自然語言處理主題

研究報告

資工二 1107110539 張仲恩

簡介 CkipTagger

CkipTagger 是中研院目前開發的最新而且也是正確率最高的中文辨識程式，它所包含的工具 斷詞(WS)、詞性標注(POS)、實體辨識(NER)。

ASBC 4.0 測試集 (50,000句)

Tool	(WS) prec	(WS) rec	(WS) f1	(POS) acc
CkipTagger	97.49%	97.17%	97.33%	94.59%
CKIPWS (classic)	95.85%	95.96%	95.91%	90.62%
Jieba-zh_TW	90.51%	89.10%	89.80%	--

實體辨識(NER)

相信前兩項應該都蠻容易理解它是做甚麼的，所以先簡單介紹一下實體辨識(NER)，它是被設計用來辨識專有名詞的，這裡的專有名詞不只包含東西還包含人名以及一些外來語，因此這對於人工智慧理解自然語言來說是一項非常重要的資訊。

斷詞(WS)

詞是最小有意義且可以自由使用的語言單位，任何語言處理的系統都必須先能分辨文本中的詞才能進行進一步的處理，這對電腦來說就有點像是標點符號，讓它知道什麼時候該中斷並理解該詞的意義。

詞性標注(POS)

詞性是詞彙基本的語法屬性，通常也稱為詞類。詞性標注就是在給定句子中判定每個詞的語法範疇，確定其詞性並加以標註的過程。

使用方法

下載完相關的套件之後，就可以開始使用它了，這裡先用它的 demo 做測試

步驟一 ：

下載模型並將它載入至程式

模型檔可在幾個不同地方下載。

- iis-ckip
- gdrive-ckip
- gdrive-jacobvsdanniel

您可以用內建 API 下載並解壓縮到指定路徑。

```
# 下載至 ./data.zip (2GB) 然後解壓縮至 ./data/
# data_utils.download_data_url("./") # iis-ckip
data_utils.download_data_gdown("./") # gdrive-ckip
```

```
# 使用 GPU：
#     1. 安裝 tensorflow-gpu (請見安裝說明)
#     2. 設定 CUDA_VISIBLE_DEVICES 環境變數，例如：os.environ["CUDA_VISIBLE_DEVICES"] = "0"
#     3. 設定 disable_cuda=False，例如：ws = WS("./data", disable_cuda=False)
# 使用 CPU：
ws = WS("./data")
pos = POS("./data")
ner = NER("./data")
```

步驟二 ：

按照順序執行斷詞(WS)、詞性標注(POS)、實體辨識(NER)

```
sentence_list = [
    "傅達仁今將執行安樂死，卻突然爆出自己20年前遭緯來體育台封殺，他不懂自己哪裡得罪到電視台。",
    "美國參議院針對今天總統布什所提名的勞工部長趙小蘭展開認可聽證會，預料她將會很順利通過參議院支持，成為
    "",
    "土地公有政策?？還是土地婆有政策。.",
    "… 你確定嗎… 不要再騙了……",
    "最多容納59,000個人,或5.9萬人,再多就不行了.這是環評的結論.",
    "科長說:1,坪數對人數為1:3。2,可以再增加。",
]

word_sentence_list = ws(
    sentence_list,
    # sentence_segmentation = True, # To consider delimiters
    # segment_delimiter_set = {",", "。", ":", "?", "!", ";"}), # This is the defualt set of de
    # recommend_dictionary = dictionary1, # words in this dictionary are encouraged
    # coerce_dictionary = dictionary2, # words in this dictionary are forced
)

pos_sentence_list = pos(word_sentence_list)

entity_sentence_list = ner(word_sentence_list, pos_sentence_list)
```

步驟三(可有可無) :

釋放記憶體

```
del ws
del pos
del ner
```

步驟四 :顯示結果

```
def print_word_pos_sentence(word_sentence, pos_sentence):
    assert len(word_sentence) == len(pos_sentence)
    for word, pos in zip(word_sentence, pos_sentence):
        print(f"{word}({pos})", end="\u3000")
    print()
    return

for i, sentence in enumerate(sentence_list):
    print()
    print(f"'{sentence}'")
    print_word_pos_sentence(word_sentence_list[i],  pos_sentence_list[i])
    for entity in sorted(entity_sentence_list[i]):
        print(entity)
```

簡介 jieba(結巴)

除了中研院研發的 CkipTagger 之外，結巴也是中文的自然語言處理中常見的程式之一，在 CkipTagger 出現之前結巴是中文辨識成功率最高的。它的主要功能有 分詞、關鍵詞提取、詞性標注、添加自定義辭典、並行分詞、命令行分詞等等，其中分詞、關鍵詞提取、詞性標注跟 CkipTagger 的功能比較類似就不再說明了。

添加自定義辭典

使用者可以指定自己自定義的辭典，雖然 jieba 有新詞辨識的功能，但是還是可以自行添加新詞以確保有更高的正確率。

用法 : jieba.load_userdict(file_name) 其中 file_name 為文件名或自定義辭典的路徑，辭典的格式為一個詞佔一行，一行分三個部分詞語、詞頻(可省略,用來計算詞的係數)、詞性(可省略)，空格隔開，順序不可顛倒。

```
创新办 3 i
云计算 5
凱特琳 nz
台中
```

並行分詞

將文本按行分隔後，把各行的文本分配到多個 python 進程並行分詞，以此加快分詞的速度。

用法 ：

- `jieba.enable_parallel(4)` # 开启并行分词模式，参数为并行进程数
- `jieba.disable_parallel()` # 关闭并行分词模式

命令行分詞

簡單來講就是打一串文字來進行分詞。

用法 :其中 new.txt 為檔案名。

使用示例：`python -m jieba news.txt > cut_result.txt`

這邊順便也將 jieba 的命令行介面也展示出來

```
固定参数:
  filename              输入文件

可选参数:
  -h, --help            显示此帮助信息并退出
  -d [DELIM], --delimiter [DELIM]
                        使用 DELIM 分隔词语，而不是用默认的' / '。
                        若不指定 DELIM，则使用一个空格分隔。
  -p [DELIM], --pos [DELIM]
                        启用词性标注；如果指定 DELIM，词语和词性之间
                        用它分隔，否则用 _ 分隔
  -D DICT, --dict DICT  使用 DICT 代替默认词典
  -u USER_DICT, --user-dict USER_DICT
                        使用 USER_DICT 作为附加词典，与默认词典或自定义词典配合使用
  -a, --cut-all         全模式分词（不支持词性标注）
  -n, --no-hmm          不使用隐含马尔可夫模型
  -q, --quiet           不输出载入信息到 STDERR
  -V, --version         显示版本信息并退出

如果没有指定文件名，则使用标准输入。
```