



温州大學  
WENZHOU UNIVERSITY

# 深度学习-自然语言处理和词嵌入

黄海广 副教授

2022年05月

# 本章目录

2

**01 词汇表征和文本数据处理**

**02 词嵌入**

**03 Word2Vec**

**04 GloVe**

**05 情感分类**

# 1.词汇表征

3

## 01 词汇表征和文本数据处理

02 词嵌入

03 **Word2Vec**

04 **GloVe**

05 情感分类

# 1. 词汇表征和文本数据处理

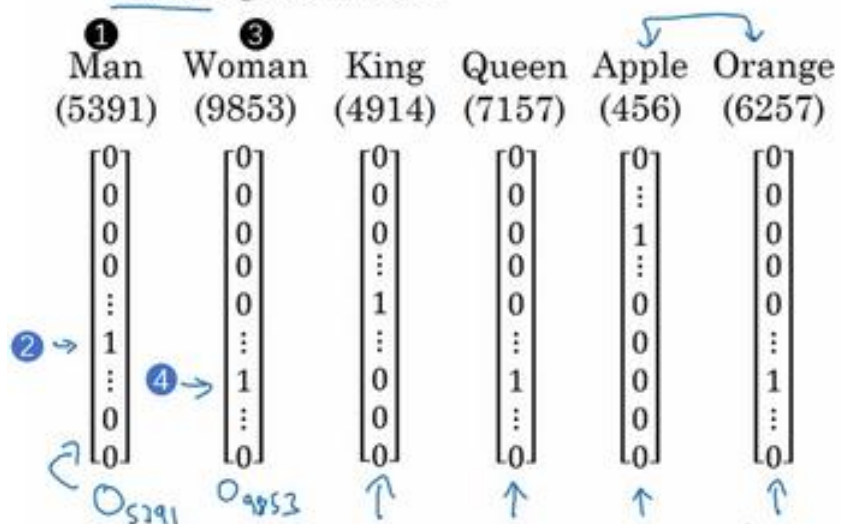
4

## Word representation

$V = [a, aaron, \dots, zulu, <UNK>]$

$|V| = 10,000$

1-hot representation



I want a glass of orange juice.

I want a glass of apple \_\_\_\_\_.

Andrew Ng

# 1. 词汇表征和文本数据处理

5

## Analogies

	Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
Gender	-1	1	-0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.70	0.69	0.03	-0.02
Food	0.09	0.01	0.02	0.01	0.95	0.97

①  $e_{5391}$   
 $e_{\text{man}}$

②  $e_{\text{woman}}$

$\text{Man} \rightarrow \text{Woman} \quad \Leftrightarrow \quad \text{King} \rightarrow ? \text{ Queen}$

$e_{\text{man}} - e_{\text{woman}} \approx e_{\text{king}} - e_{?}$

$e_{\text{man}} - e_{\text{woman}} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

$e_{\text{king}} - e_{\text{queen}} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

[Mikolov et. al., 2013, Linguistic regularities in continuous space word representations]

