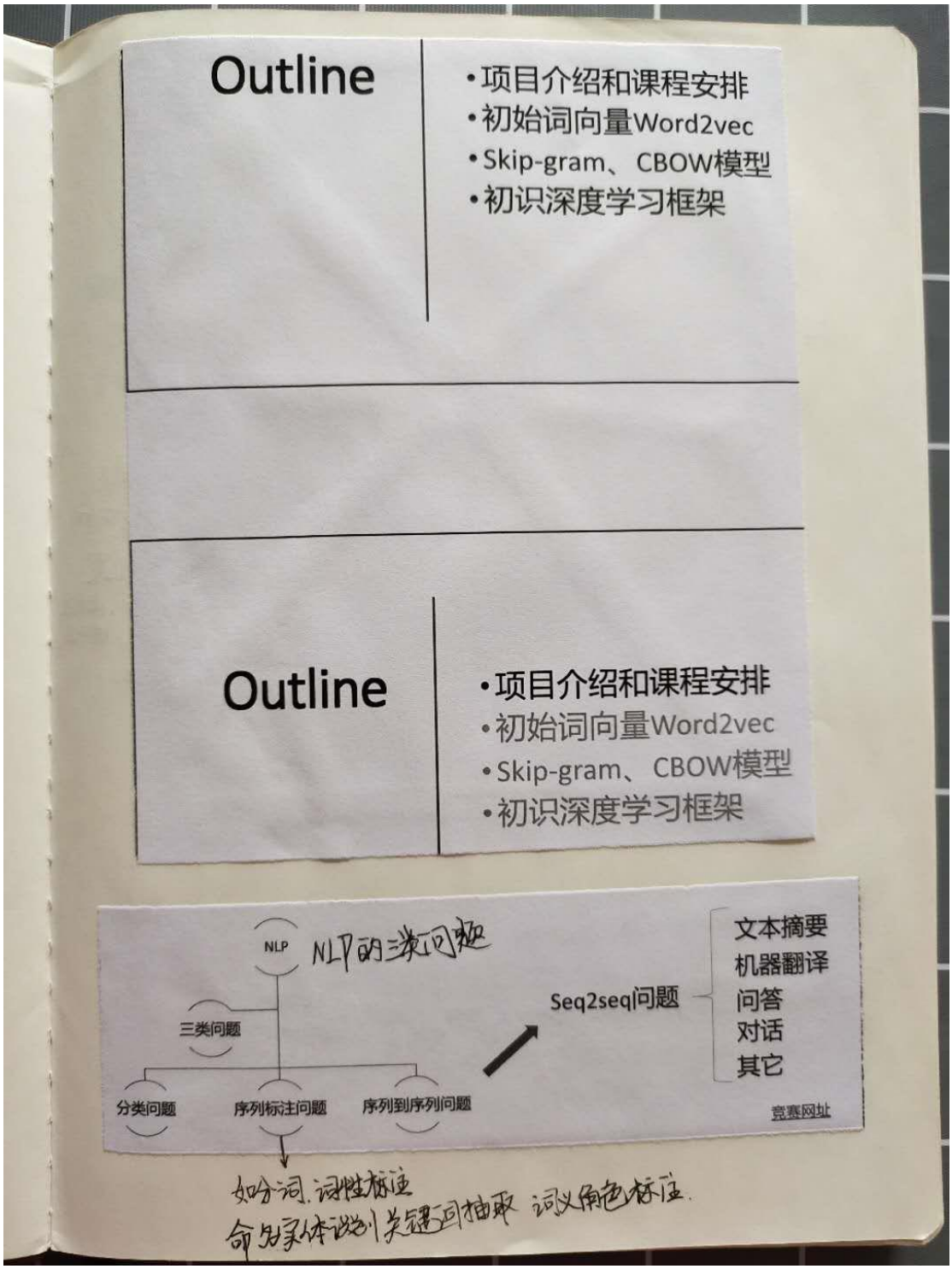
问答摘要与推理-项目简介



Outline

- 项目介绍和课程安排
- **初始词向量Word2vec**
- Skip-gram、CBOW模型
- 初识深度学习框架

语言模型:

