一文读懂NLP中的语言模型（附代码）

小明 小明AI学习 2017-11-14

导读

本文的大部分知识点整理于Michael Collins 教授的Language Modeling ，文末会附上我实现的python代码github链接（github上会有较详细的代码说明，欢迎follow & star & fork 三部曲）

本文主要分四部分讲解：

1. 通俗地讲解 “什么是语言模型”
2. 什么是“马尔可夫模型”
3. 通过例子讲述“如何实地操作”
4. 如何评估模型

语言模型广泛用于各种自然语言处理问题，最早是用于语音识别的。
通俗地说，语言模型是用来计算一个句子的概率，利用语言模型可以确定哪个句子发生的概率更大，或者给定若干词，可以预测下一个词。
举个例子吧：假设你用质量不错的语料库（“质量不错”可以这样认为：语料库中的句子是通顺的，没有错别字的，且内容覆盖面广的）训练了一个语言模型，接下来你要计算给定句子的发生的概率，比如说有两个句子：

1. 小明 爱 学习
2. 小明 学 爱 习

从肉眼看，肯定是句子1发生的概率大，语言模型可能会得出两个句子发生的概率

$$p(\text{小明 爱 学习}) > p(\text{小明 学 爱 习})$$

那么问题来了如何计算一个句子的概率呢？以sentence = “小明 爱 学习”为例：

```
p ( sentence )
= p ( 小明, 爱, 学习 )
= p ( 小明 ) * p ( 爱 | 小明 ) * p ( 学习 | 小明, 爱)
```

下面给出**语言模型的定义**：
1. 首先， 定义一个词典V，假如 V = {the, dog, laughs, saw, barks, cat, ...}, 在这里我们假设V是有限的
2. 那么由词典组成的字符串（是任意的组合,不管通不通顺）是无限的, 假设这些字符串的集合为V+，比如可能是：

```
a. the dog barks STOP
b. the cat laughs STOP
c. the cat saw the dog STOP
d. the STOP
.....
(STOP是人为约定的结束符，标记句子的结束)
```

由上面两点可以给出定义：
一个语言模型包含一个有限的V，以及一个函数

$$p(x1, x2, ...xn), x1..xn \in V$$

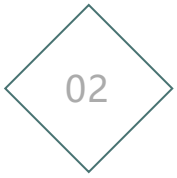
1. 对于任意

$$<x1x2...xn> \in V+, p (x1, x2, ...xn) >= 0$$

2.

$$\Sigma(<x1x2..xn> \in V+)p(x1, x2, ... , xn) = 1$$

因此，p(x1, x2, ..., xn)是V+上的概率分布



马尔可夫模型

Markov Models

读了上文后，相信你已经对语言模型已经有了一定的认识了，那么现在问题来了，上面的语言模型能解决实际问题吗？假如给了你一个庞大的训练语料，可以想象如果从整个句子角度来考虑的话，训练语料中那么将有大部分的句子得到 p(x) = 0，这是非常稀疏的，不利于模型的训练及存储。

接下来将讲解n元的马尔科夫模型如何解决这个问题。

对于一个句子

$$sentence = x1x2x3...xn\ (n>=1,\ xi \in V)$$

则我们的目的是要求

$$p(X1=x1,\ X2 = x2,\ ...,\ Xn = xn)$$

可知

$$p(X1=x1,\ X2 = x2,\ ...,\ Xn = xn)$$
$$= P(X1 = x1)\ \Pi P(Xn = xn \mid X1 = x1,\ .\ .\ .\ ,\ Xn-1 = xn-1)$$

可知随着n的增大计算难度也会随着增大，那么我们有没有好的解决方呢？肯定是有的！

我们可以假设当前词出现的概率仅仅于前面n-1个词有关，叫 **(n-1) 阶**马尔科夫模型—**n-gram**，n一般取1，2，3

1. **n = 1** 时，为零阶马尔科夫模型，一般称为**unigram**，此时式子可以简化为：

$$p\ (X1=x1,\ X2 = x2,\ ...,\ Xn = xn) = \Pi\ p\ (Xn = xn\ )$$

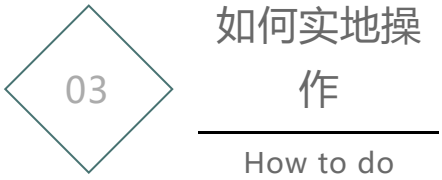
2. **n = 2**时, 为一阶马尔科夫模型, 一般称为**bigram**，此时式子可以简化为：

$$p\ (X1{=}x1,\ X2\ =\ x2,\ \ldots,\ Xn\ =\ xn)\ =\ \Pi\ p\ (Xn\ =\ xn\ |\ Xn{-}1\ =\ xn{-}1)$$