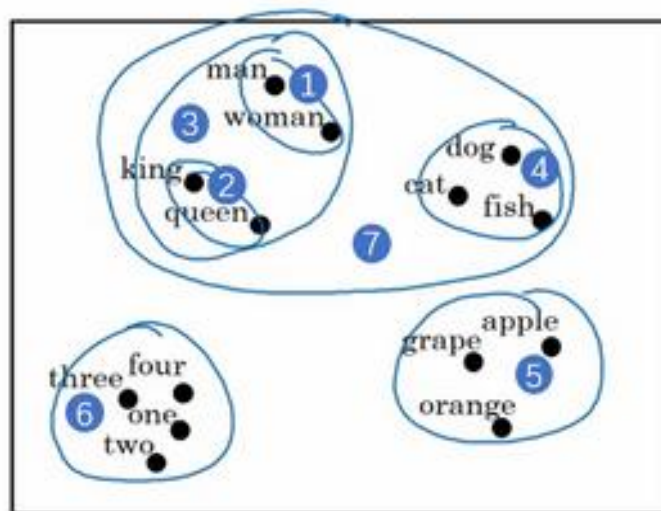
Andrew Ng



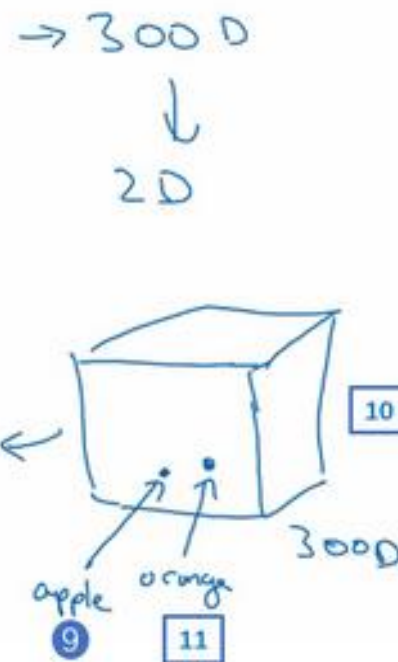
# 1.词汇表征和文本数据处理

6

Visualizing word embeddings

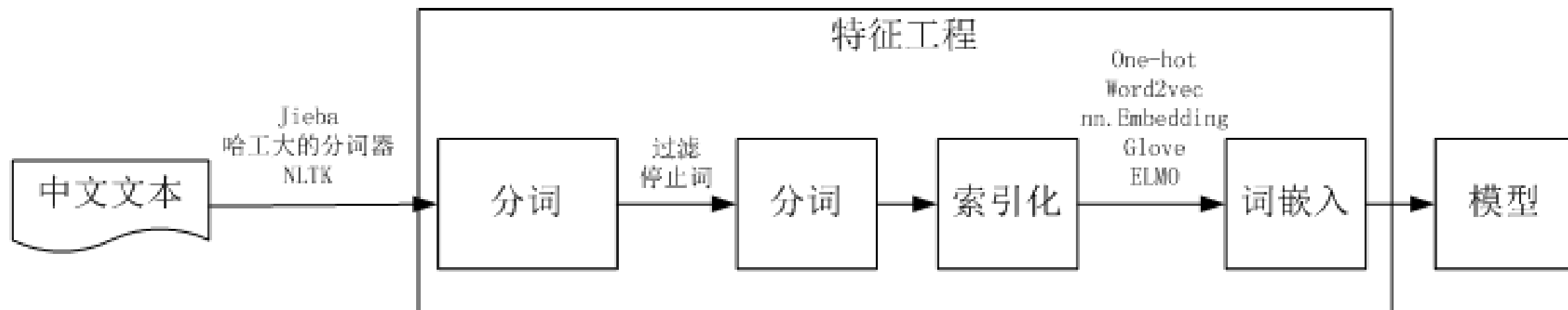


t-SNE



# 1.词汇表征和文本数据处理

7



## 2.词嵌入

8

**01 词汇表征和文本数据处理**

**02 词嵌入**

**03 Word2Vec**

**04 GloVe**

**05 情感分类**



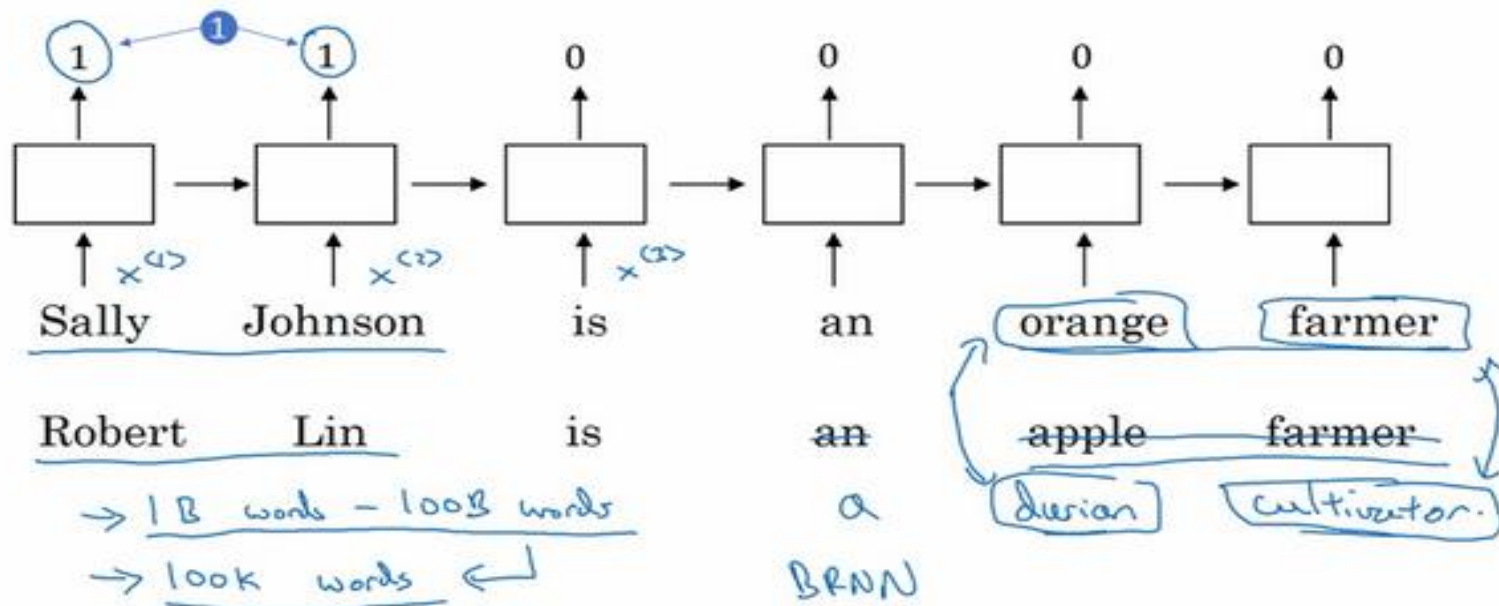
## 2.词嵌入

9

“Sally Johnson is an orange farmer.”

“Robert Lin is an apple farmer.”

Named entity recognition example



Andrew Ng

## 2.词嵌入

10

如何用词嵌入做迁移学习的步骤。

第一步，先从大量的文本集中学习词嵌入。

第二步，你可以用这些词嵌入模型把它迁移到你的新的只有少量标注训练集的任务中，比如说用这个300维的词嵌入来表示你的单词。这样做的一个好处就是你可以用更低维度的特征向量代替原来的10000维的**one-hot**向量，现在你可以用一个300维更加紧凑的向量。

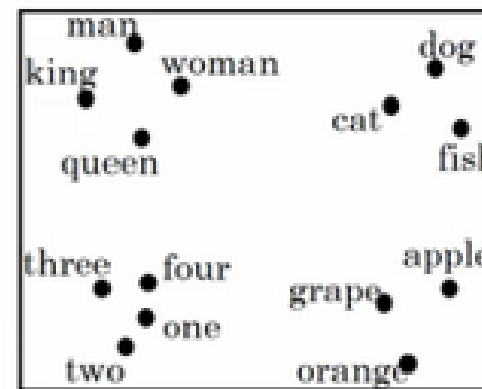
第三步，当你在你新的任务上训练模型时，在你的命名实体识别任务上，只有少量的标记数据集上，你可以自己选择要不要继续微调，用新的数据调整词嵌入。