今天|欢迎|同学们|来到|北京大学|学习

某个位置(s)上的词的输出概率: $P(s) = P(w_1, w_2, \dots w_n)$

$\Leftarrow \quad = P(w_1)P(w_2|w_1)P(w_3|w_1,w_2)\cdots P(w_n|w_1,w_2 \dots w_{n-1})$

$\Downarrow$

目标函数: $L = \sum_{w \in C} \log P(w|context(w))$

注:

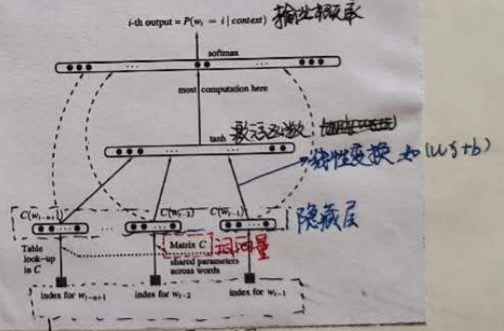
① 其中C为词表

② w_i为词表中(某位置)的词

③ 为什么用log表示: 连乘法为加法, 方便运算; 便于求和; 避免溢出.

Neural Network Language Model

神经网络语言模型(NNLM)



词表中对应位置的词

语言的表达形式

1. discrete symbols (离散模型)

缺点：不能准确表征语义的相似性。

如不能很好的区别：我爱北京 和 我喜欢北京。

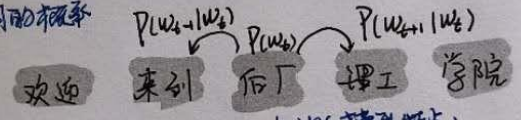
2. by their context

我爱北京：通过我和北京来计算爱（的向量）

我喜欢北京：通过我和北京来计算喜欢（的向量）

Word2vec → word embedding (词嵌入)

