关于 NLP 文本分析中词向量模型的文献调研

陈鸿绪

（中国科学技术大学 大数据学院 安徽合肥）

1 背景与现状综述：

NLP（Nature Language Processing 自然语言处理）是语言学、计算机科学和人工智能交叉的子领域，关注于计算机进行编程以处理和分析大量自然语言数据。在诸多研究领域都有 NLP 任务的存在，如情绪分析，文本数据挖掘（包括文本分类、文本聚类、信息抽取等），关系语义学等。

对于 NLP 常见任务中的文本数据挖掘领域，其任务大致分为：获取语料、语料预处理（语料清洗，分词，词性标注，去停用词）、特征工程、特征选择与构造、训练模型。而在以上流程中，特征工程尤为重要，在这个过程中需要利用词向量模型提取出语料库中的文本特征，才可以语料库数据转化为机器可以学习识别的数据。词向量模型的好坏将关乎整个文本数据挖掘所能达到的高度。

现有的词向量模型有基于统计频率的传统词向量模型、基于神经网络的词向量模型、兼顾局部与全局统计信息的词向量模型等。基于统计频率的传统模型有 One-Hot（热独码模型），TF-IDF（term frequency-inverse document frequency），N-Gram 模型，共现矩阵表示模型。基于神经网络的词向量模型的典型模型有 Word2Vec 训练模型，兼顾到全局统计信息和局部特点的 Glove（Global Vector）模型等。本文将从各类模型的具有一定代表性的文章或论文入手，阐述这些模型的基本思想与内容。

2 调研结果综述

- **One Hot 模型**

热独码是所有词向量模型中最基本最简单的词向量模型，这种表示方法是将语料库中的词编码成一个多维向量，其中有且只有一个分量为 1，其余全部为 0。假设语料库中共有 n 个词，将每个词按一定顺序排列，每个词在自己所在位置上的分量为 1，其余为 0，该向量表示成[0,0,...,1,...,0,0]。

模型评价：One Hot Representation 是最简单，易于理解的词向量表达方式，然而对于比较庞大的语料库，会导致不同词的数量非常多，这时会引起向量维数相应变高，引发“维数灾难”，同时向量编码稀疏，造成矩阵稀疏，不利于空间存储，也不利于维护。同时这种表示方法很明显导致所有词语的距离都是一样，无法度量语义上的差别与相似之处，造成“语义鸿沟”。

- **TF-idf 模型**

TF-idf 模型是由康奈尔大学计算机科学教授 Salton 在论文《A vector space model for automatic indexing》[1]最先提出介绍，在这篇文章里 Salton 引用了 Spärck Jones 提出的 IDF 的数学概念[2]。对于 IDF 的数学概念，Spärck 在论文内只将其与 Zipf 经验定理相联系，其实该方法的理论基础在很长一段时间内也没有得到证明。Idf 定义如下：

$$idf = \log\left(\frac{N}{\{d \in D,\ t \in d\}}\right)$$

N 表示为总文档数，分母表示对于 t 关键词有$\{d \in D,\ t \in d\}$个文档包含该词，然而对某些词有$\{d \in D,\ t \in d\}$为 0，此时为了平滑

$$idf = \log\left(\frac{N}{\{d \in D,\ t \in d\} + 1}\right)$$

DF 可以定义为关键词 t 出现在文档中的相对频率：

$$TF = \frac{f_{t,d}}{\sum f_{t_i,d}}$$

则得到 TF-idf 的数学表达式：

$$TFidf = TF \times idf$$

$$= \log\left(\frac{N}{\{d \in D,\ t \in d\} + 1}\right) \times \frac{f_{t,d}}{\sum f_{t_i,d}}$$

特别地，TF 可以有多种定义方式[3]：

Term frequency

natural	$tf_{t,d}$
logarithm	$1+\log(tf_{t,d})$
augmented	$0.5+\frac{0.5\times tf_{t,d}}{\max_t(tf_{t,d})}$
boolean	$\begin{cases}1 & if\ tf_{t,d}>0\\0 & otherwise\end{cases}$
log *ave*	$\frac{1+\log(tf_{t,d})}{1+\log(ave_{t\in d}(tf_{t,d}))}$