## 2.词嵌入

11

### Analogies

	Man (5391)	Woman (9853)	King (4914)	Queen (7157)	Apple (456)	Orange (6257)
Gender	-1	1	-0.95	0.97	0.00	0.01
Royal	0.01	0.02	0.93	0.95	-0.01	0.00
Age	0.03	0.02	0.70	0.69	0.03	-0.02
Food	0.09	0.01	0.02	0.01	0.95	0.97



$$e_{\text{man}} - e_{\text{woman}} = \begin{bmatrix} -1 \\ 0.01 \\ 0.03 \\ 0.09 \end{bmatrix} - \begin{bmatrix} 1 \\ 0.02 \\ 0.02 \\ 0.01 \end{bmatrix} = \begin{bmatrix} -2 \\ -0.01 \\ 0.01 \\ 0.08 \end{bmatrix} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$e_{\text{king}} - e_{\text{queen}} = \begin{bmatrix} -0.95 \\ 0.93 \\ 0.70 \\ 0.02 \end{bmatrix} - \begin{bmatrix} 0.97 \\ 0.95 \\ 0.69 \\ 0.01 \end{bmatrix} = \begin{bmatrix} -1.92 \\ -0.02 \\ 0.01 \\ 0.01 \end{bmatrix} \approx \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

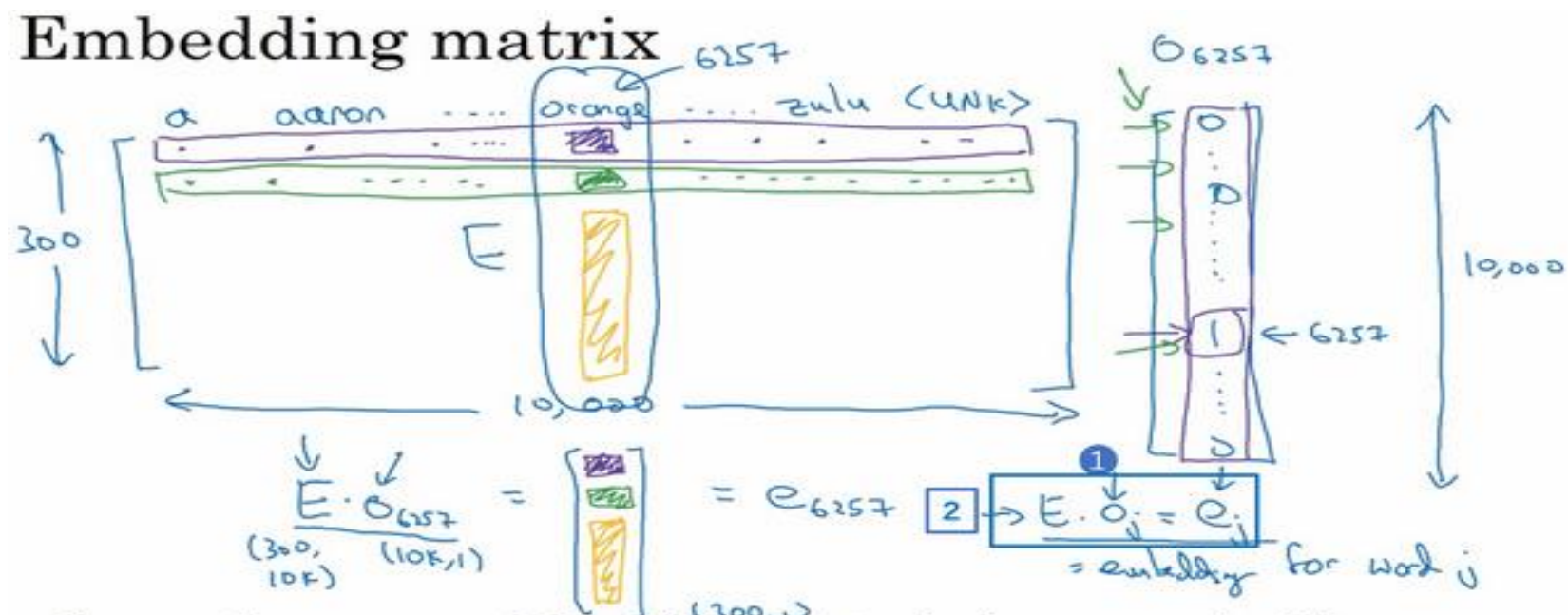
类似的，假如你用 $e_{\text{king}}$ 减去 $e_{\text{queen}}$ ，最后也会得到一样的结果

这个结果表示，**man**和**woman**主要的差异是**gender**（性别）上的差异

# 2.词嵌入

12

## 嵌入矩阵



In practice, use specialized function to look up an embedding.  
 $\rightarrow \text{Embedding}$