3. **n = 3**时，为二阶马尔科夫模型, 一般称为**trigram**，此时式子可以简化为：

$$p\ (X1{=}x1,\ X2\ =\ x2,\ \ldots,\ Xn\ =\ xn)$$
$$=\ \Pi\ p\ (Xn\ =\ xn\ |\ Xn\text{-}2\ =\ xn\text{-}2,\ Xn{-}1\ =\ xn{-}1)$$

好了，有了n-gram 是不是这时候计算就好算多了。



有了上面的理论知识，那么我们具体该如何计算呢？**下面以trigram为例讲解。**

首先用频率来作为概率值：

$$p\ (Xn\ =\ xn\ |\ Xn\text{-}2\ =\ xn\text{-}2,\ Xn{-}1\ =\ xn{-}1)$$
$$=\ q\ (Xn\ =\ xn\ |\ Xn\text{-}2\ =\ xn\text{-}2,\ Xn{-}1\ =\ xn{-}1)$$

接下来我们可以用**最大似然估计(**Maximum-Likelihood Estimates)来求函数q。
现在假设有三个词u v w

$c\ (u,\ v,\ w)$ 为三元组$(u,\ v,\ w)$ 出现的次数
$c\ (u,\ v)$ 为二元组$(u,\ v)$ 出现的次数
...
$c\ ()$ 为字典的大小

那么trigram有：

$$q(w \mid u, v) = c(u , v, w) / c(u, v)$$

bigram有：

$$q(w \mid v) = c(v, w) / c(v)$$

unigram有：

$$q(w) = c(w) / c()$$

好了，有了上面这个假设，我们就可以实际操作了，在这里，举个例子帮助大家更好的理解整个过程是怎样的。**下面还是以trigram为例**

1. 假设你有语料库：

```
你 好 STOP
你 好 吗 STOP
好 了 吗 STOP
不 好 了 STOP
```

2. 我们需要对语料库进行统计次数操作了，因为要实现trigram，所以我们需要统计三元组 (u, v, w) 和二元组 (u, v) 出现的次数
为了理解下面的数据，我们先给个单例：
“ 你 好STOP ” 二元组的情况为 (你, 好), (好，STOP)

#	二元组	次数
1	(你，好)	2
2	(好, STOP)	1
3	(好, 吗)	1
4	(吗, STOP)	2

5	(好, 了)	2
6	(了, 吗)	1
7	(不, 好)	1
8	(了, STOP)	1

#	三元组	次数
1	(你, 好, STOP)	1
2	(你, 好, 吗)	1
3	(好, 吗, STOP)	1
4	(好, 了, 吗)	1
5	(了, 吗, STOP)	1
6	(不, 好, 了)	1
7	(好,了,STOP)	1

好了有了上面的数据，我们可以计算q了
比如现在有一个句子 “ 你好 ”

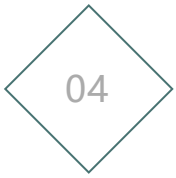
```
q(STOP| 你, 好)
= c(你 , 好, STOP) / c(你, 好)
= 1 / 2 = 0.5
```

上面只是一个很简单的例子，主要是帮助大家理解实际情况下怎么操作。
从例子中不知道你有没有发现新的问题呢？就拿上面的例子来说吧，比如说我要算 “ 你了 ” ，
也就是：

```
q (STOP| 你, 了)
```

```
= c (你，了, STOP) / c(你, 了)
= 0 / 0
= ?
```

问题出来了吧，二元组(u, v)不存在，也就是分母为零怎么办？接下来我们就要引入一些**平滑(smooth)的方法**去应对这种情况了，比如说给分子分母加上一个常数δ（0<δ≤1），
在这里就不过多讲解平滑方法了，只列举一些：加法平滑、线性插值、katz平滑...更多方法大家自行学习了



如何评估

How to evaluate

在这里，引入困惑度（Perplexity）评估方法，困惑度计算方法为

$$2^{(-l)}$$
$$l = 1/M \left(\Sigma_{(i->m)} \log_2 p(x(i)) \right)$$

其中M代表语料库中词的总数，m代表语料中总共含有的句子数目x(i) 代表第i个句子

后记

点击下面原文链接即可到达github项目，现在实现的主要是统计自然处理的language model， 过段时间会基于CNN/RNN实现，还请关注！

长按，识别二维码，加关注