通过中心词来预测输出词的概率



某个词的词向量 →

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

Word2vec 模型特点：

① 有一个很大的词表库

② 在词表中的每一个词都可以通过向量表征

③ 有一个中心词c，有一个输出词o

④ 用词c和o的相似度来计算他们之间同时出现的概率

⑤ 调整这个词向量来获得最大的输出概率

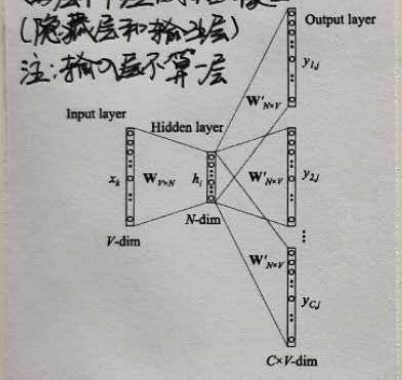
Outline

- 项目介绍和课程安排
- 初始词向量Word2vec
- Skip-gram、CBOW模型
- 初识深度学习框架

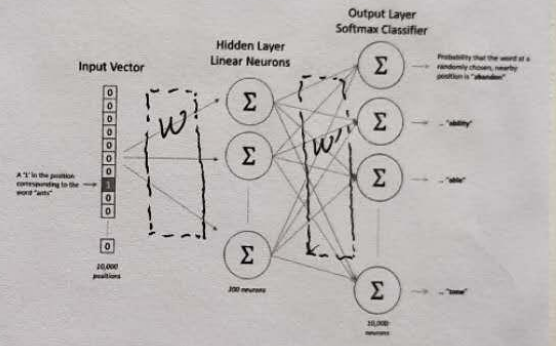
无监督学习

Skip-gram模型

两层神经网络模型
(隐藏层和输出层)
注:输入层不算一层



Skip-gram模型



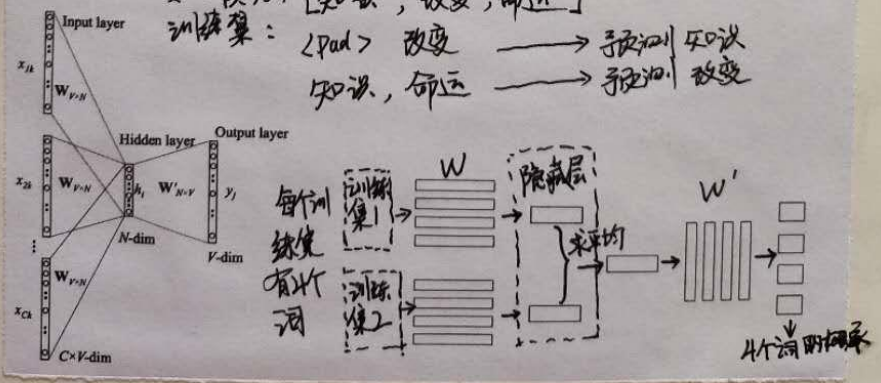
CBOW模型

输入是两边的词,输出是中间的词

如词典为: [知识, 改变, 命运]

训练集: <Pad> 改变 → 预测 知识

知识, 命运 → 预测 改变



附：

（1）《神经网络与深度学习》，期附件中数学基础知识讲解。Github 链接如下：
https://github.com/nndl/nndl.github.io

（2）优秀学员 github：
https://github.com/Light2077/

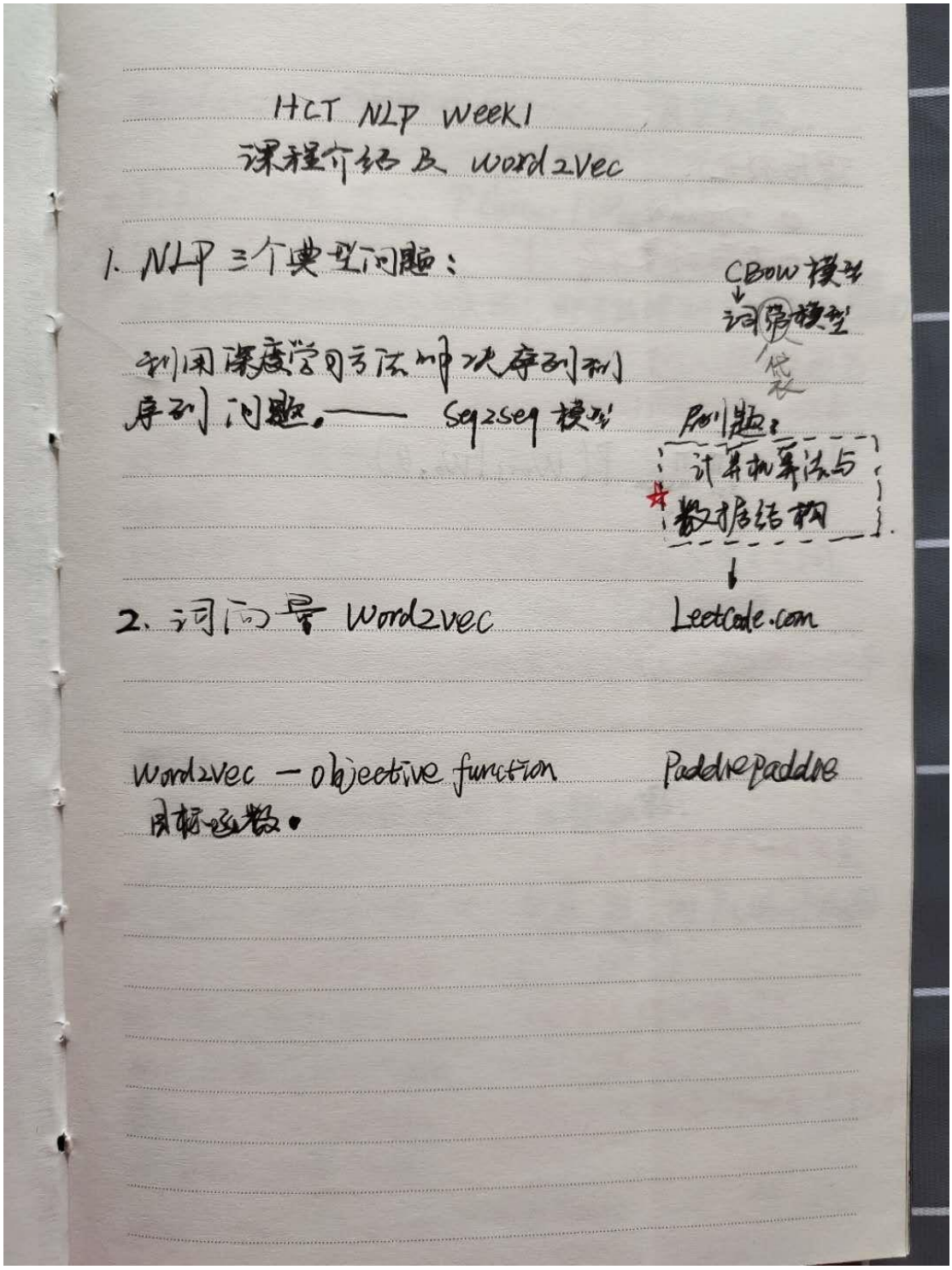
（3）推荐一个查阅资料的网站：https://medium.com/　　和 cs224n