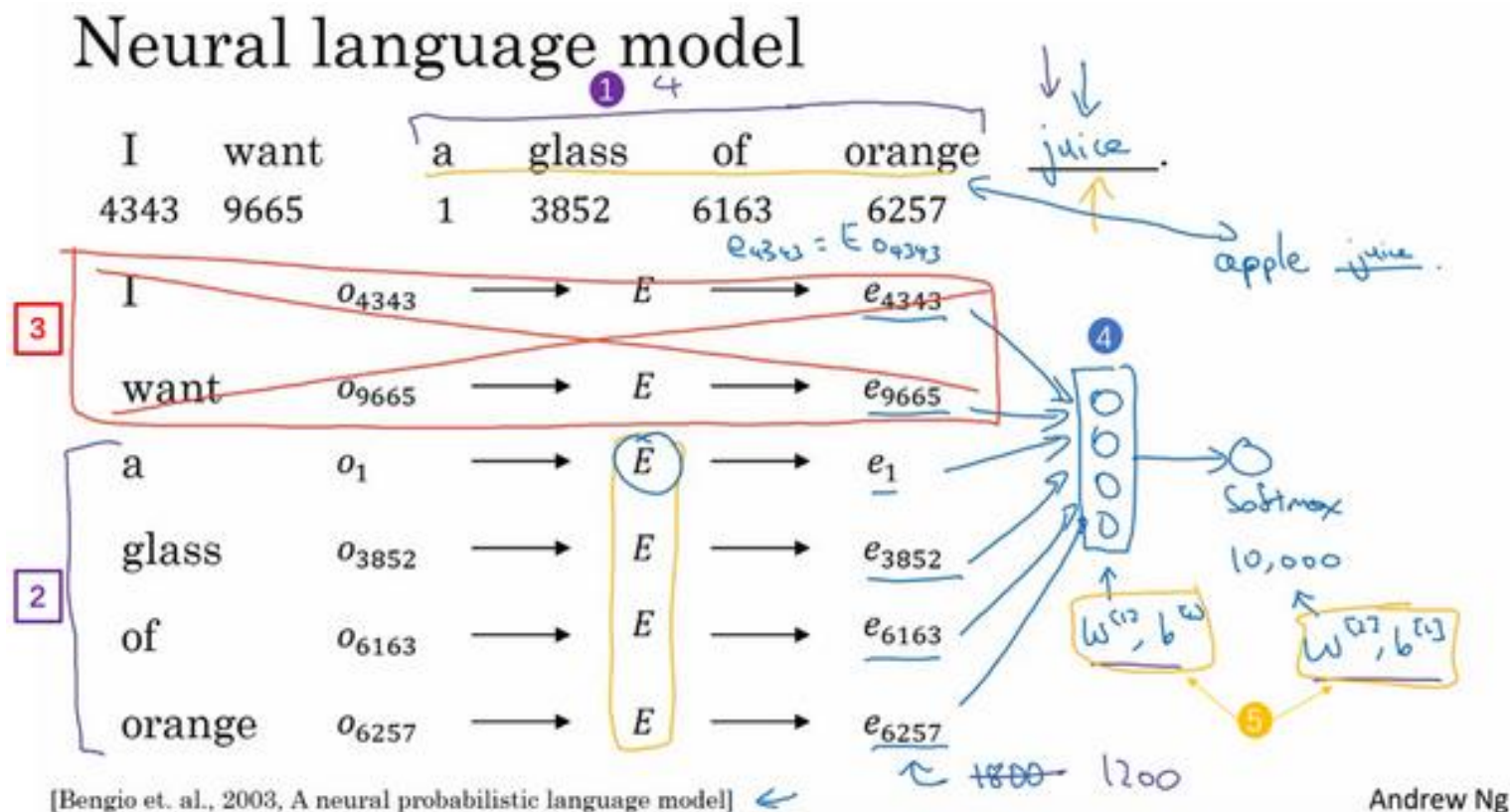
Andrew Ng

$e_w$  = embedding for word  $w$

# 2.词嵌入

13

## 嵌入矩阵



# 3.Word2Vec

14

**01** 词汇表征和文本数据处理

**02** 词嵌入

**03 Word2Vec**

**04 GloVe**

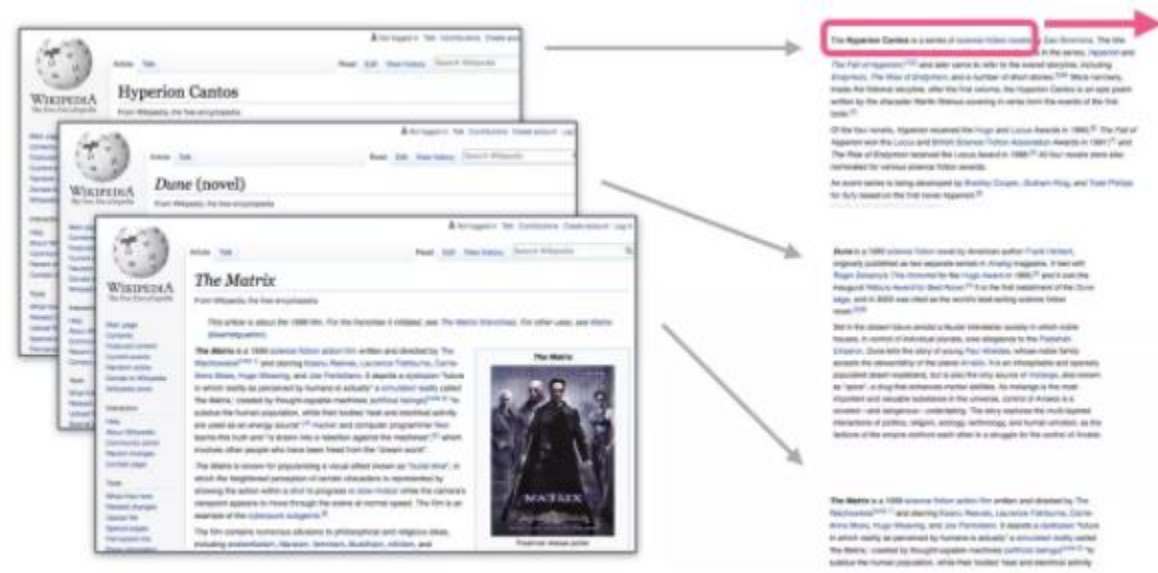
**05** 情感分类

# 3.Word2Vec

15

语言模型的训练机制就是这样

- 1.我们获得了大量文本数据（例如，所有维基百科文章）。然后
- 2.我们有一个窗口（比如说三个单词），我们会对所有文本进行滑动。
- 3.滑动窗口为我们的模型生成训练样本

