使用ElasticSearch辅助的智能客服

杨文俊

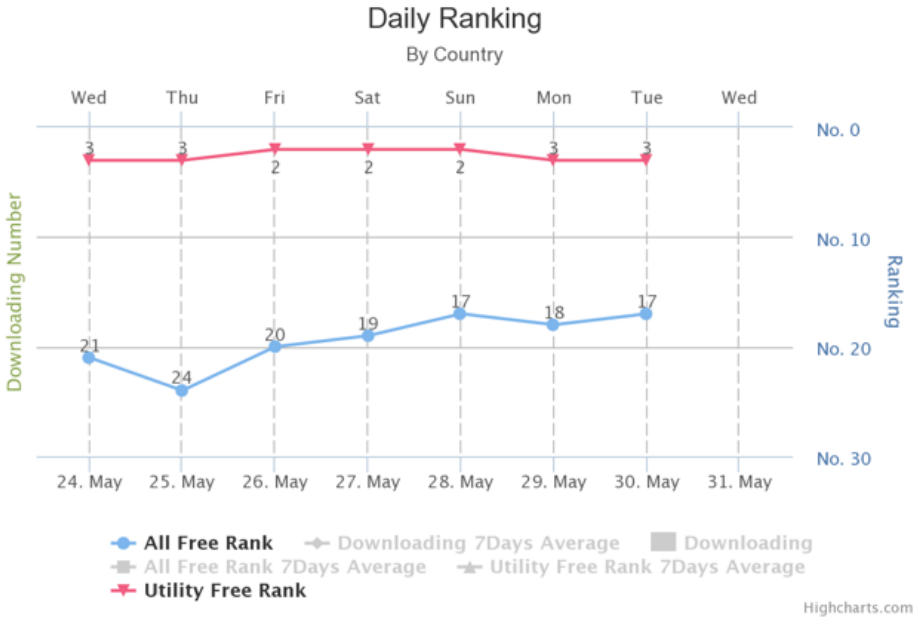
Topic

- 项目介绍与为什么引入ElasticSearch
- NLP 在ES之中的一些实践经验
- 一些其他的优化方法

项目介绍与ES的作用

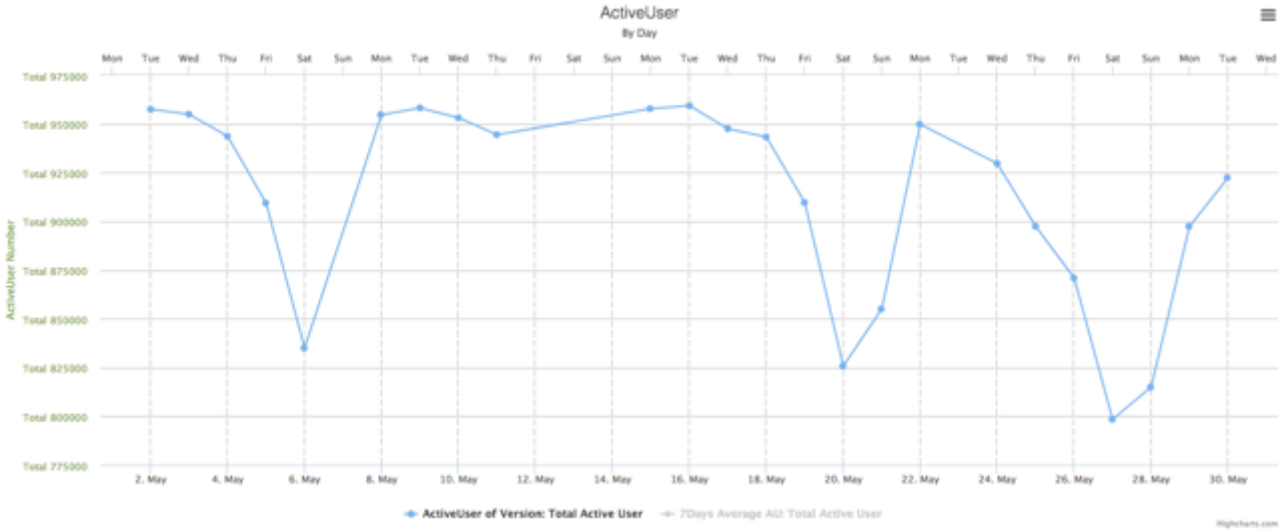
简介 – Dr. Cleaner / Dr. X 系列产品

- Dr Cleaner
  - 在多个国家、地区排名第一的清理类Mac App



简介 – Dr. Cleaner / Dr. X 系列产品

- Dr Cleaner
  - 日活接近百万



幸福的烦恼：客服！

- 多语言、跨时区：
  - 主要客户在海外，其中美国是主要客户，同时也有其他国家跟地区的用户
- 数量跟不上：
  - 随着用户数的急剧增加，客服的数量并不能急剧的增强

