西安交通大学
XI'AN JIAOTONG UNIVERSITY

分布式表示 2

李辰

2024年10月

1 PART

2 分布式表示

- 动机
- 神经网络语言模型
- 词嵌入（Word2vec）

1 PART

2.1 动机

- 分布式表示是一种形式紧凑的表示和低维表示
- 向量表示的每一个单一成分都没有其自身的任何意义
- 可解释的特征隐藏并分布在不可解释的向量分量中

1 PART

2.2 语言模型

- 语言模型为每一词序列赋予一个概率$P_r(w_1, w_2, \ldots, w_n)$

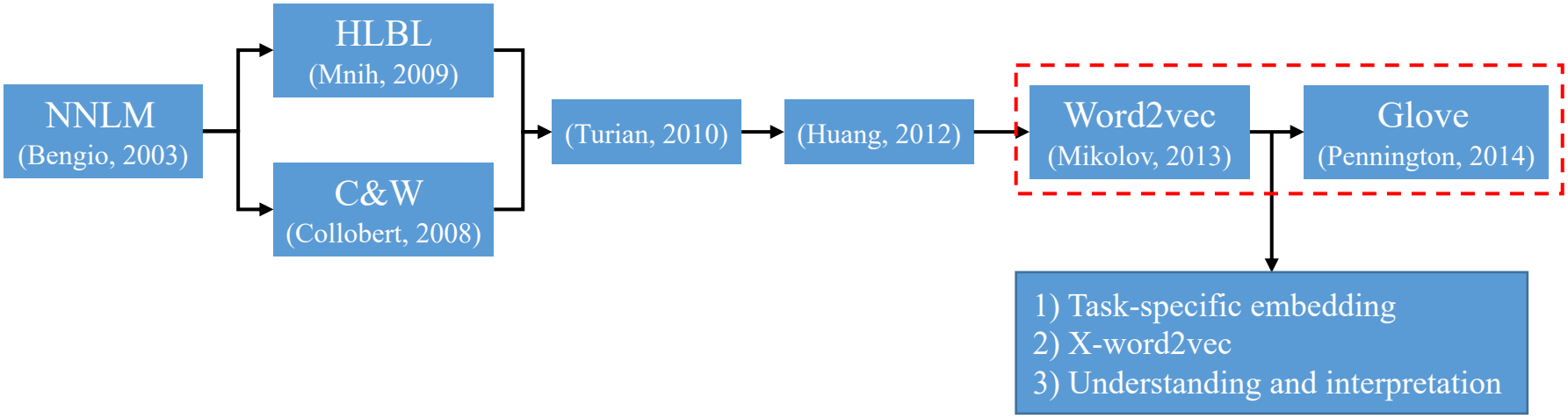
- 模型种类
  - Unigrams(1-gram)： $P_r(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} P_r(w_i)$
  - Bigrams(2-gram)： $P_r(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} P_r(w_i|w_{i-1})$
  - NNLM：神经网络语言模型

- 举例
  - S1 = “the cat jumped over the dog”, $P_r(S_1) = 1$
  - S2 = “jumped over the the cat dog”, $P_r(S_2) = 0$

2.2 神经网络语言模型

- NNLM：用神经网络实现语言模型

- NNLM路线图：



1 PART

2.3 词嵌入（Word2vec）

- 词嵌入：将词作为基本的语言单位，并通过一些方法得到词表示

- Word2vec：基于上下文窗口的方法。提出了两种算法和两种训练方法，以最大化词表示质量和最小计算复杂度。

1 PART

2.3 词嵌入（Word2vec）

- 词嵌入：将词作为基本的语言单位，并通过一些方法得到词表示
- Word2vec：基于上下文窗口的方法。提出了两种算法和两种训练方法，以最大化词表示质量和最小计算复杂度。
- 两种训练方法
  - CBOW：连续词袋模型
  - Skip-gram

1 PART

2.3 词嵌入（Word2vec）

- 词嵌入：将词作为基本的语言单位，并通过一些方法得到词表示
- Word2vec：基于上下文窗口的方法。提出了两种算法和两种训练方法，以最大化词表示质量和最小计算复杂度。
- 两种训练方法
  - CBOW：连续词袋模型
  - Skip-gram
- 两种效率提升方法
  - 层次 Softmax
  - 负采样

1 PART

2.3 CBOW

- 什么是 bag-of-words?

1 PART

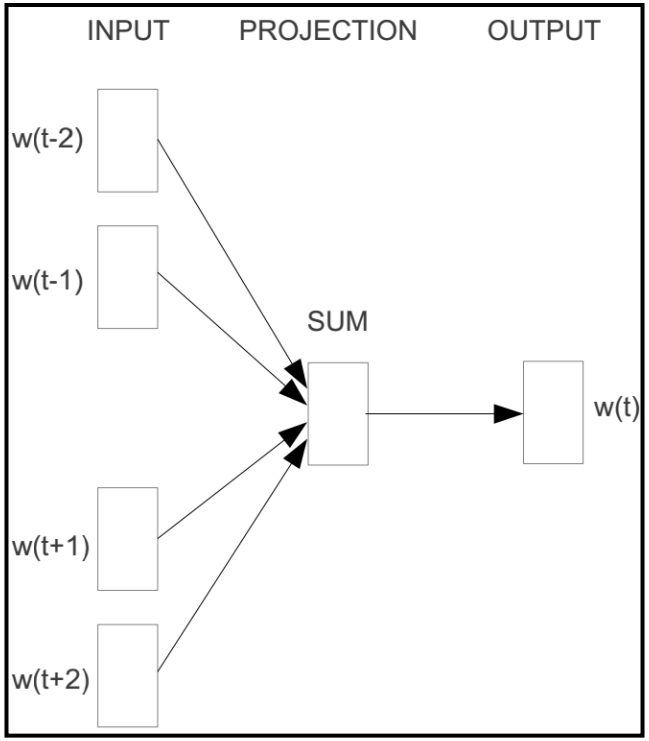
2.3 CBOW

- 什么是 bag-of-words?
- Continuous bag-of-words：根据上下文预测中心词

1 PART

2.3 CBOW

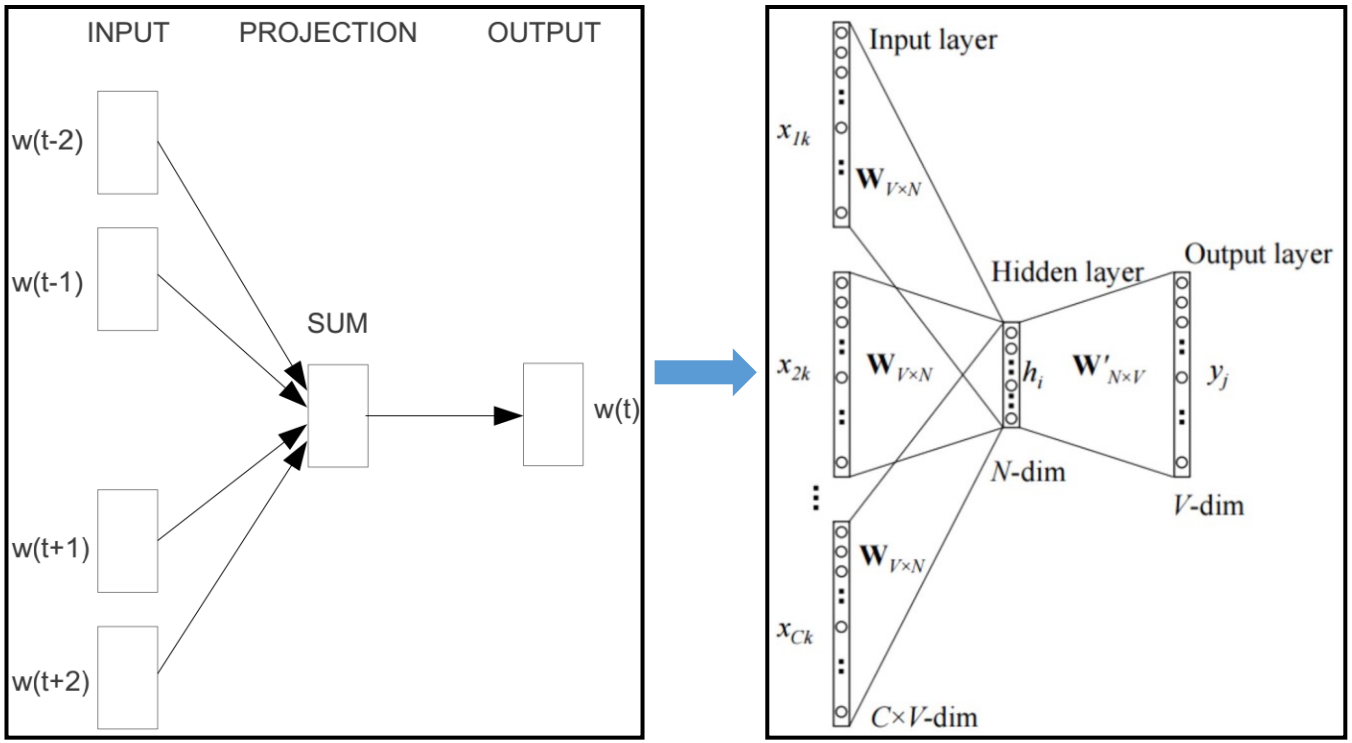
- 什么是 bag-of-words?
- Continuous bag-of-words：根据上下文预测中心词
- 架构