（4）华为云 AI 平台：modelarts

（5）百度 AI：黄埔学院

（6）需要补充学些的知识点：交叉熵、反向求导

之前的笔记


HCT NLP Week1
课程介绍及 word2vec
1. NLP 三个典型问题：
利用深度学习方法解决序列和
序列问题。—— Seq2seq 模型
CBOW 模型
词向量模型
刷题：
计算机算法与
数据结构
LeetCode.com
2. 词向量 Word2vec
Word2vec - objective function
目标函数。
PaddlePaddle

课后补充：

① softmax

② 反向梯度求导

$$L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \ne 0}} P(w_{t+j} \mid w_t, \theta)$$

word2vec - objective function 目标函数.

① $$L(\theta) = \prod_{t=1}^{T} \prod_{\substack{-m \le j \le m \\ j \ne 0}} P(w_{t+j} \mid w_t, \theta)$$

$L(\theta)$：概率

t：t位置的输入词

θ：建立神经网络的所有参数，即可能得到的所有词向量

$\prod_{t=1}^{T}$：所有位置的词，即对每个位置都要进行计算

窗口大小：$[-m, m]$

$P(w_{t+j} \mid w_t, \theta)$：通过$t$位置的输入，预测另外一个位置（$t+j$位置）词出现的概率

对于$[-m, m]$之间，对于给定位置t的词，其他位置的词出现的概率。

即：对于给定位置t的词，要计算所有位置上的词出现的概率

优化以上目标函数得如下目标函数：

目标函数

② $$J(\theta) = -\frac{1}{T} \log L(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \sum_{\substack{-m \le j \le m \\ j \ne 0}} \log P(w_{t+j} \mid w_t, \theta)$$

$\frac{1}{T}$：T个样本的均值

θ：即为所需求的词向量

T表示取了T个训练样本

$\sum_{\substack{-m \le j \le m \\ j \ne 0}}$：计算$t$位置左边m个词、右边m个词出现的概率（不计算$t$位置本身）

词向量问题即变为优化函数$J(\theta)$的问题。最优化时对应的θ即为所需求的词向量

此时的词向量θ被称之为Fake Task，即

总结：

① 词向量问题，并不是为了求得最优解（P_i），而是为了通过求解最优解，得到求解最优解的过程中计算得到的权重矩阵 W_1 和 W_2。

② 权重（矩阵）W_1、W_2 与词向量的关系

W_1

$$\begin{bmatrix} [\text{词向量}1] \\ [\text{词向量}2] \\ \vdots \\ [\text{词向量}n] \end{bmatrix}$$

