Sentiment analysis using product review data

Beilei Bian

2017/07/01

NLP入门

- 什么是自然语言处理?

  自然语言处理的目标：设计算法使计算机能够理解自然语言并执行一些任务。

- Easy
  - Spell Checking
  - Keyword Search(tf-idf)
  - Finding Synonyms
- Medium
  - Parsing information from websites, documents, etc.
- Hard
  - Machine Translation
  - Semantic Analysis
  - Coreference(共指解析)
  - Question Answering

词向量

The first and arguably most important common denominator across all NLP tasks is how we represent words as input to any and all of our models.

- one-hot vector: 为词库中的词建立索引，每个词都表示为一个 $\mathbb{R}^{|V|\times 1}$ 向量，其中|V|为词库中词的个数。

$$w^{aardvark} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{a} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, w^{at} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \cdots w^{zebra} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

$$(w^{hotel})^T w^{motel} = (w^{hotel})^T w^{cat} = 0$$

one-hot存在的问题：1.词与词是正交的，无法度量相似度 2.高维且稀疏

词向量

- 基于特征值分解的方法(SVD)

  we first loop over a massive dataset and accumulate word co-occurrence counts in some form of a matrix X, and then perform Singular Value Decomposition on X to get a USV^T decomposition.

  Use the rows of U as the word embeddings

1. Word-Document Matrix

   $\mathbb{R}^{|V| \times M}$的矩阵（large matrix）

   缺点：文档多，词的数量多，矩阵过大，存储困难

2. Window based Co-occurrence Matrix

   缺点：1.矩阵的维数经常变化 2.矩阵过于稀疏，因为许多词和词不会共同出现 3.高维

1. I enjoy flying.
2. I like NLP.
3. I like deep learning.

The resulting counts matrix will then be:

$$X = \begin{array}{c} \\ I \\ like \\ enjoy \\ deep \\ learning \\ NLP \\ flying \\ . \end{array} \begin{array}{c} \begin{array}{cccccccc} I & like & enjoy & deep & learning & NLP & flying & . \end{array} \\ \begin{bmatrix} 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix} \end{array}$$

词向量

- Iteration Based Methods
  - 统计语言模型：一个词序列的概率

    比如："The cat jumped over the puddle."

