

刘建平Pinard

十年研发，对数学统计学，数据挖掘，机器学习，大数据平台，大数据平台应用开发，大数据可视化感兴趣。

博客园 首页 新随笔 联系 订阅 管理

word2vec原理(二) 基于Hierarchical Softmax的模型

[word2vec原理\(一\) CBOW与Skip-Gram模型基础](#)

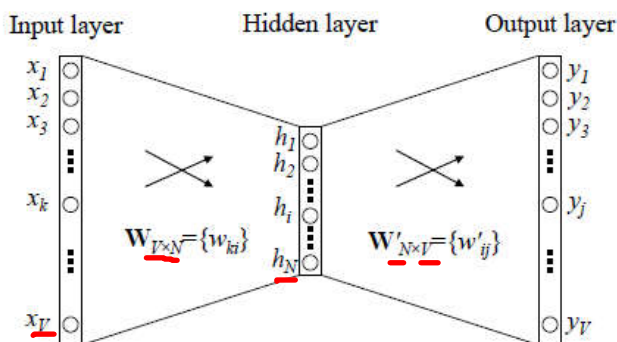
[word2vec原理\(二\) 基于Hierarchical Softmax的模型](#)

[word2vec原理\(三\) 基于Negative Sampling的模型](#)

在word2vec原理(一) CBOW与Skip-Gram模型基础中，我们讲到了使用神经网络的方法来得到词向量语言模型的原理和一些问题，现在我们开始关注word2vec的语言模型如何改进传统的神经网络的方法。由于word2vec有两种改进方法，一种是基于Hierarchical Softmax的，另一种是基于Negative Sampling的。本文关注于基于Hierarchical Softmax的改进方法，在下一篇讨论基于Negative Sampling的改进方法。

1. 基于Hierarchical Softmax的模型概述

我们先回顾下传统的神经网络词向量语言模型，里面一般有三层，输入层（词向量），隐藏层和输出层（softmax层）。里面最大的问题在于从隐藏层到输出的softmax层的计算量很大，因为要计算所有词的softmax概率，再去找概率最大的值。这个模型如下图所示。其中 V 是词汇表的大小，

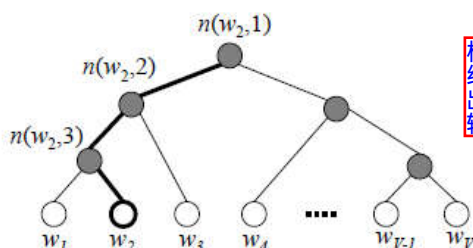


word2vec对这个模型做了改进，首先，对于从输入层到隐藏层的映射，没有采取神经网络的线性变换加激活函数的方法，而是采用简单的对所有输入词向量求和并取平均的方法。比如输入的是三个4维词向量：

$(1, 2, 3, 4), (9, 6, 11, 8), (5, 10, 7, 12)$ ，那么我们word2vec映射后的词向量就是 $(5, 6, 7, 8)$ 。由于这里是从多个词向量变成了一个词向量。

第二个改进就是从隐藏层到输出的softmax层这里的计算量改进。为了避免要计算所有词的softmax概率，word2vec采样了霍夫曼树来代替从隐藏层到输出softmax层的映射。我们在上一节已经介绍了霍夫曼树的原理。如何映射呢？这里就是理解word2vec的关键所在了。

由于我们把之前所有都要计算的从输出softmax层的概率计算变成了一颗二叉霍夫曼树，那么我们的softmax概率计算只需要沿着树形结构进行就可以了。如下图所示，我们可以沿着霍夫曼树从根节点一直走到我们的叶子节点的词 w_2 。



和之前的神经网络语言模型相比，我们的霍夫曼树的所有内部节点就类似之前神经网络隐藏层的神经元，其中，根节点的词向量对应我们的投影后的词向量，而所有叶子节点就类似于之前神经网络softmax输出层的神经元，叶子节点的个数就是词汇表的大小。在霍夫曼树中，隐藏层到输出层的softmax映射不是一下子完成的，而是沿着霍夫曼树一步步完成的，因此这种softmax取名为"Hierarchical Softmax"。

