

# 國立高雄師範大學

軟體工程系

專題製作報告

以 Pypinyin 實作歌詞產生器

指導教授：李文廷 博士

學 生：鍾弘浩 撰

中 華 民 國 111 年 10 月

# 目錄

壹、前言	3
一、研究動機	3
二、研究目的	3
三、研究流程	3
貳、文獻探討	4
一、Python 中拼音庫 PyPinyin 的用法	4
二、參考 python 中 collections 資料結構	5
三、選擇利用 collections 導入 defaultdict 型別	5
四、LSTM 介紹	7
五、RNN 計算方式	7
六、LSTM 回推方式	8
參、研究方法	9
一、系統架構	9
二、程式片段說明	9
肆、研究分析與結果	15
一、系統流程	16
伍、研究結論與建議	17
陸、參考文獻	18

# 壹、前言

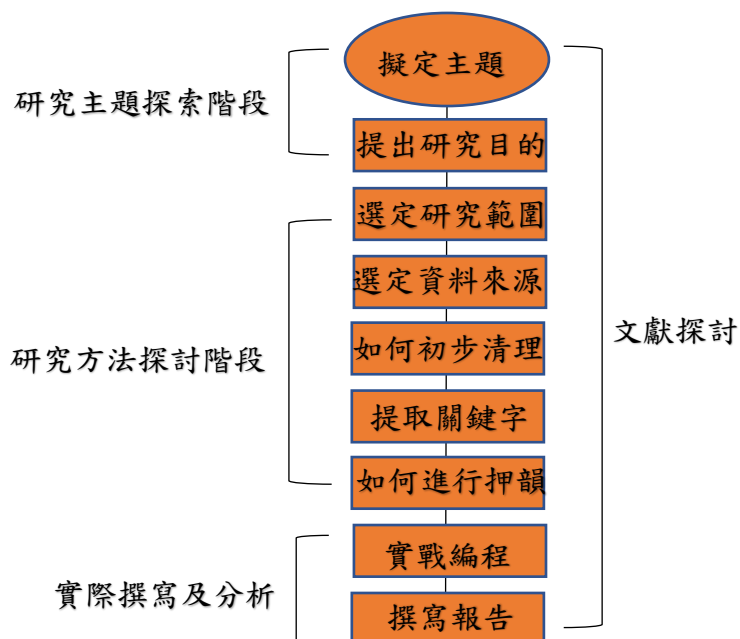
## 一、研究動機

AI 世代的來臨，令我想到的就是許多產業將以 AI 取代，許多需要智慧思考的事物不免俗的都會面臨到 AI 的訓練範疇，當然歌手也一樣，作曲家的判斷思維、創作思維以及如何判斷一句歌詞從產出到完成是如何進行，當然我對如何構詞也有一點興趣，畢竟從以前就是吉他社，如果能夠自己作詞作曲很厲害也很敬佩，所以就產生了這個想法，但是經過自己思考實在不知道怎麼作詞，只知道需要押韻，但是又不知道該如何拼拼湊湊，畢竟不止詞尾需要押韻，就連歌詞長度也有一定規範，隨著韻律去把歌詞套在裡面，但這篇不會針對隨著韻律去舞動歌詞，而是創作出有規則並且自動押韻的詞段。至於斷句以及詞句，我的想法是輸入一個關鍵詞，然後生成第一段第二段第三段歌詞。這次將以仙俠小說招式名構成歌詞。

## 二、研究目的

- (一) 以自動分類押韻將資料庫中的資料分類好詞首以及詞尾的押韻規則。
- (二) 能夠輸入關鍵詞就生成一句歌詞，因應自己需要的詞段從中可以找出靈感，或者運用此些詞段。

## 三、研究流程



## 貳、文獻探討

一、Python 中拼音库 PyPinyin 的用法(出處:華為網)

