Gensim进阶教程：训练word2vec与doc2vec模型

本篇博客是Gensim的进阶教程，主要介绍用于词向量建模的word2vec模型和用于长文本向量建模的doc2vec模型在Gensim中的实现。

Word2vec

Word2vec并不是一个模型——它其实是2013年Mikolov开源的一款用于计算词向量的工具。关于Word2vec更多的原理性的介绍，可以参见我的另一篇博客：word2vec前世今生

在Gensim中实现word2vec模型非常简单。首先，我们需要将原始的训练语料转化成一个sentence的迭代器；每一次迭代返回的sentence是一个word（utf8格式）的列表：

```
class MySentences(object):
    def __init__(self, dirname):
        self.dirname = dirname

    def __iter__(self):
        for fname in os.listdir(self.dirname):
            for line in open(os.path.join(self.dirname, fname)):
                yield line.split()

sentences = MySentences('/some/directory') # a memory-friendly iterator
```

接下来，我们用这个迭代器作为输入，构造一个Gensim内建的word2vec模型的对象（即将原始的one-hot向量转化为word2vec向量）：

```
model = gensim.models.Word2Vec(sentences)
```

如此，便完成了一个word2vec模型的训练。

我们也可以指定模型训练的参数，例如采用的模型（Skip-gram或是CBoW）；负采样的个数；embedding向量的维度等。具体的参数列表在这里

同样，我们也可以通过调用 `save()` 和 `load()` 方法完成word2vec模型的持久化。此外，word2vec对象也支持原始bin文件格式的读写。

Word2vec对象还支持online learning。我们可以将更多的训练数据传递给一个已经训练好的word2vec对象，继续更新模型的参数：

```
model = gensim.models.Word2Vec.load('/tmp/mymodel')
model.train(more_sentences)
```

若要查看某一个word对应的word2vec向量，可以将这个word作为索引传递给训练好的模型对象：

```
model['computer']  # raw NumPy vector of a word
```

Doc2vec

Doc2vec是Mikolov在word2vec基础上提出的另一个用于计算长文本向量的工具。它的工作原理与word2vec极为相似——只是将长文本作为一个特殊的token id引入训练语料中。在Gensim中，doc2vec也是继承于word2vec的一个子类。因此，无论是API的参数接口还是调用文本向量的方式，doc2vec与word2vec都极为相似。

主要的区别是在对输入数据的预处理上。Doc2vec接受一个由LabeledSentence对象组成的迭代器作为其构造函数的输入参数。其中，LabeledSentence是Gensim内建的一个类，它接受两个List作为其初始化的参数：word list和label list。

```
from gensim.models.doc2vec import LabeledSentence
sentence = LabeledSentence(words=[u'some', u'words', u'here'], tags=[u'SENT_1'])
```

类似地，可以构造一个迭代器对象，将原始的训练数据文本转化成LabeledSentence对象：

```
class LabeledLineSentence(object):
    def __init__(self, filename):
        self.filename = filename

    def __iter__(self):
        for uid, line in enumerate(open(filename)):
            yield LabeledSentence(words=line.split(), labels=['SENT_%s' % uid])
```

准备好训练数据，模型的训练便只是一行命令：

```
from gensim.models import Doc2Vec
model = Doc2Vec(dm=1, size=100, window=5, negative=5, hs=0, min_count=2, workers=4)
```

该代码将同时训练word和sentence label的语义向量。如果我们只想训练label向量，可以传入参数 `train_words=False` 以固定词向量参数。更多参数的含义可以参见这里的API文档。

注意，在目前版本的doc2vec实现中，每一个Sentence vector都是常驻内存的。因此，模型训练所需的内存大小同训练语料的大小正相关。

分类: 码农碎笔

标签: word2vec, Gensim

公子天
关注 - 0
粉丝 - 32
+加关注

2 0

« 上一篇：Gensim入门教程
» 下一篇：深度学习开发环境搭建教程（Mac篇）

CS224d(3)
deep learning(3)
Gensim(2)
Pattern Recognition and Machine Learning(2)
VSM(2)
Bias-Variance Decomposition(1)
CNN(1)
Cross Entropy(1)
更多

随笔分类(11)
读书笔记(2)
课程笔记(3)
论文笔记(3)
码农碎笔(3)

随笔档案(11)
2017年5月 (2)
2017年4月 (1)
2016年9月 (2)
2016年8月 (1)
2016年7月 (3)
2016年5月 (1)
2016年4月 (1)

posted on 2016-09-28 21:01 公子天 阅读(18114) 评论(2) 编辑 收藏

评论

#1楼 2017-08-09 20:14 zangyu00544

请问，在doc2vec模型中最后训练的时候语料是怎么传入的啊？
model = Doc2Vec(dm=1, size=100, window=5, negative=5, hs=0, min_count=2, workers=4)，从这句上完全看不出来啊

支持(0) 反对(0)

#2楼[楼主] 2017-08-11 11:22 公子天

@ zangyu00544
Sorry，文章里没有说清楚。Doc2Vec和Word2Vec类一样，可以在初始化的时候传入一个documents（比如上面的LabeledLineSentence迭代器）对象，或者是直接调用train方法进行训练。

支持(0) 反对(0)

刷新评论 刷新页面 返回顶部

注册用户登录后才能发表评论，请 登录 或 注册，访问网站首页。

阅读排行榜
1. word2vec前世今生(29677)
2. Gensim进阶教程：训练word2vec与doc2vec模型(18114)
3. Gensim入门教程(13627)
4. AlphaGo原理浅析(7207)
5. (Stanford CS224d) Deep Learning and NLP 课程笔记（三）：GloVe与模型的评估(5304)

评论排行榜
1. word2vec前世今生(3)
2. Gensim进阶教程：训练word2vec与doc2vec模型(2)
3. Gensim入门教程(1)
4. (Stanford CS224d) Deep Learning and NLP 课程笔记（二）：word2vec(1)
5. (Stanford CS224d) Deep Learning and NLP 课程笔记（一）：Deep NLP(1)

推荐排行榜
1. word2vec前世今生(10)
2. 深度学习开发环境搭建教程（Mac篇）(2)
3. Gensim进阶教程：训练word2vec与doc2vec模型(2)
4. (Stanford CS224d) Deep Learning and NLP 课程笔记（三）：GloVe与模型的评估(1)
5. (Stanford CS224d) Deep Learning and NLP 课程笔记（二）：word2vec(1)

最新知识库文章:

· 以操作系统的角度述说线程与进程
· 软件测试转型之路
· 门内门外看招聘
· 大道至简，职场上做人做事做管理
· 关于编程，你的练习是不是有效的？
» 更多知识库文章...