对于不同的语料库，选择相适应的数学表达来处理会有更佳的效果。

模型评价：该方法简单快速，可以较好的拟合实际情况。TF-idf 方法实际上是意在压制噪声的加权方法，对于文本频率较低的词就越重要，文本频率越低的词就越越平凡，实际上这对有一些文本情况是不正确的。当在一个语料库中，同一类的文本占比很大的时候，由于该加权方法弊端性导致这一类文本特征被忽略。同时 TF-idf 方法实际上只考虑了语料库中词的出现频率，并没有考察其相邻词之间的关系和词语的位置关系。导致相同词的贡献度实际上一致。

- **N-gram 模型**[4]

N-gram 模型来源于 Claude Shannon 的一个信息论实验：给定一个字母序列，下一个字母出现的概率是多少。N-gram 模型是理解以 RNNS 为基本的神经网络语言模型的基础。即对于一个语句：$w_1,w_2,...,w_n$，如何计算出$P(w_1,w_2,...,w_n)$。首先用概率论中链式法则将$P(w_1,w_2,...,w_n)$展开：

$$P(w_{1:n})=P(w_1)P(w_2|w_1)P(w_3|w_{1:2})...P(w_n|w_{1:n-1})$$

$$=\prod_{k=1}^{n}P(w_k|w_{1:k-1})$$

引入 Markov assumption：

$$P(w_k|w_{1:k-1}) \approx P(w_k|w_{k+1-N:k-1})$$

此时对于计算该句子出现的概率只需：

$$P(w_{1:n}) = \prod_{k=1}^{n} P(w_k|w_{1:k-1}) \approx \prod_{k=1}^{n} P(w_k|w_{k+1-N:k-1})$$

为计算$P(w_k|w_{k+1-N:k-1})$，引入最大似然估计来进行计算：

$$P(w_k|w_{k+1-N:k-1}) = \frac{COUNT(w_k w_{k+1-N:k-1})}{COUNT(w_{k+1-N:k-1})}$$

N=1 的 N-gram 模型称为 unigram 模型，而一元模型可以看作是几个单状态有限状态机的组合，N=2 的 N-gram 模型 bigram 模型，N=3 的 N-gram 模型成为 trigram 模型。以下考虑 bigram 模型：

$$P(w_{1:n}) = \prod_{k=1}^{n} P(w_k|w_{1:k-1}) \approx \prod_{k=1}^{n} P(w_k|w_{k-1})$$

此时我们可以列出关于 bigram 模型词向量矩阵：

Bigram

	w_1	w_2	…	w_n
w_1	$P(w_1\|w_1)$	$P(w_2\|w_1)$	…	$P(w_n\|w_1)$
w_2	$P(w_1\|w_2)$	$P(w_2\|w_2)$	…	$P(w_n\|w_2)$
…	…	…	…	…
w_n	$P(w_1\|w_n)$	$P(w_2\|w_n)$	…	$P(w_n\|w_n)$

注意对一个句子而言开头与末尾的边界情况是需要特殊处理的，若不进行处理会在计算句子概率时有$P(w_1|w_0) = P(w_1)$，与后面的项不对称又不易写入矩阵之中，所以为了方便起见可以将语料库中每个句子的开头结尾加上标识$<s>$、$</s>$，这样可以保证矩阵每一行每一列加起来全部为 1。即如果一个句子为“I love study”，则改写之后的句子为”<s>I love study</s>”，这时只需计算：

$$P(<s>\ I\ love\ study\ </s>) = P(I|<s>)P(love|I)P(study|love)P(</s>|study)$$

如果句子过长会导致计算该句概率的时候连乘会使句子概率趋于 0，这在计算机里面对于浮点数表示会导致精度缺失，所以为了避免这种情况发生，采取取对数形式：

$$\prod_{k=1}^{n} P_k = \exp\left\{\sum_{k=1}^{n} \log P_k\right\}$$

模型评价：对于 n 而言，n 越大，则对下一个词的约束能力就越大，但是其词向量矩阵就更为稀疏。n 越小，训练语料库的代价也越小，且矩阵也一般不会非常稀疏。在实际应用中 n=2、3 的实际使用情况就已经不错，n 再大对文本分析的结果提升并不显著。N-gram 局限性在于它对长程语句的无力（n 非常大时参数会爆炸式增长），且对于语料库词量巨大时也会面临着“维度灾难”的问题。