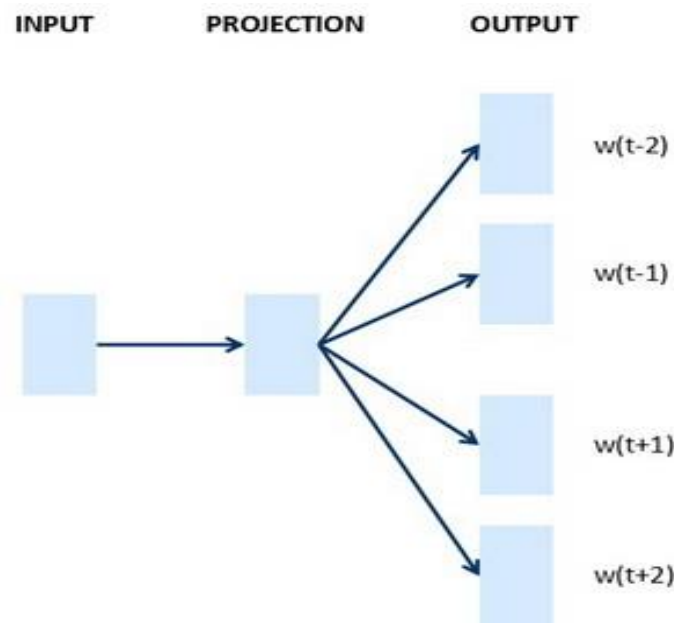
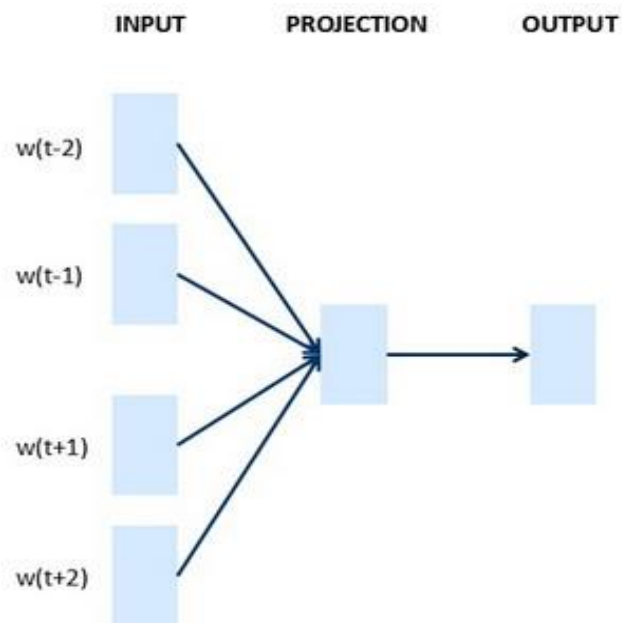




# 3.Word2Vec

16

(下图左边为CBOW，右边为Skip-Gram)



**CBOW**对小型数据库比较合适，而**Skip-Gram**在大型语料中表现更好。

# 3.Word2Vec

17

Jay was hit by a \_\_\_\_\_ bus in...

by	a	red	bus	in
----	---	-----	-----	----

我们实际构建和训练模型的数据集将如下所示：

input 1	input 2	input 3	input 4	output
by	a	bus	in	red

这被称为连续词袋结构，并在word2vec论文 one of the word2vec papers 中进行过描述。

# 3.Word2Vec

18

## 负采样

计算的角度来看，SkipGram非常消耗资源：尤其是我们将在数据集中为每个训练样本做一次（很可能数千万次）。我们需要做一些事情来提高效率。

一种方法是将目标分成两个步骤：

- 1.生成高质量的单词嵌入（不用担心下一个单词预测）。
- 2.使用这些高质量的嵌入来训练语言模型（进行下一个单词预测）。

# 3.Word2Vec

19

## 负采样

并不是每次迭代都训练全部10,000个，我们只训练其中的5个，我们要训练对应真正目标词那一个分类器，再训练4个随机选取的负样本，这就是 $K = 4$ 的情况。所以不使用一个巨大的10,000维度的**softmax**，因为计算成本很高，而是把它转变为10,000个二分类问题，每个都很容易计算，每次迭代我们要做的只是训练它们其中的5个，一般而言就是 $K + 1$ 个，其中 $K$ 个负样本和1个正样本。这也是为什么这个算法计算成本更低，因为只需更新 $K + 1$ 个逻辑单元， $K + 1$ 个二分类问题，相对而言每次迭代的成本比更新10,000维的**softmax**分类器成本低。

Selecting negative examples

context	word	target?
orange	juice	1
orange	king	0
orange	book	0
orange	the	0
orange	of	0

the, of, and, ...