21 個聲母：b p m f d t n l g k h j q x zh ch sh r z c s (y, w 不是聲母)

i 行的韻母，前面沒有聲母的時候，寫成 yi (衣)，ya (呀)，ye (耶)，yao (腰)，you (憂)，yan (煙)，yin (因)，yang (央)，ying (英)，yong (雍)。(y 不是聲母)

u 行的韻母，前面沒有聲母的時候，寫成 wu (烏)，wa (蛙)，wo (窩)，wai (歪)，wei (威)，wan (彎)，wen (溫)，wang (汪)，weng (翁)。(w 不是聲母)

ü 行的韻母，前面沒有聲母的時候，寫成 yu (迂)，yue (約)，yuan (冤)，yun (暈)；ü 上兩點省略。(韻母相關風格下還原正確的韻母 ü)

ü 行的韻跟聲母 j, q, x 拼的時候，寫成 ju (居)，qu (區)，xu (虛)，ü 上兩點也省略；但是跟聲母 n, l 拼的時候，仍然寫成 nü (女)，lǜ (呂)。(韻母相關風格下還原正確的韻母 ü)

iou, uei, uen 前面加聲母的時候，寫成 iu, ui, un。例如 niu (牛)，gui (歸)，lun (論)。(韻母相關風格下還原正確的韻母 iou, uei, uen)

首先就是這個庫的安裝了，透過 pip 安裝即可：

```
pip3 install pypinyin
```

進行基本的拼音轉換，直接調用 pinyin 方法即可：

```
from pypinyin import pinyin
print(pinyin('中心'))
```

運行結果：

```
[['zhōng'], ['xīn']]
```

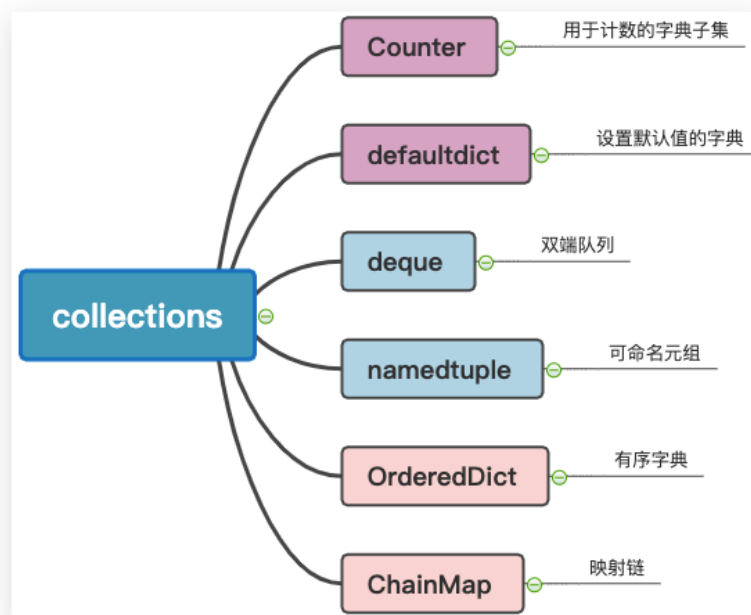
我們可以對結果進行一些風格轉換，比如不帶聲調風格、標準聲調風格、聲調在拼音之後、聲調在韻母之後、注音風格……等等，比如我們想要聲調放在拼音後面，可以這樣實現：

```
from pypinyin import lazy_pinyin, Style
style = Style.TONE3
print(lazy_pinyin('聰明的小兔子', style=style))
```

運行結果：

```
['cong1', 'ming2', 'de', 'xiao3', 'tu4', 'zi']
```

## 二、參考 python 中 collections 資料結構(出處:IT 人)



## 三、選擇利用 collections 導入 defaultdict 型別(出處:IT 幫幫忙)

通常在我們取用調用某個 dict 的 key 值的時候，都必須事先檢查這個 key 值是否存在，否則如果直接調用不存在的 key 值，會直接拋出一個 `KeyError`，比如說：

