1、词汇向量表

之前的 RNN 模型在处理文字的时候，是将文字转换为一个个 onehot 编码之后再进行计算的，在序列生成过程中，其实每个单词/字母经过这样的编码之后是毫无关联的，因为它们的内积=0，虽然训练过程是将所有单词都依次输入进去，但内部并无关联，模型训练的是它们特征的简单合并。在其他应用场景下，比如：

I want a glass of orange juice.

I want a glass of apple juice.

虽然都能通过序列生成模型得到 juice，但第一句话的训练对第二句并无真正意义上的帮助，因为模型并不知道 apple 其实和 orange 很类似。

2、词汇表征

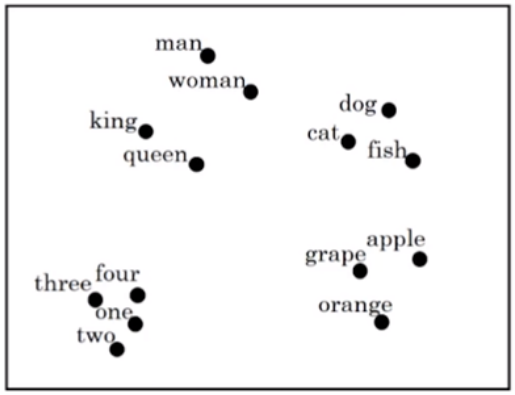
基于以上原因，我们需要一种新的区别于 onehot 方式的编码过程，这里称之为词汇表征，在下方这幅图里，第一行表示不同实体名词，第一列表示每个单词可以具有的不

	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	0.7	0.69	0.03	-0.02
Food	0.04	0.01	0.02	0.01	0.95	0.97
⋮						
size						
cost						
alive						

同特征，这样一来每个单词的编码就会详细丰富很多，如果相似的单词，它们的内积就会很大，不相似的单词内积就会很小。参考三维向量空间的内积运算。

3、词汇嵌入

将这些具有众多特征的单词聚类起来，可以使用 t-SNE 算法，进行词汇嵌入，基本理念是将高维特征映射到低维特征，然后利用欧式距离/余弦相似度模型得到最终嵌入的词汇。



4、词汇表征为什么有效

个人理解词汇表征是用内容更丰富的特征值来表示的一种方式，相比于 onehot 编码，特征复杂使得不同词汇之间有了关联，能够互相学习，这是它有效最关键的原因。其次，我们可以使用迁移学习的方式，通过使用别人已经编码好的巨型词汇表特征，将自己的项目的词汇“映射”其中，并且可以减小特征数量，使其更紧密。

从原理上来分析，词与词之间之所以有了关联，是因为特征余弦相似度，如果将 u 定义为第一个词汇的特征，v 定义为第二个词汇的特征，那么它们之间的相似度可以通过该一下公式来计算：

$$sim(u,v) = \frac{u^T v}{\|u\|_2 \|v\|_2}$$

当二者特征相似时，该值趋近于 1（而只有当夹角为 0° 时，余弦为 1），反之趋向于 0，如果特征方向相反，则趋近于-1，该模型很好的测量了不同实体之间的相似度，而人脸识别领域中的欧式距离计算公式更侧重于相异度。

5、后续步骤

得到了词汇的特征之后，那么剩下的算法就和之前的 RNN 算法类似了，对于词汇生成模型，我们可以通过将前面几个单词的特征输入到 RNN 网络中，进行下一个单词的预测，抑或是将前四个单词的特征同时输入到一对一神经网络中，通过 softmax 输出预测单词，比起将四个单词的特征作为输入，skip gram 更为简单有效，且只需要用到前面的一个单词作为输入。

6、Word2Vec

Word2vec 模型有两种方式，一种是 CBOW，是通过两边的词汇预测中间的词汇，而另一种是 skip gram，通过中间的词汇预测两边（词距<=α ）的词汇，比如输入为 orange，输出为 juice，那么训练的时候首先要进行特征编码，获得特征值 e，然后通过 softmax 进行预测，然而这里有一个严重的问题，就是在进行 softmax 分类预测的时候，会将分母进行加和，而分母有很多数量（商业级别的在百万条词库以上）。

针对上述问题，有两种解决方案，一种是通过非平衡二叉树寻找，常见词汇在树的浅部，不常见词汇在树的深部，另一种更有效的方法是负采样,该方案的原理是使用多级 softmax 分类预测。

7、负采样

负采样所用的方法与 skip gram 类似，不过更能体现非常见词汇的训练，即在一轮采样过程中，上下文、目标词选择一组，标记为 1，及正采样，然后从语料库中随机选取多组目标词与该上下文构成负采样，这样一组训练集中即有正采样，又有负采样，使得非常见词汇训练频率大大提高。另外关于样本组数量的选择 k，训练集越大，k 越小。

在进行训练的时候，比如我们有如下一组训练集，其中 1 个正样本，4 个负样本，

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

那么训练的时候不再以通过上下文来预测目标词，而是训练一个二分类的 logistic 回归模型，当输入是 orange、juice 时，输出为 1，其余情况为 0，相比于使用 softmax 预测分类，该方法不会进行大数量的加和操作，所以运算效率较高。

总结一下负采样的优点：

A、非常见词训练频率变高

B、训练过程（输出）运算效率提高

最后说明一下负样本的选取，一种方法是完全根据词汇出现频率来选择，但会出现极端情况，另一种是在语料库中完全均匀分布，但会使得词汇缺乏代表性，准确性大打折扣。所以现在普遍的做法是用，wi 是目标词汇，wj 是总体词汇，通过该公式得到每个词的频率，最后根据该词频选择负样本进行采样。

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

8、Glove 采样法

之前的 Word2Vec 模型是一种根据词向量特征来进行词汇预测的，通过 RNN 网络进行 softmax 预测或者 logistic 分类决定下一个词汇。

本次的 Glove 算法首先定义的是一个 X_{it}，i 代表上下文，t 代表目标词，左右 α 距离的词汇，不难发现它们具有对称性，即 $X_{it}=X_{ti}$，该算法的实现思路是通过学习词汇出现频率来进行序列生成的，具体的优化公式是通过：

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

X_{ij} 代表词汇出现频次，如果越频繁，则该值越大，$\theta_i^T e_j$ 代表两个词汇的特征值，$f(X_{ij})$ 进行常见词汇和不常见词汇的频分权重，如果 xij 过大，则 f 值适当缩小，使得算法不要过于预测常见词汇的出现频率，反之 xij 越小，则 f 适当放大，以加大非常见词汇的训练力度。最终的训练效果是根据不同词汇的特征预测出它们出现一起的频次有多大，进而根据上下文选择目标词汇。

关于 $\theta_i^T e_j$ 的初始化，由于 θe 是完全对称的，即该算法中上下文和目标词汇是同等地位的，因此初始化过程应当一并随机初始化，而后经过训练，再用公式：

初始化所有词汇的特征，这样就可以很好的预测出不同词汇间出现的频次了。

到目前为止，已经介绍了所有的词汇嵌入的方法，需要提醒的是，这仅是在**序列生成**中所使用的算法，即根据上下文词汇特征输出目标词 onehot 编码，或者根据上下文和目标词的特征输出概率/频次。