解决方案：客服机器人

- 首要能够解决产品相关的问题
- 其次要能解决Mac / iOS 相关的技术问题
- 多语言的问题，通过翻译API翻译成英语再尝试给出解决方案

知识库的构成

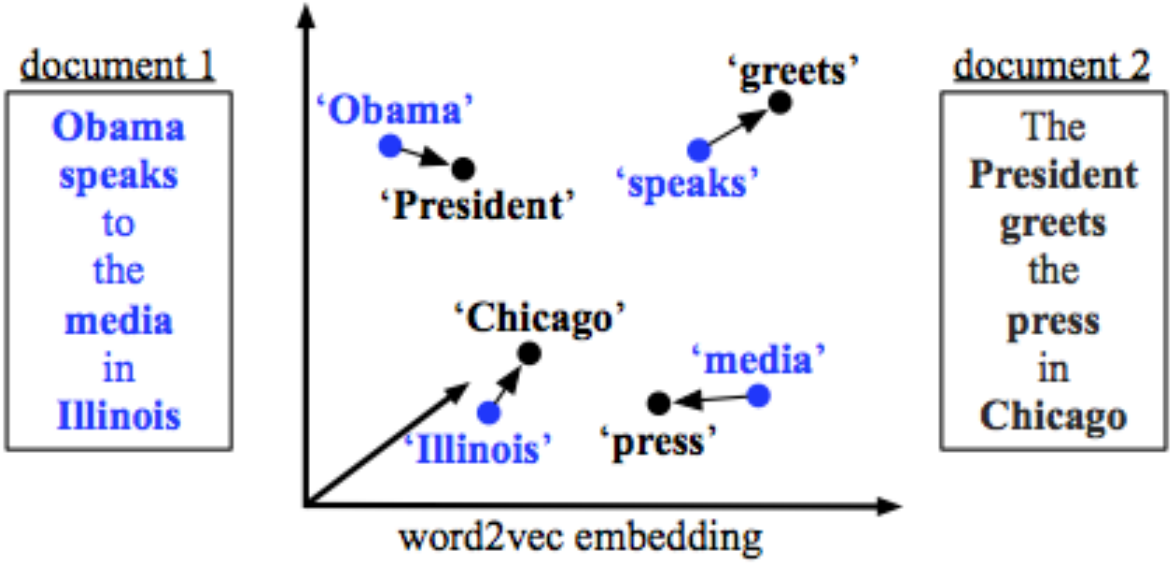
- 各种爬虫：
  - StackExcange Apple分论坛（公开数据源）
  - Apple Discussion
  - Apple Support Center
  - Mac world
  - WikiHow
  - ….

