

实验报告

小组成员

- 雷雨轩 PB18111791
- 裴启智 PB18111793

算法及优化的描述

semantic_search.py

- 函数功能请参考 README，这里主要说一些细节方面以及优化
- `tf_idf_normalize(tf_idf_matrix)`:
 - 优化：通过 numpy 数组，对 tf-idf 矩阵按列进行归一化处理，每次都对列进行整体操作（计算一列的模长，对一列整体进行修改），可以加快运行速度
- `compare(input_word_vec, normalize_matrix)`：利用 numpy 的 `inner` 方法直接求查询向量和归一过的文档向量求内积，并根据内积进行降序排列
- `cal_input_query(query_str)`：首先建立一个 1000 维的零向量，在对前 1000 个词项进行遍历，查询中在前1000个词中出现的词对应权重为1，否则为0，得到查询向量

bool_search.py

- 利用生成好的倒排表来做一个二值匹配。
- 读取形如 `(contact OR power) AND (price OR market)` 的bool查询语句并对此字符串做处理，然后按其对应的逻辑运算规则，用倒排表所得文档列表进行筛选。
- bool语句的处理采用自顶向下的递归算法，考虑到算符的优先级 `NOT > AND > OR`，有括号先处理括号。所以将运算大体分为了三层:exp,item,factor
 - `def exp`：表达式，由单独的"项"或者"项"与"OR"运算符连接形成；
 - 此函数调用一次或多次 `item()` 来得到项并进行 OR 运算
 - `def item`：项，由单独的"因子"或"因子"和"AND"运算符连接而成；
 - 此函数调用一次或多次 `factor()` 来得到因子并进行 AND 运算
 - `def factor`：因子，可以是单独的词，也可以是用括号括起来的"表达式"。
 - 遇到 "("：调用 `exp()` 处理括号内的一个新的表达式
 - 遇到 "NOT"：调用 `notExp()` 表示处理一个需要取补集的表达式
 - 否则遇到的是一个单词，查询倒排表并返回其文档id列表
 - `def notExp`：对NOT做专门处理的，与factor函数差不多，只是多了一层NOT操作
 - 遇到 "("：调用 `NOT(exp())` 处理括号内的一个新的表达式，对exp()函数返回后用一次 `NOT()` 取补集
 - 遇到 "NOT"：调用 `NOT(notExp())` 表示处理一个需要取双重补集的表达式
 - 否则遇到的是一个单词，查询倒排表并做NOT()后返回其文档id列表
 - 需注意查倒排表前需对bool查询语句的单词作词根化处理
- 求并、交、补的函数
 - `def AND(p1,p2)`：主要思想同ppt，对两个倒排表各备一个指针，然后自左向右扫描一遍即可，相同的文档id加入到返回值list里
 - `def OR(p1,p2)`：

- 一开始是考虑简单的利用python内置函数实现列表的拼接+去重

```
p=p1+p2
answer=list(set(p))
answer.sort()
```

- 优化：但因为实际上这样一来内置实现的复杂度较高，至少 $O(nlgn)$ ，这对于可能五十多万文档集合来说查询时间就会显得有点慢

所以考虑仍然是两个倒排表各一个指针，然后自左向右扫描一遍，相同文档id只加入一次即可

- `def NOT(p1)`

- 一开始考虑生成一个50多万的列表，然后把在p1中的项remove()，但是这样时间复杂度太高，开销过大
- 所以后续考虑同样看成两个列表的问题，从左到右扫描一遍即可完成求补

build.py



- 函数功能请参考 README，这里主要说一些细节方面以及优化
- `add_inverted_table(word, freq)`：根据传入的词条和频率更新倒排表，如果倒排表中没有该 word 则扩充倒排表中的词条，如果有更新该 word 对应行的信息，将文档 id 和词频作为一个列表存入内存中的倒排表。
 - 最开始时采用了边建立边判断，因为一篇文章中一个词会有多次出现，不同的调用传入的参数 (word, freq) 会相同，这是如果采用只 append 不同的id 和词频的话，则每次都要遍历当前行来判断是否有重复，这样建立过程会越来越慢，因为一个词条后面的 (id, 词频) 对会越来越多，加上新的词条不断出现，导致算法复杂度很高。
 - 优化：采用空间换时间的思想，将重复的 (id, 词频) 对也存储下来，后续再通过 $O(n)$ 复杂度的去重算法来去重，这大大降低了建倒排表的时间
 - 说明：程序（内存）中的倒排表保存了其他信息（用以建立 tf-idf 矩阵），在输出倒排表文件时只输出了倒排表本身所需要的信息
- `build_invert_list(path)`：功能参见README。由于不同的文件可能有不同的编码方式，故通过 `try...except...` 尝试采用不同的编码方式打开文件
- `de_repetition2(old_list)`：
 - 优化：这里采用了 $O(n)$ 时间复杂度的去重算法。即记录当前的元素，对后续的元素如果相同则删去，否则更新当前元素，这么循环读下去。每个元素只需要读取一次，故时间复杂度为 $O(n)$ 。最开始采用了如下算法：

```
def de_repetition(old_list):
    new_list = []
    for i in old_list:
        if i not in new_list:
            new_list.append(i)
    return new_list
```

可以看到这个算法进行了两重遍历，时间复杂度为 $O(n^2)$ ，这会导致去重所需要的运行时间大幅上升

- `tf_idf_matrix_build(list_sorted)`：
 - 优化：因为 tf-idf 矩阵的很多项都是 0，相当于是一个稀疏矩阵，故可以采用压缩存储的方式，只存储 1 所在的位置的索引，这样可以大大减低存储所需要的空间。优化主要是通过 Python 的 `scipy.sparse.save_npz` 以及 `scipy.sparse.load_npz` 来实现压缩和解压缩的

- 例子：采用 maildir/allen-p 中的 3034 个文件进行测试，比较 txt 和 npz 格式的大小，可以看到：

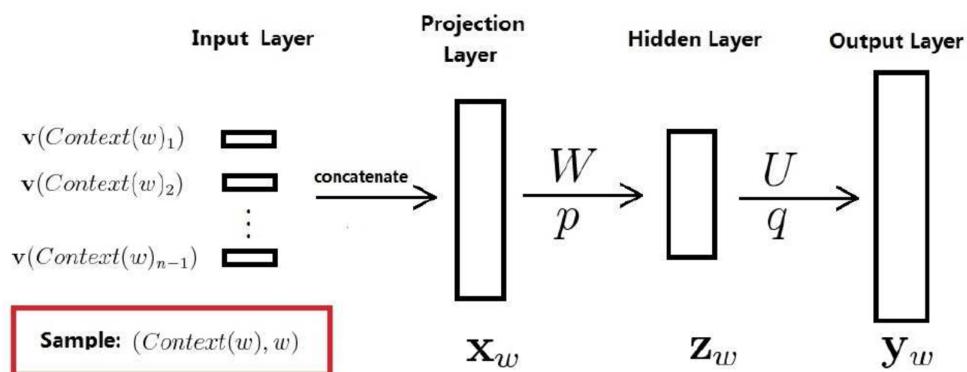
 tf_idf_matrix.npz	2020/12/3 22:31	NPZ 文件	232 KB
 tf_idf_matrix.txt	2020/12/1 22:19	文本文档	30,277 KB