词向量为权重矩阵 W_1（W_2）中的一个向量

W_2

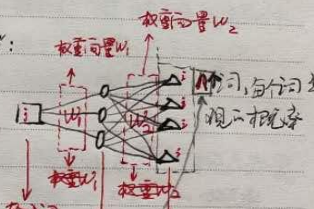
$$\begin{bmatrix} [\text{词向量}1] \\ [\text{词向量}2] \\ \vdots \\ [\text{词向量}n] \end{bmatrix}$$

此求最优解问题最终并不是想要得到最优解 P_i，而是想要得到的权重值（权重值代表每个词在空间中的意义）W_1, W_2。

简单神经网络模型：

j位置的词做为输入，

i位置的词做为输出



假设有n个词

$$\text{softmax} = \frac{e^{x}}{\sum_{i=1}^{n} e^{x_i}}$$

则：

$$P(i|j) = \frac{e^{(U_i^T V_j)}}{\sum_{r \in n} e^{(U_r^T V_j)}}$$

$P(i|j)$：j位置输入词，i位置的词出现的概率

其中 U 即权重向量 W_1，U_i 即权重向量 W_1 中i位置的词的权重，V_j 即权重向量 W_2

$\sum_{r \in n} e^{(U_r^T V_j)}$：表示对于任意位置j做为输入词，所有词都要做为输出计算一遍

3. Skip-gram. CBOW模型.

Skip-gram模型：（即已知中间词，预测其前后周围词）

CBOW模型即已知周围词的词，预测中间词.

例：我 爱 北京 ~~天安门~~ 天安门.

对于Skip-gram模型为输入北京，预测输入为爱和天安门的概率

对于CBOW模型，即为输入为爱和天安门.
输出为北京的概率

Skip-gram模型：

例：the dog barked at the mailman

以dog作为输入.

设 Skip_windows = 2，即滑动窗口大小为2，即在dog的左右分别取2个词.

得：训练样本 ["the" "dog" "barked" "at"]

