Word Embedding & Word2vec

詞嵌入以及word2vec介紹

Sean,王家祥

CONTENTS

/01 Word Embedding

What is this ?

Word Embedding - 什麼是詞嵌入(Word Embedding)

- Word Embedding 的概念是建立字詞向量（Word Vector），例如我定義一個向量的每個維度對應到什麼字，並且將句子中每個字轉換為向量，最後結合起來變成矩陣。

```
["I", "like", "apple", "mango"]


I     => [1, 0, 0, 0]
like  => [0, 1, 0, 0]
apple => [0, 0, 1, 0]
mango => [0, 0, 0, 1]


"I like apple" =>
[[1, 0, 0, 0],
 [0, 1, 0, 0],
 [0, 0, 1, 0]]


"I like mango" =>
[[1, 0, 0, 0],
 [0, 1, 0, 0],
 [0, 0, 0, 1]]
```

Word Embedding - 什麼是詞嵌入(Word Embedding)

- 自然语言是一套用来表达含义的复杂系统。在这套系统中，词是表义本单元。
- 顾名思义，词向量是用来表示词的向量，也可被认为是词的特征向量或表征。
- **把词映射为实数域向量的技术也叫词嵌入（word embedding）。**
- 在NLP(自然语言处理)领域，文本表示是第一步，也是很重要的一步，通俗来说就是把人类的语言符号转化为机器能够进行计算的数字，因为普通的文本语言机器是看不懂的，必须通过转化来表征对应文本。早期是**基于规则**的方法进行转化，而现代的方法是**基于统计机器学习**的方法。

/02 離散表示

Discrete for word

2.离散表示 - One-hot表示

- One-hot简称读热向量编码，也是特征工程中最常用的方法。其步骤如下：
- 构造文本分词后的字典，每个分词是一个比特值，比特值为0或者1。
- 每个分词的文本表示为该分词的比特位为1，其余位为0的矩阵表示。
- One-hot表示文本信息的**缺点**：
- 随着语料库的增加，数据特征的维度会越来越大，产生一个维度很高，又很稀疏的矩阵。
- 这种表示方法的分词顺序和在句子中的顺序是无关的，不能保留词与词之间的关系信息。
- 例：**John likes to watch movies. Mary likes too**
- **John also likes to watch football games.**
- 以上两句可以构造一个词典，**{"John": 1, "likes": 2, "to": 3, "watch": 4, "movies": 5, "also": 6, "football": 7, "games": 8, "Mary": 9, "too": 10}**
- 每个词典索引对应着比特位。那么利用One-hot表示为：
- **John: [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]**
- **likes: [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]** .....等等，以此类推。

2.离散表示 - 词袋模型

- 词袋模型(Bag-of-words model)，像是句子或是文件这样的文字可以用一个袋子装着这些词的方式表现，这种表现方式不考虑文法以及词的顺序。

- **文档的向量表示可以直接将各词的词向量表示加和。**例如：
- **John likes to watch movies. Mary likes too**
- **John also likes to watch football games.**

- 以上两句可以构造一个词典，**{"John": 1, "likes": 2, "to": 3, "watch": 4, "movies": 5, "also": 6, "football": 7, "games": 8, "Mary": 9, "too": 10}**

- 那么第一句的向量表示为：**[1,2,1,1,1,0,0,0,1,1]**，其中的2表示**likes**在该句中出现了2次，依次类推。
- 词袋模型同样有一下**缺点**：
- 词向量化后，词与词之间是有大小关系的，不一定词出现的越多，权重越大。
- 词与词之间是没有顺序关系的。
-

2.离散表示 - TF-IDF

- TF-IDF（term frequency–inverse document frequency）是一种用于信息检索与数据挖掘的常用加权技术。TF意思是词频(Term Frequency)，IDF意思是逆文本频率指数(Inverse Document Frequency)。