1 PART

2.3 CBOW

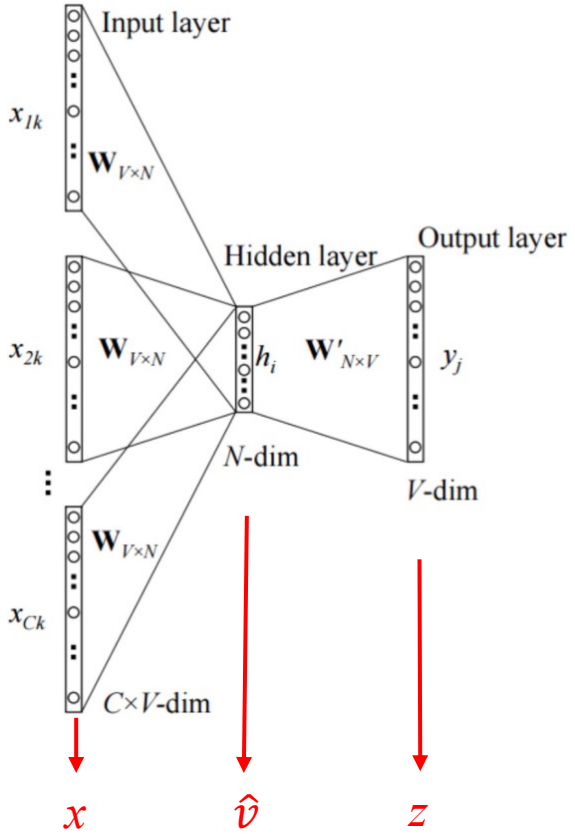
- 什么是 bag-of-words?
- Continuous bag-of-words：根据上下文预测中心词
- 架构



2.3 CBOW

- 流程

1. Generate one-hot word vectors for the input context of size m:
   $(x^{c-m}, \ldots, x^{c-1}, x^{c+1}, \ldots, x^{c+m} \in R^{|V|})$.

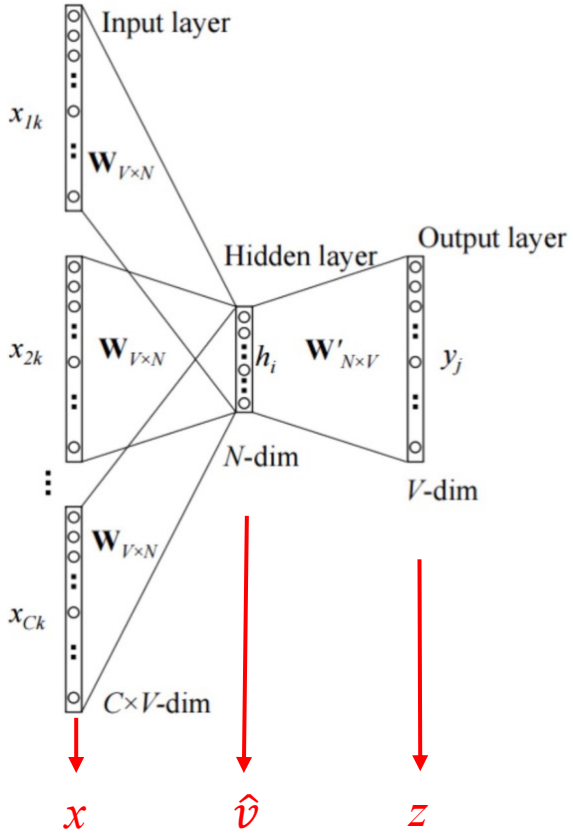


2.3 CBOW

- 流程

1. Generate one-hot word vectors for the input context of size m:
   $(x^{c-m}, \ldots, x^{c-1}, x^{c+1}, \ldots, x^{c+m} \in R^{|V|})$.

2. Get embedded word vectors for the context.
   $v_{c-m} = x^{c-m} \cdot W, \ldots, v_{c+m} = x^{c+m} \cdot W \in R^N$



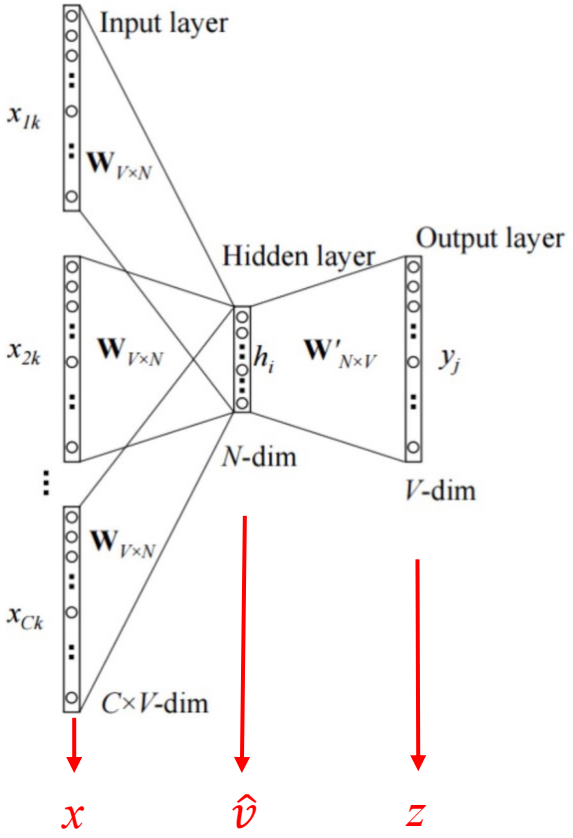
2.3 CBOW

- 流程

1. Generate one-hot word vectors for the input context of size m:
   $(x^{c-m}, \ldots, x^{c-1}, x^{c+1}, \ldots, x^{c+m} \in R^{|V|})$.

2. Get embedded word vectors for the context.
   $v_{c-m} = x^{c-m} \cdot W, \ldots, v_{c+m} = x^{c+m} \cdot W \in R^N$

3. Average context word vectors get $\hat{v}$.
   $\hat{v} = \frac{v_{c-m+\cdots+} v_{c+m}}{2m} \in R^N$



2.3 CBOW

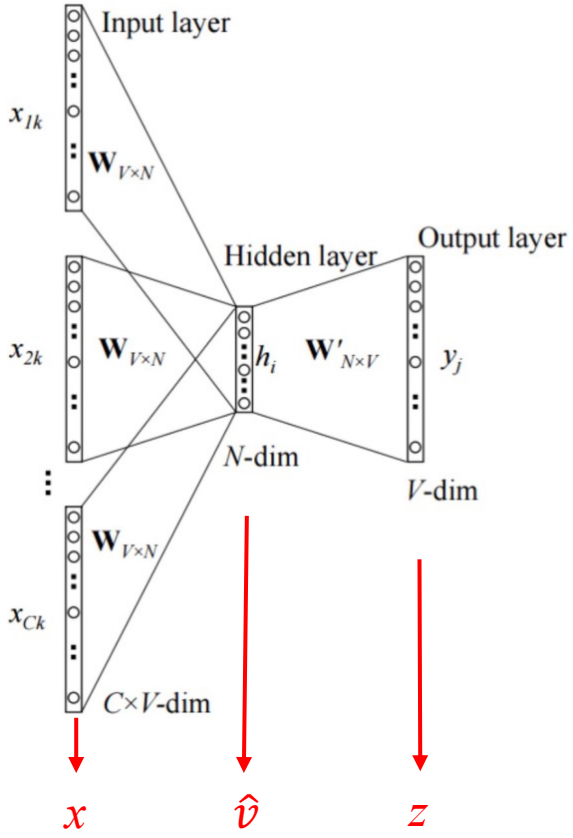
- 流程

1. Generate one-hot word vectors for the input context of size m:
   $(x^{c-m}, \ldots, x^{c-1}, x^{c+1}, \ldots, x^{c+m} \in R^{|V|})$.