文档搜索

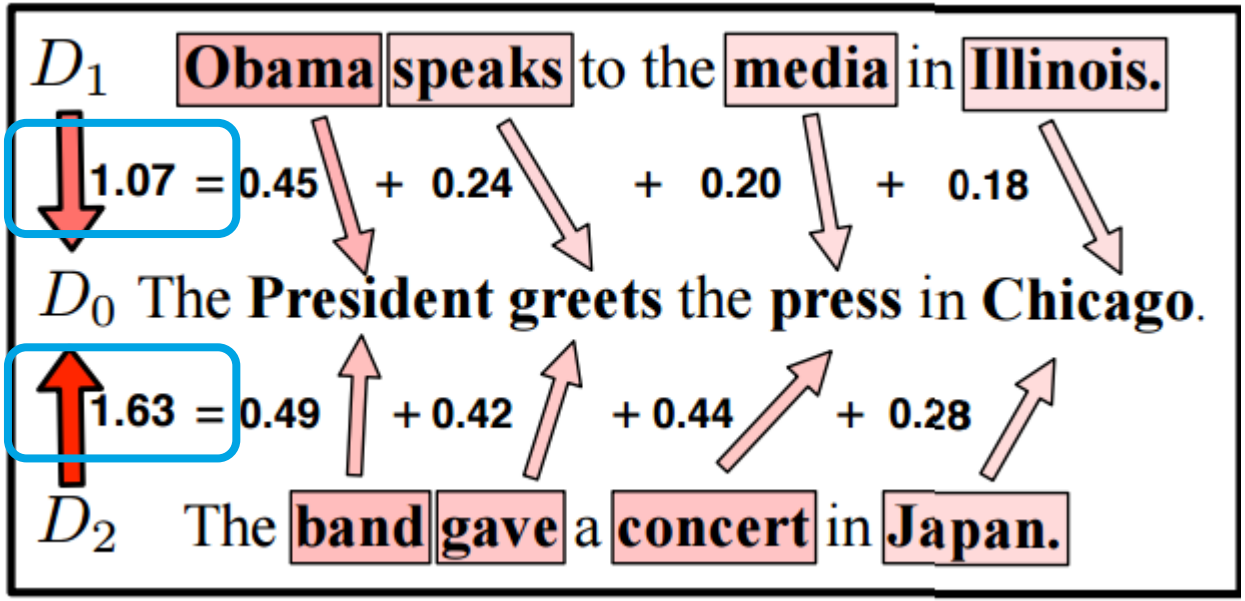
- 直接使用ES?
  - 距离语义还是太远了！

解決方案1：WMD

- Word Mover Distance
  - Word distance + Word2vec

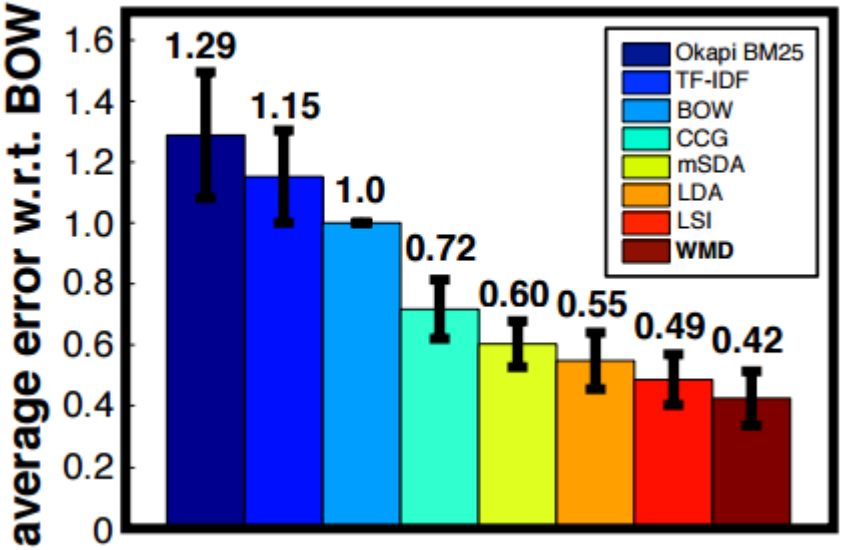


Sample:



WMD 是个好算法

- Stat-of-the-art

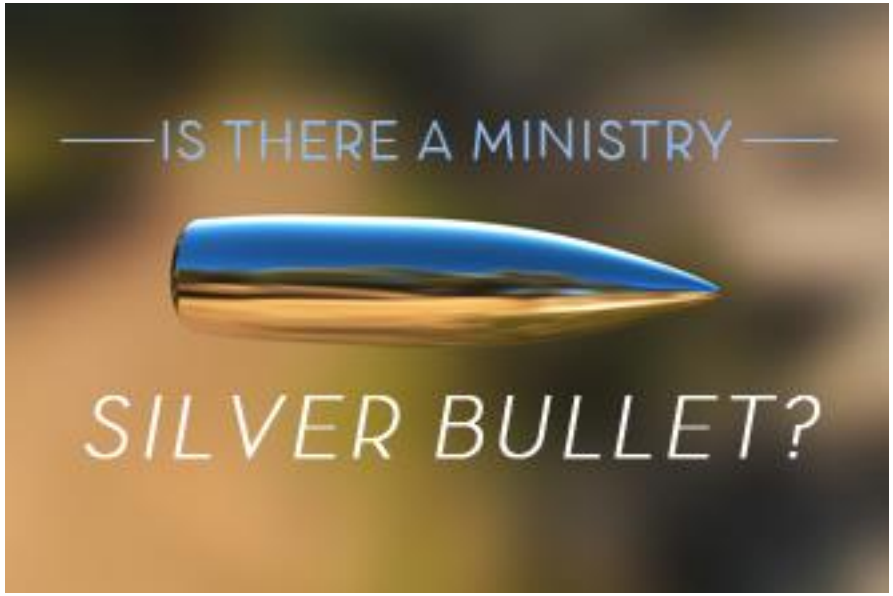


WMD也有明显缺点：慢！

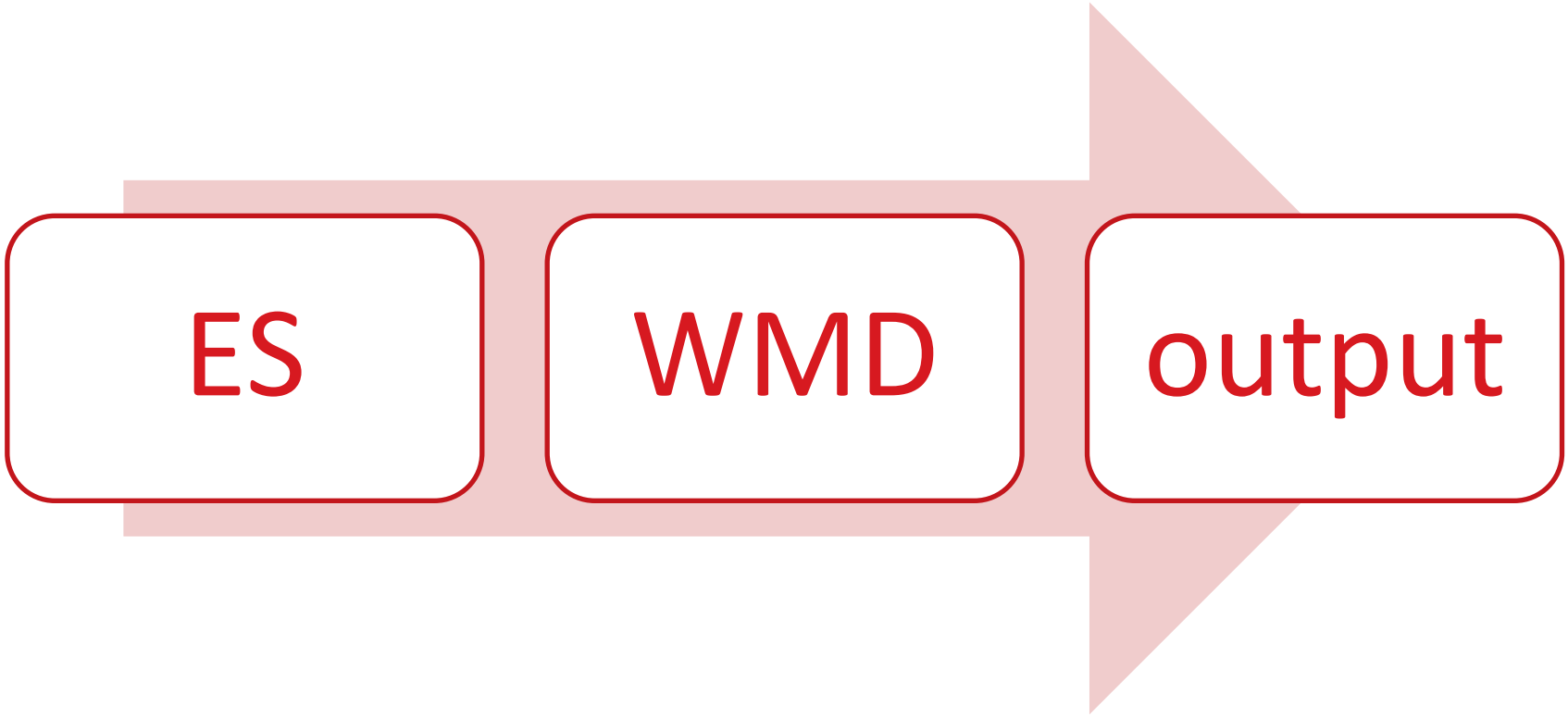
- 算法复杂度:
  - $O(p^3 \log p)$
  - p: 需要计算的词的个数

WMD 也不是银弹

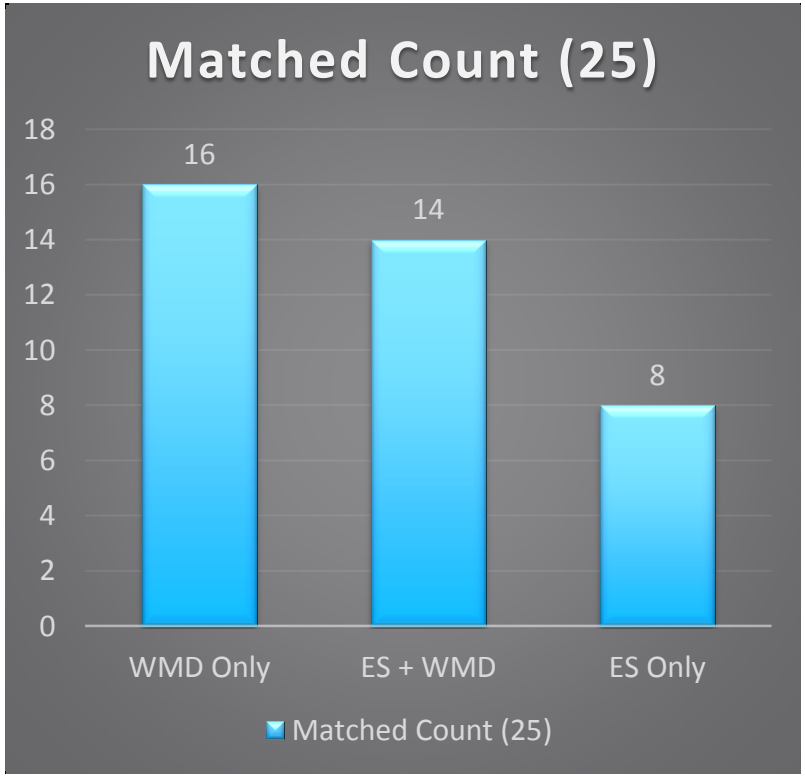
- 查询:
  - How to diagnose WI-FI connection
- 一些不太好的结果:
  - *10.10.5 wifi poor*
  - *How to turn off calls, iMessage and texts, but stay on Wifi?*