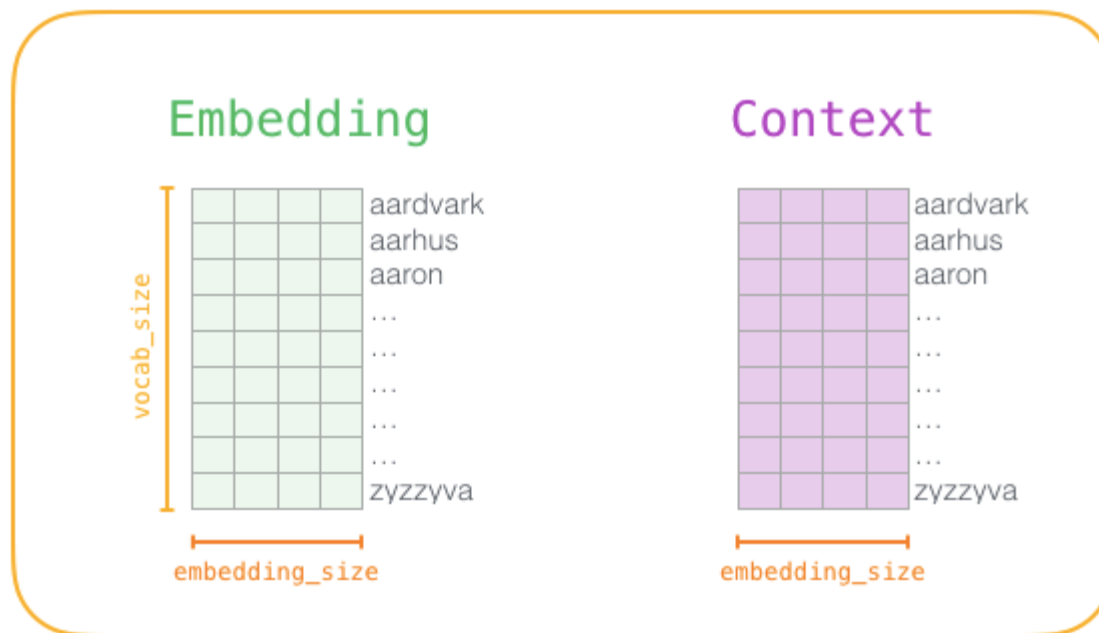
$$P(w_i) = \frac{f(w_i)^{\frac{3}{4}}}{\sum_{j=1}^{10,000} f(w_j)^{\frac{3}{4}}}$$

# 3.Word2Vec

20

## 训练流程

在训练过程开始之前，我们预先处理我们正在训练模型的文本。在这一步中，我们确定词汇量的大小（我们称之为`vocab_size`，比如说，将其视为10,000）以及哪些词属于它。在训练阶段的开始，我们创建两个矩阵 - `Embedding`矩阵和`Context`矩阵。这两个矩阵在我们的词汇表中嵌入了每个单词（这`vocab_size`是他们的维度之一）。第二个维度是我们希望每次嵌入的时间长度（`embedding_size`- 300是一个常见值）。



# 3.Word2Vec

21

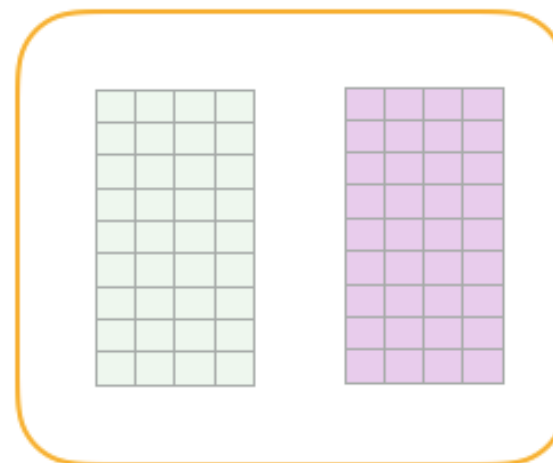
## 训练流程

在训练过程开始时，我们用随机值初始化这些矩阵。然后我们开始训练过程。在每个训练步骤中，我们采取一个正样本及其相关的负样本。我们来看看我们的第一组：

dataset

input word	output word	target
not	thou	1
not	aaron	0
not	taco	0
not	shalt	1
not	mango	0
not	finglonger	0
not	make	1
not	plumbus	0
...	...	...