(1)統計一個 list 裡每個元素出現的個數

```
a_list = ['a','b','x','a','a','b','z']
counter_dict = {}
for element in a_list:
    if element not in counter_dict:
        counter_dict[element] = 1
    else:
        counter_dict[element] += 1
```

(2) 建立一個一對多的 multidict:

```

key_values = [('even',2),('odd',1),('even',8),('odd',3),('float',2.4),('odd',
7)]

multi_dict = {}
for key,value in key_values:
    if key not in multi_dict:
        multi_dict[key] = [value]
    else:
        multi_dict[key].append(value)

```

dict 的 key 值存不存在無關這件事，為何 dict 不會聰明一點，沒有 key 值就先生一個 default 值出來，這樣方便多了，所幸對於這種調用 key 值的議題，collections 提供的 defaultdict 給我們一個很好的解決方案，顧名思義，defaultdict 對於我們調用一個不存在的 key 值，他會先建立一個 default 值給我們，而這個 default 值必須由一個可呼叫的函數產生，在我們初始化一個 defaultdict 時，必須先指定一個產生 default 值的函數：

```

from collections import defaultdict

better_dict = defaultdict(list) # default值以一個list()方法產生
check_default = better_dict['a']
print(check_default) # 會輸出list()方法產生的空串列[]

better_dict['b'].append(1) # [1]
better_dict['b'].append(2) # [1,2]
better_dict['b'].append(3) # [1,2,3]
print(better_dict['b'])

```

因此若想要建立一個 multidict，只要用 defaultdict(list) 就可以很輕鬆的達成，不需要事先檢查 key 值存不存在，進而提高程式的可讀性：

```

from collections import defaultdict

multi_dict = defaultdict(list)
key_values = [('even',2),('odd',1),('even',8),('odd',3),('float',2.4),('odd',
7)]

for key,value in key_values:
    multi_dict[key].append(value)

print(multi_dict) # 會輸出defaultdict(<class 'list'>, {'float': [2.4], 'even':
[2, 8], 'odd': [1, 3, 7]})

```

但若我們想要直接給予一個固定的值給 defaultdict 是不行的，會產生 TypeError 的例外，比如說是在統計元素個數的狀況下，我想要讓 default 值

為 0，那有一個方法就是建構一個生成 0 的函數：

```
from collections import defaultdict

def zero():
    return 0

counter_dict = defaultdict(zero) # default值以一個zero()方法產生
a_list = ['a','b','x','a','a','b','z']

for element in a_list:
    counter_dict[element] += 1

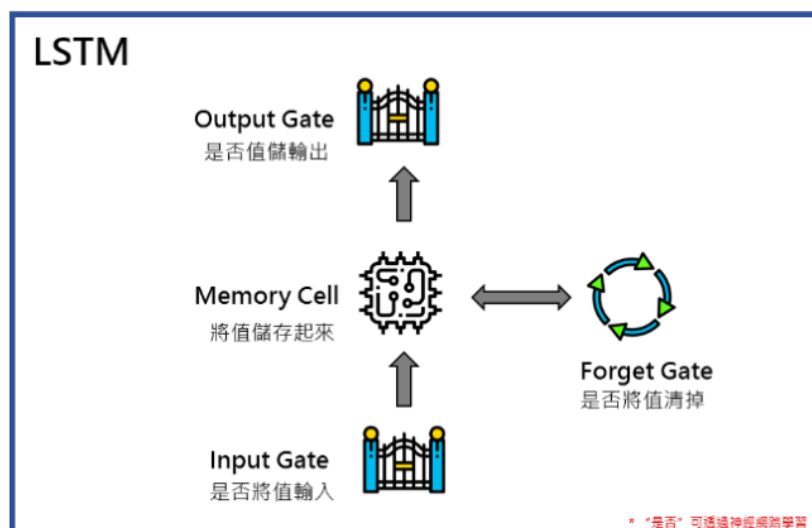
print(counter_dict) # 會輸出defaultdict(<function zero at 0x7fe488cb7bf8>,
{'x': 1, 'z': 1, 'a': 3, 'b': 2})
```