2. Get embedded word vectors for the context.
   $v_{c-m} = x^{c-m} \cdot W, \ldots, v_{c+m} = x^{c+m} \cdot W \in R^N$

3. Average context word vectors get $\hat{v}$.
   $\hat{v} = \frac{v_{c-m+\cdots+} v_{c+m}}{2m} \in R^N$

4. Generate a score vector z.
   $z = \hat{v} \cdot W' \in R^{|V|}$



2.3 CBOW

- 流程

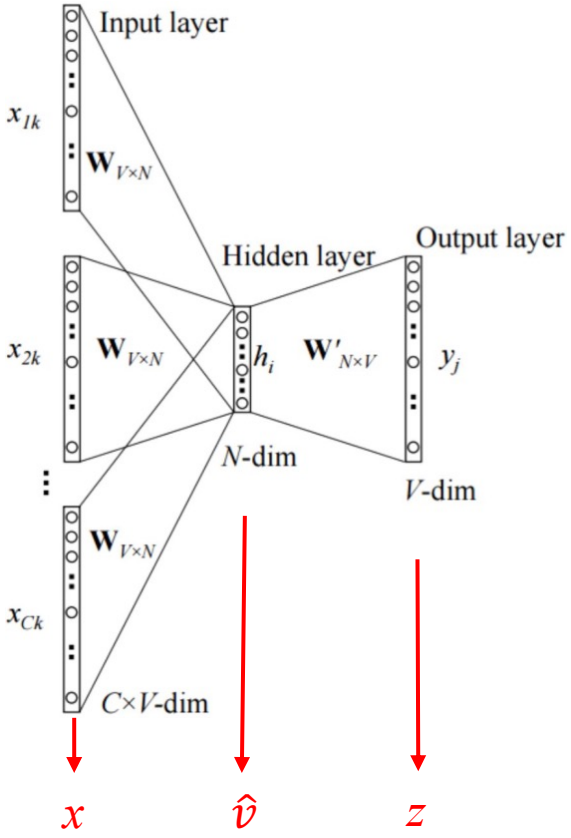
1. Generate one-hot word vectors for the input context of size m:
   $(x^{c-m}, \ldots, x^{c-1}, x^{c+1}, \ldots, x^{c+m} \in R^{|V|})$.

2. Get embedded word vectors for the context.
   $v_{c-m} = x^{c-m} \cdot W, \ldots, v_{c+m} = x^{c+m} \cdot W \in R^N$

3. Average context word vectors get $\hat{v}$.
   $\hat{v} = \frac{v_{c-m+\cdots+} v_{c+m}}{2m} \in R^N$

4. Generate a score vector z.
   $z = \hat{v} \cdot W' \in R^{|V|}$

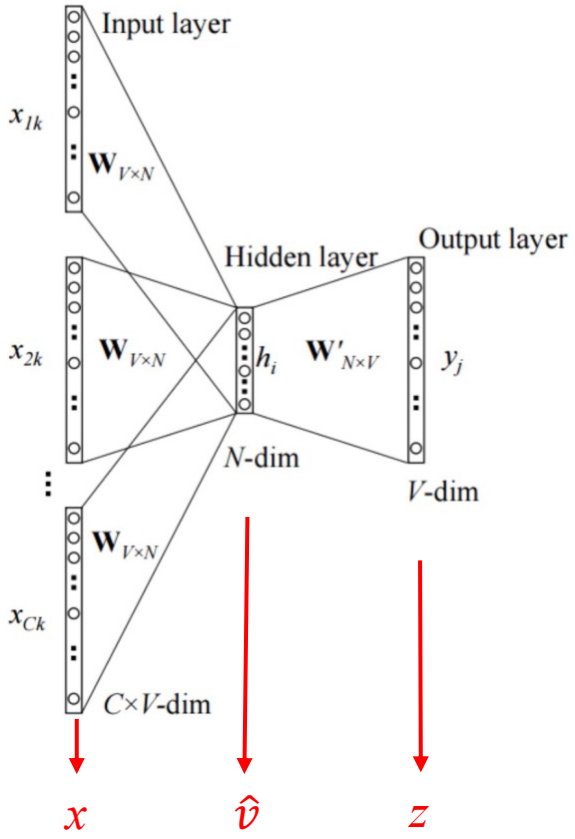
5. Turn the score vector into probabilities $\hat{y}$.
   $\hat{y} = softmax(z)$



2.3 CBOW

- 损失函数

Maximize $\mathrm{P}\left(x_c | x_{c-m}, \ldots, x_{c+m}\right)$



2.3 CBOW

- 损失函数

Maximize $\mathrm{P}(x_c|x_{c-m},\dots,x_{c+m})$

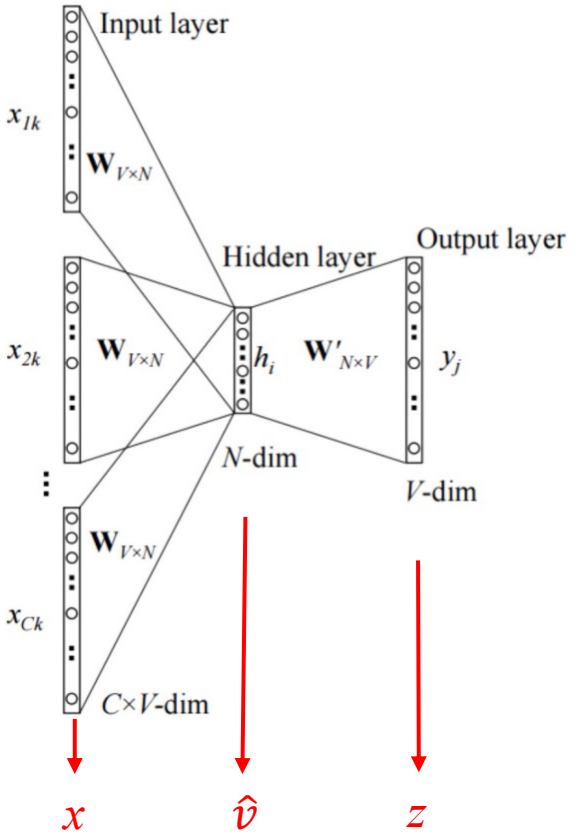
$\Updownarrow$

Minimize $\mathrm{J} = -\log \mathrm{P}(x_c|x_{c-m},\dots,x_{c+m})$

$= -\log \mathrm{P}(x_c|\hat{v})$

$= -\log \frac{\exp(x_c^T \cdot \hat{v})}{\sum_{j=1}^{|V|} \exp(x_j^T \cdot \hat{v})}$

$= -x_c^T \cdot \hat{v} + \log \sum_{j=1}^{|V|} \exp(x_j^T \cdot \hat{v})$



2.3 CBOW

- 损失函数

Maximize $\mathrm{P}(x_c|x_{c-m},\dots,x_{c+m})$

$\Updownarrow$

Minimize $\mathrm{J} = -\log \mathrm{P}(x_c|x_{c-m},\dots,x_{c+m})$

$= -\log \mathrm{P}(x_c|\hat{v})$

$= -\log \frac{\exp(x_c^T\cdot\hat{v})}{\sum_{j=1}^{|V|}\exp(x_j^T\cdot\hat{v})}$

$= -\ x_c^T\cdot\hat{v} + \log\sum_{j=1}^{|V|}\exp(x_j^T\cdot\hat{v})$

$\Updownarrow$

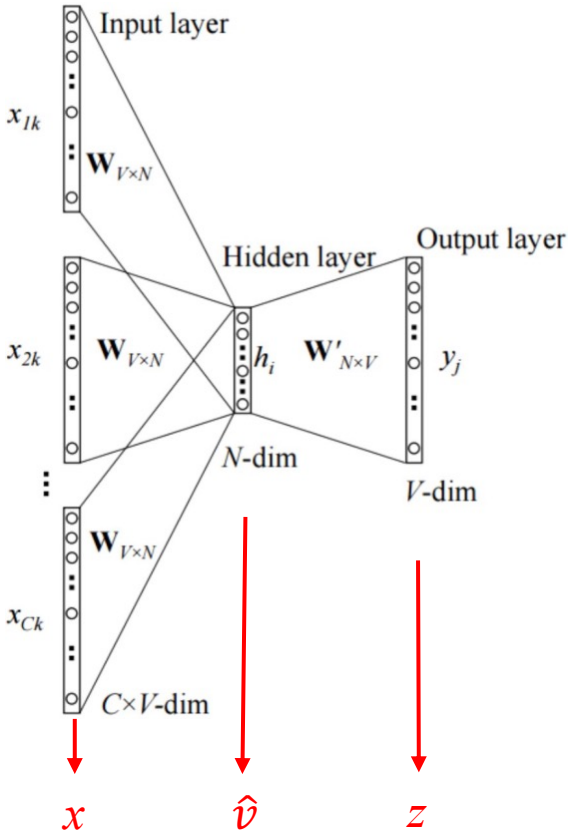
Minimize $H(\hat{y},y) = -\sum_{j=1}^{|V|} y_j \log\hat{y}_j$