公告

★珠江追梦，饮岭南茶，恋鄂北家★

昵称：刘建平Pinard

园龄：2年

粉丝：2385

关注：15

+加关注

随笔分类(120)

- 0040. 数学统计学(4)
- 0081. 机器学习(69)
- 0082. 深度学习(11)
- 0083. 自然语言处理(23)
- 0084. 强化学习(11)
- 0121. 大数据挖掘(1)
- 0122. 大数据平台(1)

随笔档案(120)

- 2018年10月 (3)
- 2018年9月 (3)
- 2018年8月 (4)
- 2018年7月 (3)
- 2018年6月 (3)
- 2018年5月 (3)
- 2017年8月 (1)
- 2017年7月 (3)
- 2017年6月 (8)
- 2017年5月 (7)
- 2017年4月 (5)
- 2017年3月 (10)
- 2017年2月 (7)
- 2017年1月 (13)
- 2016年12月 (17)
- 2016年11月 (22)
- 2016年10月 (8)

常用的机器学习网站

52 NLP
Analytics Vidhya
机器学习库
机器学习路线图
强化学习入门书
深度学习进阶书
深度学习入门书

积分与排名

积分 - 353402
排名 - 547

阅读排行榜

如何“沿着霍夫曼树一步步完成”呢？在word2vec中，我们采用了二元逻辑回归的方法，即规定沿着左子树走，那么就是负类(霍夫曼树编码1)，沿着右子树走，那么就是正类(霍夫曼树编码0)。判别正类和负类的方法是使用sigmoid函数，即：

$$P(+) = \sigma(x_w^T \theta) = \frac{1}{1 + e^{-x_w^T \theta}}$$

其中 x_w 是当前内部节点的词向量，而 θ 则是我们需要从训练样本求出的逻辑回归的模型参数。

使用霍夫曼树有什么好处呢？首先，由于是二叉树，之前计算量为 V ，现在变成了 $\log_2 V$ 。第二，由于使用霍夫曼树是高频的词靠近树根，这样高频词需要更少的时间会被找到，这符合我们的贪心优化思想。

容易理解，被划分为左子树而成为负类的概率为 $P(-) = 1 - P(+)$ 。在某一个内部节点，要判断是沿左子树还是右子树走的标准就是看 $P(-)$ ， $P(+)$ 谁的概率值大。而控制 $P(-)$ ， $P(+)$ 谁的概率值大的因素一个是当前节点的词向量，另一个是当前节点的模型参数 θ 。

对于上图中的 w_2 ，如果它是一个训练样本的输出，那么我们期望对于里面的隐藏节点 $n(w_2, 1)$ 的 $P(-)$ 概率大， $n(w_2, 2)$ 的 $P(-)$ 概率大， $n(w_2, 3)$ 的 $P(+)$ 概率大。

回到基于Hierarchical Softmax的word2vec本身，我们的目标就是找到合适的所有节点的词向量和所有内部节点 θ ，使训练样本达到最大似然。那么如何达到最大似然呢？

2. 基于Hierarchical Softmax的模型梯度计算

我们使用最大似然法来寻找所有节点的词向量和所有内部节点 θ 。先拿上面的 w_2 例子来看，我们期望最大化下面的似然函数：

$$\prod_{i=1}^3 P(n(w_i), i) = (1 - \frac{1}{1 + e^{-x_w^T \theta_1}}) (1 - \frac{1}{1 + e^{-x_w^T \theta_2}}) \frac{1}{1 + e^{-x_w^T \theta_3}}$$

对于所有的训练样本，我们期望最大化所有样本的似然函数乘积。

为了便于我们后面一般化的描述，我们定义输入的词为 w ，其从输入层词向量求和平均后的霍夫曼树根节点词向量为 x_w ，从根节点到 w 所在的叶子节点，包含的节点总数为 l_w ， w 在霍夫曼树中从根节点开始，经过的第 i 个节点表示为 p_i^w ，对应的霍夫曼编码为 $d_i^w \in \{0, 1\}$ ，其中 $i = 2, 3, \dots, l_w$ 。而该节点对应的模型参数表示为 θ_i^w ，其中 $i = 1, 2, \dots, l_w - 1$ ，没有 $i = l_w$ 是因为模型参数仅仅针对于霍夫曼树的内部节点。