然後因為這個 defaultdict 是 dict 的一個子類別，也就是說他繼承了 dict 的所有方法，或是現在這個 defaultdict 是 dict 的擴充，一般用在 dict 的使用方法在 defaultdict 也可以使用。

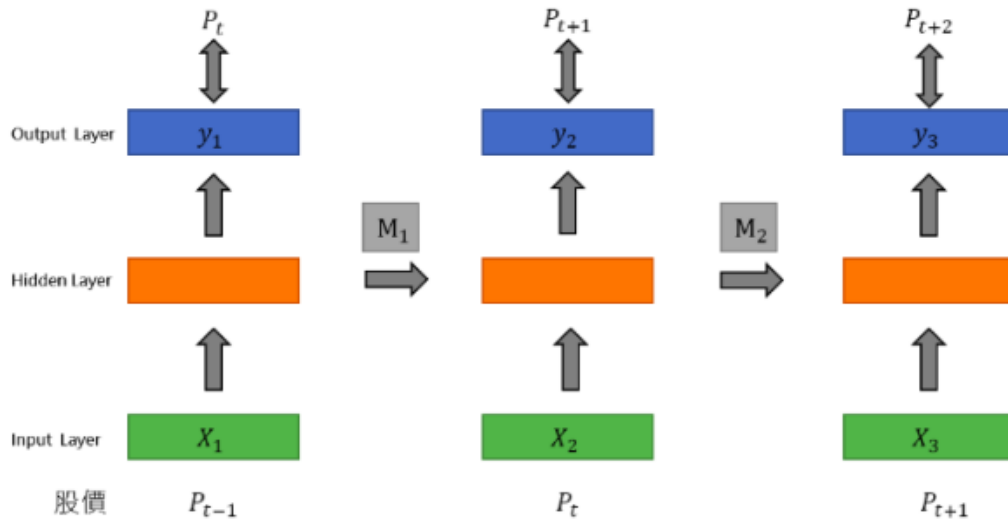
(1) default\_factory：就是 defaultdict 在初始化的過程中，第一個參數所接受的函數對象(也就是上述的 list()或是 zero())，而第二個之後的參數都比照一般 dict 傳入參數的格式。

(2) \_\_missing\_\_(key)：在我們調用不存在的 key 值時 defaultdict 會調用 \_\_missing\_\_(key)方法，這個方法會利用 default\_factory 創造一個 default 值給我們使用。

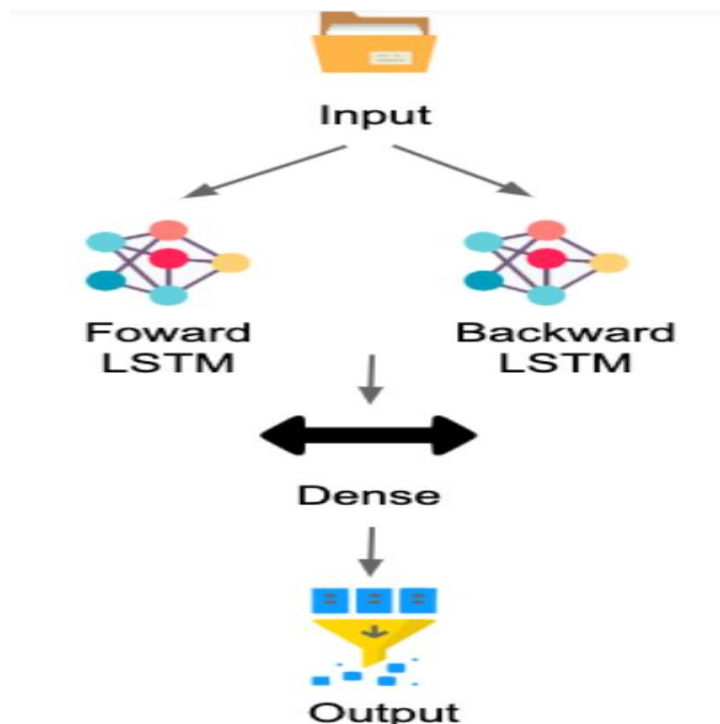
#### 四、LSTM 介紹(出處:IT 幫幫忙)



