

Word2Vec：一种 Embedding，将原始文本中的词嵌入(embedding)到一个新的多维空间中，用词向量方式表示词的语义信息。

语义相近的词在该空间内距离近

目的：Neural Network 只接受 vector. (tensor, matrix) 作为输入，因此要将文本等价转换为数字向量。

原理：Word2Vec 从大量的文本语料中以无监督学习的方式  
诗、文章...

学习语义结构，大量应用于 NLP 领域。

模型：① Skip - Gram ② CBOW

(Continuous Bag of Words)

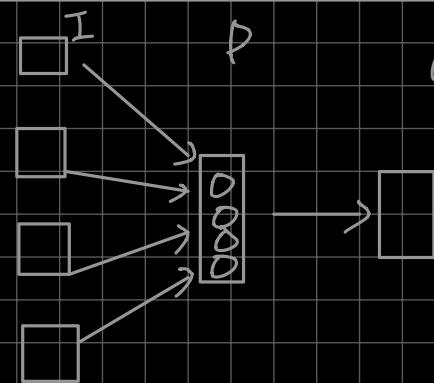
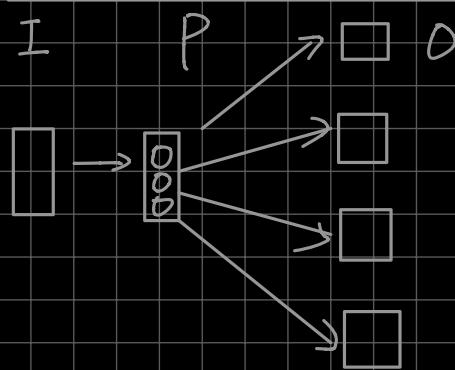
二者对比：

Skip Gram	CBOW
给定中心词 (input word)	给定上下文
预测上下文 (context)	预测中心词
适用于数据量较少，或生僻词出现次数较少	适用于数据量较大的场景
计算开销大，速度慢	计算开销小，速度快

类比：

1个学生 VS K个老师    K个学生 VS 1个老师  
    中心词                周围词                周围词                中心词

结构:



每个词作为中心词时，要使  
用  $K$  个周围词进行  $K$  次预测  
、调整，每个词都会受周围词  
影响，因此词向量相对准确

每个词作为中心词对周  
围  $K$  个词向量进行一次  
相同的调整

By Gradient Descent

Skip-Gram Model:

I. 建立模型  $\Rightarrow$  Fake Task.

原理：为了获取词向量的表示（即隐层权重），  
需要构建一个完整  $NN$  模型进行训练。  
通过模型 ~~获取~~ 取出嵌入词向量

Components:

Input word: 即中心词（用 one-hot 编码表示）

skip-window: input word - 一个 ~~词~~ 选取周围词的数量

span: 2 \* skip-window . 即  $K$

Training samples:  $K$  个 (中心词, 周围词) 组成

Output layer: Softmax classifier, 大小为 (anc directory)

Improve points (优化之处):

① 隐层大小设为 300 (用 300 维向量来表示一个单词)  $\Rightarrow$  减小了隐层参数量 ( $10000 \rightarrow 300$ )

② Look up table: 进行矩阵计算时, 直接去查输入向量中取值为 1 的下标, 对应的权重值进行计算 ( $1 \times 10000 @ 10000 \times 300 \rightarrow 1 @ 300$ )

## II. 高效训练

word pairs and phases

对高频频词进行抽样  $\Rightarrow$  获取中心词

negative sampling

对高频频词进行抽样:

$$P(w_i) = \underbrace{\left( \sqrt{\frac{Z(w_i)}{e^{-3}}} + 1 \right)}_{\text{保留单词 } w_i \text{ 的概率}} \times \underbrace{\frac{e^{-3}}{Z(w_i)}}_{w_i \text{ 单词的频率}}$$

保留单词  $w_i$  的概率

sample 值,

越小  $\Leftrightarrow w_i$  删除概率越大

$$\Rightarrow \lceil Z(w_i) \rceil = 0.0026, P(w_i) = -0$$

此类 words 必被保留

$Z(w_i) = 0.00746$ ,  $P(w_i) = 0.5$   
此类 words 可能被保留 (50% 根率)

$Z(w_i) = 1.0$ ,  $P(w_i) = 0.033$

太常出现的单词 几乎不被保留

目的：对原始训练文本中遇到的每个中心词，  
均有一定根率被删除，且此根率与单  
词频率成正比。

Negative sampling (负采样)

$$P(w_i) = \frac{f(w_i)^{\frac{3}{4}}}{\sum_{j=0}^n f(w_j)^{\frac{3}{4}}}$$

选为 negative words 的根率

$\Rightarrow$  经验指数

$\Rightarrow$  原本 每个训练样本 更新所有权重 (计算开销大，训练缓慢)