model

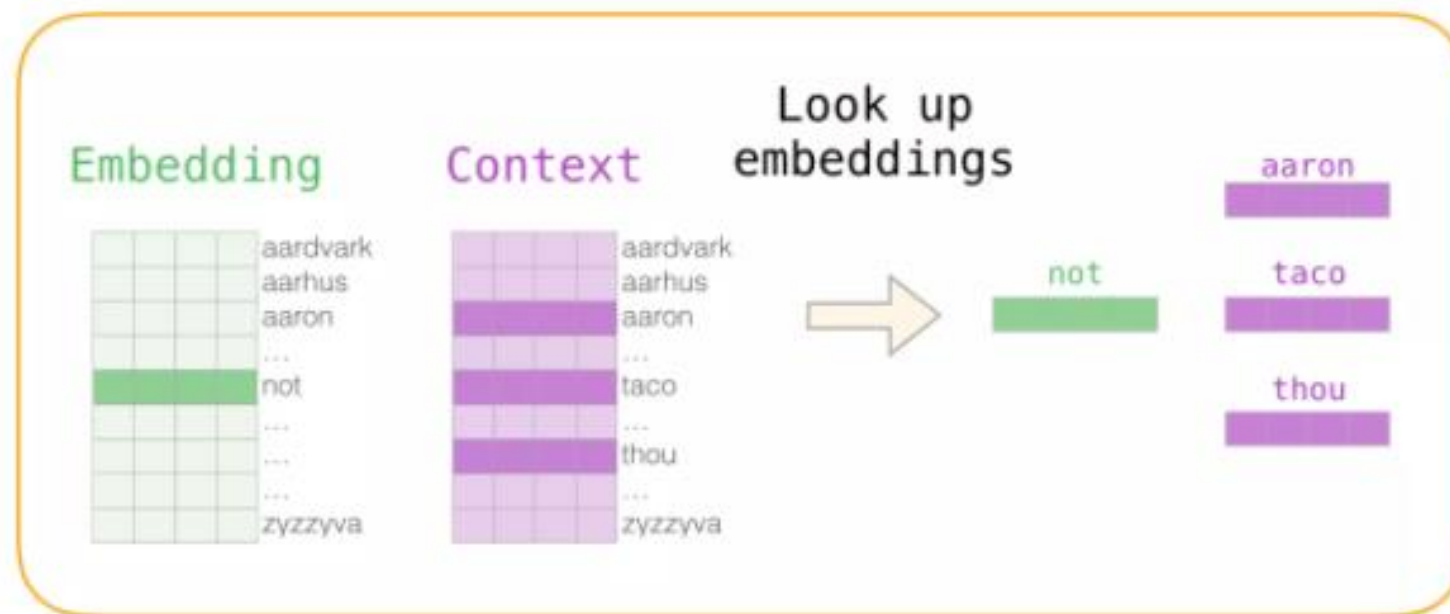


# 3.Word2Vec

22

## 训练流程

现在我们有四个单词：输入单词`not`和输出/上下文单词:(`thou`实际邻居),`aaron`, 和`taco` (负样本)。我们继续查找它们的嵌入 - 对于输入词, 我们查看`Embedding`矩阵。对于上下文单词, 我们查看`Context`矩阵 (即使两个矩阵都在我们的词汇表中嵌入了每个单词)。



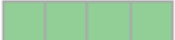













# 3.Word2Vec

23

**训练流程** 然后，我们计算输入嵌入与每个上下文嵌入的点积。在每种情况下，会产生一个数字，该数字表示输入和上下文嵌入的相似性。

input word	output word	target	input • output
not 	thou 	1	0.2
not 	aaron 	0	-1.11
not 	taco 	0	0.74

现在我们需要一种方法将这些分数转化为看起来像概率的东西：  
使用sigmoid函数把概率转换为0和1。







input word	output word	target	input • output	sigmoid()
not 	thou 	1	0.2	0.55
not 	aaron 	0	-1.11	0.25
not 	taco 	0	0.74	0.68

# 3.Word2Vec

24

## 训练流程

现在我们可以将sigmoid操作的输出视为这些样本的模型输出。您可以看到taco得分最高aaron，并且在sigmoid操作之前和之后仍然具有最低分。既然未经训练的模型已做出预测，并且看到我们有一个实际的目标标签要比较，那么让我们计算模型预测中的误差。为此，我们只从目标标签中减去sigmoid分数

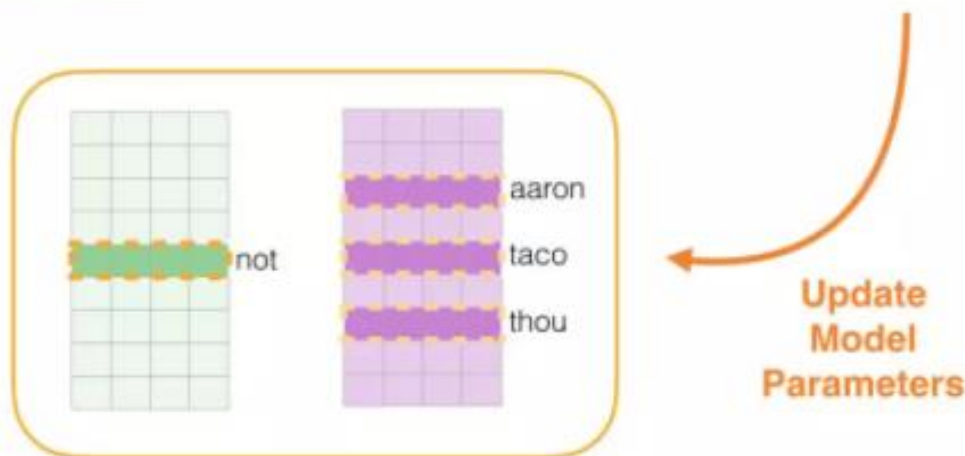
input word	output word	target	input • output	sigmoid()	Error
not 	thou 	1	0.2	0.55	0.45
not 	aaron 	0	-1.11	0.25	-0.25
not 	taco 	0	0.74	0.68	-0.68

# 3.Word2Vec

25

**训练流程** 这是“机器学习”的“学习”部分。现在，我们可以利用这个误差分数调整`not`，`thou`，`aaron`和`taco`的嵌入,使下一次我们做出这一计算，结果会更接近目标分数。

input word	output word	target	input • output	sigmoid()	Error
not	thou	1	0.2	0.55	0.45
not	aaron	0	-1.11	0.25	-0.25
not	taco	0	0.74	0.68	-0.68



# 3.Word2Vec

26

**训练流程** 训练步骤到此结束。我们从这一步骤中得到稍微好一点的嵌入（`not`，`thou`，`aaron`和`taco`）。我们现在进行下一步（下一个正样本及其相关的负样本），并再次执行相同的过程。

dataset			model	
input word	output word	target		
not	thou	1		
not	aaron	0		
not	taco	0		
not	shalt	1		
not	mango	0		
not	finglonger	0		
not	make	1		
not	plumbus	0		
...	...	...		
...	...	...		