$= -y_c\log\hat{y}_c$

$= -\log\hat{y}_c$

$= -\log\frac{\exp(x_c^T\cdot\hat{v})}{\sum_{j=1}^{|V|}\exp(x_j^T\cdot\hat{v})}$

$= -x_c^T\cdot\hat{v} + \log\sum_{j=1}^{|V|}\exp(x_j^T\cdot\hat{v})$



1 PART

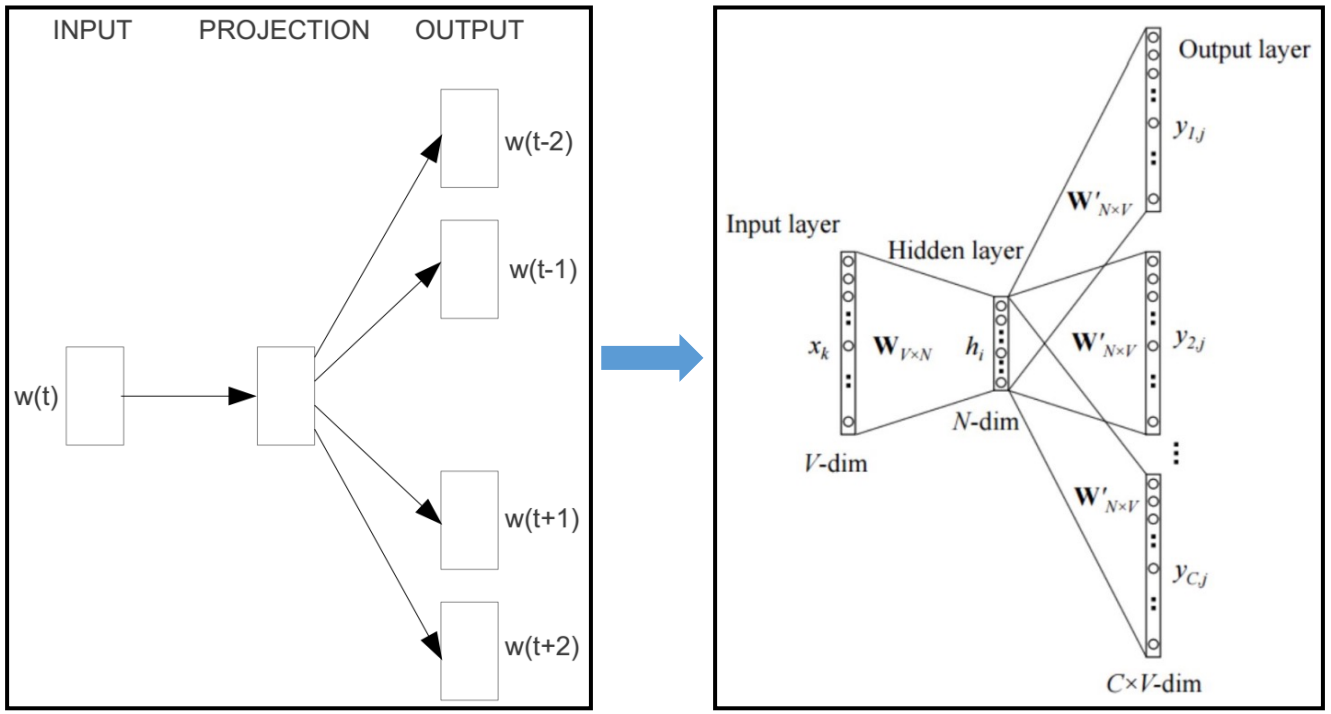
2.3 Skip-gram

- Skip-gram：根据中心词预测上下文

1 PART

2.3 Skip-gram

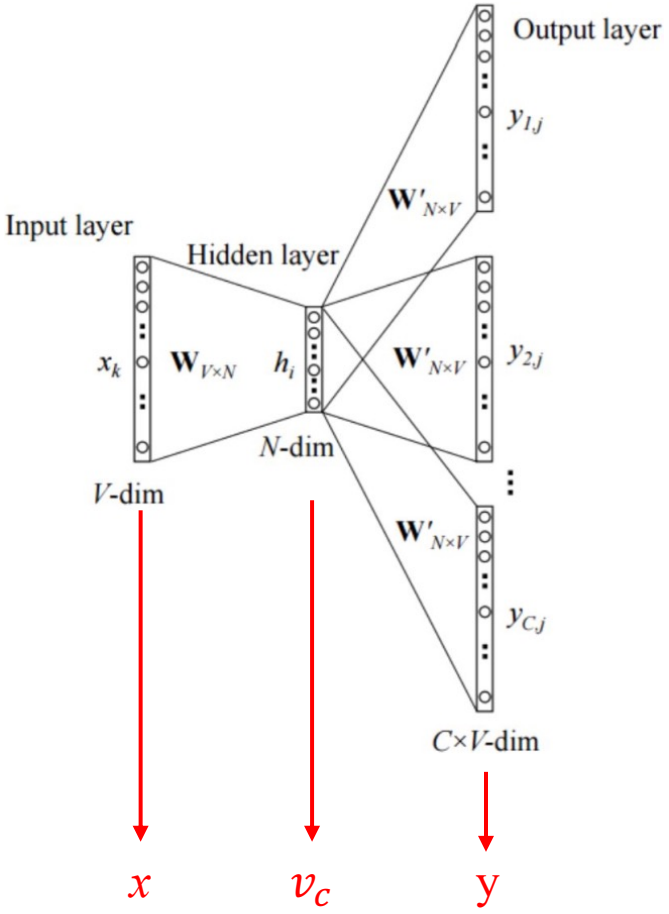
- Skip-gram：根据中心词预测上下文
- 架构



2.3 Skip-gram

- 流程

1. Generate one-hot input vector $x \in R^{|V|}$ of the center word.



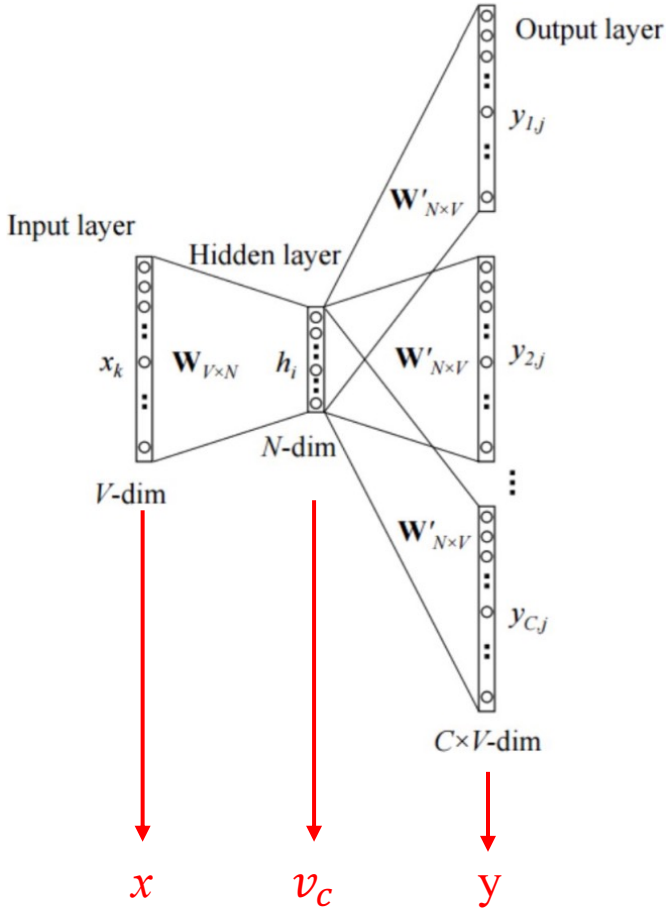
2.3 Skip-gram

- 流程

1. Generate one-hot input vector $x \in R^{|V|}$ of the center word.

2. Get embedded word vectors for the center word.

   $v_c = x \cdot W \in R^N$



2.3 Skip-gram

- 流程

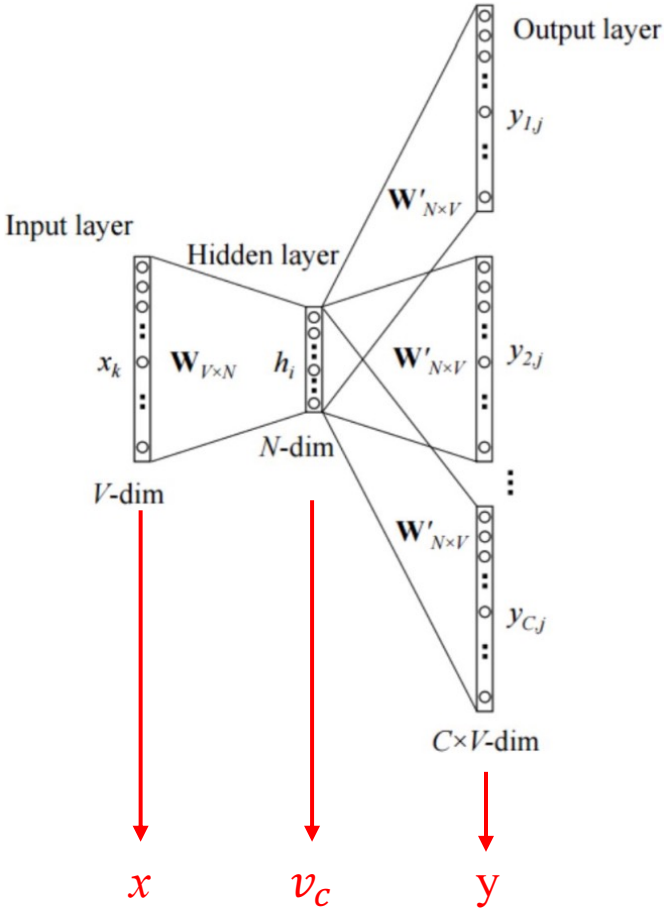
1. Generate one-hot input vector $x \in R^{|V|}$ of the center word.