二者大小有 130 倍的差距，优化的效果是很明显的

采用Word2vec模型的优化

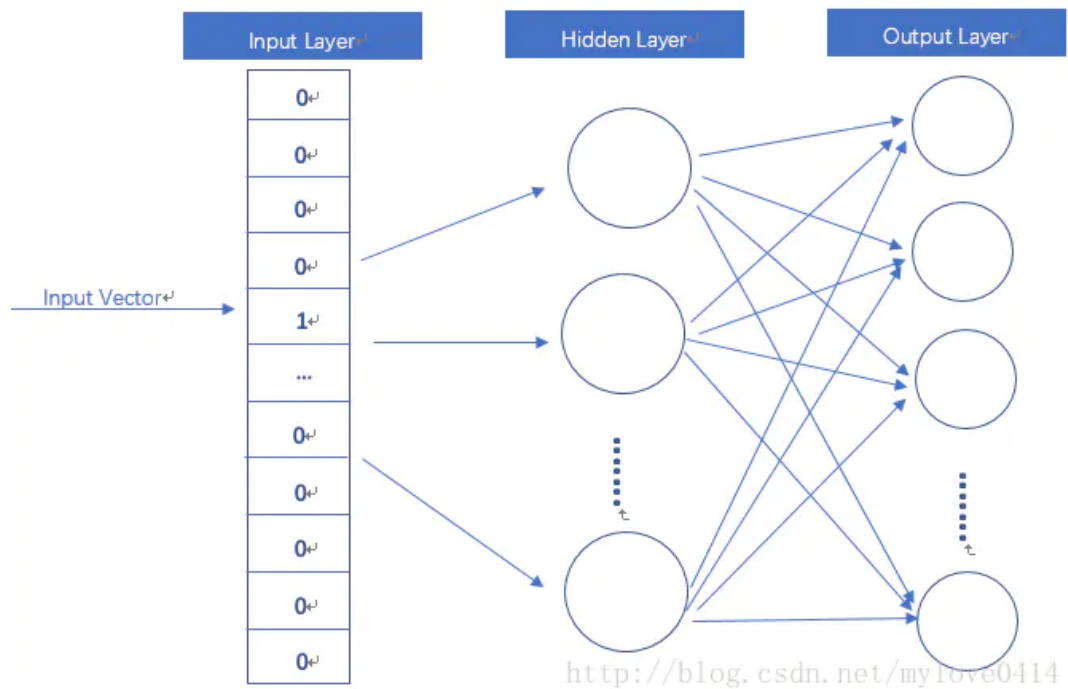
word2vec原理调研

- Word2vec的模型以大规模语料库作为输入，然后生成一个向量空间（通常为几百维）。词典中的每个词都对应了向量空间中的一个独一无二的向量，而且语料库中拥有共同上下文的词映射到向量空间中的距离会更近。又可称为word embedding,即将高维词向量嵌入到一个低维空间。
- 发展：
 - N-gram模型：此为课上所讲知识，不过多总结
 - 神经概率语言模型
 - 神经概率语言模型的网络结构



参数有包括：『词向量』以及『神经网络参数』。一旦确定了这些参数，就相当于确定了『函数』的参数，也就相当于知道了参数，继而能求得整个句子的概率。

- 词向量：即对词典中任意词，指定一个固定长度的实值向量。则即称为的词向量，是词向量的长度。
- 词向量有两种表现形式：
 - One-hot Representation：用维度为字典长度的向量表示一个词，仅一个分量为 1，其余为 0。缺点是容易收到维度灾难的困扰，而且不能很好的刻画词与词之间的关系。
 - Distributed Representation：每个词映射为固定长度的短向量。通过刻画两个向量之间的距离来刻画两个向量之间的相似度。
- 词向量的优点
 - 词语与词语间的相似度可以通过词向量来体现
 - 基于词向量的模型自带『平滑化』功能，无需额外处理。
- 词向量的缺点：计算量太大
- Word2Vec相比于神经概率语言模型的改进
 - Word2vec的网络网络更简单，因此训练起来更加容易，能训练更多的数据



- 优化Softmax归一化
 - 使用Hierarchical Softmax优化
 - CBOW模型：给定上下文，来预测当前的词
 - Skip-gram模型：给定当前词，预测上下文
 - 使用Negative Sampling优化

基于word2vec来表征查询和文档

- 训练以及查询的代码均在 `src/word2vec.py`
- 主要考虑对电子邮件文档作预处理后得到训练语料，利用gensim包提供的API来进行训练，最后用生成的模型来判断查询词组与各个文档的相似度并返回相似度在前10的文档编号
- gensim提供的word2vec的API的操作详见[官网](#)
- 对电子邮件的预处理：
 - 通过分词、词根化以及去停用词后，把每个文档的词按空格隔开，写入 `output/word2vec_data.txt`（每个文档的词占一行）
- 然后将该文件作为语料输入到Word2Vec模型训练
 - 参数考虑：
 - 在一定尝试与调参后，有了如下考量
 - 因为考虑与原来的tf-idf相比，word2vec的优势在于其能充分的利用文档上下文的关系来生成词向量，所以考虑直接用预处理后的所有数据进行训练，而不用 `max_vocab_size` 来把训练的词限制在前1000频率的词里
 - 此外，考虑到许多文档就短短几个词的一句话，所以考虑不对词频作筛选，`min_count=1`
 - 词向量长度 `size=100`
 - 前后文参数 `window=5`
- 训练好的模型保存在 `output/word2vec.model`

在查询时只需把训练部分代码注释掉（上传文件里 `src/word2vec.py` 已被注释），只进行查询即可

- 查询时，利用函数 `model.wv.n_similarity()` 计算查询词组(list形式)与每个文档的词汇列表的相似度，最后返回相似度最高的10个文档编号

运行示例与对比

布尔查询

- 因为一个bool查询可能满足条件的文档很多，所以代码里只是挑了有代表性的结果文档打印出来作为报告中的演示。

```
PS E:\Web_lab\exp1\src> python -u "e:\Web_lab\exp1\src\bool_search.py"
power AND NOT energy
[56, 107, 146, 175, 206, 245, 275, 276, 292, 296]
NOT contact AND power OR price
[0, 1, 10, 11, 12, 13, 14, 21, 28, 29]
( contact OR power ) AND ( price OR market )
[0, 1, 10, 11, 12, 35, 48, 107, 153, 175]
issue AND contact AND price
[0, 1, 231, 630, 1055, 1057, 1137, 1698, 2025, 2026]
contact OR power OR price
[0, 1, 10, 11, 12, 13, 14, 19, 21, 28]
```

如图为返回了每个bool查询满足条件的前10小的文档编号(注意，此处是从0开始编号，比output/dictionary.txt里的行号少1)

- power AND NOT energy 这一组里，编号56的文档对应 maildir\allen-p\all_documents\153_，如下图可以看到有power而没有energy

```
Cooper,
Can you give access to the new west power site to Jay Reitmeyer. He is an
analyst in our group.
```

- 再比如 NOT contact AND power OR price，编号为10的文档对应 maildir\allen-p\all_documents\10_，里面没有contact，而出现了power和price

Here is today's copy of Bloomberg **Power** Lines. Adobe Acrobat Reader is required to view the attached pdf file. You can download a free version of Acrobat Reader at <http://www.adobe.com/products/acrobat/readstep.html>

If you have trouble downloading the attached file it is also located at <http://www.bloomberg.com/energy/daily.pdf>

Don't forget to check out the Bloomberg **PowerMatch** West Coast indices, the most accurate indices anywhere. Index values are calculated from actual trades and can be audited by all **PowerMatch** customers.

Our aim is to bring you the most timely electricity market coverage in the industry and we welcome your feedback on how we can improve the product=20 further.

Bloomberg Energy Department

12/13 Bloomberg Daily **Power** Report

Table

				Bloomberg U.S. Regional Electricity Prices
				(\$/MWh for 25-50 MWh pre-scheduled packages, excluding transmission=20 costs)

- 最后看 issue AND contact AND price,编号为231的文档, 对应maildir\allen-p\all_documents\312_, 其中三个词均含有

Price of Reliability: The Value and Strategy of Gas Transportation.

Price of Reliability, delivered in 6 parts (with each part representin=0

The attached prospectus explains the various options for subscribing.?Plea=se=20

note two key **issues** in regards to your subscribing options:?One, there is =a=20

10% savings for PIRA retainer clients who order before February 25, 2000;=20

10% savings for PIRA retainer clients who order before February 25, 20=20

and?two, there are discounts for purchasing?more than one region.

If you have any questions, please do not hesitate to **contact** me.