五、RNN 主要是透過將隱藏層的 output 存在 Memory 裡，當下次 input 資料進去 train 的時候，會同時考慮上一次存在 Memory 裡的值進行計算。(出處:IT 幫幫忙)



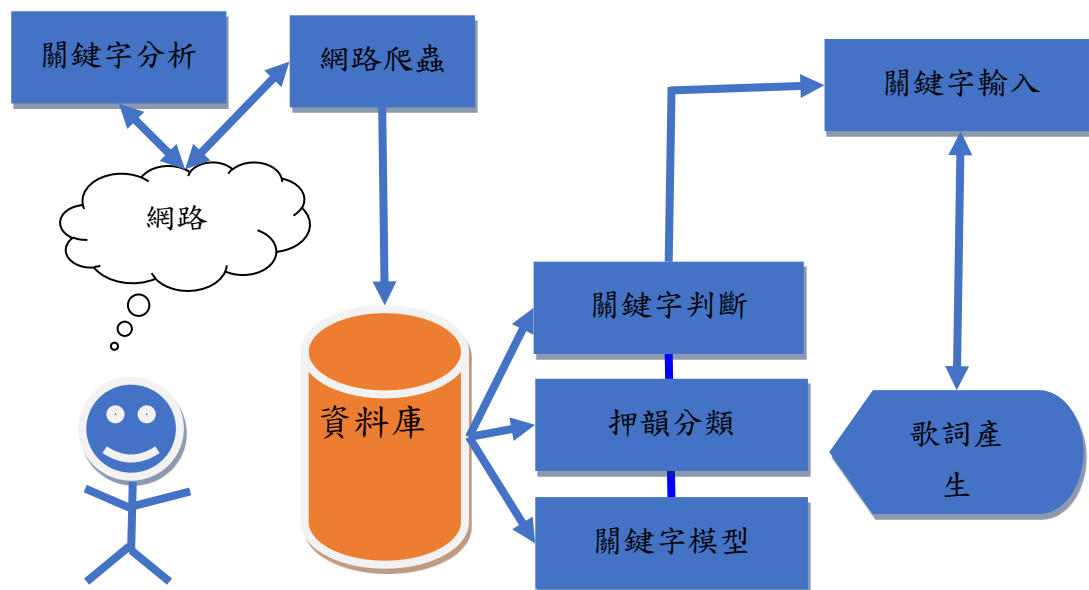
六、除了 input output 的變化，LSTM 也可以後面 train 回來 (Bidirectional LSTM)，舉個例子來說，當有一個句子，一般的 LSTM 只會從前面掃過去，但 Bidirectional LSTM 除了從前面掃過去，也會從後面掃過來，類似從後文來回推前文。與一般的 LSTM 相比 Bidirectional LSTM 在一些語音辨識或者 NLP 應用上有更好的效果。但還是要注意應用場景上是否合乎資料邏輯 (Ex: 往回推的時候是否符合資料邏輯) (出處:IT 幫幫忙)





## 叁、研究方法

### 一、系統架構



### 二、程式片段說明

(一)首先先在網路上搜索一些玄幻小說常出現的詞語或者名子，那這邊我選擇天下武功大全(上)作為我的第一個爬取的資料，我要得到它的內容。

這邊使用的套件為 beautiful soup 強大的爬蟲套件。

```
pipenv install beautifulsoup4
```

## 天下武功大全（上）

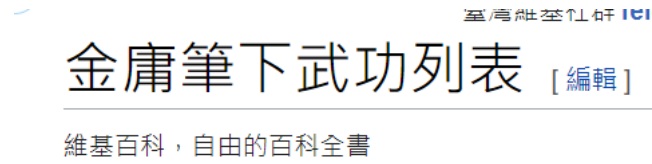
小说杂谈 2020-05-08 13:50 1419阅读 · 64喜欢 · 13评论