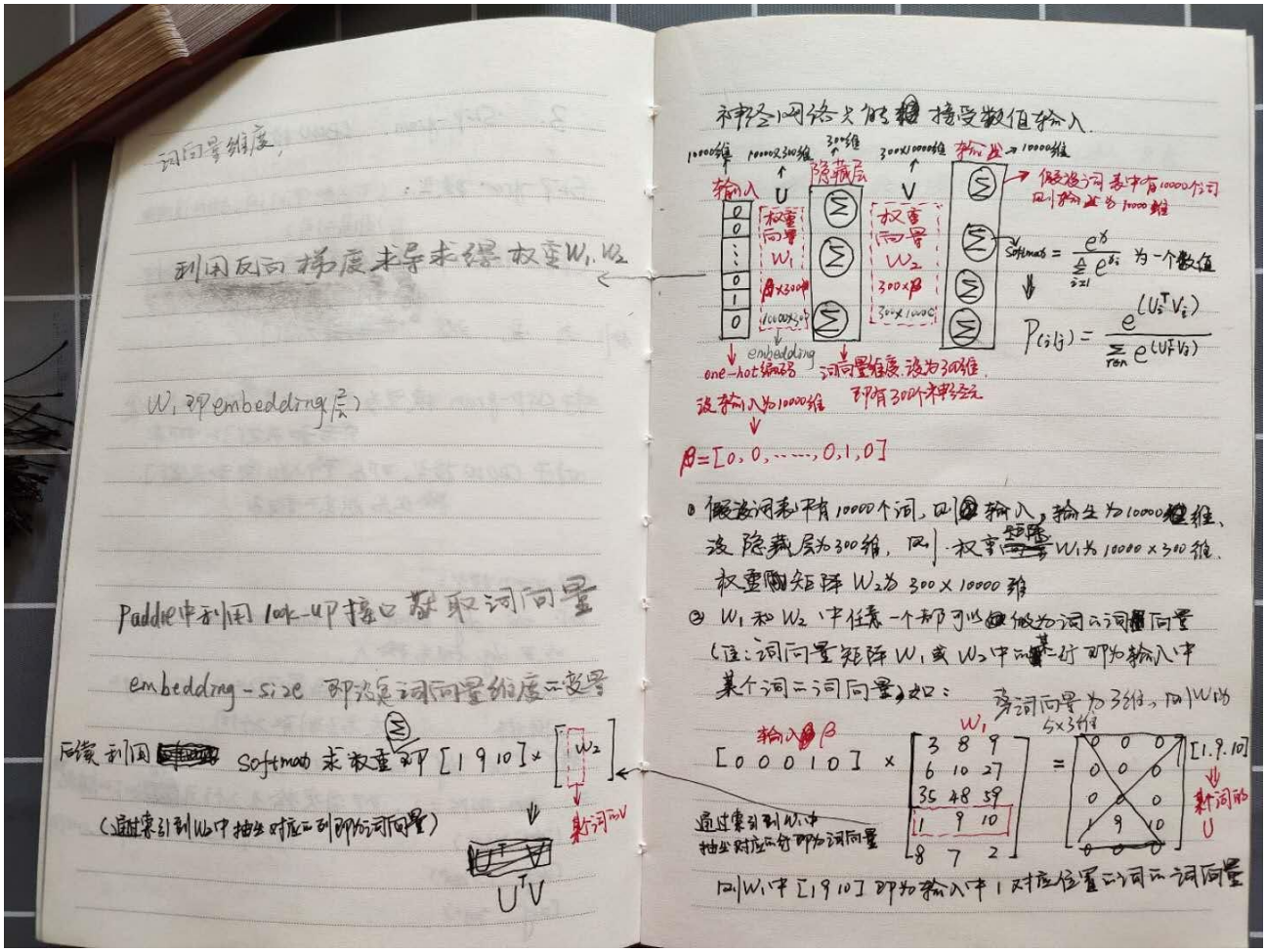
设：num_skips = 2，即每次输出2个词，得到如下训练数据

("dog" "the")

("dog" "barked")

("dog" "at")

注：输出格式(input, output)


词向量维度
利用反向梯度求导求得权重W1,W2
W1即embedding层
Paddle中利用lookup接口获取词向量
embedding_size 即决定词向量维度的变量
后续利用 Softmax 求权重即 [1 9 10] x
(通过索引到W2中抽出对应行到即为词向量)
新词向量
UTV
神经网络只能接受数值输入
10000维
10000×300维
300维
隐藏层
300×10000维
输入
权重向量 W1
权重向量 W2
embedding
one-hot编码器
词向量维度,设为300维
即有300个神经元
设输入为10000维
β=[0,0,……,0,1,0]
Softmax = e^xi / Σ e^xi 为一个数值
P(i|j) = e^(U_i^T V_j) / Σ e^(U^T V_j)
①假设词表有10000个词,则输入,输出为10000维,设隐藏层为300维,则权重矩阵W1为10000×300维,权重矩阵W2为300×10000维
②W1和W2中任意一个都可以被做为词向量(注:词向量矩阵W1或W2中的某一行即为输入中某个词的词向量),如:
输入β
W1
5×3维
通过索引到W1中抽出对应行即为词向量
新词向量 U
则W1中[1 9 10]即为输入中1对应位置的词向量

CBOW 模型

CBOW 模型与 Skip-gram 模型计算就相反，只是 CBOW 模型是给出周围词，计算中间词出现~~概率~~概率.

Skip-gram与CBOW区别.

1. 计算复杂度不同:

Cbow的计算复杂度与字典大小(V)有关, Cbow的计算复杂度为O(V)
(即字典中每个词都要计算一遍)

Sg的计算复杂度与滑窗大小K有关, 复杂度为O(KV) (Sg与Cbow复杂度大)

2. 周围词的影响不同:

Cbow的调整是根据周围词一起调整的 (预测出词的概率为周围词计算结果的平均值)

Sg是由中心词预测输出词, 受周围词的影响较小

CBOW优点: 训练集小的时候更准确 (周围词的平均)