当我们循环遍历整个数据集多次时，嵌入继续得到改进。然后我们可以停止训练过程，丢弃`Context`矩阵，并使用`Embeddings`矩阵作为下一个任务的预训练嵌入。

# 4.GloVe

27

**01** 词汇表征和文本数据处理

**02** 词嵌入

**03** Word2Vec

**04** GloVe

**05** 情感分类

## 4.GloVe

28

**GloVe**代表用词表示的全局变量 (**global vectors for word representation**) 。

对于**GloVe**算法，我们可以定义上下文和目标词为任意两个位置相近的单词，假设是左右各10词的距离，那么 $X_{ij}$ 就是一个能够获取单词 $i$ 和单词 $j$ 出现位置相近时或是彼此接近的频率的计数器。

**GloVe**模型做的就是进行优化，我们将他们之间的差距进行最小化处理：

$$\text{minimize } \sum_{i=1}^{10,000} \sum_{j=1}^{10,000} f(X_{ij}) (\theta_i^T e_j + b_i + b'_j - \log X_{ij})^2$$

# 5.情感分类

29

**01 词汇表征和文本数据处理**

**02 词嵌入**

**03 Word2Vec**

**04 GloVe**

**05 情感分类**



# 5.情感分类

30

## Sentiment classification problem

$x$   $\longrightarrow$   $y$

The dessert is excellent.



Service was quite slow.



Good for a quick meal, but nothing special.



Completely lacking in good taste, good service, and good ambience.



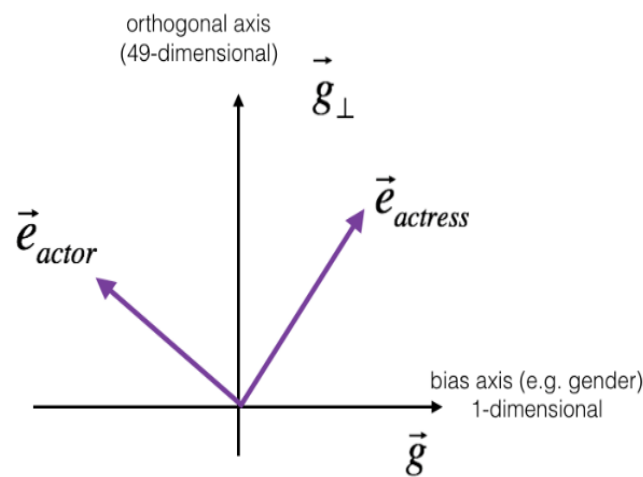
10,000  $\rightarrow$  100,000 words

# 5.情感分类

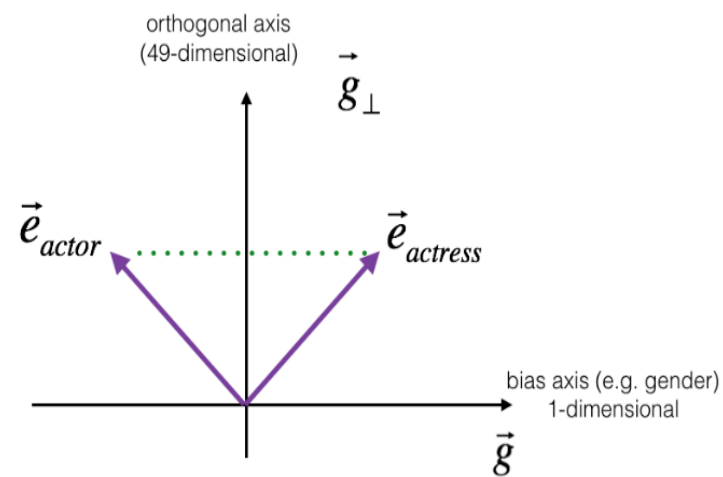
31

## 词嵌入除偏 (Debiasing Word Embeddings)

现在机器学习和人工智能算法正渐渐地被信任用以辅助或是制定极其重要的决策，因此我们想尽可能地确保它们不受非预期形式偏见影响，比如说性别歧视、种族歧视等等。



**before equalizing,**  
"actress" and "actor" differ  
in many ways beyond the  
direction of  $\vec{g}$



**after equalizing,**  
"actress" and "actor" differ  
only in the direction of  $\vec{g}$ , and further  
are equal in distance from  $\vec{g}_\perp$

# 5.情感分类

32

## 词嵌入除偏 (Debiasing Word Embeddings)

$$\mu = \frac{e_{w1} + e_{w2}}{2} \quad \mu_B = \frac{\mu * \text{bias}_a \text{xis}}{\|\text{bias}_a \text{xis}\|_2} + \|\text{bias}_a \text{xis}\|_2 * \text{bias}_a \text{xis}$$

$$\mu_{\perp} = \mu - \mu_B \quad e_{w1B} = \sqrt{|1 - \|\mu_{\perp}\|_2^2|} * \frac{(e_{w1} - \mu_{\perp}) - \mu_B}{|(e_{w1} - \mu_{\perp}) - \mu_B|}$$

$$e_{w2B} = \sqrt{|1 - \|\mu_{\perp}\|_2^2|} * \frac{(e_{w2} - \mu_{\perp}) - \mu_B}{|(e_{w2} - \mu_{\perp}) - \mu_B|}$$

$$e_1 = e_{w1B} + \mu_{\perp} \quad e_2 = e_{w2B} + \mu_{\perp}$$

1. IAN GOODFELLOW等, 《深度学习》, 人民邮电出版社, 2017
2. Andrew Ng, <http://www.deeplearning.ai>
3. <https://twitter.com/jalammar>

谢谢!