MC红白机 **LV6**  
粉丝：1.4万 文章：5

(圖一)

以及金庸筆下武功列表，我也要得到他的內容。



(圖二)

(二)創建一個 python 文件，parser.py，目的是為了將網路爬蟲得到的資料進行解析。

```
data_parser > parser.py > pinyin_correction
17     return parsed_pinyins
18
19
20 def pinyin_correction(pinyin: str) -> str:
21
22     # z/c/s + i -> z/c/s + I
23     if re.match(r'[zcs]i$', pinyin):
24         return pinyin.replace('i', 'I')
25
26     # zh/ch/sh/r + i -> zh/ch/sh/r + II
27     elif re.match(r'(?:(zh|ch|sh|r)i$', pinyin):
28         return pinyin.replace('i', 'II')
29
30     # j/q/x/y + u/ue/un/uan -> j/q/x + v/ve/vn/van | v/ve/vn/van
31     elif re.match(r'[jqxy]u', pinyin):
32         return re.sub(r'y*(.+)', r'\1', pinyin.replace('u', 'v'))
33
34     # y + a/e/ao/ou/an/ in/ian/ing/iong -> ia/ie/iao/iou/ian/ in/ian/ing/iong
35     elif pinyin.startswith("y"):
36         return re.sub(r'y*(.*)', r'i\1', pinyin)
```

(圖三)

以下為處理好的資料，生成結果為：

```
400
401 奪狼命爪：獵狼兔、野狼孤鷹、餓刺虎、獵狼屠獅
402
403 赤煉鬼爪：猛鬼叩門、餓鬼投胎、冤鬼纏身、惡鬼凌遲、鬼火燎原
404
405 閻羅八殺爪：招魂殺、絕代殺、分屍殺、追魂殺、驚天殺、動地殺、奪命殺、閻羅殺
406
407 刻骨銘心爪：刻骨銘心爪：破腦、破腦、刻骨破腕、刻骨無骨折斷腕、刻骨銘心、骨分肩、刻骨離襟配合步法：無
    刻痕跡
408
409 紫焰七勢：神鷹版、惡獅擒、餓狼噬、靈蛇吐
410
411 蝕月三殺：撕心、斷肢、破腦
412
413 四絕白骨爪：抽屍剝膚、破頭蝕腦、搜索枯腸、屍骨如山
414
415 神鷹八段錦爪法：神鷹奪魄、雄鷹沖天、鷹爪挖心
416
417 玄陰鬼爪：夜叉探爪、厲鬼招魂、鬼手追魂
418
```

(圖四)

(三)創建一個 data.prep.py，導入剛剛處理好的文件，將每個招式名按標點符號斷開，使用 word\_parser 獲得每個詞語，按聲母韻母分解好的拼音，並將每組詞按其結尾韻母分好組保存起來。

```
12 > for word in words:
13     pinyins = word_parser(word)
14     look_up_length[len(word)].append(word)
15 > if len(word) > 1:
16     look_up_vowel[tuple([pinyin[1][-1] for pinyin in pinyins[-2:]])].append(word)
17 > if len(word) > 2:
18     look_up_vowel[tuple([pinyin[1][-1] for pinyin in pinyins[-3:]])].append(word)
19 > if len(word) > 3 and word[-4] != ',':
20     look_up_vowel[tuple([pinyin[1][-1] for pinyin in pinyins[-4:]])].append(word)
```