语义查询

查询词

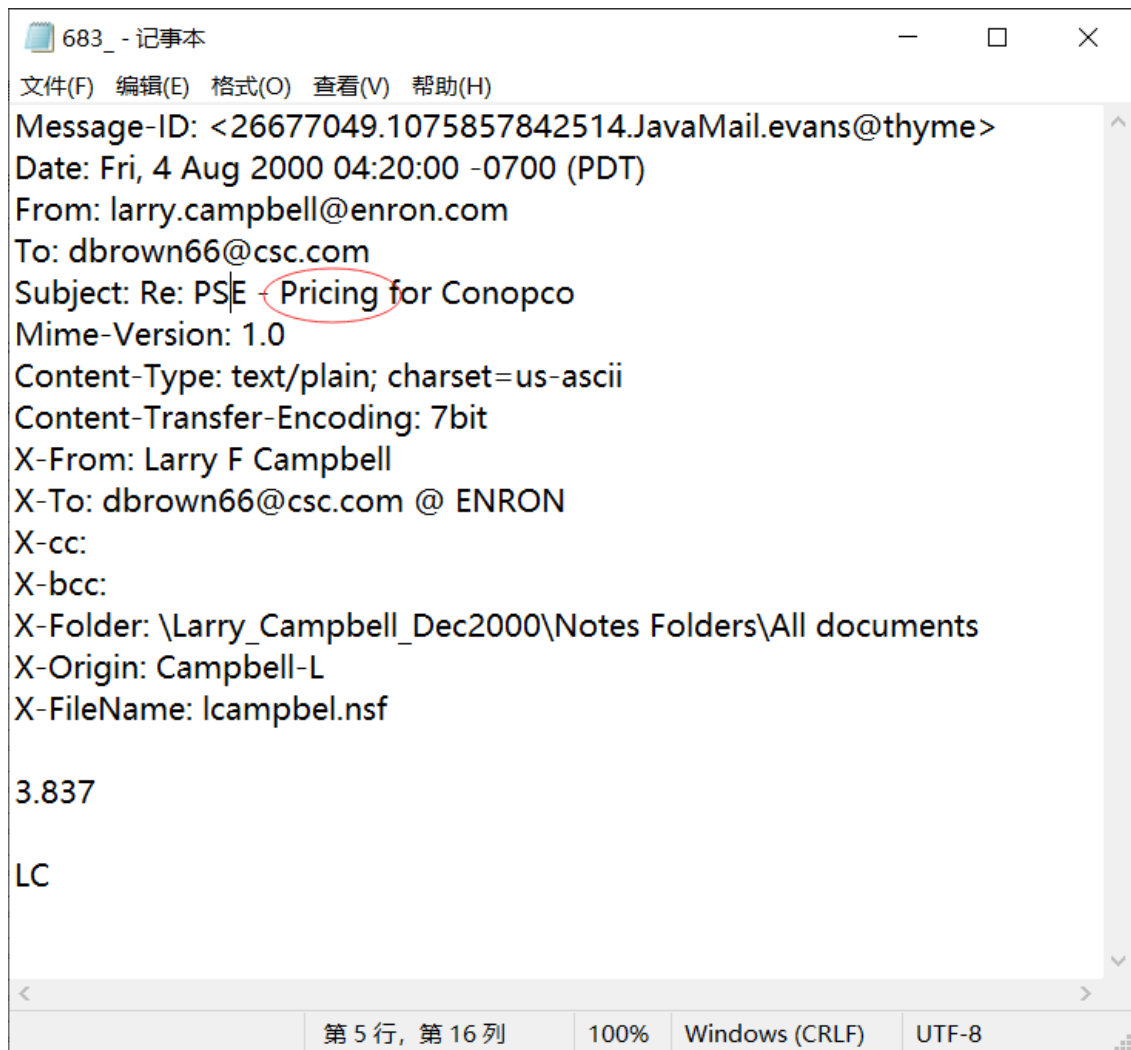
构造一个组合和四个单个单词, 主要用来从不同维度解释语义查询和wordvec2查询的区别和联系

contact issue price
business
offer
financial
issues

结果

```
PS E:\Web_lab\exp1\src> python -u "e:\Web_lab\exp1\src\semantic_search.py"  
[('13887', 1.0), ('130079', 1.0), ('357152', 1.0), ('515406', 1.0), ('60265', 0.7947135883024297), ('81726', 0.7947135883024297), ('103108', 0.75886  
1318716418), ('500269', 0.7260167642363606), ('517269', 0.683732340962702), ('260222', 0.6784573304117147)]  
[('161322', 1.0), ('162004', 1.0), ('371923', 1.0), ('372885', 1.0), ('373241', 1.0), ('373962', 1.0), ('446603', 1.0), ('447722', 1.0), ('448002',  
1.0), ('126484', 0.7440469019230798)]  
[('10101', 1.0), ('12994', 1.0), ('15514', 1.0), ('17331', 1.0), ('493100', 1.0), ('332315', 0.7524295425741687), ('291624', 0.7431720601554569), ('  
294069', 0.7431720601554569), ('295569', 0.7431720601554569), ('291982', 0.7097721287977076)]  
[('421906', 1.0), ('122767', 0.7916788504065422), ('477024', 0.7750619863445768), ('481751', 0.7750619863445768), ('161399', 0.7042958945070275), ('  
162098', 0.7042958945070275), ('453746', 0.6956463440881376), ('475279', 0.6682818019787303), ('483262', 0.6682818019787303), ('121787', 0.646461983  
5306805)]  
[('248', 1.0), ('1711', 1.0), ('2684', 1.0), ('256928', 0.815688866136106), ('256941', 0.815688866136106), ('297530', 0.815688866136106), ('161320',  
0.7387822353380893), ('162002', 0.7387822353380893), ('16181', 0.7050202698972577), ('46211', 0.6892952636586334)]
```

- contact issue price (前三个文档依次如下)



231_ - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

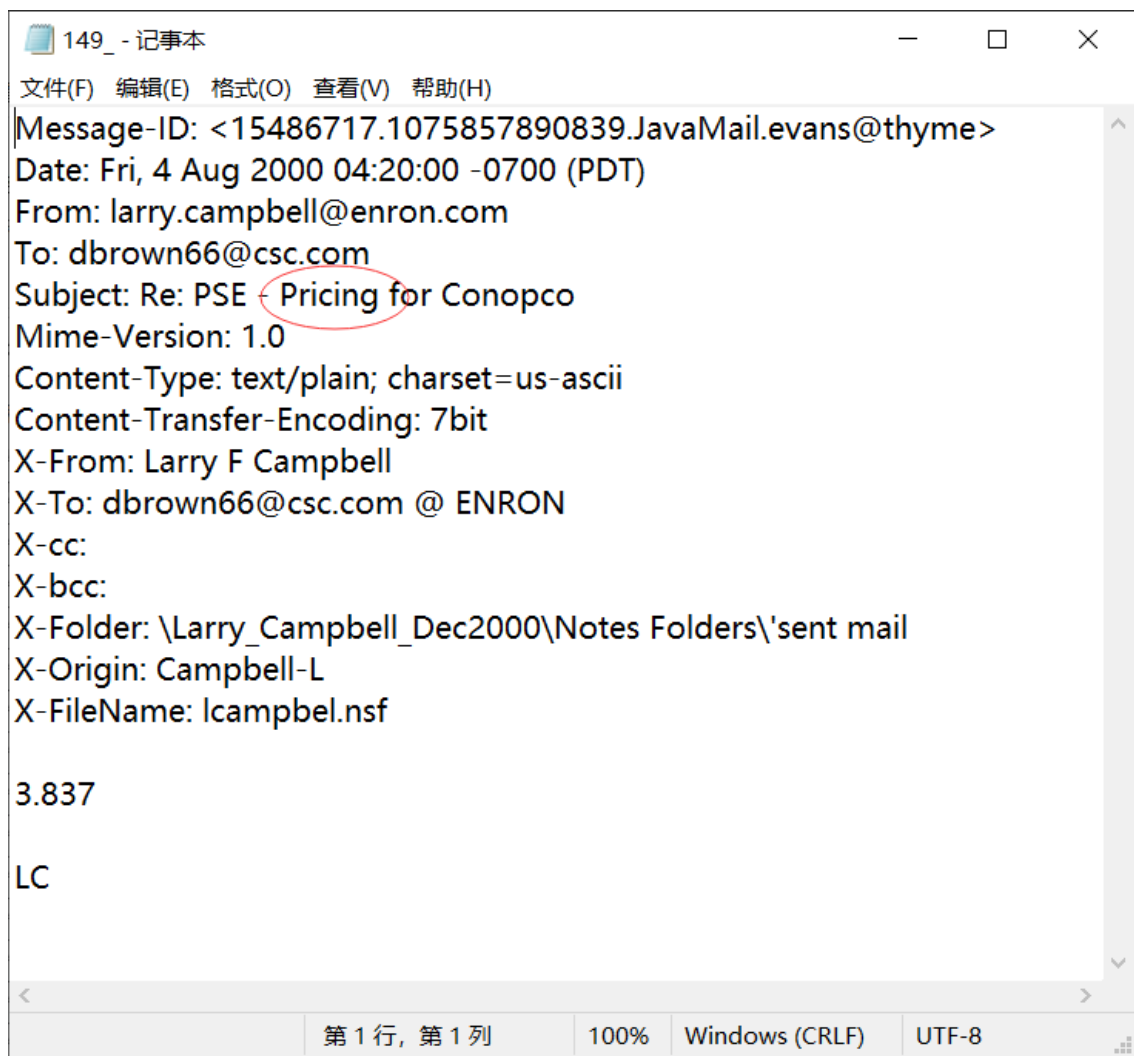
Message-ID: <2722746.1075857886649.JavaMail.evans@thyme>
Date: Fri, 4 Aug 2000 04:20:00 -0700 (PDT)
From: larry.campbell@enron.com
To: dbrown66@csc.com
Subject: Re: PSE - Pricing for Conopco
Mime-Version: 1.0
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
X-From: Larry F Campbell
X-To: dbrown66@csc.com @ ENRON
X-cc:
X-bcc:
X-Folder: \Larry_Campbell_Dec2000\Notes Folders\Sent
X-Origin: Campbell-L
X-FileName: lcampbel.nsf