定义 w 经过的霍夫曼树某一个节点 j 的逻辑回归概率为 $P(d_j^w | x_w, \theta_{j-1}^w)$ ，其表达式为：

$$P(d_j^w | x_w, \theta_{j-1}^w) = \begin{cases} \sigma(x_w^T \theta_{j-1}^w) & d_j^w = 0 \\ 1 - \sigma(x_w^T \theta_{j-1}^w) & d_j^w = 1 \end{cases}$$

那么对于某一个目标输出词 w ，其最大似然为：

$$\prod_{j=2}^{l_w} P(d_j^w | x_w, \theta_{j-1}^w) = \prod_{j=2}^{l_w} [\sigma(x_w^T \theta_{j-1}^w)]^{1-d_j^w} [1 - \sigma(x_w^T \theta_{j-1}^w)]^{d_j^w}$$

在word2vec中，由于使用的是随机梯度上升法，所以并没有把所有样本的似然乘起来得到真正的训练集最大似然，仅仅每次只用一个样本更新梯度，这样做的目的是减少梯度计算量。这样我们可以得到 w 的对数似然函数 L 如下：

$$L = \log \prod_{j=2}^{l_w} P(d_j^w | x_w, \theta_{j-1}^w) = \sum_{j=2}^{l_w} ((1 - d_j^w) \log[\sigma(x_w^T \theta_{j-1}^w)] + d_j^w \log[1 - \sigma(x_w^T \theta_{j-1}^w)])$$

要得到模型中 w 词向量和内部节点的模型参数 θ ，我们使用梯度上升法即可。首先我们求模型参数 θ_{j-1}^w 的梯度：

$$\frac{\partial L}{\partial \theta_{j-1}^w} = (1 - d_j^w) \frac{(\sigma(x_w^T \theta_{j-1}^w)(1 - \sigma(x_w^T \theta_{j-1}^w))}{\sigma(x_w^T \theta_{j-1}^w)} x_w - d_j^w \frac{(\sigma(x_w^T \theta_{j-1}^w)(1 - \sigma(x_w^T \theta_{j-1}^w))}{1 - \sigma(x_w^T \theta_{j-1}^w)} x_w \quad (1)$$

$$= (1 - d_j^w)(1 - \sigma(x_w^T \theta_{j-1}^w)) x_w - d_j^w \sigma(x_w^T \theta_{j-1}^w) x_w \quad (2)$$

$$= (1 - d_j^w - \sigma(x_w^T \theta_{j-1}^w)) x_w \quad (3)$$

如果大家看过之前写的逻辑回归原理小结，会发现这里的梯度推导过程基本类似。

同样的方法，可以求出 x_w 的梯度表达式如下：

$$\frac{\partial L}{\partial x_w} = \sum_{j=2}^{l_w} (1 - d_j^w - \sigma(x_w^T \theta_{j-1}^w)) \theta_{j-1}^w$$

有了梯度表达式，我们就可以用梯度上升法进行迭代来一步步的求解我们需要的所有的 θ_{j-1}^w 和 x_w 。

3. 基于Hierarchical Softmax的CBOW模型

由于word2vec有两种模型：CBOW和Skip-Gram，我们先看看基于CBOW模型时，Hierarchical Softmax如何使用。

首先我们要定义词向量的维度大小 M ，以及CBOW的上下文大小 $2c$ ，这样我们对于训练样本中的每一个词，其前面的 c 个词和后面的 c 个词作为CBOW模型的输入，该词本身作为样本的输出，期望softmax概率最大。

1. 梯度下降 (Gradient Descent) (98)
2. 梯度提升树(GBDT)
3. 线性判别分析LDA
4. word2vec原理(-模型基础(62711))
5. scikit-learn决策(29)

评论排行榜

1. 梯度提升树(GBDT)原理小结(225)
2. 集成学习之Adaboost算法原理小结(121)
3. 谱聚类 (spectral clustering) 原理总结(112)
4. 梯度下降 (Gradient Descent) 小结(104)
5. word2vec原理(二) 基于Hierarchical Softmax的模型(98)