2. Get embedded word vectors for the center word.

   $v_c = x \cdot W \in R^N$

3. Generate a score vector z.

   $z = v_c \cdot W'$



2.3 Skip-gram

- 流程

1. Generate one-hot input vector $x \in R^{|V|}$ of the center word.

2. Get embedded word vectors for the center word.

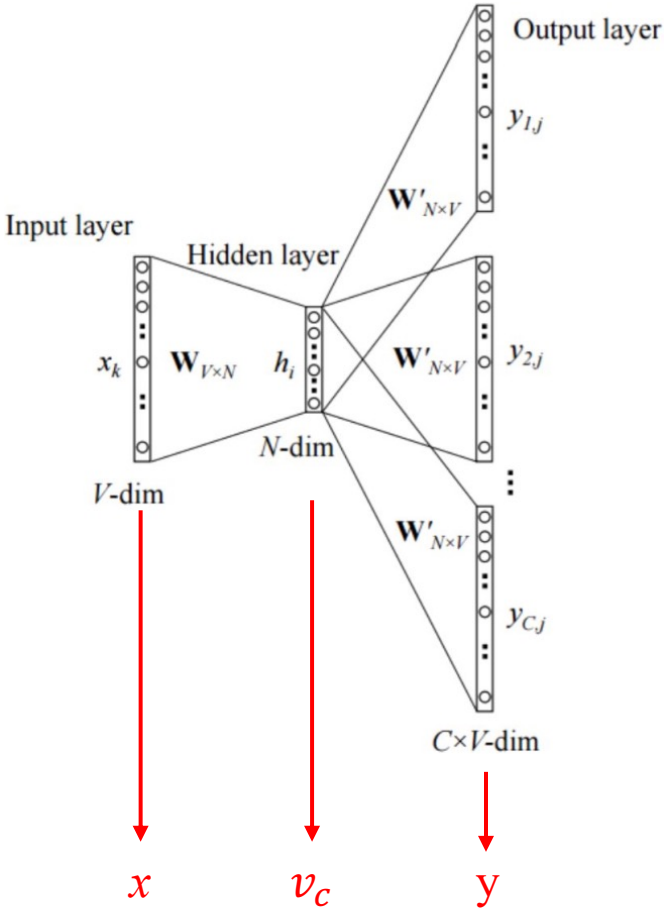
   $v_c = x \cdot W \in R^N$

3. Generate a score vector z.

   $z = v_c \cdot W'$

4. Turn the score vector into probabilities $\hat{y}$.

   $\hat{y} = softmax(z)$



2.3 Skip-gram

- 流程

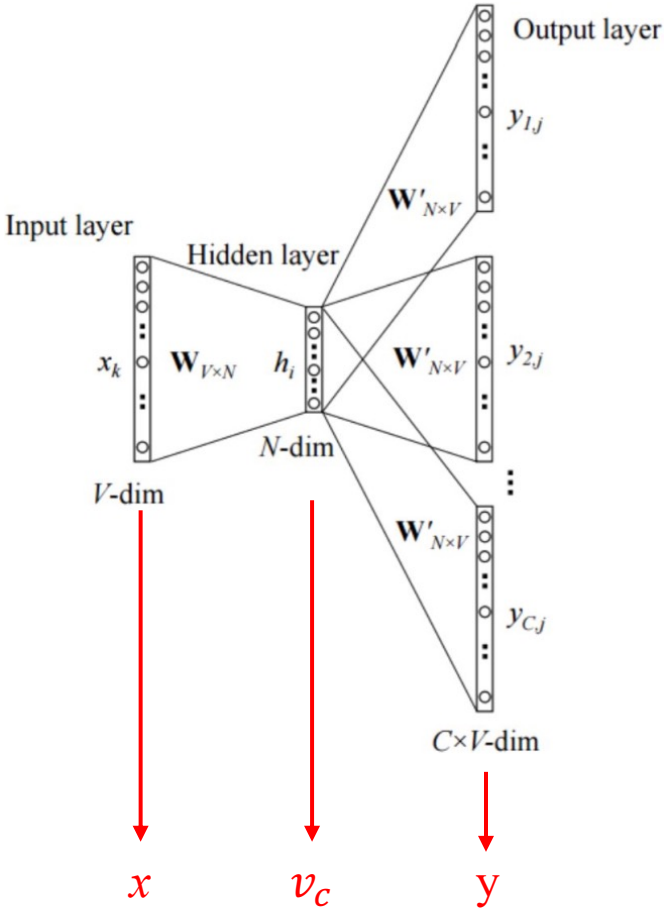
1. Generate one-hot input vector $x \in R^{|V|}$ of the center word.

2. Get embedded word vectors for the center word.
   $v_c = x \cdot W \in R^N$

3. Generate a score vector z.
   $z = v_c \cdot W'$

4. Turn the score vector into probabilities $\hat{y}$.
   $\hat{y} = softmax(z)$

5. Note that $\hat{y}_{c-m}, \ldots, \hat{y}_{c-1}, \hat{y}_{c+1}, \ldots, \hat{y}_{c+m}$ are the probabilities of observing context word.

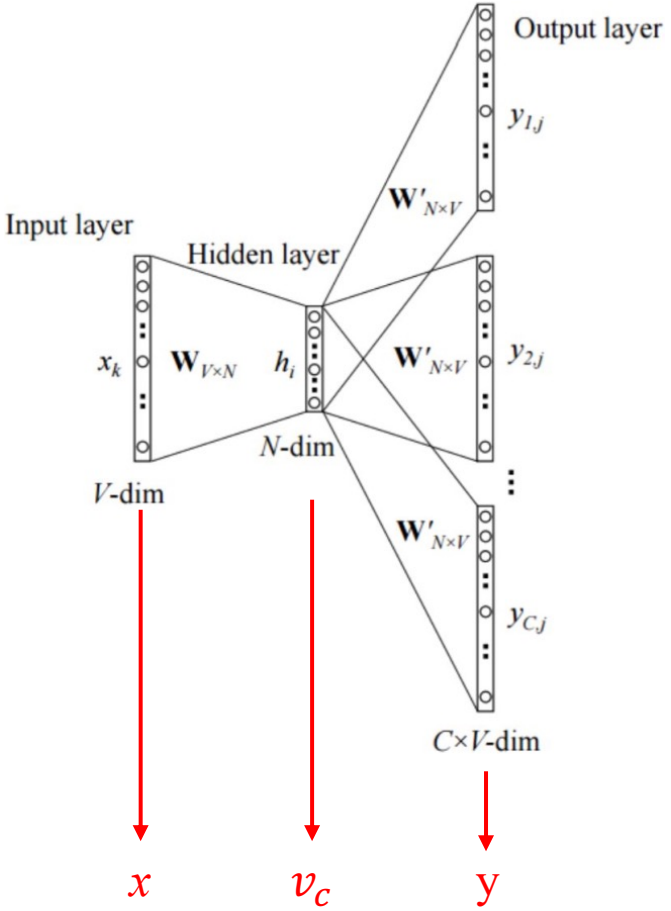


1 PART

2.3 Skip-gram

- 损失函数

  Maximize $\mathrm{P}(x_{c-m}, \ldots, x_{c-1}, x_{c+1}, \ldots, x_{c+m} \mid x_c)$



2.3 Skip-gram

- 损失函数

Maximize $\mathrm{P}(x_{c-m}, \ldots, x_{c-1}, x_{c+1}, \ldots, x_{c+m} \mid x_c)$