- **字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在语料库中出现的频率成反比下降。一个词语在一篇文章中出现次数越多, 同时在所有文档中出现次数越少, 越能够代表该文章。**

$$TF_w = \frac{\text{在某一类中词条}w\text{出现的次数}}{\text{该类中所有的词条数目}}$$

$$IDF = log(\frac{\text{语料库的文档总数}}{\text{包含词条}w\text{的文档总数}+1})$$

- 分母之所以加1，是为了避免分母为0。
- **缺点：**还是没有把词与词之间的关系顺序表达出来

2.離散表示 - TF-IDF

- tf-idf = tf x idf

詞彙	文件1	文件2	…	文件 D
文字 1	0.48	0.03	…	0.00
文字 2	0.00	0.37	…	0.38
文字 3	0.05	0.08	…	0.22
…	…	…	…	…
文字 T	0.12	0.19	…	0.00

2.离散表示 - n-gram模型

- n-gram模型为了保持词的顺序，做了一个滑窗的操作，这里的n表示的就是滑窗的大小，
- 例如2-gram模型，也就是把2个词当做一组来处理，然后向后移动一个词的长度，再次组成另一组词，把这些生成一个字典，按照词袋模型的方式进行编码得到结果。

- 例如：
- **John likes to watch movies. Mary likes too**
- **John also likes to watch football games.**

- 以上两句可以构造一个词典，**{"John likes”: 1, "likes to”: 2, "to watch”: 3, "watch movies”: 4, "Mary likes”: 5, "likes too”: 6, "John also”: 7, "also likes”: 8, “watch football”: 9, "football games": 10}**
- 那么第一句的向量表示为：**[1, 1, 1, 1, 1, 1, 0, 0, 0, 0]**，其中第一个1表示**John likes**在该句中出现了1次，依次类推。

- **缺点：**随着n的大小增加，词表会成指数型膨胀，会越来越大。

/03 分布式表示

Relations

3. 分布式表示

- 科学家们为了提高模型的精度，又发明出了分布式的表示文本信息的方法，这就是这一节需要介绍的。

- **用一个词附近的其它词来表示该词，这是现代统计自然语言处理中最有创见的想法之一。**

- 当初科学家发明这种方法是基于人的语言表达，认为一个词是由这个词的周边词汇一起来构成精确的语义信息。
- 就好比，物以类聚人以群分，如果你想了解一个人，可以通过他周围的人进行了解，因为周围人都有一些共同点才能聚集起来。

3.分布式表示 - 共现矩阵

- 共现矩阵顾名思义就是共同出现的意思
- 局域窗中的word-word共现矩阵可以挖掘语法和语义信息，**例如：**
- I like deep learning.
- I like NLP.
- I enjoy flying
- 有以上三句话，设置滑窗为2，可以得到一个词典：**{"I like","like deep","deep learning","like NLP","I enjoy","enjoy flying","I like"}。**
- 我们可以得到一个共现矩阵(对称矩阵)：

3.分布式表示 - 共现矩阵

- 中间的每个格子表示的是行和列组成的词组在词典中共同出现的次数，也就体现了**共现**的特性。

counts	I	like	enjoy	deep	learning	NLP	flying	.
I	0	2	1	0	0	0	0	0
like	2	0	0	1	0	1	0	0
enjoy	1	0	0	0	0	0	1	0
deep	0	1	0	0	1	0	0	0
learning	0	0	0	1	0	0	0	1
NLP	0	1	0	0	0	0	0	1
flying	0	0	1	0	0	0	0	1
.	0	0	0	0	1	1	1	0

3.分布式表示 - 共现矩阵

- **存在的问题：**
- 向量维数随着词典大小线性增长。
- 存储整个词典的空间消耗非常大。
- 一些模型如文本分类模型会面临稀疏性问题。
- **模型会欠稳定，每新增一份语料进来，稳定性就会变化。**
-