解決方案2: 使用ES + WMD



结果对比



ES具体操作

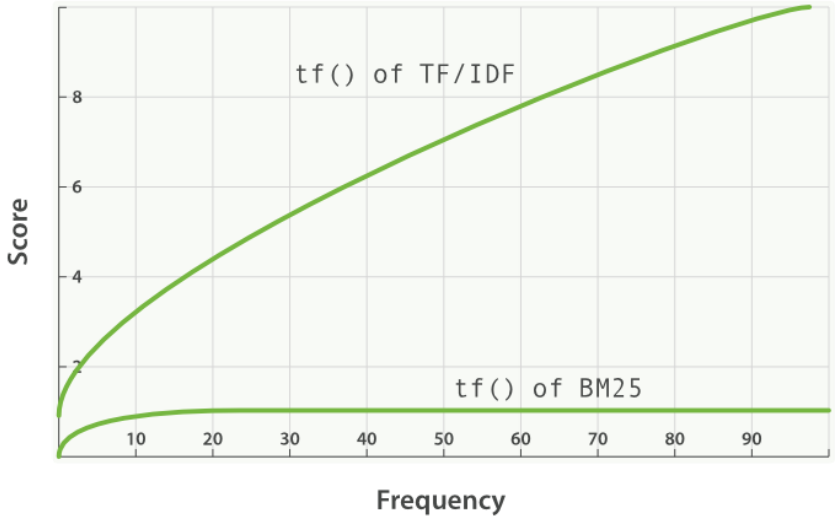
基础：数据的存储与搜索

```
"mappings": {
  "type_name": {
    "properties": {
      "url": {
        "type": "keyword"
      },
      "site_name": {
        "type": "keyword"
      },
      "title": {
        "type": "text",
        "analyzer": "english"
      }
```

优化：BM25 or TFIDF

BM25 or TFIDF

- 实验方法:
  - 修改Mapping，分别使用BM25 与 TFIDF
  - 从知识库之中随机抽取了100 * 10条记录送给ES进行查询
  - 对比两边的结果
- 实验数据:
  - 下一页
- 实验结论:
  - BM25 的作用确实还是比较明显的。推荐使用！



结果对比

Round	Duplicated Count	Total Count	Dup-Ratio
0	937	1000	93.70%
1	904	1000	90.40%
2	921	1000	92.10%
3	916	1000	91.60%
4	907	1000	90.70%
5	893	1000	89.30%
6	935	1000	93.50%
7	926	1000	92.60%
8	934	1000	93.40%
9	906	1000	90.60%
10	905	1000	90.50%
AVG	916.7272727	1000	91.67%

BM25 应用

- BM25 在Mapping之中需要特殊指出

```
'title': {
        'type': 'text',
        'analyzer': 'english',
        "similarity" : "BM25"
}
```

优化：拼写检查与纠错

Google 的例子：


diagonse
全部
图片
新闻
地图
图书
更多
找到约 86,400,000 条结果（用时 0.30 秒）重定向已删除
显示的是以下查询字词的结果： diagnose
仍然搜索： diagonse

我们的

- 一个非常
- Analysis

```
# custom analyzer for analyzing file paths
'analysis': {
    'filter' : {
        "english_possessive_stemmer": {
            "type":       "stemmer",
            "language":   "possessive_english"
        }
    },
    "char_filter": {
        "punc_del": {
            "type": "pattern_replace",
            "pattern": "\\pP|\\pS",
            "replacement": "\s"
        }
    },
    'analyzer': {
        'suggest_analyzer': {
            'type': 'custom',
            'tokenizer': 'whitespace',
            'filter': ['lowercase', 'english_possessive_stemmer
```

我们的方案 ctn'd

- Mapping:

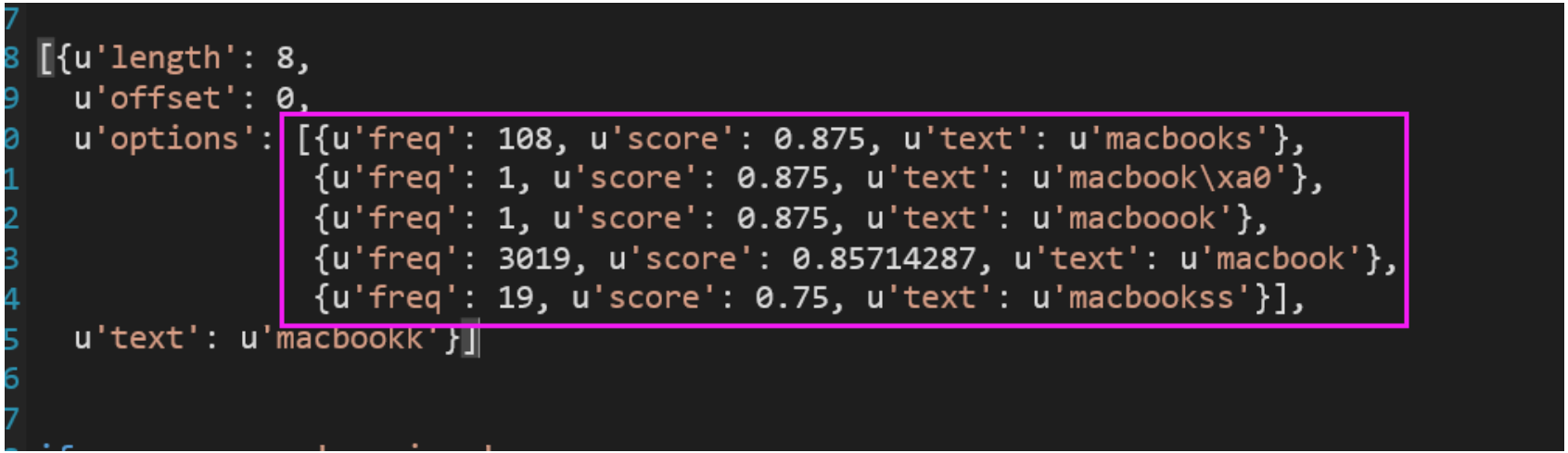
```
},
'mappings': {
    type_name: {
        'properties': {
            'url':       {"type": "keyword"},
            'site_name': {"type": "keyword"},
            'title': {
                'type': 'text',
                'analyzer': 'english',
                'similarity' : "BM25",
                'copy_to' : ["title_4_suggest"]
            },
            'title_4_suggest': {
                'type' : "text",
                'analyzer' : "suggest_analyzer"
            }
        }
    }
```