$\Updownarrow$

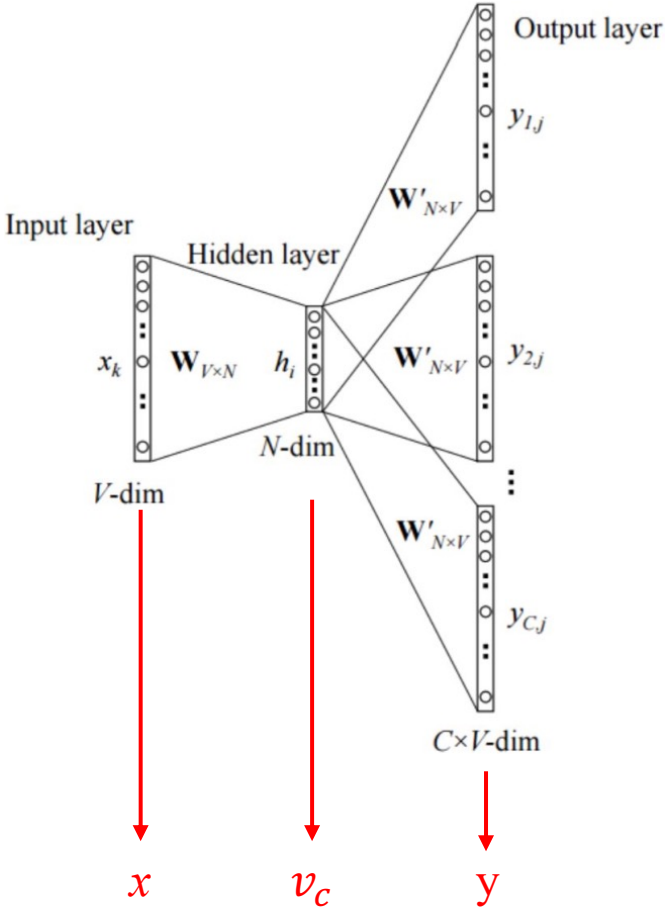
Minimize $\mathrm{J} = -\log \mathrm{P}(x_{c-m}, \ldots, x_{c+m} \mid x_c)$

$$= -\log \prod_{j=0, j\neq m}^{2m} p(x_{c-m+j} \mid x_c)$$

$$= -\log \prod_{j=0, j\neq m}^{2m} p(v_{c-m+j} \mid v_c)$$

$$= -\log \prod_{j=0, j\neq m}^{2m} \frac{\exp(v_{c-m+j}^T v_c)}{\sum_{k=1}^{|V|} exp(v_k^T v_c)}$$

$$= -\sum_{j=0, j\neq m}^{2m} v_{c-m+j}^T v_c + 2m \log \sum_{k=1}^{|V|} \exp(v_k^T v_c)$$



2.3 Skip-gram

- 损失函数

Maximize $\mathrm{P}(x_{c-m}, \ldots, x_{c-1}, x_{c+1}, \ldots, x_{c+m} \mid x_c)$

$\Updownarrow$

Minimize $\mathrm{J} = -\log \mathrm{P}(x_{c-m}, \ldots, x_{c+m} \mid x_c)$

$= -\log \prod_{j=0, j\neq m}^{2m} p(x_{c-m+j} \mid x_c)$

$= -\log \prod_{j=0, j\neq m}^{2m} p(v_{c-m+j} \mid v_c)$

$= -\log \prod_{j=0, j\neq m}^{2m} \frac{\exp(v_{c-m+j}^T v_c)}{\sum_{k=1}^{|V|} exp(v_k^T v_c)}$

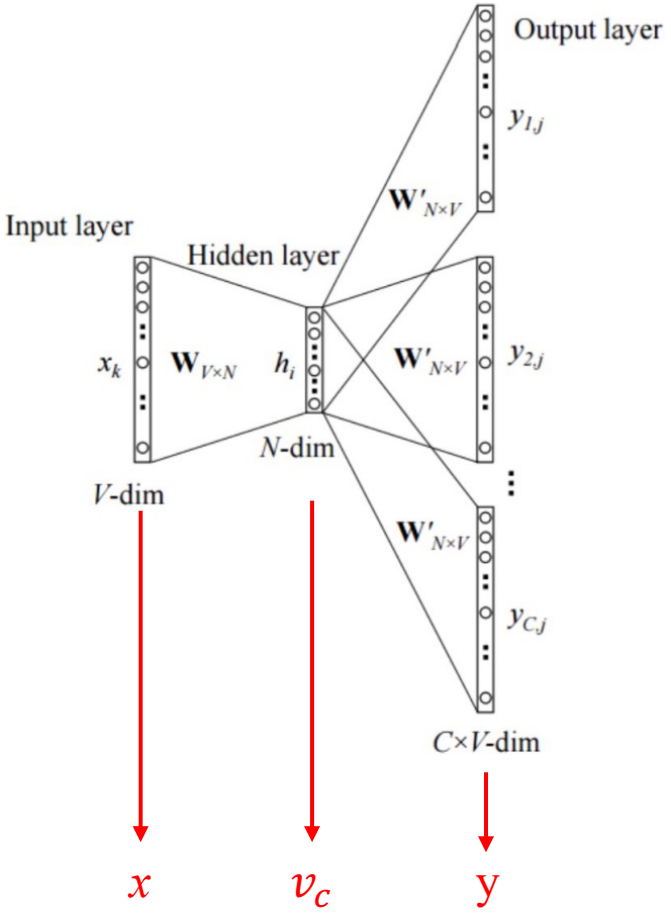
$= -\sum_{j=0, j\neq m}^{2m} v_{c-m+j}^T v_c + 2m \log \sum_{k=1}^{|V|} \exp(v_k^T v_c)$

$\Updownarrow$

Minimize $H(\hat{y}, y) = \sum_{j=0, j\neq m}^{2m} H(\hat{y}, y_{c-m+j})$

$= \sum_{j=0, j\neq m}^{2m} \log P(v_{c-m+j} \mid v_c)$

$= \mathrm{J}$



1 PART

2.3 问题及解决方法

- 缺点
  - 计算量太大

$$J(CBOW) = -\log \frac{\exp(x_c^T \cdot \hat{v})}{\sum_{j=1}^{|V|} \exp(x_j^T \cdot \hat{v})} \qquad J(SG) = -\log \prod_{j=0, j\neq m}^{2m} \frac{\exp(v_{c-m+j}^T v_c)}{\sum_{k=1}^{|V|} exp(v_k^T v_c)}$$

对于每一个窗口的计算，需要计算整个词汇表，|V|通常百万级

1 PART

2.3 问题及解决方法

- 缺点
  - 计算量太大

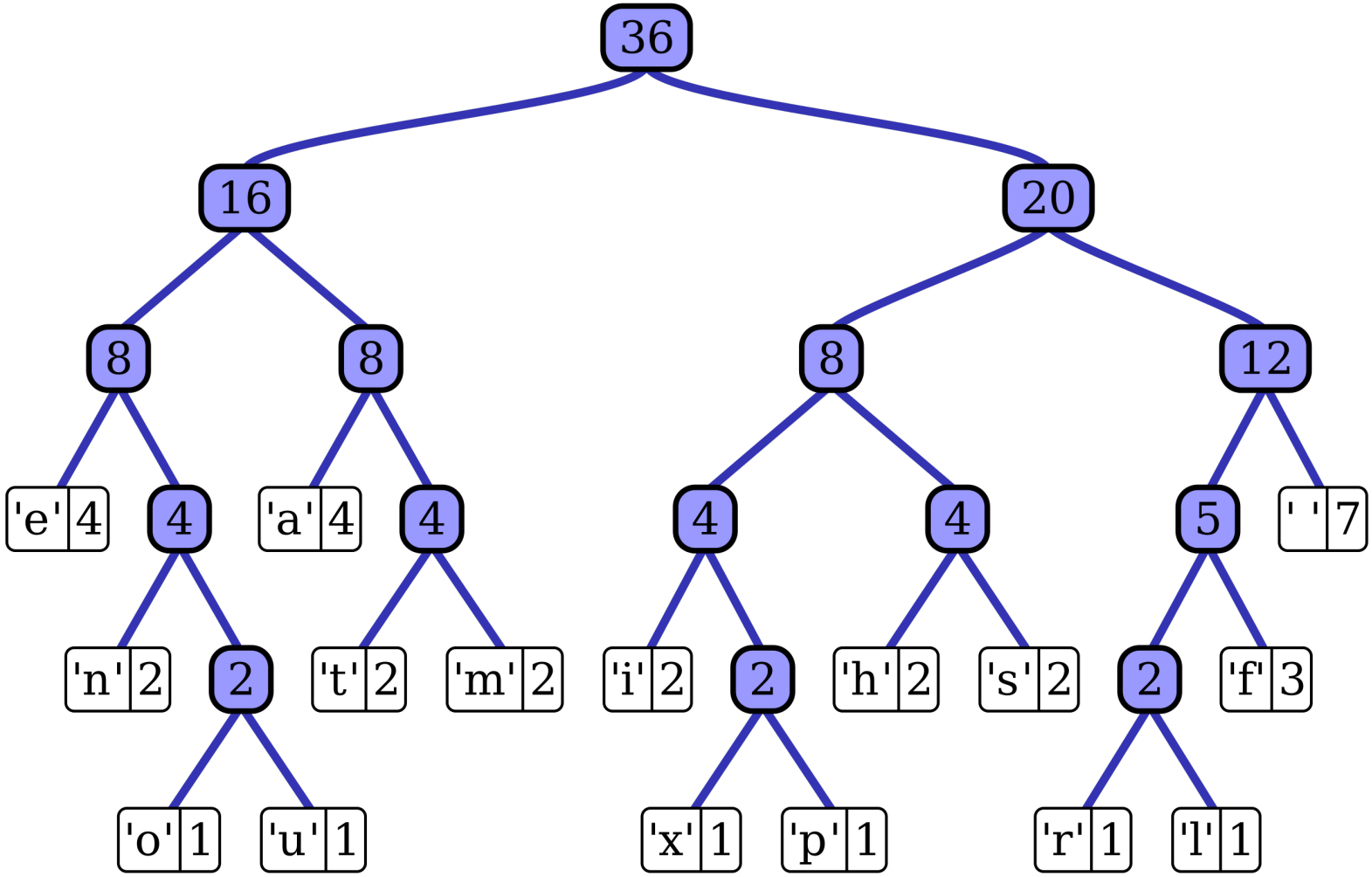
$$J(CBOW) = -\log \frac{\exp(x_c^T \cdot \hat{v})}{\sum_{j=1}^{|V|} \exp(x_j^T \cdot \hat{v})} \qquad J(SG) = -\log \prod_{j=0, j\neq m}^{2m} \frac{\exp(v_{c-m+j}^T v_c)}{\sum_{k=1}^{|V|} exp(v_k^T v_c)}$$