推荐排行榜

1. 梯度下降 (Gradient Descent) 小结(60)
2. 奇异值分解(SVD)原理与在降维中的应用(35)
3. 集成学习原理小结(21)
4. 卷积神经网络(CNN)反向传播算法(20)
5. 梯度提升树(GBDT)原理小结(19)

在 CBOW 模型中，输出层是 hierarchical softmax，以霍夫曼树的叶子节点表示输出单词。用霍夫曼树使得叶子节点的输出概率，累加和等于1。

在做CBOW模型前，我们需要先将词汇表建立成一颗霍夫曼树。

对于从输入层到隐藏层（投影层），这一步比较简单，就是对 w 周围的 $2c$ 个词向量求和取平均即可，即：

$$x_w = \frac{1}{2c} \sum_{i=1}^{2c} x_i$$

第二步，通过梯度上升法来更新我们的 θ_{j-1}^w 和 x_w ，注意这里的 x_w 是由 $2c$ 个词向量相加而成，我们做梯度更新完后会用梯度项直接更新原始的各个 $x_i (i = 1, 2, \dots, 2c)$ ，即：

$$\theta_{j-1}^w = \theta_{j-1}^w + \eta(1 - d_j^w - \sigma(x_w^T \theta_{j-1}^w)) x_w$$

$$x_w = x_w + \eta \sum_{j=2}^{l_w} (1 - d_j^w - \sigma(x_w^T \theta_{j-1}^w)) \theta_{j-1}^w \quad (i = 1, 2, \dots, 2c)$$

其中 η 为梯度上升法的步长。

这里总结下基于Hierarchical Softmax的CBOW模型算法流程，梯度迭代使用了随机梯度上升法：

输入：基于CBOW的语料训练样本，词向量的维度大小 M ，CBOW的上下文大小 $2c$ ，步长 η

输出：霍夫曼树的内部节点模型参数 θ ，所有的词向量 w

1. 基于语料训练样本建立霍夫曼树。
2. 随机初始化所有的模型参数 θ ，所有的词向量 w
3. 进行梯度上升迭代过程，对于训练集中的每一个样本 $(context(w), w)$ 做如下处理：

a) $e=0$ ，计算 $x_w = \frac{1}{2c} \sum_{i=1}^{2c} x_i$

b) for $j = 2$ to l_w ，计算：

$$f = \sigma(x_w^T \theta_{j-1}^w)$$

$$g = (1 - d_j^w - f) \eta$$

$$e = e + g \theta_{j-1}^w$$

$$\theta_{j-1}^w = \theta_{j-1}^w + g x_w$$

c) 对于 $context(w)$ 中的每一个词向量 x_i (共 $2c$ 个) 进行更新：

$$x_i = x_i + e$$

d) 如果梯度收敛，则结束梯度迭代，否则回到步骤3继续迭代。

4. 基于Hierarchical Softmax的Skip-Gram模型

现在我们先看看基于Skip-Gram模型时，Hierarchical Softmax如何使用。此时输入的只有一个词 w ，输出的为 $2c$ 个词向量 $context(w)$ 。

我们对于训练样本中的每一个词，该词本身作为样本的输入，其前面的 c 个词和后面的 c 个词作为了Skip-Gram模型的输出，期望这些词的softmax概率比其他的词大。