    一元模型：

$$P(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} P(w_i)$$

    但是，下一个词的出现是强依赖于上一个词的。

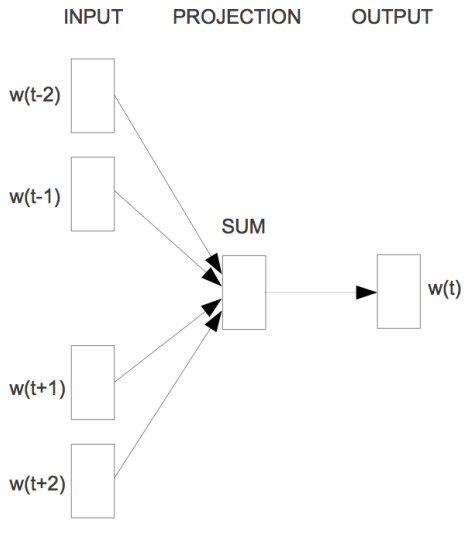
    二元模型：

$$P(w_1, w_2, \ldots, w_n) = \prod_{i=1}^{n} P(w_i | w_{i-1})$$

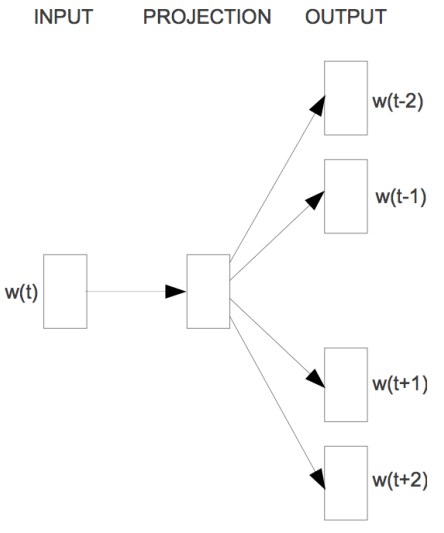
实际应用中最多的是N=3的情况

词向量

- word2vec
  - CBOW
  - Skip-gram



CBOW



Skip-gram

三层的神经网络：输入层，投影层，输出层

涉及到的数学原理：统计语言模型、二叉树（Huffman树）、Hierarchical softmax、negative sampling、梯度下降法

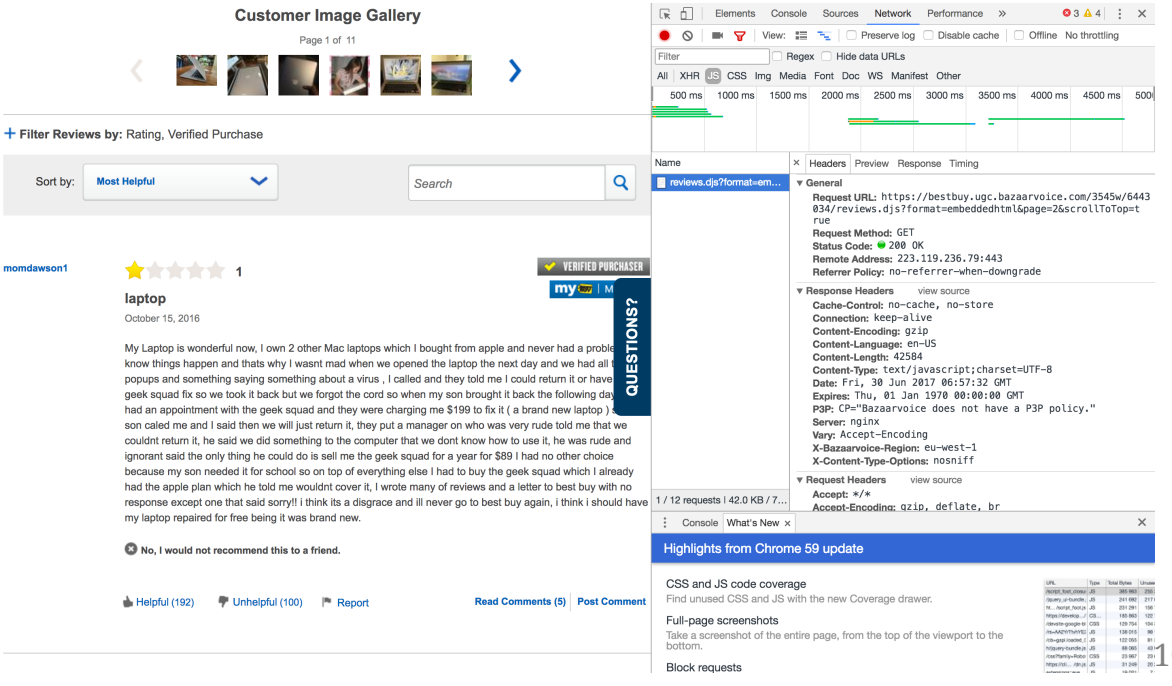
词向量是帮助构造目标函数的辅助参数，训练完成后，它是统计语言模型的一个副产物。

Data Analysis

以bestbuy电子商品评论为例进行情感分析

Part 1

数据爬取：以Mac为例



找到资源所在的地址：

Request URL:https://bestbuy.ugc.bazaarvoice.com/3545w/6443034/reviews.djs?format=embeddedhtml&page=1&scrollToTop=true

找出URL的规律，对整个页面进行下载并解析。

用到的工具：

- urllib
- json
- lxml.html
- CSSSelector
- BeautifulSoup

Part 2

数据清洗

- regular expression

```
import re
review_letters = re.sub("[^a-zA-Z]", " ", review_text)
```

- 分词 / 字母大小写问题

```
words = review_letters.lower().split()
```

- nltk去停顿词

```
from nltk.corpus import stopwords
stops = set(stopwords.words("english"))
words = [w for w in words if not w in stops]
```

Part 3

建立特征

- 方法一：Bag of words（工具：sklearn）

  词袋法：不考虑语法，不考虑词的顺序，只考虑词的多样性。常用于文档分类。每个词出现的频率作为特征输入分类器。（也可以用于图像分类：Bag of words for computer vision https://github.com/bikz05/bag-of-words）

```
from sklearn.feature_extraction.text import CountVectorizer
vectorizer = CountVectorizer(analyzer = "word", tokenizer = None, pre
train_data_features = vectorizer.fit_transform(clean_train_reviews)
train_data_features = train_data_features.toarray()
```

- 方法二：word2vec（工具：gensim）

```
from gensim.models import word2vec
tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
all_sentences = []
for sum in data["review"]:
    all_sentences += sum_to_sent(sum, tokenizer)
model = word2vec.Word2Vec(all_sentences, size=300, min_count = 1, wi
model.save(model_name)
```

Part 4

建立分类模型(sklearn)

- Random Forest

```
from sklearn.ensemble import RandomForestClassifier
forest = RandomForestClassifier(n_estimators = 100)
forest = forest.fit(trainDataVecs, train["star"])
result = forest.predict(testDataVecs)
```

- Naive Bayes
- SVM

Part 5

模型评估

将数据集随机分为训练集用来建模，测试集用来测试。

评估方法：

- Accuracy
- TPR/TNR
- ROC/AUC

```
from sklearn import metrics
print(metrics.classification_report(test['star'], result))
```

Deep learning

- CNN CNN用于文本分类的开山之作：Kim, Yoon. "Convolutional neural networks for sentence classification." arXiv preprint arXiv:1408.5882 (2014).

  CNN最初用于图像分类，在自然语言处理中，可以把word embedding看作image

- RNN RNN引入了定向循环，可以处理序列数据，网络对前面的数据进行记忆并应用于当前输出的计算中。应用最广泛的为：LSTM

一些资料

关于word2vec: http://blog.csdn.net/itplus/article/details/37969635

NLP tutorial：http://cs224d.stanford.edu/

数学之美

常用工具：

- NLTK
- gensim
- Stanford NLP
- openNLP