  对于每一个窗口的计算，需要计算整个词汇表，|V|通常百万级

- 解决方法
  - 层次Softmax
  - 负采样

1 PART

2.3 层次Softmax



1 PART

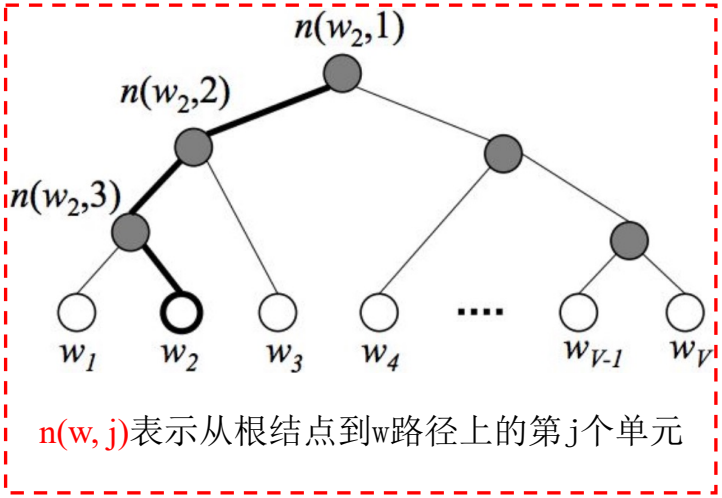
2.3 层次Softmax

- 层次Softmax：使用一个二叉树（例如Huffman Tree），其中叶子是单词。单词作为输出单词的概率被定义为从根到该单词的叶的随机游走的概率。

$$p(w = w_o) = \prod_{j=1}^{L(w)-1} \sigma([\![n(w, j+1) = ch(n(w, j))]\!] \cdot v_{n(w,j)}^T h)$$

$$[\![x]\!] = \begin{cases} 1 & \text{If } (j+1)^{th} \text{ node is left child of } j^{th} \text{ node} \\ -1 & \text{Else} \end{cases}$$

$v_{n(w,j)}^T h$： 隐层向量与内节点向量之间的向量积.

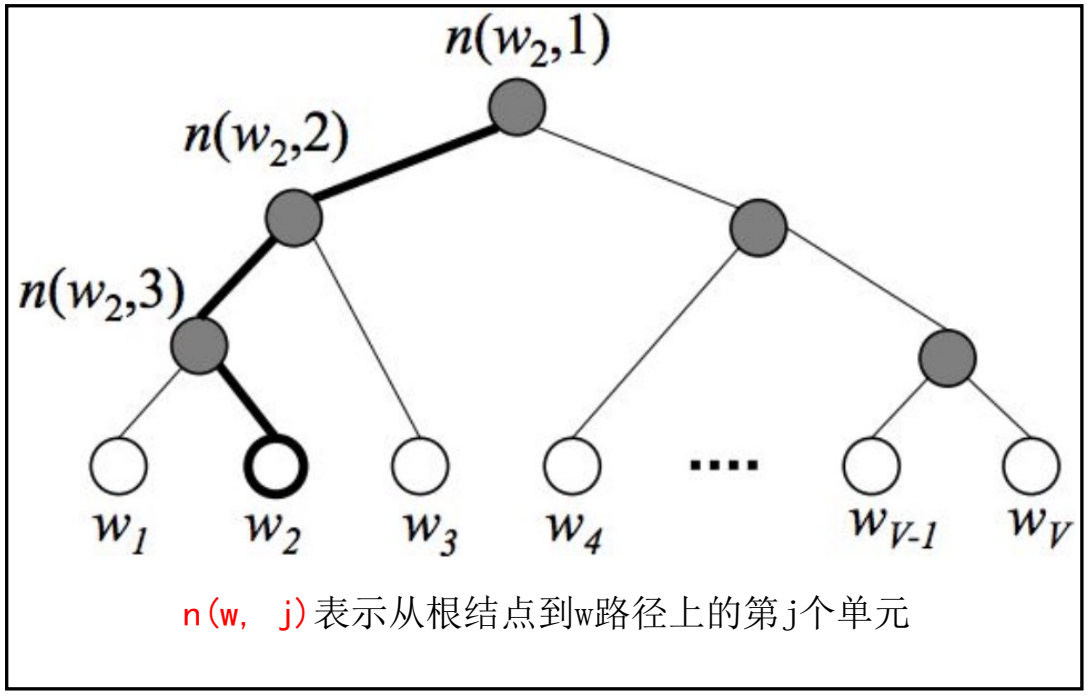


n(w, j)表示从根结点到w路径上的第j个单元

2.3 层次Softmax

- 举例。

$$p(w = w_2) = p(n(w_2, 1), left) \cdot p(n(w_2, 2), left) \cdot p(n(w_2, 3), right)$$
$$= \sigma(v^T_{n(w_2,1)} h) \cdot \sigma(v^T_{n(w_2,2)} h) \cdot \sigma(-v^T_{n(w_2,3)} h)$$



n(w, j)表示从根结点到w路径上的第j个单元

1 PART

2.3 负采样

- 负采样：不遍历整个词汇表，只从一个与词汇频率顺序匹配的噪声概率分布$p_n(w)$中采样几个negative例子

1 PART

2.3 负采样

- 负采样：不遍历整个词汇表，只从一个与词汇频率顺序匹配的噪声概率分布$p_n(w)$中采样几个negative例子

- 动机：

1. 定义:

(w, c): 词和上下文

P(D=1|w, c): (w, c)在语料库中的赋予的概率

P(D=0|w, c): (w, c)不在语料库中的赋予的概率

$$p(D=1|w,c)=\frac{1}{1+\exp(-v_c^T v_w)}$$

2. Objective Function (J):

Maximize P(D=1|w, c): 如果 (w, c) 在语料库中.

Maximize P(D=0|w, c): 如果 (w, c) 不在语料库中.

2.3 负采样

- 损失函数

$$
\begin{aligned}
\theta &= \underset{\theta}{\operatorname{argmax}} \prod_{(w,c)\in D} P(D=1|w,c,\theta) \prod_{(w,c)\in \tilde{D}} P(D=0|w,c,\theta) \\
&= \underset{\theta}{\operatorname{argmax}} \prod_{(w,c)\in D} P(D=1|w,c,\theta) \prod_{(w,c)\in \tilde{D}} (1-P(D=1|w,c,\theta)) \\
&= \underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\in D} \log P(D=1|w,c,\theta) + \sum_{(w,c)\in \tilde{D}} \log(1-P(D=1|w,c,\theta)) \\
&= \underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\in D} \log \frac{1}{1+\exp(-u_w^T v_c)} + \sum_{(w,c)\in \tilde{D}} \log(1-\frac{1}{1+\exp(-u_w^T v_c)}) \\
&= \underset{\theta}{\operatorname{argmax}} \sum_{(w,c)\in D} \log \frac{1}{1+\exp(-u_w^T v_c)} + \sum_{(w,c)\in \tilde{D}} \log(\frac{1}{1+\exp(u_w^T v_c)})
\end{aligned}
$$

$$
J = -\sum_{(w,c)\in D} \log \frac{1}{1+\exp(-u_w^T v_c)} - \sum_{(w,c)\in \tilde{D}} \log(\frac{1}{1+\exp(u_w^T v_c)})
$$

1 PART

2.3 负采样

- 损失函数

1. CBOW.

$$J = -\log \sigma(v_c^T \cdot \hat{v}) - \sum_{k=1}^{K} \log \sigma(-\tilde{v}_k^T \cdot \hat{v})$$

2.3 负采样

- 损失函数

1. CBOW.

$$J = -\log \sigma(v_c^T \cdot \hat{v}) - \sum_{k=1}^{K} \log \sigma(-\tilde{v}_k^T \cdot \hat{v})$$

2. Skip-Gram.

$$J = -\log \sigma\left(v_{c-m+j}^T \cdot v_c\right) - \sum_{k=1}^{K} \log \sigma(-\tilde{v}_k^T \cdot v_c)$$

1 PART

2.3 负采样

- 损失函数

1. CBOW.

$$J = -\log \sigma(v_c^T \cdot \hat{v}) - \sum_{k=1}^{K} \log \sigma(-\tilde{v}_k^T \cdot \hat{v})$$

2. Skip-Gram.

$$J = -\log \sigma\left(v_{c-m+j}^T \cdot v_c\right) - \sum_{k=1}^{K} \log \sigma(-\tilde{v}_k^T \cdot v_c)$$

3. Illustration

a) $\{\tilde{v}_k | k = 1 \ldots K\}$ are sampled from $P_n(w)$.

b) $P_n(w)$ = Unigram distribution raised to the power 3/4.

c) Usually K = 20~30 works well.