- **Co-occurrence_matrix 模型**[5]

共现矩阵一般运用在图像分析学上，但在 NLP 领域共现矩阵也有很大的作用。相较于 n-gram 方法中取的窗口大小为 N-1，在共现矩阵中窗口大小取为 1。共现矩阵模型通过滑动窗口对语料库进行统计。比如下面的例子：

1. I enjoy music
2. I like NLP
3. I like machine learning

对以上三句话统计后共现矩阵如下：

I like enjoy machine learning NLP music 分别对应于 1-7 行与列

$$X = \begin{pmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

事实上由于共现矩阵行数列数仍然与语料库中词的数目有关，所以为了提取出最明显的特

征，可以对 X 做 PCA 降维，具体实现：

求 X 的协方差矩阵：

$$COV = \frac{1}{n} XX^T$$

对该矩阵求 SVD 分解：

$$X = U\Sigma V^T$$

U是COV的特征向量矩阵，V是COV^T的特征向量矩阵

$$\Sigma = \left(\mathrm{diag}\left(\sqrt{\lambda_1}, \sqrt{\lambda_2}, ..., \sqrt{\lambda_n}\right)\right)_{n \times m}$$

选取前 k 个特征值最大的λ_i，然后将原矩阵乘上 U 转置后的矩阵 k 个λ_i对应的向量，从而得到 X 的投影，即得到 X 的 PCA 降维。

模型评价：由于语料库会有变化，所以会有新的单词的增加或者减少，共现矩阵维数变化就会非常频繁，导致维护难度增大；由于许多词语都不是共现的，这会导致矩阵的稀疏度非常大，浪费了空间；SVD 降维代价随矩阵维数的增加而变得非常大；需要一些操作解决急剧不平衡的词语共现频率。为了解决这个问题有如下做法：忽略一些常用词，比如“he”,“she”,“and”等；对该模型应用 ramp window，即不再对窗口里的值视为同一权重，比如根据文档中词之间的距离相应改变其权重；使用皮尔逊相关稀疏取代计数。

- **Word2Vec 模型**[6]

Word2vec 是用于产生词向量的相关模型，其是在 2013 年由 Tomas Mikolov 的 Google 团队首创，传统词向量模型大多数都无法表示出词语与词语之间的相似性。这个问题可以由使用神经网络的方法结合降低维度来有效实现。Word2vec 的两种基本模型：CBOW 模型，skip-gram 模型，CBOW 利用周围词预测中间词，skip-gram 利用周围词来预测中间词。下面结合所调研的文献总结出两个模型的基本思路：

(1). CBOW 模型

CBOW 由三层神经网络组成：输入层，隐藏层，输出层。对语料库中的词语编码为热独码，输入层输入 2m 个词语对应的热独码组成的向量矩阵

$$(x^{(c-m)},...,x^{(c-1)},x^{(c+1)},...,x^{(c+m)})$$

我们需要完成的是预测$x^{(c)}$，即前 m 个词语与后 m 个词语之间的一个词的具体内容。 下设ω_i是语料库中第 i 个词语，Λ是输入层与隐藏层之间的 n×|V|矩阵，Π是隐藏层与输出层之间的|V|×n 矩阵。求解过程如下：

a. 将Λ左乘输入的热独码向量矩阵得到：

$$(v_{c-m}=\Lambda x^{(c-m)},v_{c-m+1}=\Lambda x^{(c-m+1)},...,v_{c+m}=\Lambda x^{(c+m)})$$