Skip-Gram模型和CBOW模型其实是反过来的，在上一篇已经讲过。

在做CBOW模型前，我们需要先将词汇表建立成一颗霍夫曼树。

对于从输入层到隐藏层（投影层），这一步比CBOW简单，由于只有一个词，所以，即 x_w 就是词 w 对应的词向量。

第二步，通过梯度上升法来更新我们的 θ_{j-1}^w 和 x_w ，注意这里的 x_w 周围有 $2c$ 个词向量，此时如果我们期望 $P(x_i|x_w), i = 1, 2, \dots, 2c$ 最大。此时我们注意到由于上下文是相互的，在期望 $P(x_i|x_w), i = 1, 2, \dots, 2c$ 最大化的同时，反过来我们也期望 $P(x_w|x_i), i = 1, 2, \dots, 2c$ 最大。那么是使用 $P(x_i|x_w)$ 好还是 $P(x_w|x_i)$ 好呢，word2vec使用了后者，这样做的好处就是在一个迭代窗口内，我们不是只更新 x_w 一个词，而是 $x_i, i = 1, 2, \dots, 2c$ 共 $2c$ 个词。这样整体的迭代会更加的均衡。因为这个原因，Skip-Gram模型并没有和CBOW模型一样对输入进行迭代更新，而是对 $2c$ 个输出进行迭代更新。

这里总结下基于Hierarchical Softmax的Skip-Gram模型算法流程，梯度迭代使用了随机梯度上升法：

输入：基于Skip-Gram的语料训练样本，词向量的维度大小 M ，Skip-Gram的上下文大小 $2c$ ，步长 η

输出：霍夫曼树的内部节点模型参数 θ ，所有的词向量 w

1. 基于语料训练样本建立霍夫曼树。
2. 随机初始化所有的模型参数 θ ，所有的词向量 w ，
3. 进行梯度上升迭代过程，对于训练集中的每一个样本 $(w, context(w))$ 做如下处理：

输入为一个单词向量，经过 continuous projection layer，输入到 log-linear classifier，得到前 c 、后 c 个单词。

a) for $i = 1$ to $2c$:

i) $e = 0$

ii) for $j = 2$ to l_w , 计算:

$$f = \sigma(x_i^T \theta_{j-1}^w)$$

$$g = (1 - d_j^w - f)\eta$$

$$e = e + g\theta_{j-1}^w$$

$$\theta_{j-1}^w = \theta_{j-1}^w + gx_i$$

iii)

$$x_i = x_i + e$$

b) 如果梯度收敛, 则结束梯度迭代, 算法结束, 否则回到步骤a继续迭代。

5. Hierarchical Softmax的模型源码和算法的对应

这里给出上面算法和word2vec源码中的变量对应关系。

在源代码中, 基于Hierarchical Softmax的CBOW模型算法在435-463行, 基于Hierarchical Softmax的Skip-Gram的模型算法在495-519行。大家可以对着源代码再深入研究下算法。

在源代码中, `neule`对应我们上面的 e , `syn0`对应我们的 x_w , `syn1`对应我们的 θ_{j-1}^i , `layer1_size`对应词向量的维度, `window`对应我们的 c 。

另外, `vocab[word].code[d]`指的是, 当前单词`word`的, 第 d 个编码, 编码不含Root结点。`vocab[word].point[d]`指的是, 当前单词`word`, 第 d 个编码下, 前置的结点。


以上就是基于Hierarchical Softmax的word2vec模型, 下一篇我们讨论基于Negative Sampling的word2vec模型。

(欢迎转载, 转载请注明出处。欢迎沟通交流: liujianping-ok@163.com)

分类: 0083. 自然语言处理

标签: 自然语言处理





刘建平Pinard
关注 - 15
粉丝 - 2385
[+加关注](#)

10

0

« 上一篇: word2vec原理(一) CBOW与Skip-Gram模型基础

» 下一篇: word2vec原理(三) 基于Negative Sampling的模型

posted @ 2017-07-27 17:26 刘建平Pinard 阅读(26115) 评论(98) 编辑 收藏

< Prev 1 2

评论列表

51楼 2018-04-15 08:37 whoisleilei

@ 刘建平Pinard
谢谢刘老师, 我再学习一下。

支持(0) 反对(0)

52楼 2018-05-29 00:34 super_super

您好, 我想请教下这两种方法得到的模型是一致的吗? 我的理解是通过训练得到的CBOW模型可用于预测几个词的中心词; 通过Skip-Gram模型可得到某个词上下文相关的若干个词, 二者应该不能反过来用吧? 还有一个问题是, 训练出每个词的词向量后, 是通过什么方法得到每个词最相似的 n 个词的呢? 使用余弦定理还是比较欧式距离或者有更高效的方法?

支持(0) 反对(0)