一些文本预处理方法

使用gensim训练词向量

- Word2vec

  Word2vec的原理见实验课的PPT，这里只介绍如何自己训练词向量。

```
# 读入测试集文本
fr = open('/home/liujy/LR/data/MulLabelTest.ss')
text = [line.split('\t\t')[1].replace("<ssss>","") for line in fr] #
[doc1, doc2, doc3...]

# 将每句话拆成词
sentences = [line.split(' ') for line in text] # [[word1,
word2,word3...], [word1, word2, word3...].....]

# 训练Word2Vec
model = gensim.models.Word2Vec(sentences, min_count=1, size=100,
window=5, iter=100)

# 查看某个单词的词向量
print(model['time'])

# 查看与某个单词最相似的topn个单词 [可以用来检验模型训练的效果]
model.most_similar(u"time", topn=9)
"""
output:
[('visit', 0.5926637053489685),
 ('day', 0.5672399401664734),
 ('sightline', 0.5185168385505676),
 ('week', 0.515683114528656),
 ('chance', 0.5015542507171631),
 ('while\n', 0.49100732803344727),
 ('ceaser', 0.4863746166229248),
 ('person/couple', 0.4802390933036804),
 ('replay', 0.4801928699016571)]
"""
```

word2vec的相关参数设置可以参考这里

另外，Google已经采用超大语料库训练了一个词向量库，里面包含了很多常用词的词向量。

可以在这里的**Where to obtain the training data**部分下载到

自己训练的向量与pre-trian的向量可能在语义上会有所不同，大家可以自己尝试使用两种向量。

- Doc2vec

  Doc2vec的原理与Word2vec相似，输入是文档的合集，输出是各个文档的向量。

  同样的，**训练数据越大，模型越准确**。

  下面介绍如何用只用**测试集**文本训练doc向量，建议自行改为使用训练集+测试集文本。

```
import gensim
import numpy as np
from gensim.models.doc2vec import Doc2Vec,LabeledSentence
def labelize(texts, label_type):
    labelized = []
    for i,v in enumerate(texts):
        label = '{}_{}'.format(label_type,i)
        labelized.append(LabeledSentence(v, [label]))
    return labelized

# 读入文本
fr = open('/home/liujy/LR/data/MulLabelTest.ss')
test_text = [line.split('\t\t')[1].replace("<ssssss>","") for line in
fr]

# 根据gensim的文档要求， 将文本转为 doc,test_i 组
test_text_labelized = labelize(test_text,"test")

# 训练
model = gensim.models.Doc2Vec(test_text_labelized, size=100, window=3)
model.train(test_text_labelized, total_examples=model.corpus_count,
epochs=model.iter)

# 得到测试集文本的向量
test_vec = model.docvecs[np.arange(len(train_text))]
```