/04 神经网络表示 - NNLM

Not to talk to much for this chapater

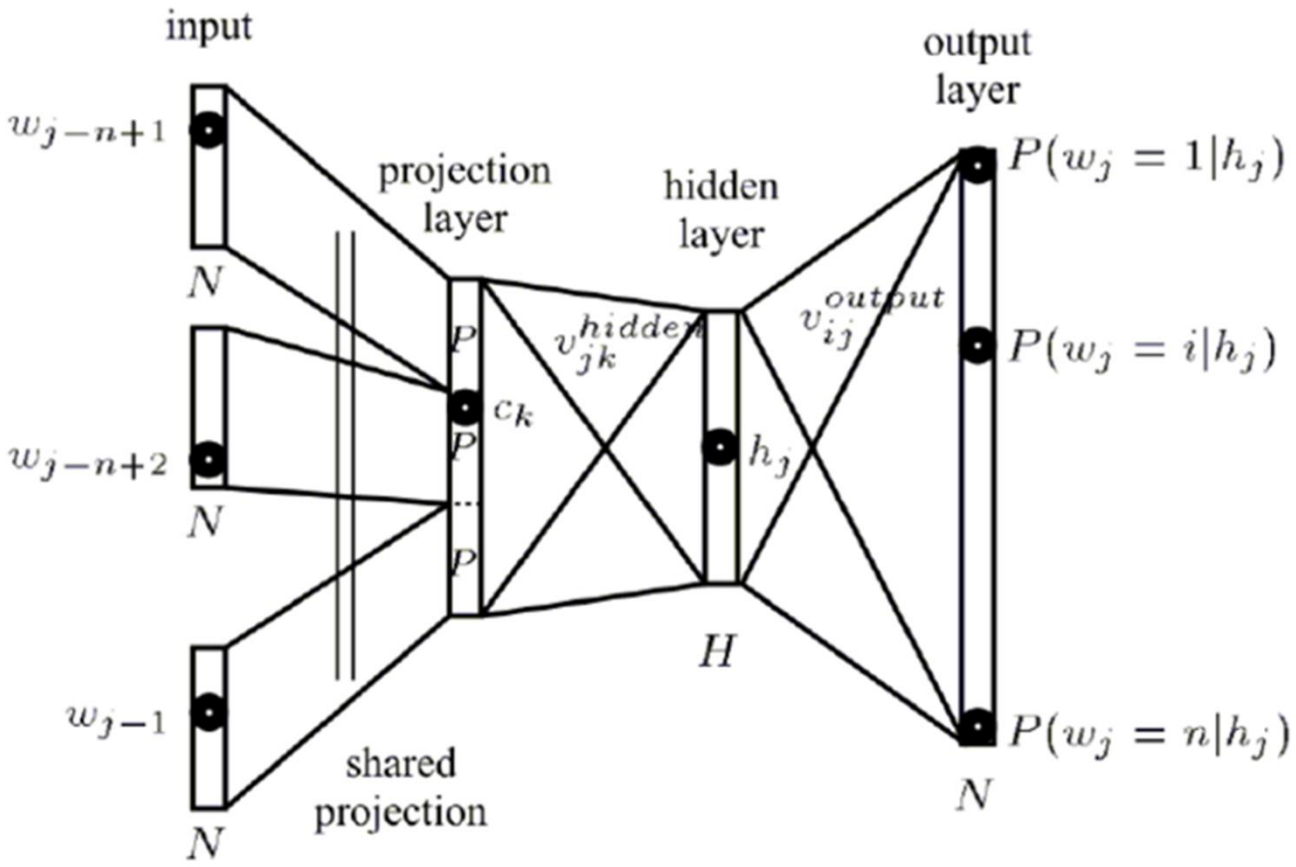
4.神经网络表示 - NNLM

- NNLM (Neural Network Language model)，神经网络语言模型是03年提出来的，
- 通过训练得到中间产物--词向量矩阵，这就是我们要得到的文本表示向量矩阵。
- NNLM说的是定义一个前向窗口大小，其实和上面提到的窗口是一个意思。
- 把这个窗口中最后一个词当做y，把之前的词当做输入x，通俗来说就是预测这个窗口中最后一个词出现概率的模型。