3.837

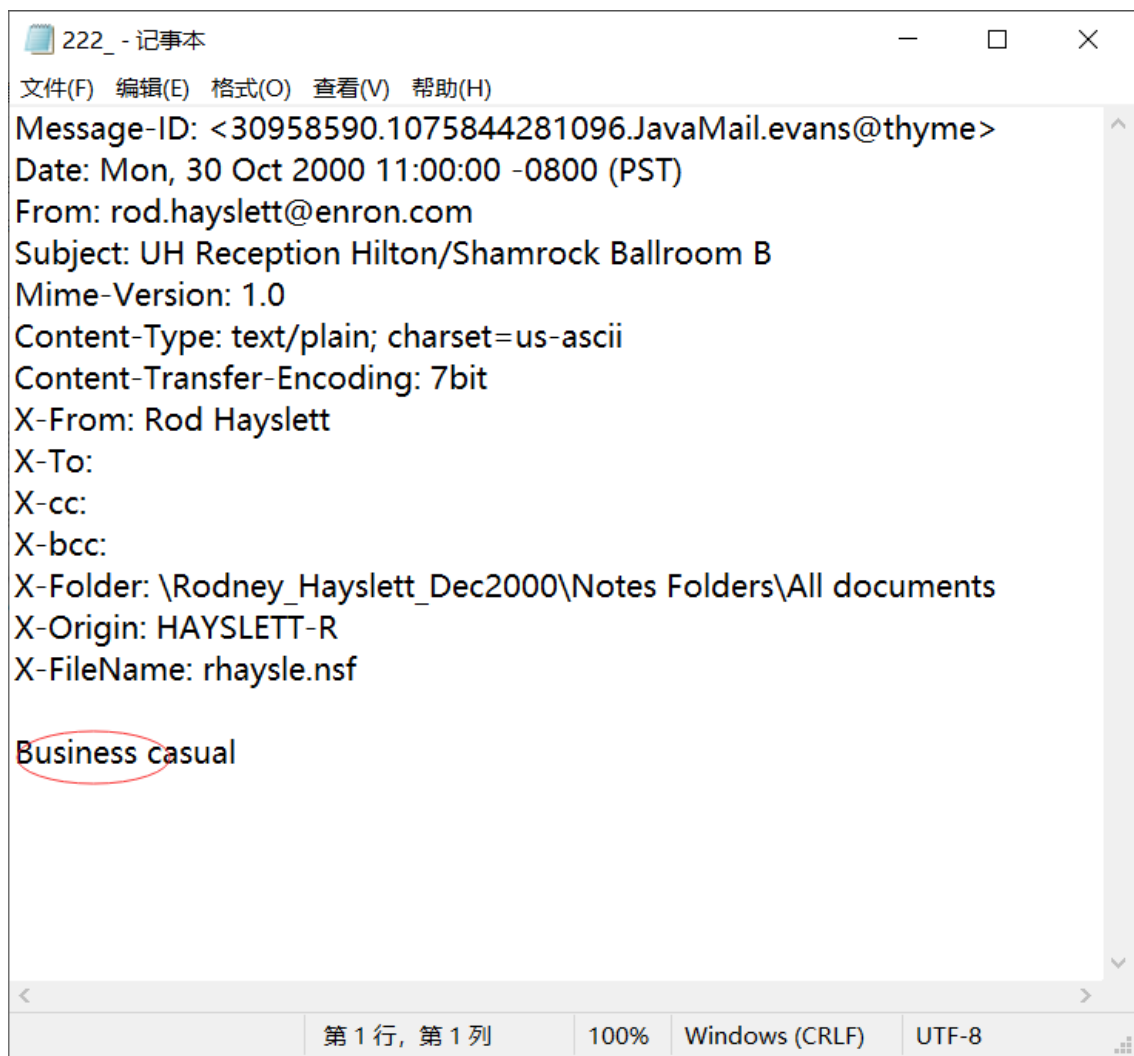
LC

< >

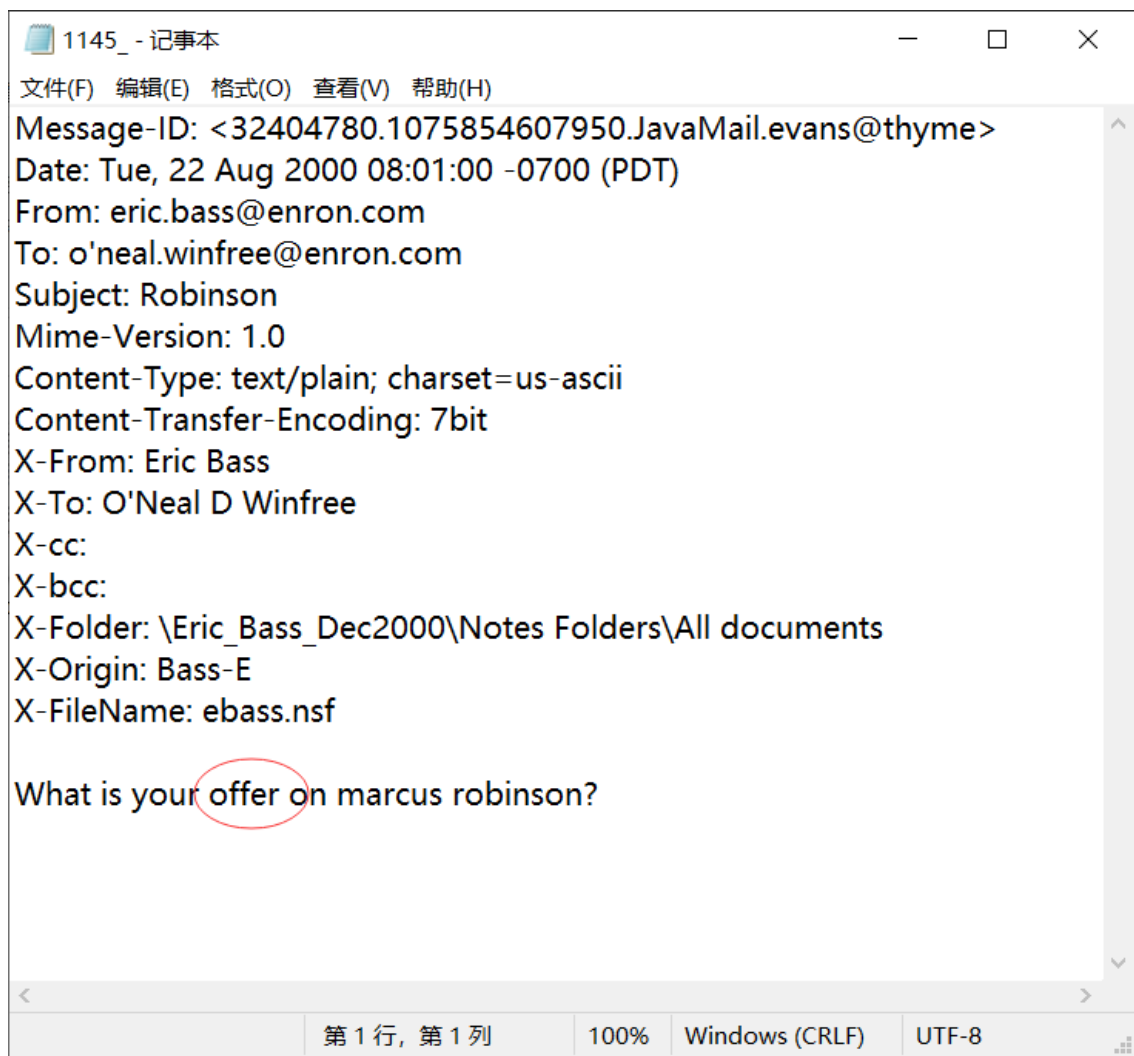
第 1 行, 第 1 列	100%	Windows (CRLF)	UTF-8
--------------	------	----------------	-------



