理解 NNLM

# 1.目的

如何判断一句话是否通顺合理？比如下面两句话：

A: 一群美国人抗议中餐馆杀鸡，老板反驳说你们平时不都吃鸡肉的吗，众人说我们吃的是鸡肉不是鸡，你杀鸡是侵犯鸡权

B: 中餐馆一群杀鸡美国人，，反驳你们平时不都吃鸡肉的吗老板说，说我们众人吃的不是鸡是鸡肉，侵犯鸡权你杀鸡是

我们如何用一个算法模型来判断上面哪句话比较通顺合理。

用 {W1, W2, … , Wt} 表达一句话的每个词汇， P(W1t)=P(W1, W2, … , Wt)表示这句话的概率即通顺合理的程度.

根据条件概率公式得到

P(W1, W2, … , Wt)

= P(Wt | Wt-1, … , W2, W1) \* P(Wt-1, … , W2, W1))

= P(Wt | Wt-1, … , W2, W1) \* P(Wt-1 | Wt-2, …, W2, W1) \* P(Wt-2, …, W2, W1)

= P(Wt | Wt-1, … , W2, W1) \* P(Wt-1 | Wt-2, …, W2, W1) \* … \* P(W2 | W1) \* P(W1)

P(Wt | Wt-1, … , W2, W1) 如何算呢？

先通过引入下面的假设降低 P(Wt | Wt-1, … , W2, W1) 的维数：

{Wt-1, … , W2, W1} 中越靠近 Wt 的越相关，Wt-1和Wt最相关，其次是Wt-2, 依次类推，因而我们进一步假设 Wt 只和靠近它的 n - 1 个词汇相关，即

P(Wt | Wt-1, … , W2, W1) = P(Wt | Wt-1, Wt-2, … , Wt-n+1)

n 一般取值都比 t 小不少，这样，求t个的条件概率就变成了求n个的条件概率;

以上推理就是

1. 用词汇的联合概率表达一个语句是否通顺；
2. 将计算联合概率转换为计算条件概率；
3. 将条件概率由不定长度的且一般较大的t维降到一般较小的n-1维;

# 2.模型架构

## 2.1 总架构

我们取前面描述的 n-1 的值为 6, 即取n=7, 整个词汇表词汇量为 50000，模型总架构表示如下，输入和词汇 Wt 之前的 6 个词汇，得到50000个Wt为不同词汇的概率，当然概率和为1，即 P(1) + P(2) + … + P(50000) = 1，这里1, 2, … , 50000 是词汇编号.



图2.1 总架构图

## 2.2 词汇编码(特征表达)

2.2.1 one-hot

如图2.1所示，{Wt-1, Wt-2, … , Wt-6} 是模型输入，我们需要用程序实现模型，one-hot即是一种用向量表示词汇的表达方式，如表2.1所示，一个词汇对应一个50000维向量，向量中只有一个1，其余都是0，所以叫one-hot.

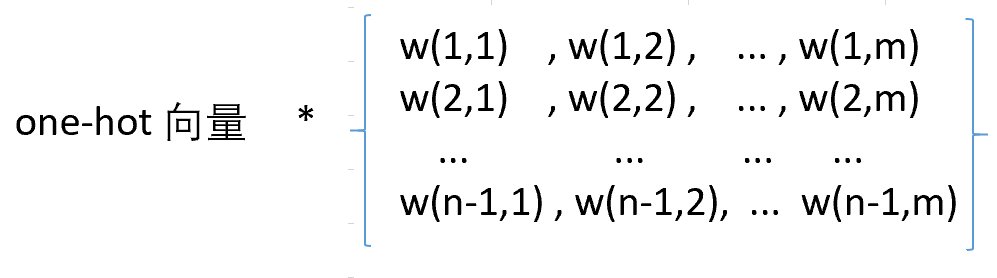
表2.1 one-hot 示例

|  |  |  |
| --- | --- | --- |
| 词汇 | 词汇库中编码 | 向量表示 |
| 众人 | 1 |  |
| 反驳 | 2 |  |
| … | … | … |
| 美国 | 50000 |  |

2.2.2 特征降维

用one-hot 表示 {Wt-1, Wt-2, … , Wt-6} 需要 6 x 50000 = 300000 维度，这是不可接受的。NNLM作者用如下的方式将语句对应的one-hot特征向量降维, one-hot 矩阵(向量) 右乘权重矩阵, 则 one-hot 向量就转换成了{ w(i,1), w(i,2), … , w(i,m) }, m即是新的特征维度，一般远小于 50000.n 比如 one-hot向量为 (0, 1, … , 0, 0) ，那么转换得到的特征向量就为

{ w(2,1) , w(2,2) , … , w(2,m) }，这个转换得到特征向量



这个矩阵乘法可以简化，直接用序号表示词汇，用序号作为转换矩阵的行序号索引得到的行向量就是所求的特征向量, 用C表示转换矩阵，则词汇Wi的词向量为C(Wi).