4.神经网络表示 - NNLM

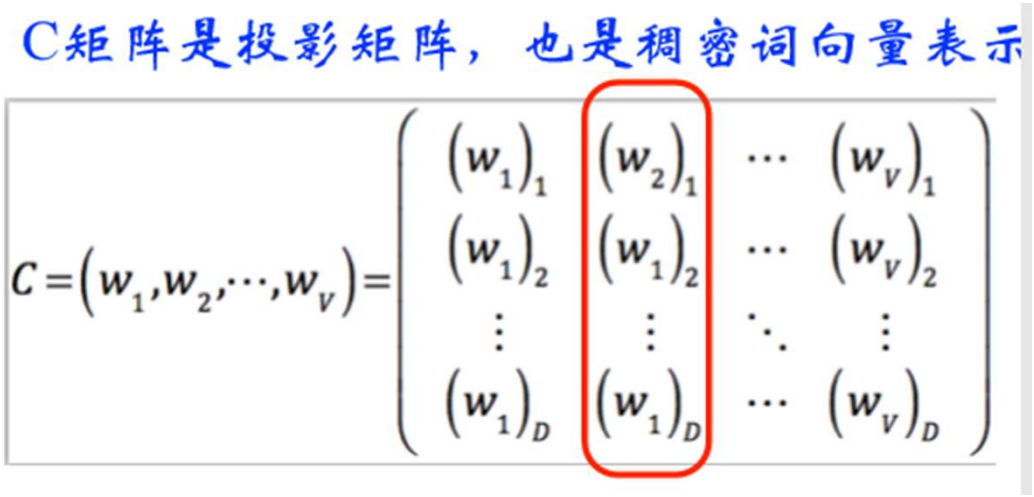
- NNLM是從語言模型出發(即計算概率角度)，構建神經網路針對目標函式對模型進行最優化，
- 訓練的起點是使用神經網路去搭建語言模型實現詞的預測任務，並且在優化過程後模型的副產品就是詞向量。
- *進行神經網路模型的訓練時，目標是進行詞的概率預測，就是在詞環境下，預測下一個該是什麼詞
- ，目標函式如下式, 通過對網路訓練一定程度後，最後的模型引數就可當成詞向量使用.

4.神经网络表示 - NNLM



4.神经网络表示 - NNLM

- input层是一个前向词的输入，是经过one-hot编码的词向量表示形式，具有V*1的矩阵。
- C矩阵是投影矩阵，也就是稠密词向量表示，在神经网络中是**w参数矩阵**，该矩阵的大小为D*V，正好与input层进行全连接(相乘)得到D*1的矩阵，采用线性映射将one-hot表示投影到稠密D维表示。
- output层(softmax)自然是前向窗中需要预测的词。
- 通过BP + SGD得到最优的C投影矩阵，这就是NNLM的中间产物，也是我们所求的文本表示矩阵，
- **通过NNLM将稀疏矩阵投影到稠密向量矩阵中。**

