系統環境

- VMWare Workstation
  - Intel® Core™ i5-7500 CPU @ 3.40GHz × 4 (4 cores for VM)
  - 8GB Ram for VM
  - 100 GB SSD for VM
  - Ubuntu 16.04 LTS

使用工具與套件

- CppJieba
- C++ 5.4
- JsonCpp
- OpenCC
  - 把 cppJieba 的字典轉成繁體
- Elastic Search 6.2.2
- Solr 7.3
- Python 3.6
- Java 8

測試資料與前置處理

ettoday 11GB file，實際有成功使用到的record數為 4812350個。

有部分 utf8 解析會失敗的 record，我在 轉json 或是 做斷詞 出現 exception 時，會直接忽略這些資料。

```
const char *recordHeading[HEADINGCOUNT] = {
    "Gais_REC",  "url",       "MainTextMD5", "UntagMD5", "SiteCode",
    "UrlCode",   "title",     "Size",        "keyword",  "image_links",
    "Fetchtime", "post_time", "Ref",         "BodyMD5",  "Lang",
    "IP",        "body",      "botVer",      "Time"
};
```

整個資料裡有的欄位非常的多，以上是我分析資料時所獲得的欄位。

我有擷取出來使用的欄位分別為 url, title, keyword, image link, body。對於 body 我有先做斷詞之後，分別每 10萬筆 record 存放到一個 json 檔案中。

C++ 轉換成 Json 格式有一個很棒的 JsonCpp library 可以用。只要宣告 json type 的變數之後，就當作 c++ map 來放資料，最後再用 dump() 即可得到 Json format std::string!

---

cppJieba 斷詞

```
./jieba  1518.52s user 105.91s system 99% cpu 27:15.16 total
```

真的非常的快！10000 筆 record ，如果只針對 body 做斷詞的話，大約是在 2.5 秒可以處理完畢。

cppJieba 內建很棒的字典，Dict, Hmm, IDF, StopWord 全數都有提供！雖然全部都是簡體中文，但是使用 OpenCC 轉換成繁體中文之後，就可以正常的拿來使用了！

使用 HMM 的時候效果我覺得好上不少，以下例子分別是 字典沒有的政府部門詞彙 和 數字。

Original	HMM On	HMM off
衛福部疾管署	衛福部 疾管署	衛 福 部 疾 管 署
共達2978人	共達 2978 人	共 達 2 9 7 8 人

---

cppJieba 坑

正常 compile cppJieba 之後，我們如果直接 include cppJieba 的 header file，會得到 fatal error: 'limonp/Logging.hpp' file not found 錯誤。我們需要複製 deps/limonp 資料夾 到 include/cppjieba 底下，這樣就能解決問題。（我不知道為何他們官方的 makefile 就可以正常 compile，也許他 cmake 有多設定搜尋路徑吧）

Elastic search 和 Solr 的 insertion 和 search 時間比較

Operation	Elastic search	Solr
Insertion	30分鐘 (python Elastic-search套件)	1 小時 10 分鐘 (POST)
Search (隨機從 keyword 欄位抓取 1000 個 不重複 keyword)	15 秒	23 秒

Elastic search 在效能上感覺比起 Solr 好上許多，但是 Solr 的文件筆 elastic search 好上非常多。

Solr 內帶有一個網頁版管理介面，不像 Elastic search 還要自己裝 Kibana。