b. 取上面列向量的平均值：

$$\hat{v}=\frac{v_{c-m}+v_{c-m+1}+...+v_{c+m}}{2m}$$

c. 经过隐藏层和输出层得到向量：

$$z=\Pi\hat{v}$$

将这个向量经过 softmax 函数处理得到：

$$\hat{y}=soft\max(z)$$

d. 取$\hat{y}$中值最大的分量为 1，其他分量均为 0，此时得到的热独码即为对应的预测值

事实上以上是已经训练好的神经网络进行求解预测，具体训练需要通过训练集利用梯度下降方法进行参数迭代，损失函数是由交叉熵构造出的函数，即求解：

$$\min imize\ J=\ -\log P(w_c|w_{c-m},...,w_{c-1},w_{c+1},...,w_{c+m})$$

$$=-\sum_{i=1}^{|V|}y_i\log(\hat{y}_i)=-\log(\hat{y}_c)$$

$$=-\log\frac{\exp(u_c\hat{v})}{\sum_{j=1}^{|V|}\exp(u_j\hat{v})}$$

$$=-u_c\hat{v}+\log\sum_{j=1}^{|V|}\exp(u_j\hat{v})$$

(2). Sikp-Gram 模型

不同于从一些词出发预测一个词，该模型从一个词出发预测前后词语。同样的该神经网

络模型也有三层：输入层、隐藏层、输出层。输入的只有一个词的热独码词向量x。下设ω_i是语料库中第 i 个词语，Λ是输入层与隐藏层之间的 n×|V|矩阵，Π是隐藏层与输出层之间的|V|×n 矩阵。求解过程如下：

a. 降维后得到词向量：$v_c = \Lambda x$

b. 由于降维后只有一个词向量，即：$\hat{v} = v_c$

c. 通过$u = Uv_c$产生 2m 个向量：$u_{c-m},...,u_{c-1},u_{c+1},...,u_{c+m}$

将这个向量经过 softmax 函数处理得到：

$$y = soft\max(u)$$

d. 对$y^{(c-m)},...,y^{(c-1)},...,y^{(c+1)},...,y^{(c+m)}$每个求出所对应的热独码，将其与编码的词语对应得出预测结果。

事实上以上是已经训练好的神经网络进行求解预测，具体训练需要通过训练集利用梯度下降方法进行参数迭代，具体损失函数构造如下：

$$\min imize\ J = -\log P(w_{c-m},...,w_{c-1},w_{c+1},...,w_{c+m}|w_c)$$

$$= -\log\left(\prod_{j=0,j\neq m}^{2m} P(w_{c-m+j}|w_c)\right)$$

$$= -\log\left(\prod_{j=0,j\neq m}^{2m} P(u_{c-m+j}|v_c)\right)$$ 注意这里的u_i指的是输出层与隐藏层之间的矩阵的第i行

$$= -\log\left(\prod_{j=0,j\neq m}^{2m} \frac{\exp(u_{c-m+j}v_c)}{\sum_{k=1}^{|V|}\exp(u_k v_c)}\right) = -\left(\sum_{j=0,j\neq m}^{2m} u_{c-m+j}v_c\right) + 2m\log\left(\sum_{k=1}^{|V|}\exp(u_k v_c)\right)$$

事实上，可以发现，以上两种模型方法实际上在隐藏层将热独码映射到一个更低维的向量空间，在这些向量中，不难发现，对于词义相近的两个词得出的低维向量理论上会比较相近。这相较于之前传统的词向量模型有了很大的突破。

- GloVe 模型[7]

目前主流的词向量模型主要分为两种：基于全局的频率统计的词向量模型和基于局部的上下窗口的词向量模型，如 CBOW、skip-gram 模型等。对全局频率统计的模型而言，“维度灾难”是其最大弊端之一，虽然有一些降维手段，但是在维度非常大的时候，降维操作带来的代价不可估量。对基于局部频率统计的模型而言，CBOW、skip-gram 模型对于文本表示具有很大的优势，然而该类模型并没有从整体上进行考察，导致整体信息缺失。Glove 模型正是结合了两种，避开了两者局限性。Glove 模型实质上是一种全局的对数线性回归模型，由斯坦福教授 Jeffrey Pennington 于 2014 年发布的论文《Global Vectors for Word Representation》提出。下面从该篇论文调研该模型。