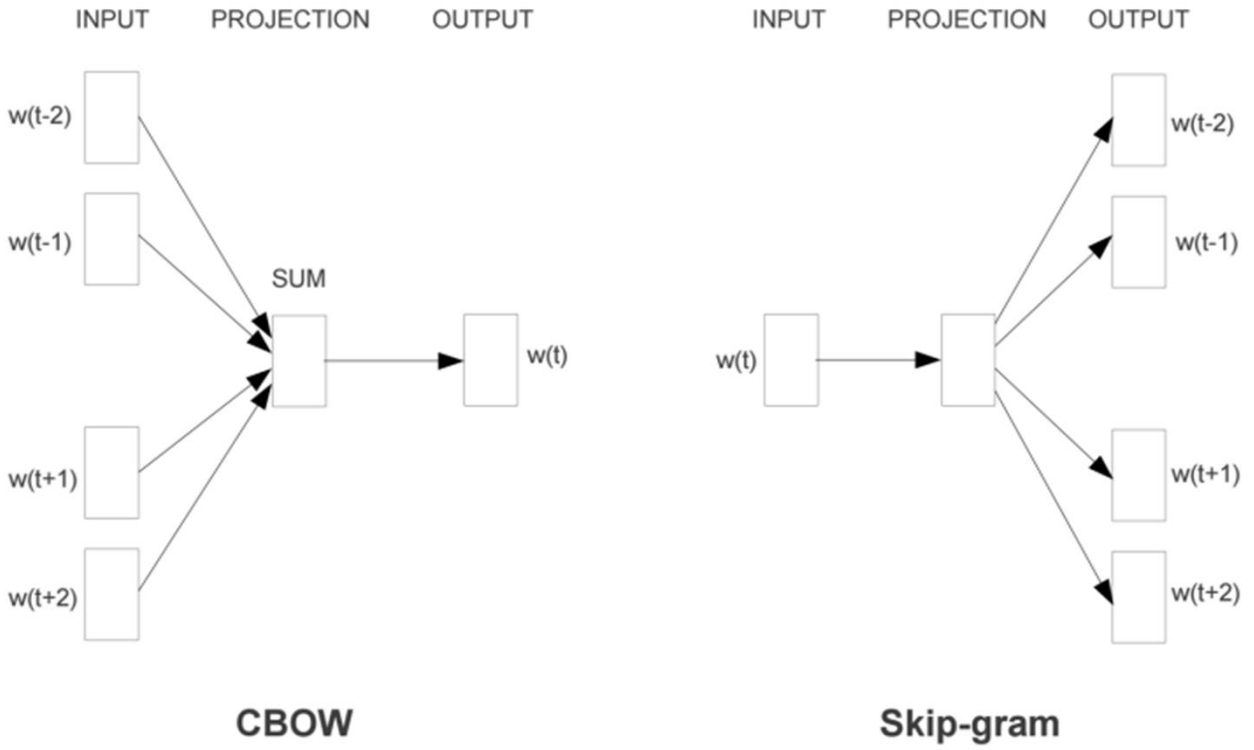


/05 Word2Vec

Simple introuce

5.Word2Vec (NNLM優化)

- 谷歌2013年提出的Word2Vec是目前最常用的词嵌入模型之一。Word2Vec实际是一种浅层的神经网络模型，它有两种网络结构，**分别是CBOW（Continues Bag of Words）连续词袋和Skip-gram。**Word2Vec和上面的NNLM很类似，但比NNLM简单。



5.Word2Vec - Skip-gram

- Skip-gram是通过当前词来预测窗口中上下文词出现的概率模型，
- 把当前词当做x，把窗口中其它词当做y，
- 依然是通过一个隐层接一个Softmax激活函数来预测其它词的概率。

5.Word2Vec - CBOW

- CBOW获得中间词两边的的上下文，然后用周围的词去预测中间的词，把中间词当做y，
- 把窗口中的其它词当做x输入，x输入是经过one-hot编码过的，然后通过一个隐层进行求和操作，
- 最后通过激活函数softmax，可以计算出每个单词的生成概率，
- 接下来的任务就是训练神经网络的权重，使得语料库中所有单词的整体生成概率最大化，而求得的权重矩阵就是文本表示词向量的结果。

5.Word2Vec - 词嵌入为何不采用one-hot向量

- 虽然one-hot词向量构造起来很容易，但通常并不是一个好选择。
- 一个主要的原因是，one-hot词向量无法准确表达不同词之间的相似度，如我们常常使用的余弦相似度。
- 由于任何两个不同词的one-hot向量的余弦相似度都为0，多个不同词之间的相似度难以通过onehot向量准确地体现出来。
- word2vec工具的提出正是为了解决上面这个问题。它将每个词表示成一个定长的向量，并使得这些向量能较好地表达不同词之间的相似和类比关系。

- 需要说明的是：Word2vec 是上一代的产物（18 年之前）， 18 年之后想要得到最好的效果，已经不使用 Word Embedding 的方法了，所以也不会用到 Word2vec。
- 优点：
- 由于 Word2vec 会考虑上下文，跟之前的 Embedding 方法相比，效果要更好（但不如 18 年之后的方法）
- 比之前的 Embedding方 法维度更少，所以速度更快
- 通用性很强，可以用在各种 NLP 任务中

Demo

- word2vec_ex.ipynb
- word2vec_tf_ex.ipynb

Thanks
Keep Going to End

Sean, 王家祥