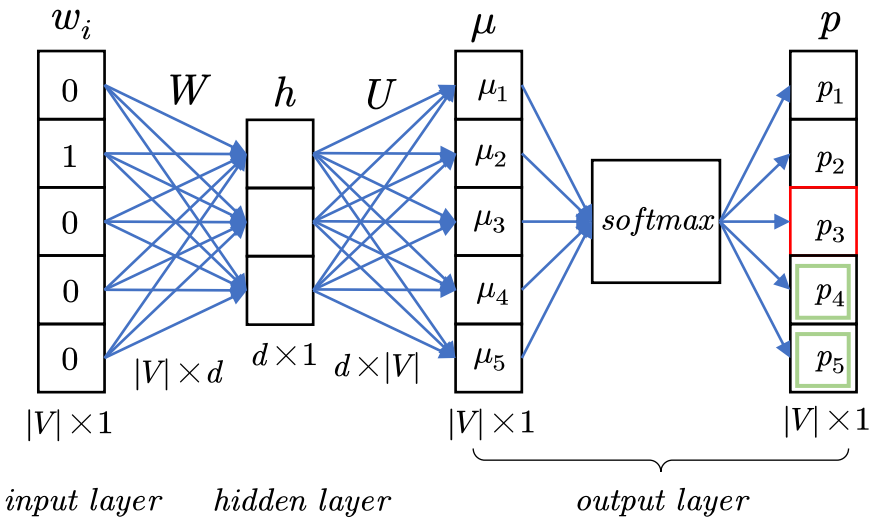
θ_j：词w在树上第j个节点的参数

三、效率优化

◆ Negative Sampling

基本思想：只预测总体类别的一个子集，提升速度

$$Loss = -\log\sigma(v_i^T \cdot u_j) + \sum_{k \in neg} \log\sigma(v_i^T \cdot u_k), k << |V|$$



Thanks for listening!