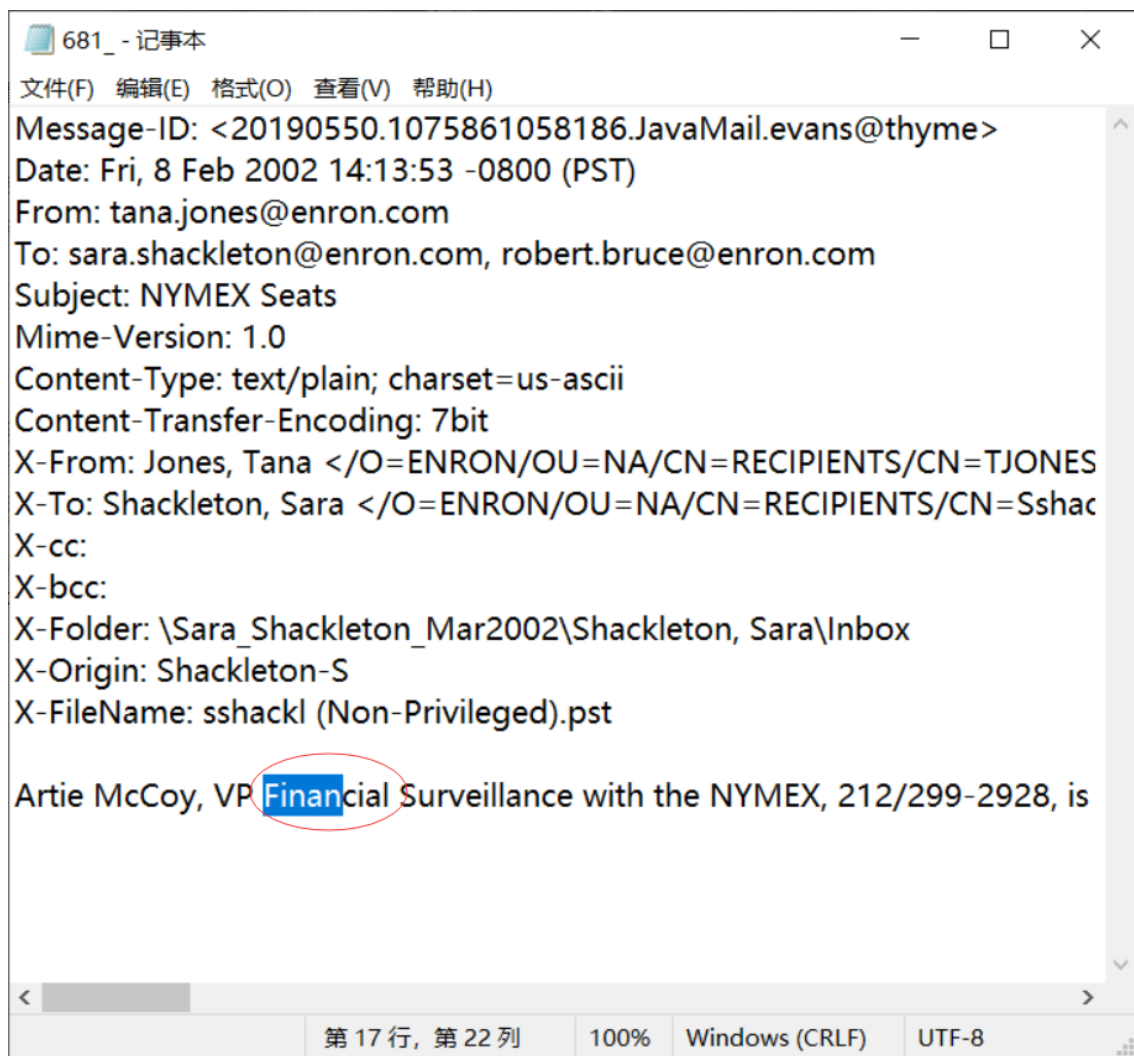
- business:



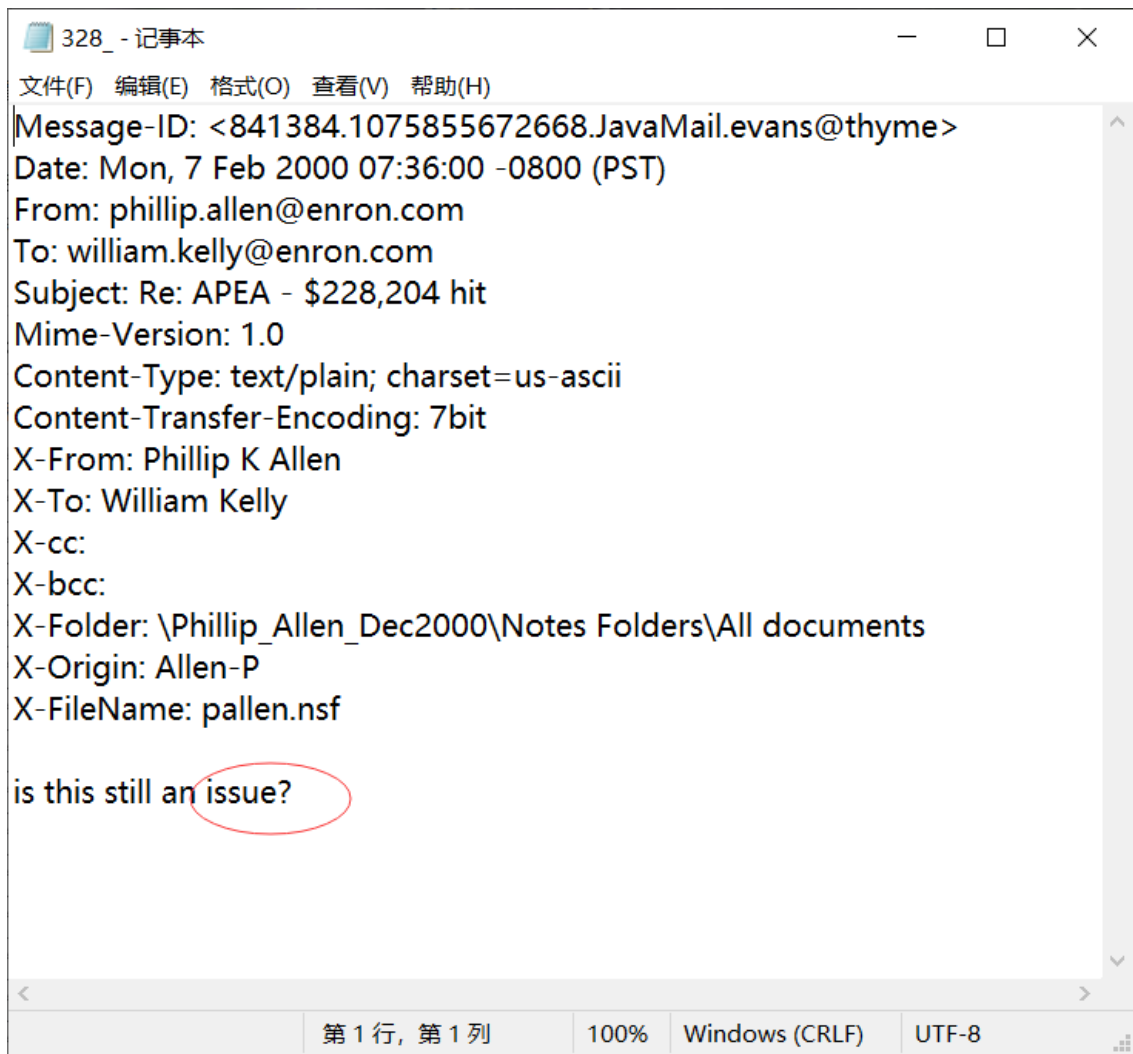
- offer:



- financial:



- issues:



Word2vec查询

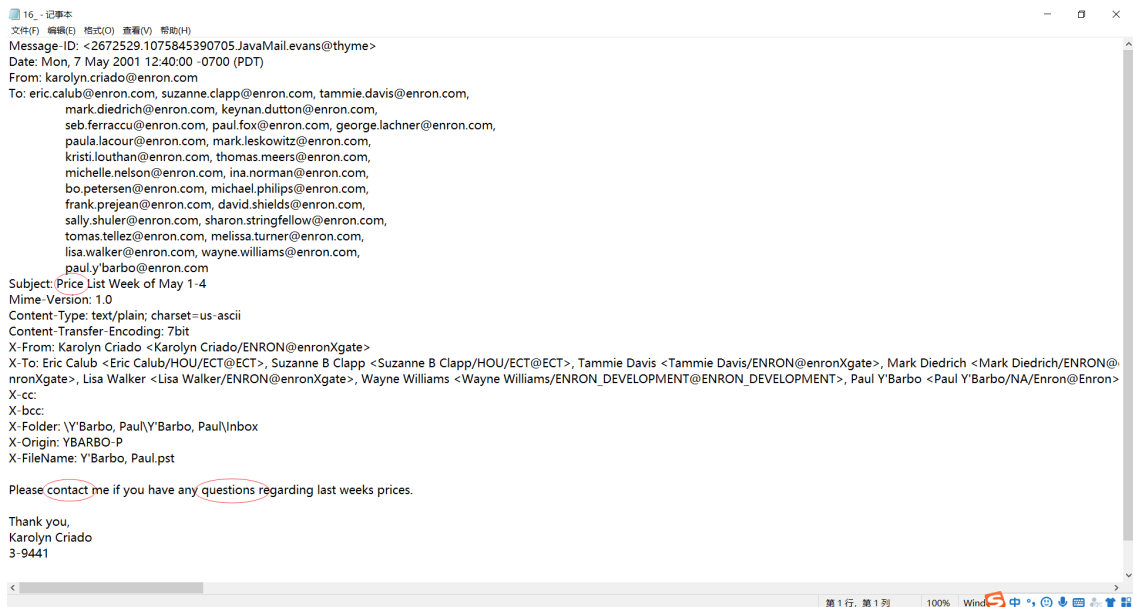
查询词

- 同语义查询

结果

```
PS E:\Web_lab\exp1\src> python -u "e:\Web_lab\exp1\src\word2vec.py"
['contact', 'issu', 'price']
[(514523, 0.84745336), (102961, 0.8360282), (105041, 0.8360282), (108290, 0.8360282), (435137, 0.8352999), (438163, 0.8352999), (438991, 0.8352999),
(102938, 0.8347613), (105042, 0.8347613), (109759, 0.8347613)]
['busi']
[(371923, 1.0), (372885, 1.0), (373241, 1.0), (373962, 1.0), (446603, 1.0), (447722, 1.0), (448002, 1.0), (298828, 0.8392502), (180915, 0.82027996),
(198011, 0.82027996)]
['offer']
[(330761, 0.91444683), (331271, 0.91444683), (331365, 0.91444683), (280676, 0.79880756), (366987, 0.79880756), (280520, 0.79212475), (366955, 0.7907
3966), (513107, 0.7905043), (339272, 0.7872046), (339395, 0.78698313)]
['financi']
[(187325, 0.7457735), (197052, 0.7457735), (255624, 0.7186647), (419897, 0.7114633), (429524, 0.7114633), (187324, 0.7091155), (197051, 0.7091155),
(182791, 0.7033069), (195160, 0.7033069), (121787, 0.70128036)]
['issu']
[(248, 0.81834465), (1711, 0.81834465), (2684, 0.81834465), (260803, 0.7944392), (64072, 0.7927178), (73449, 0.7927178), (228704, 0.7927178), (23279
2, 0.7927178), (237231, 0.7927178), (240104, 0.7927178)]
```

- contact issue price (前三个文档依次如下)



Thanks, Stephanie (3-6004)

----- Forwarded by Stephanie Gomes/HOU/ECT on 08/30/2000
 08:42 AM -----

From: Stephanie Gomes

08/28/2000 01:58 PM

To: Elizabeth L Hernandez/HOU/ECT@ECT

cc:

Subject: Pricing Issue for 7/00 production

Elizabeth,

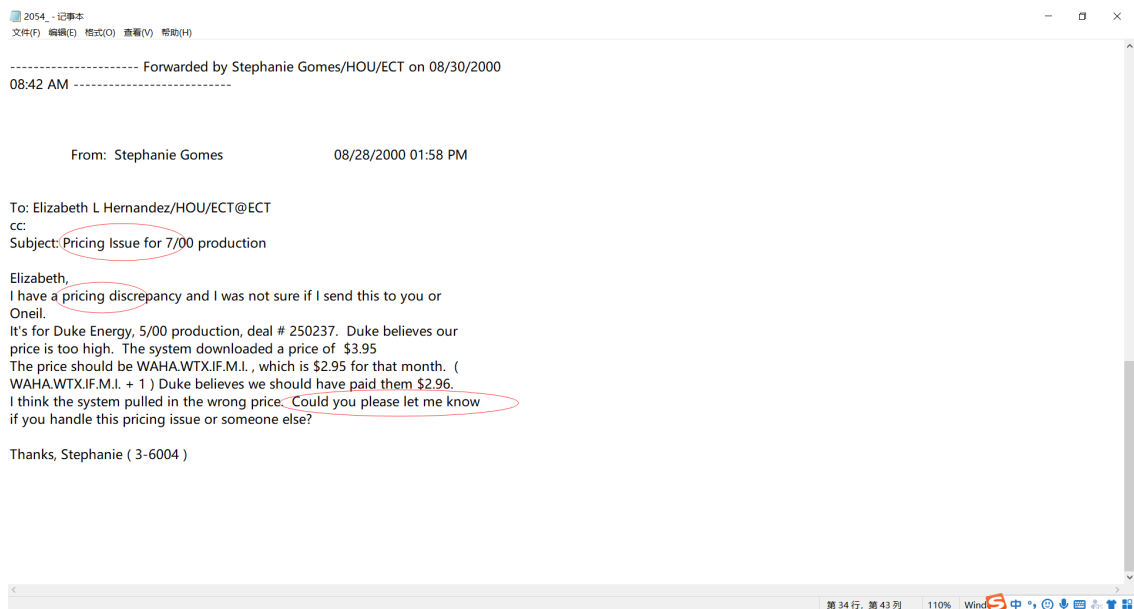
I have a pricing discrepancy and I was not sure if I send this to you or Oneil.