Glove 模型大致分为三步：

a. 根据语料库构造共现矩阵，不同于传统共现矩阵，作者构造的共现矩阵解决了对于相同窗口词语之间地位平等的弊端，而是采用了加权方式，对于词语之间的距离 d，构造衰减函数 $\frac{1}{d}$，可以发现对于距离越远的词语权重会越小

b. 作者经过一系列数学推导（笔者认为推导比较定性，数学上并不是非常严谨），得出需要构造出的词向量和和共现矩阵之间的近似关系：

$$w_i^T \tilde{w}_j + b_i + \tilde{b}_j = \log(X_{ij})$$

w_i 与 w_j 为需要构造的词向量具体值，b_i 与 $\tilde{b}_j$ 是 bias 偏置项

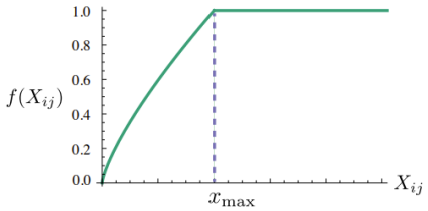
c. 对于一个语料库，直观上来说，若两个词语一起出现的频率越大，则构造损失函数的时候需要加大权重，即权重函数满足随着频率应该满足不减的函数关系，如果两个词语在语料库中没有同时在一个窗口出现过，则置该权重函数为 0。同时权重不宜过大，当频率达到一定时，应该保持稳定。从上式入手构造权重函数、损失函数：

$$J = \sum_{i,j=1}^{V} f(X_{ij}) \left(w_i^T \tilde{w}_j + b_i + \tilde{b}_j - \log(X_{ij}) \right)^2$$

论文中作者使用的权重函数为分段函数：

$$f(x) = \begin{cases} \left(\frac{x}{x_{\max}} \right)^{\alpha} & if\ x < x_{\max} \\ 1 & otherwise \end{cases}$$

在作者的实验中，其所有的α参数均为 0.75，$x^{\max}$设置为 100，下图为论文中权重函数图像：



Glove 模型训练方法：论文作者采用梯度下降进行词向量的迭代逼近。即每一步对共现矩阵进行非零的随机采样，学习速率设置成为 0.05，进行不断迭代直到收敛为止。对于最后得到的w与$\tilde{w}$理论来说由共现矩阵对称性可以确定为是对称一致的，由于初始化的不同，所以会有值的不同，但是可以通过两者相加一定程度消减随机噪声（即初始化带来的随机噪声），最终的词向量选取为$w+\tilde{w}$。作者最后通过实验得到了词向量维数为 300、窗口大小为 6-10 之间取最佳效果。

3 学习心得

对于 NPL 的文本特征分析任务而言，特征工程非常重要，我们需要进行各种方面的分析来决定选择何种词向量模型。如果语料库非常之大，则倾向于采取基于神经网络的词向量模型进行迭代，避免传统词向量模型带来的“维度灾难”，反正如果语料库规模不大，用传统词向量模型不失为一种好的选择。如果任务需求表明要体现出意思相近的词语对应的词向量也要相似，则最好采取带有窗口分析的词向量模型。如果要求模型建立在局部同时也兼顾全局，则 GloVe 模型是较好的选择。

其实这些词向量模型很多时候并没有单一的好坏之分，这还是要取决于语料库的具体实例与数据分析者的具体实现步骤。

由于时间、精力、篇幅种种限制，本篇调研文献并没有将以上词向量模型具体实现细节展示，同时还有许多更加复杂的模型也尚未加以探索列出。如今肯定也有不少此类算法尚未开源，期待后面 NLP 邻域词向量模型的进一步发展。

[参 考 文 献]

[1] Salton.G,A. vector space model for automatic indexing. November 1975

[2] Sparck Jones,K. A statistical interpretation of term specificity and its application to retrieval. J. Documen. 28, 1 (March 1972)

[3] Manning.C.D, Raghavan.P, Schutze.H. “Scoring, term weighting, and the vector space model”. Introduction to information Retrieval. p128

[4] Daniel Jurafsky. “N-gram Language Models”.Speech and Language Processing. P31

[5] Francois Chaubard, Rohit Mundra, Richard Socher.CS224D:Deep learning for NLP

[6] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space. 2013

[7] Jeffrey Pennington. Global Vectors for Word Representation. 2014