负采样 每次让一个训练样本更新  
一部分权重，大大降低了 GD 中计算量。  
negative words 对应 neurons,

当然 positive word 也进行权重更新

新

Details: Unigram table

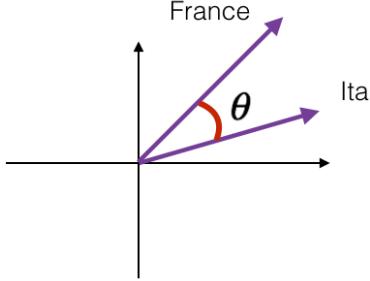
包含 1 亿个元素的 array ( $n = 1e^9$ )

每个 word 的索引号构成。

$$\#(w_i \text{ in table}) = n * P(w_i)$$

因此当选择 negative word 时，只需  $i = \text{rand}(1, n)$   
并选择 table[i] 所对应  $w_i$  即可。

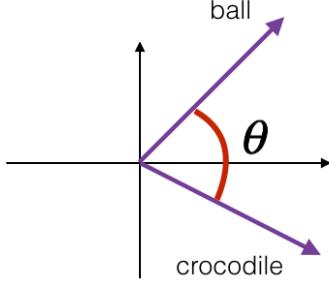
# I. Cosine Similarity



France and Italy are quite similar

$\theta$  is close to 0°

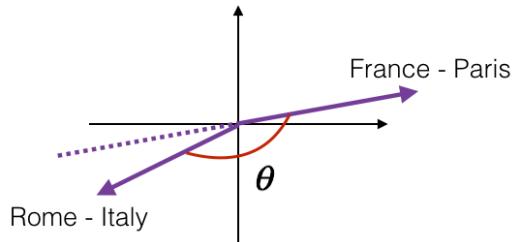
$$\cos(\theta) \approx 1$$



ball and crocodile are not similar

$\theta$  is close to 90°

$$\cos(\theta) \approx 0$$



the two vectors are similar but opposite  
the first one encodes (city - country)  
while the second one encodes (country - city)

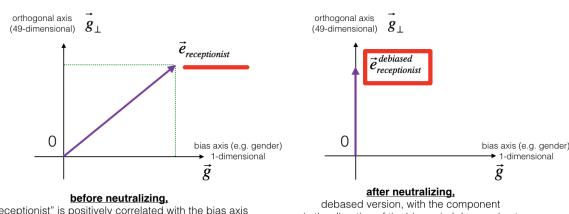
$\theta$  is close to 180°

$$\cos(\theta) \approx -1$$

应用: word analogy task ( $e_b - e_a \approx e_d - \underline{e_c}$ ) 未知向量,

## II. Neutralize bias for non-gender specific words

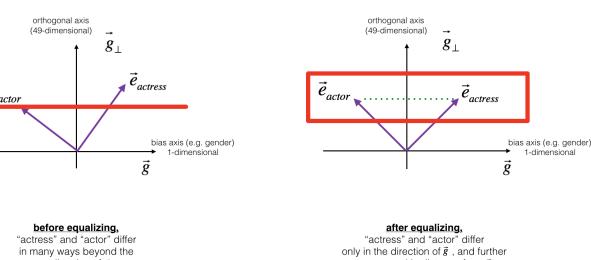
通过遍历比  
较 cosine similarity  
选择  $\cos(\theta)$  最大的



$$e^{bias\_component} = \frac{e \cdot g}{\|g\|_2^2} * g \quad (2)$$

$$e^{debiased} = e - e^{bias\_component} \quad (3)$$

## III. Equalization for gender specific words



$$\mu = \frac{e_{w1} + e_{w2}}{2} \quad (4)$$

$$\mu_B = \frac{\mu \cdot bias\_axis}{\|bias\_axis\|_2^2} * bias\_axis \quad (5)$$

$$e_{w1B} = \frac{\mu_1 - \mu_B}{\|bias\_axis\|_2^2} * bias\_axis \quad (6)$$

$$e_{w2B} = \frac{\mu_2 - \mu_B}{\|bias\_axis\|_2^2} * bias\_axis \quad (7)$$

$$e_{w1B}^{corrected} = \sqrt{[1 - \|\mu_1\|_2^2] * \frac{e_{w1B} - \mu_B}{\|(e_{w1} - \mu_1) - \mu_B\|}} \quad (8)$$

$$e_{w2B}^{corrected} = \sqrt{[1 - \|\mu_2\|_2^2] * \frac{e_{w2B} - \mu_B}{\|(e_{w2} - \mu_2) - \mu_B\|}} \quad (9)$$

$$e_1 = e_{w1B}^{corrected} + \mu_1 \quad (10)$$

$$e_2 = e_{w2B}^{corrected} + \mu_2 \quad (11)$$