使用Term Suggester

- 支持直接输入一句话:
  - How to replace ***macbookk*** SSD ?

```
body={
    "suggest" : {
        "title-suggestion" : {
            "text" : word,
            "term" : {
                "field" : "title_4_suggest"
            }
        }
    }
}
```

Results



Term Suggerter 自身调整

- 设定最小出现次数
  - min_doc_freq：3

- 修改默认的相似度/Score计算方法
  - "string_distance" : "jarowinkler",

再次优化的查询结果

- 可以直接取第一个Option了

```
[{u'length': 8,
  u'offset': 0,
  u'options': [{u'freq': 3019, u'score': 0.9875, u'text': u'macbook'},
               {u'freq': 108, u'score': 0.975, u'text': u'macbooks'},
               {u'freq': 15, u'score': 0.9652778, u'text': u'macbookss'}],
  u'text': u'macbookk'}]
```

Failed Cases

- How to user google chrome?
  - User 是一个非常常见的词。。。

改进方法：

- 增加用户行为数据的支撑
  - Google的算法很大一部分就是有用户行为数据
- “瞻前顾后”
  - 不光考虑当前单词，还考虑前后关系！
  - Sample:
    - How to diagnose wifi ***connectity***
    - Correct : ***connectivity***
    - ES result: ***connectify***

优化:输入标准化

很多输入标准不统一

- Example : WIFI
  - Apple Support： WI-FI
    - ES 默认分词器会将其分解为：WI FI 两个词

- Example 2: MacbookPro
  - 标准写法：Macbook Pro

解决方案:

- 使用Gensim生成备选词组
- 使用规则过滤出比较精确的候选词组
- 根据候选词组生成常见的错误写法
- 实时处理用户输入 + 批量处理ES存储的知识库

Gensim 生

- Word2vec

$$\text{score}(w_i, w_j)$$

  - 在gensim

	bigram_count	name
0	38271	os_x
1	11894	com_apple
2	10400	mac_os
3	10048	time_machine
4	9690	app_store
5	7261	hard_drive
6	7256	el_capitan
7	7040	251_deny
8	6964	sandbox_commcenter
9	6568	disk_utility
10	6276	system_preferences
11	6204	0_bytes
12	6058	apple_id
13	5586	macbook_air
14	5336	mountain_lion
15	5195	make_sure
16	5010	file-read-metadata_/system/library/s

规则

- 纯英文字符

- 正则:
  - 主要是品牌名 + 版本号
  - Example: iphone 4s / iphone 7+ …

常见错误写法:

- 连词
  - WIFI -> WI-FI
  - Safe Model -> Safe-Mode
- 缺少空格:
  - Macbook Pro → MacbookPro
  - Mac Mini → Macmini

POS Tagging + 词性过滤

POS Tagging 简介

- POS : part-of-speech
- 作用:
  - 获取每个词的词性
- Example:
  - How to unmount SMB in macbook ?

```
[('How', 'WRB'),
 ('to', 'TO'),
 ('unmount', 'VB'),
 ('SMB', 'NNP'),
 ('in', 'IN'),
 ('macbook', 'NN')]
```

Why ？

- 好处：
  - 1. 降低WMD的计算强度
  - 2. 提高准确率
- Example:
  - How to unmount SMB in macbook ?
  - 重要的词：
    - Unmount / SMB / macbook

解决方案:

- 当前方案：
  - Python NLTK 分析过滤
  - ES 存储结果

- 推荐方案:
  - ES 一条龙: 分析、过滤、存储
    - 需要自行编写ES的Pos插件

推荐方案的优点:

- 性能:
  - java实现的东西一般来说要比纯python的快
  - 特别是比较消耗CPU计算资源时候
- 简单:
  - 逻辑就不需要在ES + Python两边同时维护
- 节省空间,
  - NLTK的模型文件也比较大
  - 多个Docker镜像就意味着占用多个内存、磁盘

一个潜在的坑：Keey_types

- keep_types
  - 官方例子
- 并非POS Tagging
  - 自定义的一些规则
  - 官方链接

```
},
"filter" : {
    "extract_numbers" : {
        "type" : "keep_types",
        "types" : [ "<NUM>" ]
    }
}
```

优化：同义词

基于Word2vec的同义词

- 人为的定义同义词很难
  - 只能有限的几个，
  - Example: UK = United Kingdom
- 使用Word2vec生成”同义词”
  - gensim: model.most_similar_cosmul

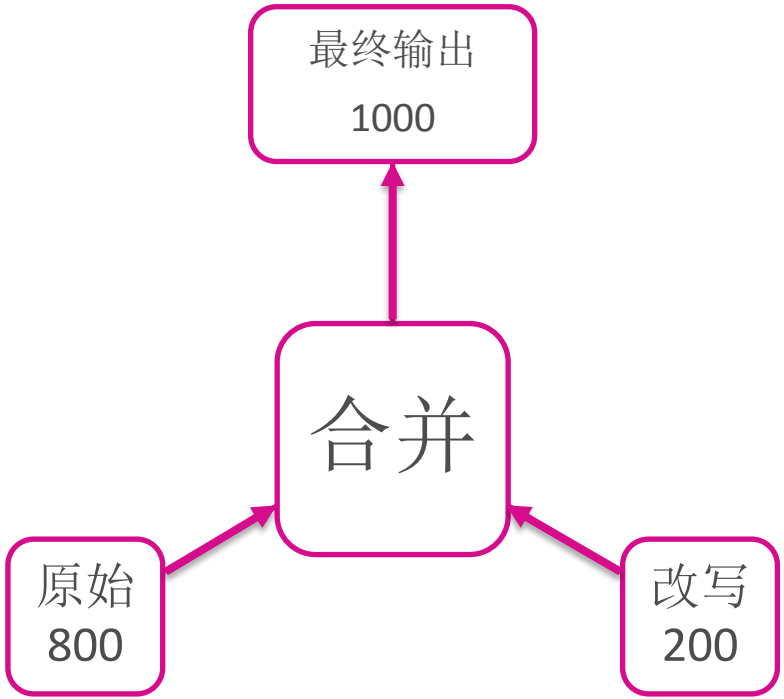
Example: Macbook

```
model.most_similar_cosmul(positive=["macbook"])
[(u'mbp', 0.7659705281257629),
 (u'imac', 0.7585741281509399),
 (u'mac', 0.7414194345474243),
 (u'macbooks', 0.7391695976257324),
 (u'mba', 0.7358759641647339),
 (u'sequel', 0.734239459037808),
 (u'mid', 0.725445032119751),
 (u'gapped', 0.7245665788650513),
```

查询改写方案:

- How to replace macbook SSD
  - POS 结果:
    - Replace macbook ssd
  - Word2vec 改写 + word count:
    - Replace → Replacing
    - Macbook → mbp
    - Ssd → HDD

组合查询:



其他一些优化

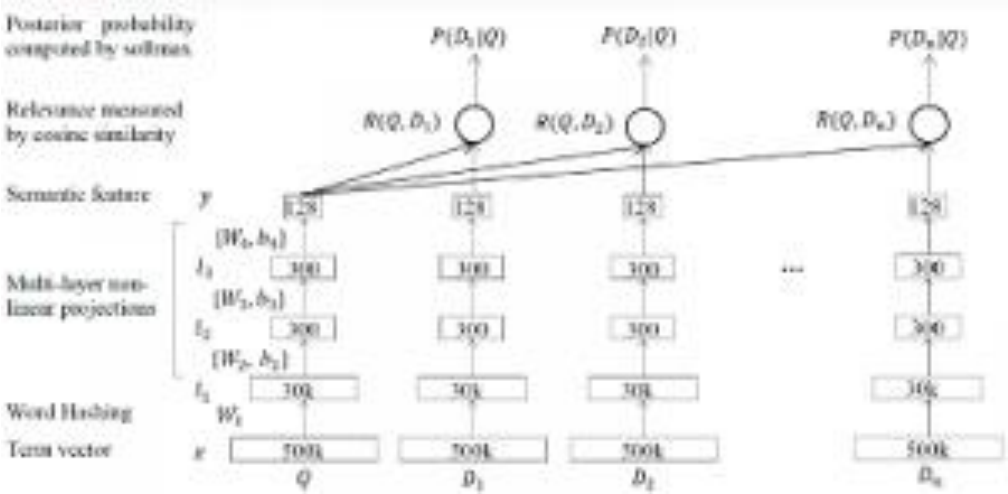
LTR: Learning to Rank

- 基于Machine Learning的重排序
  - 从Q / R 生成feature set
  - 从日志生成数据集
- 模型按照预测的点击概率重新排序

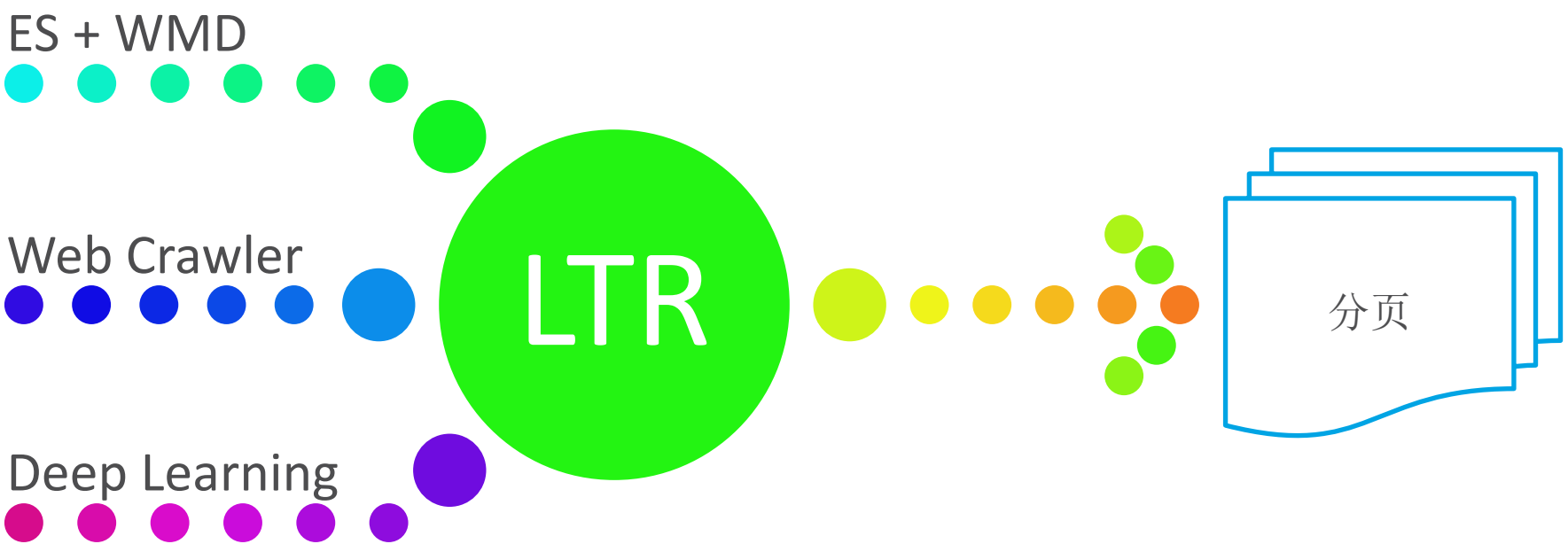
Question	Response	Score
Q1	R1	1
Q1	R2	1
Q1	R3	0
Q1	R4	0
Q1	R5	0

深度学习

- 参考微软的DSSM



3个臭皮匠



Live Demo: (EN Only)

- http://linode-us-east.flyml.net/index.html

Thank you!

TREND MICRO

backup

优化：LTR

- LTR: Learning to Rank
  - 使用ML的方式对TopN 结果重排序
- 开源插件：
  - https://github.com/o19s/elasticsearch-learning-to-rank
- 训练模型数据来源：
  - Youtube 爬虫

优化: 专有名词 / 词组

- 举例:
  - 产品名 : iphone 4s, iphone 7+
  - 默认处理结果: iphone 4 s, iphone 7 (其中+被删除了)
- 解决方案:
  - custom filter: keyword_marker

优化:专有名词 – 产生方法

- Word2vec 论文之中作者提供的方法:

$$\text{score}(w_i, w_j) = \frac{\text{count}(w_i w_j) - \delta}{\text{count}(w_i) \times \text{count}(w_j)}.$$

  - 在gensim之中有现成的实现方法

方案1: 组合

- 方案:
  - 1. Python NLTK 进行词性过滤
  - 2. ElasticSearch 存储过滤之后的结果
- 优点:
  - 实现简单
- 缺点:
  - 运维上面比较复杂,
  - 性能稍差

方案2: ES 一条龙

- 方案:
  - 编写ES的plugin实现POS Tagging Token Filter
  - 在导入数据的时候，只需修改mapping即可
- 优点:
  - 高效 / 方便 / 运维相对容易一些
- 缺点:
  - 需要自行开发，并且多个ES版本需要单独编译

引言

- 本次分享主要会介绍一下ES是如何帮忙我们完成NLP的任务的。
  - 在做NLP相关任务的时候，ES 的相似度算法并不足以支撑用户的搜索，需要使用一些与语义相关的方法进行改进。但是ES的很多特性对我们优化搜索体验是非常有帮助的。