## 刘重平Pinard

十年码农,对数学统计学,数据控据,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。

事客回 自见 玩派

### word2vec原理(三) 基于Negative Sampling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

在上一篇中我们讲到了基于Hierarchical Softmax的word2vec模型,本文我们我们再来看看另一种求解word2vec 模型的方法: Negative Sampling。

## 1. Hierarchical Softmax的缺点与改进

在讲基于Negative Sampling的word2vec模型前,我们先看看Hierarchical Softmax的的缺点。的确,使用霍夫曼 树来代替传统的神经网络,可以提高模型训练的效率。但是如果我们的训练样本里的中心词w是一个很生僻的词,那么就得在 霍夫曼树中辛苦的向下走很久了。能不能不用搞这么复杂的一颗霍夫曼树,将模型变的更加简单呢?

Negative Sampling就是这么一种求解word2vec模型的方法,它摒弃了霍夫曼树,采用了Negative Sampling(负 采样)的方法来求解,下面我们就来看看Negative Sampling的求解思路。

# 2. 基于Negative Sampling的模型概述

既然名字叫Negative Sampling(负采样),那么肯定使用了采样的方法。采样的方法有很多种,比如之前讲到的大 名鼎鼎的MCMC。我们这里的Negative Sampling采样方法并没有MCMC那么复杂。

比如我们有一个训练样本,中心词是w,它周围上下文共有2c个词,记为context(w)。由于这个中心词w,的确和 context(w)相关存在,因此它是一个真实的正例。通过Negative Sampling采样,我们得到neg个和w不同的中心词  $w_i, i=1,2,\dots neg$ ,这样context(w)和\$\$ $w_i$ \$就组成了neg个并不真实存在的负例。利用这一个正例和neg个负例,我 们进行二元逻辑回归,得到负采样对应每个词\$w\_i\$对应的模型参数\$\theta\_{i}\$, 和每个词的词向量。

从上面的描述可以看出,Negative Sampling由于没有采用霍夫曼树,每次只是通过采样neg个不同的中心词做负 例,就可以训练模型,因此整个过程要比Hierarchical Softmax简单。

不过有两个问题还需要弄明白: 1) 如果通过一个正例和neg个负例进行二元逻辑回归呢? 2) 如何进行负采样呢? 我们在第三节讨论问题1,在第四节讨论问题2.

# 3. 基于Negative Sampling的模型梯度计算

Negative Sampling也是采用了二元逻辑回归来求解模型参数,通过负采样,我们得到了neg个负例  $(context(w), w_i)i = 1, 2, \dots neg$ 。为了统一描述,我们将正例定义为 $w_0$ 。

在逻辑回归中, 我们的正例应该期望满足:

$$P(context(w_0), w_i) = \sigma(x_{w_0}^T heta^{w_i}), y_i = 1, i = 0$$

我们的负例期望满足:

$$P(context(w_0), w_i) = 1 - \sigma(x_{w_0}^T heta^{w_i}), y_i = 0, i = 1, 2, \dots neg$$

我们期望可以最大化下式:

$$\prod_{i=0}^{neg} P(context(w_0), w_i) = \sigma(x_{w_0}^T heta^{w_0}) \prod_{i=1}^{neg} (1 - \sigma(x_{w_0}^T heta^{w_i}))$$

利用逻辑回归和上一节的知识,我们容易写出此时模型的似然函数为:

$$\prod_{i=0}^{neg} \sigma(x_{w_0}^T heta^{w_i})^{y_i} (1 - \sigma(x_{w_0}^T heta^{w_i}))^{1-y_i}$$

此时对应的对数似然函数为:

$$L = \sum_{i=0}^{neg} y_i log(\sigma(x_{w_0}^T heta^{w_i})) + (1-y_i) log(1-\sigma(x_{w_0}^T heta^{w_i}))$$

#### 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)

#### 常去的机器学习网站

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

和Hierarchical Softmax类似,我们采用随机梯度上升法,仅仅每次只用一个样本更新梯度,来进行迭代更新得到我们需要的 $x_{w_i}, \theta^{w_i}, i=0,1,\dots neg$ ,这里我们需要求出 $x_{w_0}, \theta^{w_i}, i=0,1,\dots neg$ 的梯度。

首先我们计算 $\theta^{w_i}$ 的梯度:

$$\frac{\partial L}{\partial \theta^{w_i}} = y_i (1 - \sigma(x_{w_0}^T \theta^{w_i})) x_{w_0} - (1 - y_i) \sigma(x_{w_0}^T \theta^{w_i}) x_{w_0}$$
 (1)

 $= (y_i - \sigma(x_{w_0}^T \theta^{w_i})) x_{w_0} \tag{2}$ 

同样的方法,我们可以求出 $x_w$ 。的梯度如下:

$$rac{\partial L}{\partial x^{w_0}} = \sum_{i=0}^{neg} (y_i - \sigma(x_{w_0}^T heta^{w_i})) heta^{w_i}$$