## 2.4 模型结构

NNLM 的模型结构如图2.2所示，前面已经叙述由one-hot向量转换到词向量；如图，每个m维度的词向量按顺序前后相接合并成一个 (n-1)m 维向量 x . 向量 x 作为神经网络隐含层的输入, 细节上说就是对 x 的每个元素做加权求和，加上偏移后作为神经元激活函数tanh输入, 不同神经元对应加权求和权重值不同，具体参考下网上前馈神经网络讲解;

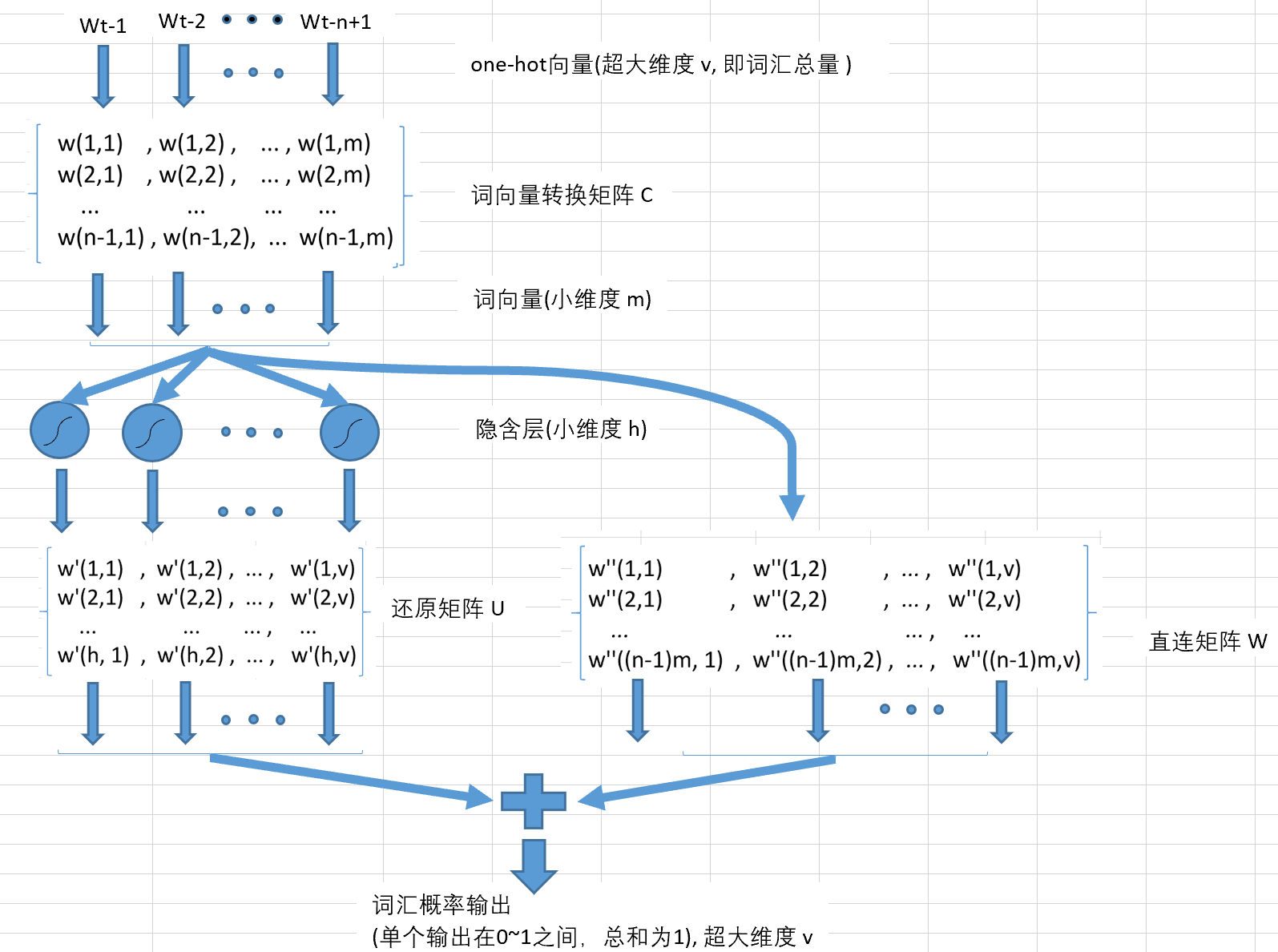


图2.2 模型结构

每个神经元激活函数 tanh 输出值组成h维向量(h x 1 矩阵) 右乘(图中为了画图方便其实是左乘U，原始论文中U实际是图中U的转置，读者意会即可，不影响理解)还原矩阵 U 即得到 v x 1 维的输出向量，这个向量的每个元素做 softmax 归一化处理后就是每个词汇对应的概率。这个过程用数学表达如下

U \* tanh(d + H \* x)

其中 x 是合并后词向量, H 是连接隐含层神经元权重矩阵, d是连接神经元偏移值矩阵;

上述过程其实就可根据目标词汇附近词汇计算出目标词汇的概率，即和邻近词汇放在一起组成一句话的合理性，但原始论文作者其实在网络结构上加了直连，如图右侧所示，合并词向量不经过隐含层，直接右乘直连矩阵 W 得到 v x 1 维输出后与前述的 v x 1 维输出向相加，得到一个最终的 v x 1 维输出向量。这个直连结构按照网上看到的说法是对模型效果提升不大，但是能加快模型的训练速度。这个过程用数学表达如下：

b + W \* x

其中, W 是直连矩阵, b是偏移向量;

由上，sotmax 归一化前的结果输出向量如下：

y = b + W \* x + U \* tanh(d + H \* x)

其中，根据 2.2.2 所述， x = ( C(Wt-1), C(Wt-2), … , C(Wt-n+1) ) .

# 2.5 训练

涉及神经网络的模型训练一般是用梯度下降法，这个我也只是知道个大概原理，具体实现也是当前一直在发展的一个方向，常用的梯度下降法实现有Adam等等，整理就不展开了。使用梯度下降法就得定义损失函数:

Loss =

y(t) 词汇t的概率, y(t)越大，logy(t) 越小，则Loss越小，R是关于要训练的参数的正则项函数(我也看不懂这个正则项函数是啥，也没有查)

# 说明

1. 转载请注明出处，加上链接；
2. 无耻小广告：作者目前在某大厂做图像处理算法(ISP)和智能算法，想跳槽，有招聘意向私信；