It's for Duke Energy, 5/00 production, deal # 250237. Duke believes our price is too high. The system downloaded a price of \$3.95

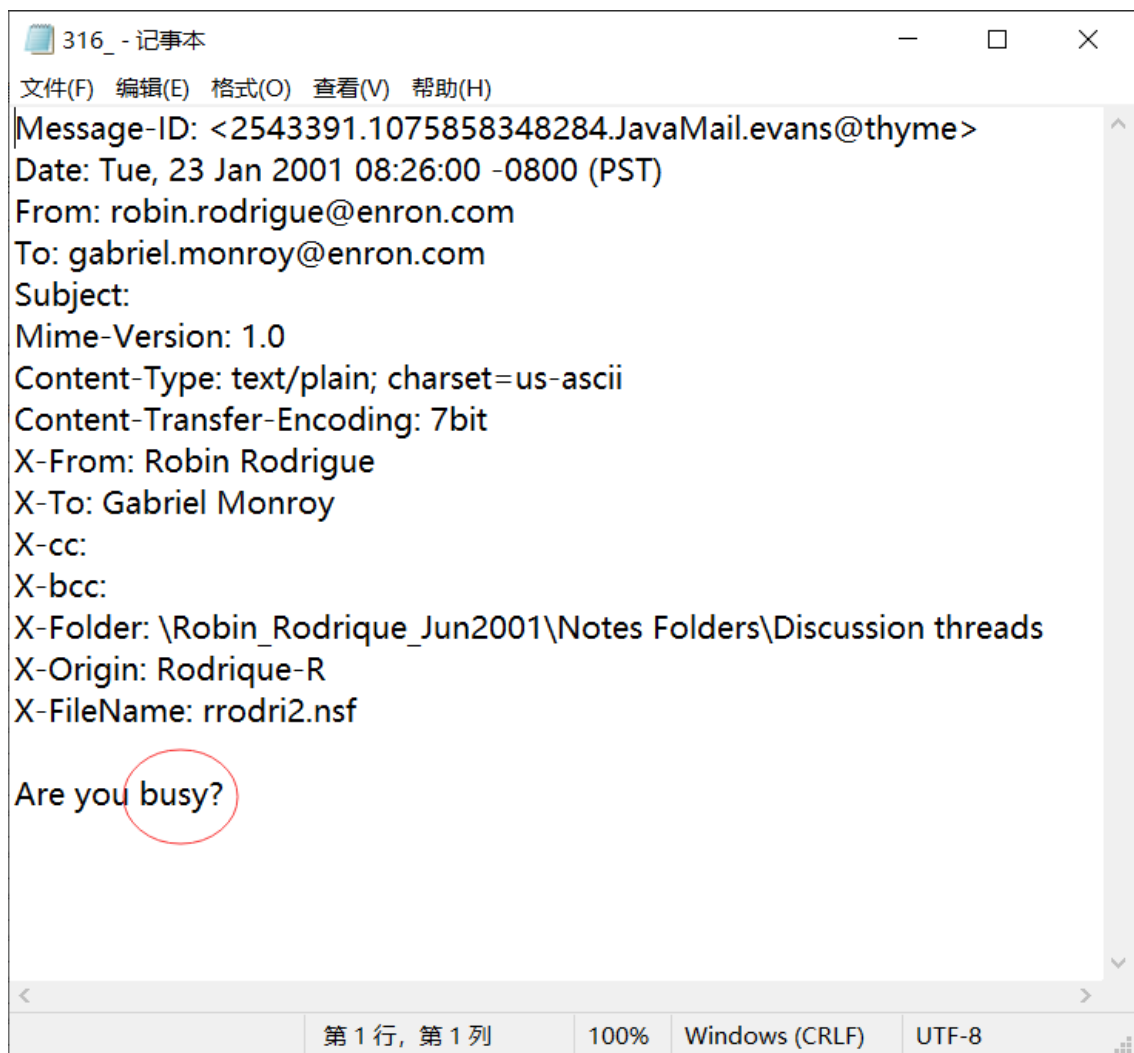
The price should be WAHA.WTX.IF.M.I. , which is \$2.95 for that month. (WAHA.WTX.IF.M.I. + 1) Duke believes we should have paid them \$2.96.

I think the system pulled in the wrong price. Could you please let me know if you handle this pricing issue or someone else?

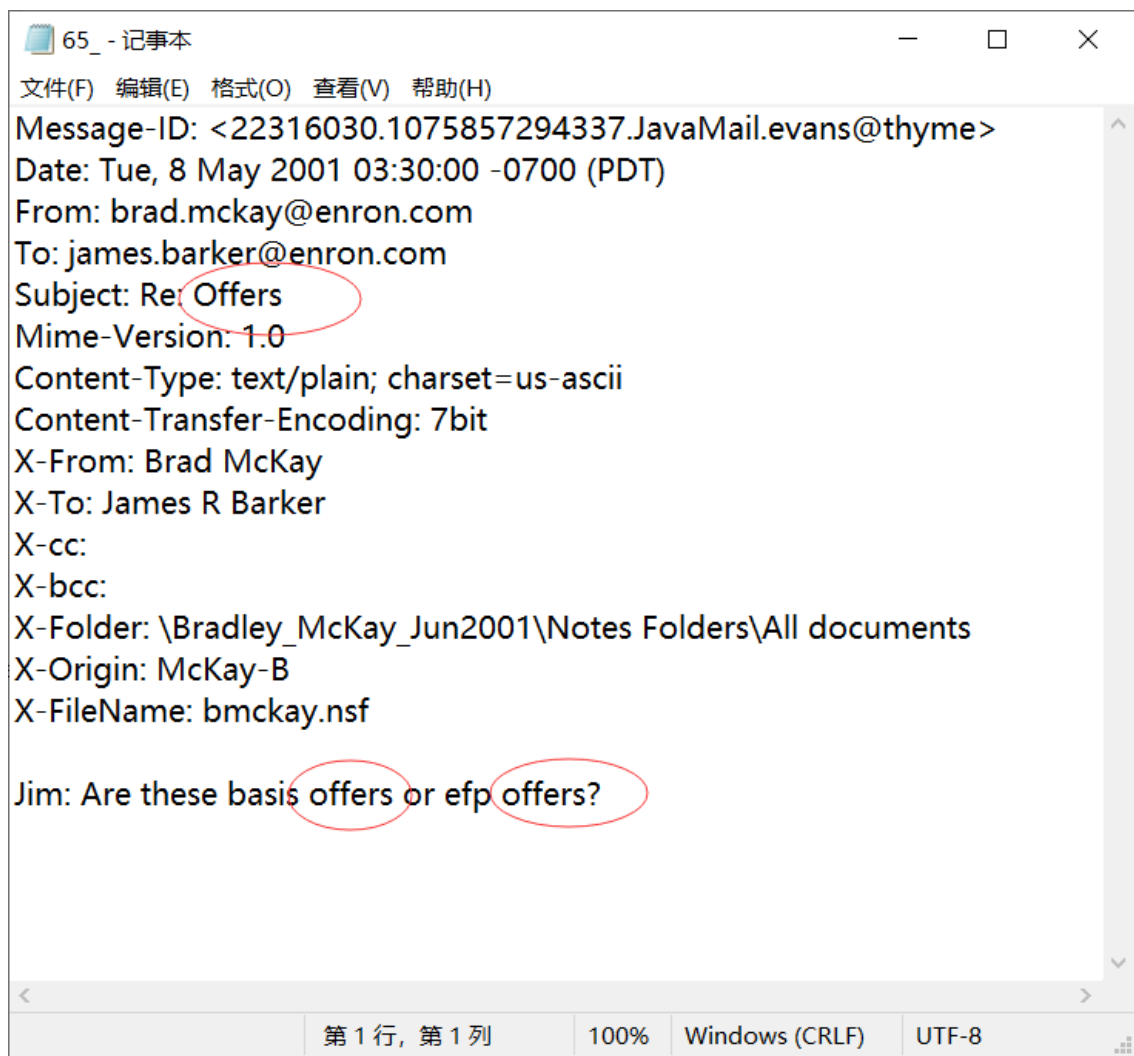
Thanks, Stephanie (3-6004)