1 PART

2.3 为什么不直接计算同现关系?

- 同现关系 co-occurrence

  1. 同现关系的两种计算单元：窗口 vs 文档

1 PART

2.3 为什么不直接计算同现关系?

- 同现关系 co-occurrence

1. 同现关系的两种计算单元：窗口 vs 文档

2. 窗口：类似于word2vec，利用窗口捕捉句法和语法信息

1 PART

2.3 为什么不直接计算同现关系?

- 同现关系 co-occurrence

1. 同现关系的两种计算单元：窗口 vs 文档

2. 窗口：类似于word2vec，利用窗口捕捉句法和语法信息

3. 文档：类似于主题提取，例如LDA

1 PART

2.3 基于同现关系向量计算的问题

1. 随着词汇表增大而增大

1 PART

2.3 基于同现关系向量计算的问题

1. 随着词汇表增大而增大

2. 非常高维：需要很大存储

1 PART

2.3 基于同现关系向量计算的问题

1. 随着词汇表增大而增大

2. 非常高维：需要很大存储

3. 后续的分类问题会存在稀疏问题

1 PART

2.3 基于同现关系向量计算的问题

1. 随着词汇表增大而增大

2. 非常高维：需要很大存储

3. 后续的分类问题会存在稀疏问题

→ 模型不稳定

1 PART

2.3 如何校验词向量

内部校验（Intrinsic） vs 外部校验（extrinsic）

1 PART

2.3 如何校验词向量

内部校验（Intrinsic） vs 外部校验（extrinsic）

内部校验：

- 在中间子任务上进行校验
- 计算便捷
- 帮助理解该系统
- 不能确保有用性，除非和子任务的关联性能够得到证实

1 PART

2.3 如何校验词向量

内部校验（Intrinsic） vs 外部校验（extrinsic）

内部校验：
- 在中间子任务上进行校验
- 计算便捷
- 帮助理解该系统
- 不能确保有用性，除非和子任务的关联性能够得到证实

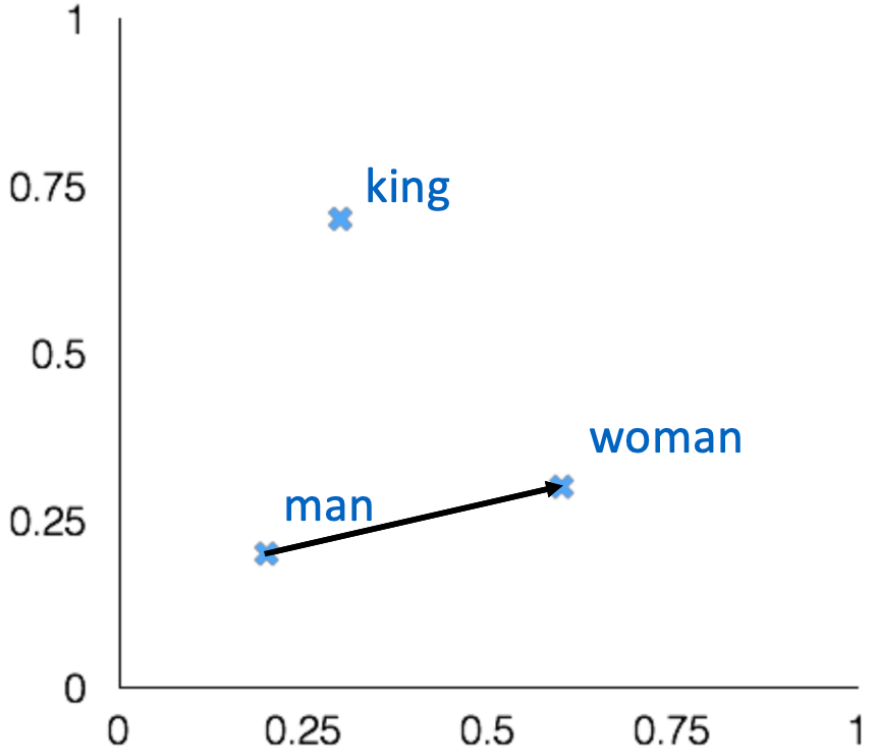
外部校验：
- 在真实任务上进行校验
- 校验计算量大
- 当有问题时，难以确定是该系统问题还是该系统与其他系统进行交换时的问题
- 如果将原系统用新系统替代后，性能提升，则证明有效性（消除实验）

2.3 内部校验

词向量相似性

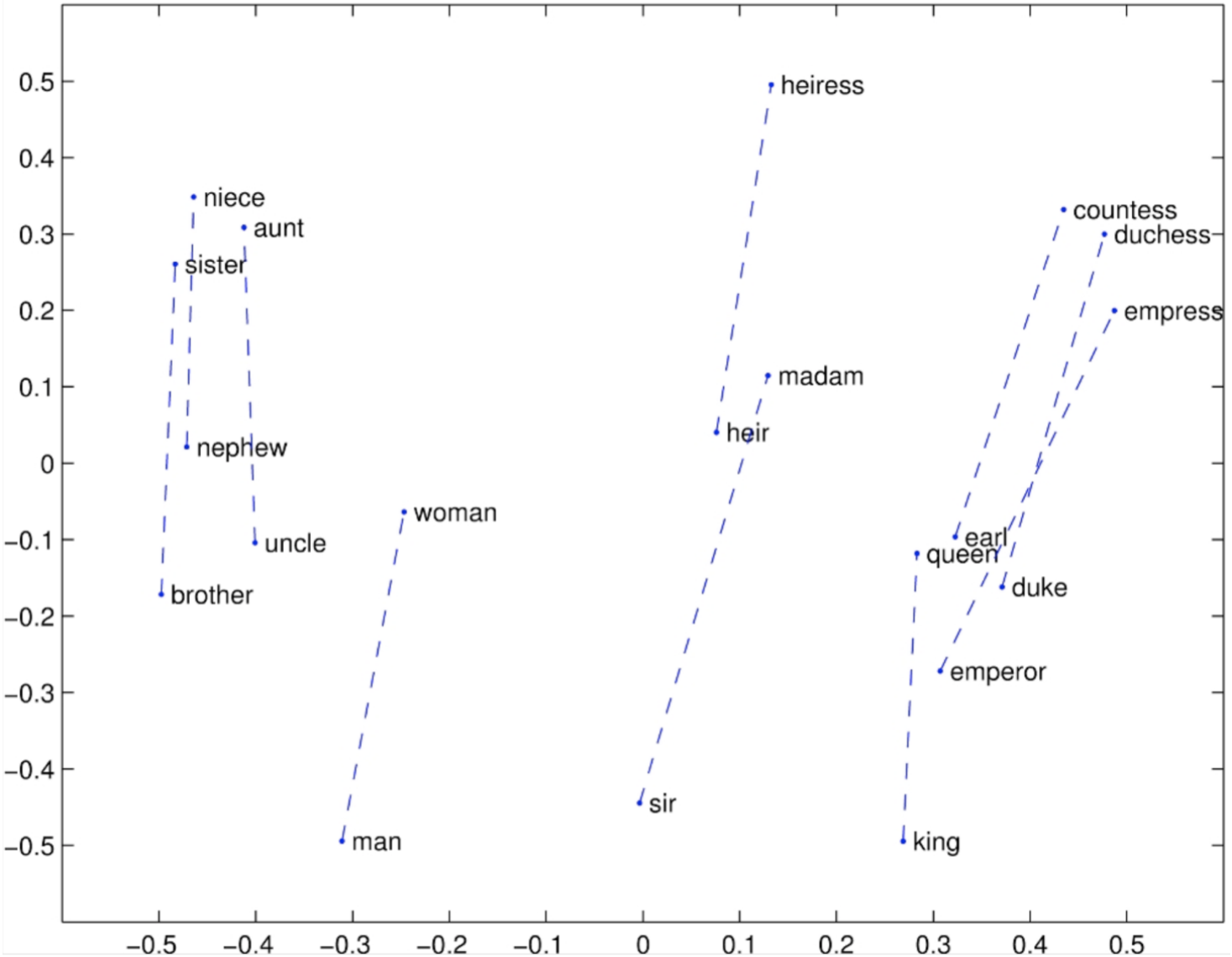
king – man + woman = ?

利用向量代数计算后的余弦相似度



1 PART

2.3 内部校验



谢谢