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 $x_{w_0}, \theta^{w_i}, i=0,1,\dots neg.$ 

# 4. Negative Sampling负采样方法

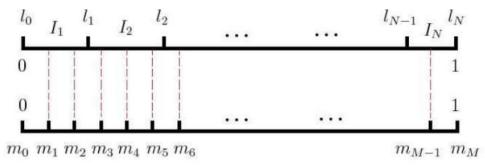
现在我们来看看如何进行负采样,得到neg个负例。word2vec采样的方法并不复杂,如果词汇表的大小为V,那么我们就将一段长度为1的线段分成V份,每份对应词汇表中的一个词。当然每个词对应的线段长度是不一样的,高频词对应的线段长,低频词对应的线段短。每个词w的线段长度由下式决定:

$$len(w) = rac{count(w)}{\sum\limits_{u \in vocab} count(u)}$$

在word2vec中,分子和分母都取了3/4次幂如下

$$len(w) = rac{count(w)^{3/4}}{\sum\limits_{u \in vocab} count(u)^{3/4}}$$

在采样前,我们将这段长度为1的线段划分成M等份,这里M>>V,这样可以保证每个词对应的线段都会划分成对应的小块。而M份中的每一份都会落在某一个词对应的线段上。在采样的时候,我们只需要从M个位置中采样出neg个位置就行,此时采样到的每一个位置对应到的线段所属的词就是我们的负例词。



在word2vec中,M取值默认为 $10^8$ 。

# 5. 基于Negative Sampling的CBOW模型

有了上面Negative Sampling负采样的方法和逻辑回归求解模型参数的方法,我们就可以总结出基于Negative Sampling的CBOW模型算法流程了。梯度迭代过程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Mcount,CBOW的上下文大小2c,步长 $\eta$ ,负采样的个数neg

输出:词汇表每个词对应的模型参数heta,所有的词向量 $x_w$ 

- 1. 随机初始化所有的模型参数heta,所有的词向量w
- 2. 对于每个训练样本 $(context(w_0),w_0)$ ,负采样出neg个负例中心词 $w_i,i=1,2,\dots neg$
- 3. 进行梯度上升迭代过程,对于训练集中的每一个样本 $(context(w_0), w_0, w_1, \dots w_{neg})$ 做如下处理:

a) e=0, 计算
$$x_{w_0}=rac{1}{2c}\sum\limits_{i=1}^{2c}x_i$$

b) for i= 0 to neg, 计算:

$$f = \sigma(x_{w_0}^T heta^{w_i})$$
  $g = (y_i - f)\eta$   $e = e + g heta^{w_i}$   $heta^{w_i} = heta^{w_i} + gx_{w_0}$ 

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

- 1. 梯度下降 (Grac 098)
- 2. 梯度提升树(GBD
- 3. 线性判别分析LD/
- 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)

 $x_k = x_k + e$ 

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。

## 6. 基于Negative Sampling的Skip-Gram模型

有了上一节CBOW的基础和上一篇基于Hierarchical Softmax的Skip-Gram模型基础,我们也可以总结出基于 Negative Sampling的Skip-Gram模型算法流程了。梯度迭代过程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Mcount,Skip-Gram的上下文大小2c,步长 $\eta$ ,,负采 样的个数neg。

输出:词汇表每个词对应的模型参数heta,所有的词向量 $x_w$ 

- 1. 随机初始化所有的模型参数heta,所有的词向量w
- 2. 对于每个训练样本 $(context(w_0),w_0)$ ,负采样出neg个负例中心词 $w_i,i=1,2,\ldots neg$
- 3. 进行梯度上升迭代过程,对于训练集中的每一个样本 $(context(w_0), w_0, w_1, \dots w_{neg})$ 做如下处理:
  - a) for i = 1 to 2c:
    - i) e=0
    - ii) for j= 0 to neg, 计算:

$$f = \sigma(x_{w_{0i}}^T heta^{w_j})$$
  $g = (y_j - f)\eta$   $e = e + g heta^{w_j}$   $heta^{w_j} = heta^{w_j} + gx_{w_{0i}}$ 

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

$$x_k = x_k + e$$

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。

# Negative Sampling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Negative Sampling的CBOW模型算法在464-494行,基于Hierarchical Softmax的Skip-Gram 的模型算法在520-542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 $x_w$ ,syn1neg对应我们的 $heta^{w_i}$ ,layer1\_size对应词向量的维 度,window对应我们的c。negative对应我们的neg, table\_size对应我们负采样中的划分数M。

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。这些和基于Hierarchical Softmax的是一样的。

以上就是基于Negative Sampling的word2vec模型,希望可以帮到大家,后面会讲解用gensim的python版 word2vec来使用word2vec解决实际问题。

(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





关注 - 15 粉丝 - 2385

« 上一篇: word2vec原理(二) 基于Hierarchical Softmax的模型

» 下一篇: <u>用gensim学习word2vec</u>

posted @ 2017-07-28 15:56 刘建平Pinard 阅读(18137) 评论(43) 编辑 收藏

5

0