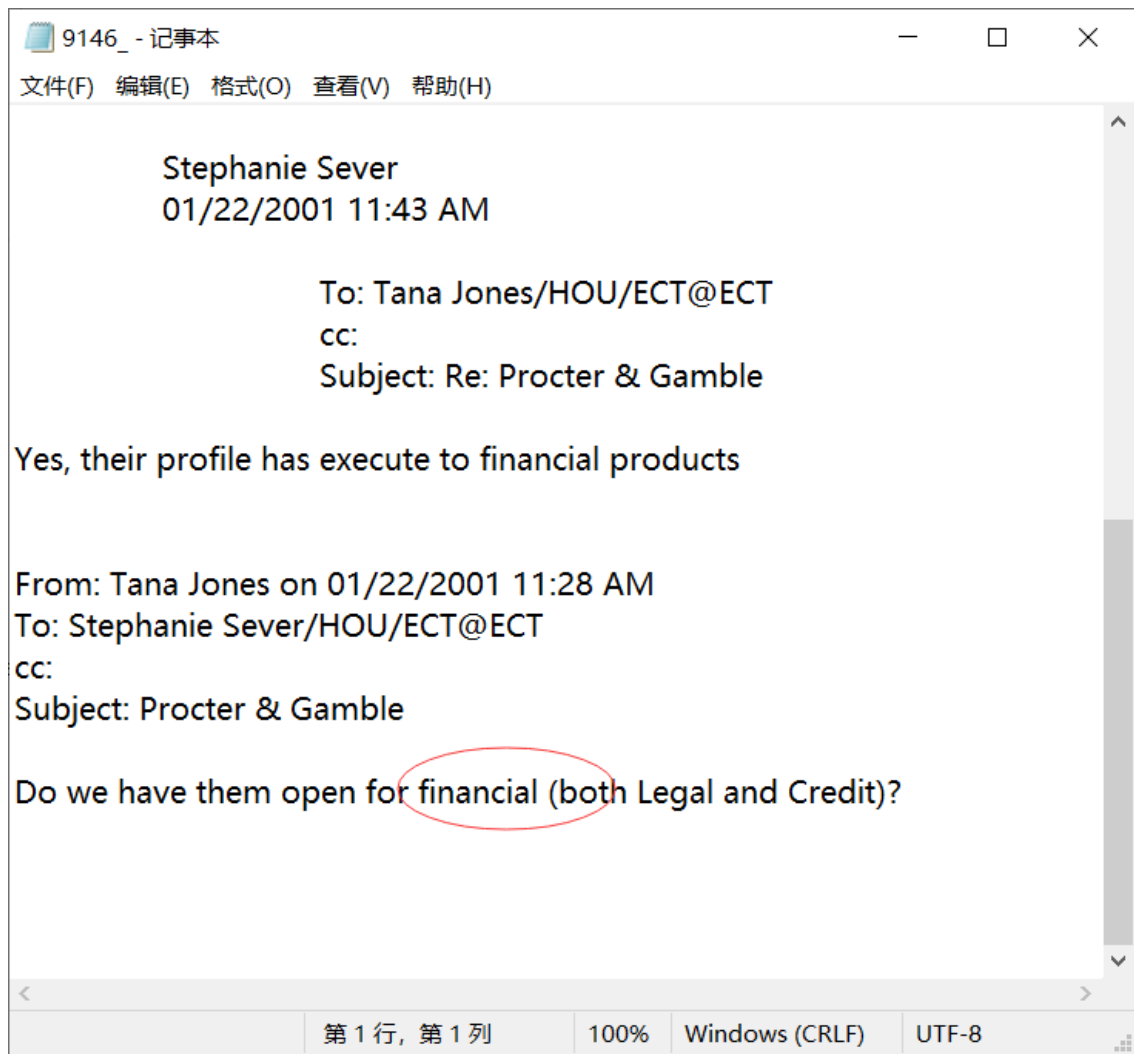
- business:



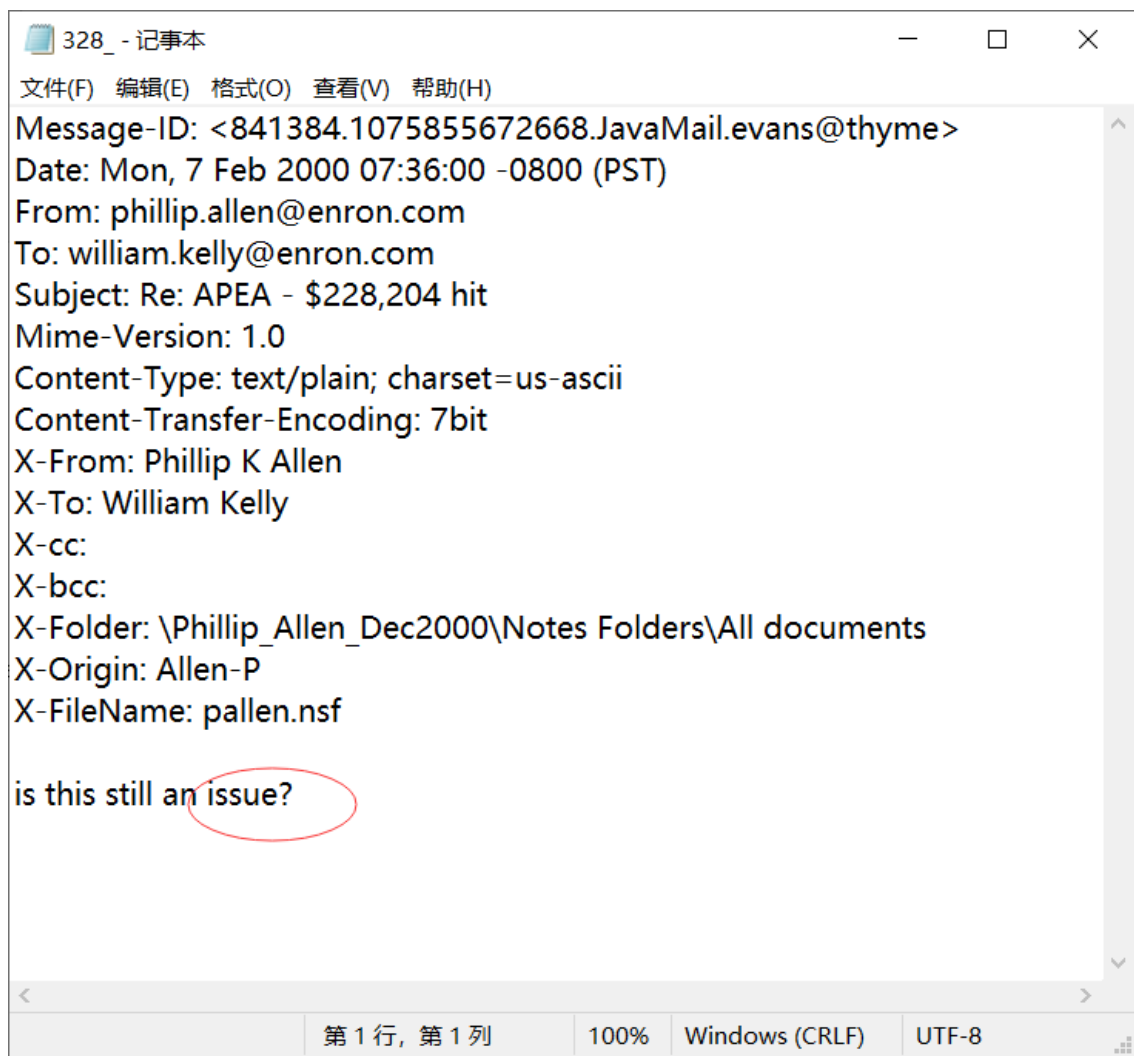
- offer:



- financial:



- issues:



布尔查询、语义查询、Word2Vec查询对比

(以下分析仅限于基于前 1000 个词的查询)

- 对于单个词的查询，后二者返回的文档有很大的重合，二者都倾向于返回“仅”含有该词的文档（即含有要查询的词且尽量少地含有其他非查询词）
- 对于多个词的查询，布尔查询会返回同时含有这些查询词的文档（即用AND连接）；语义查询仍会返回更“干净”的文档，即使该文档中只有一个要查询的词，只要这篇文档中其他非查询词的干扰较少，那么仍会返回这篇文档。这是因为在计算归一化向量时该词项的作用被过度放大。而Word2vec返回的结果更倾向于尽量多地含有查询词汇，而不要求全部出现。且返回的结果会有上下文信息，如上图中的 let me know 在语义上和 contact 很相关。
 - 故对于多个词的查询，word2vec的查询效果要比语义查询好一些，而布尔查询会尽量兼顾所有的查询词
- 对于有些词的查询，如 business，由于词根化过后该词会被变为 busi，对于word2vec来说这个词和 busy 的相似度可能超过了和 business 的相似度，故返回的结果和理想结果有差异

参考文献

- <https://www.zybuluo.com/Dounm/note/591752#1-%E4%BB%8B%E7%BB%8D>
- <https://radimrehurek.com/gensim/models/word2vec.html?highlight=word2vec#module-gensim.models.word2vec>
- <https://blog.csdn.net/u010700066/article/details/83070102>