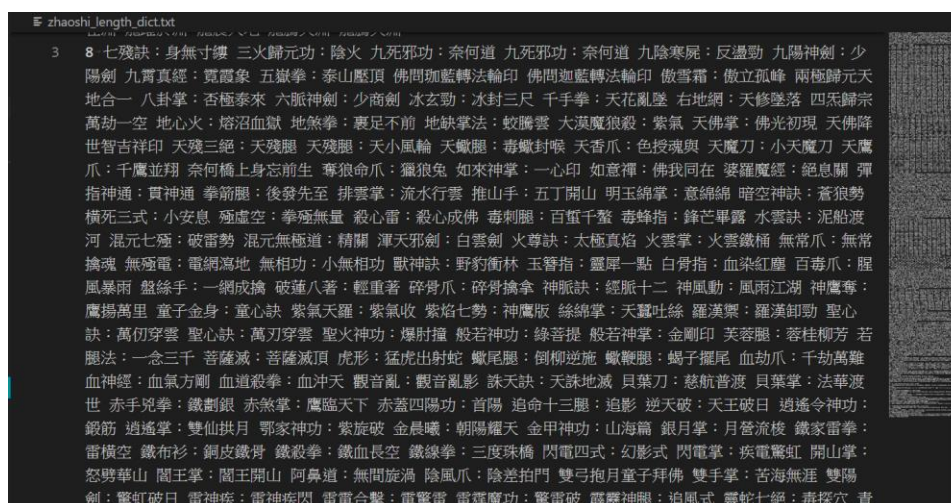
(圖五)

分別為將資料生成 zhaoshi\_length\_dict.txt 的檔案

```
22 > with open('zhaoshi_length_dict.txt', 'w', encoding="utf-8") as f:
23 >     for k, v in look_up_length.items():
24 >         f.write('{}\t{}\n'.format(k, ' '.join(sorted(v))))
25 >     print('完成!')
```

(圖六)

生成結果為：



The screenshot shows a text file named 'zhaoshi\_length\_dict.txt' with a list of martial arts moves and their corresponding pinyin characters. The moves are listed in a single column, and the pinyin characters are listed in a single column. The moves are: 3 8 七殘訣, 身無寸縷, 三火歸元功, 陰火, 九死邪功, 奈何道, 九死邪功, 奈何道, 九陰寒屍, 反盪勁, 九陽神劍, 少陽劍, 九霄真經, 寬露象, 五嶽拳, 泰山壓頂, 佛問迦藍轉法輪印, 佛問迦藍轉法輪印, 傲雪霜, 傲立孤峰, 兩極歸元, 天地合一, 八卦掌, 否極泰來, 六脈神劍, 少商劍, 冰玄勁, 冰封三尺, 千手拳, 天花亂墜, 右地網, 天修墜落, 四岳歸宗, 萬劫一空, 地心火, 熔沼血獄, 地煞拳, 裹足不前, 地缺掌法, 蛟騰雲, 大漠魔狼殺, 紫氣, 天佛掌, 佛光初現, 天佛降世, 智吉祥印, 天殘三絕, 天殘腿, 天殘腿, 天小風輪, 天蠟腿, 毒蠟封喉, 天香爪, 色授魂與, 天魔刀, 小天魔刀, 天魔爪, 千鷹並翔, 奈何橋上, 身忘前生, 奪狼命爪, 獵狼兔, 如來神掌, 一心印, 如意禪, 佛我同在, 婆羅經, 絕息關, 彈指神通, 貫神通, 拳筋腿, 後發先至, 排雲掌, 流水行雲, 推山手, 五丁開山, 明玉綿掌, 意綿綿, 暗空神訣, 蒼狼勢, 橫死三式, 小安息, 殞虛空, 拳殞無量, 殺心雷, 殺心成佛, 毒刺腿, 百重千疊, 毒蜂指, 鋒芒畢露, 水雲訣, 泥船渡河, 混元七傷, 破雷勢, 混元無極道, 精關, 渾天邪劍, 白雲劍, 火尊訣, 太極真焰, 火雲掌, 火雲鐵桶, 無常爪, 無常擒魂, 無極電, 電網瀉地, 無相功, 小無相功, 獸神訣, 野豹衝林, 玉簪指, 靈犀一點, 白骨指, 血染紅塵, 百毒爪, 腥風暴雨, 盤絲手, 一網成擒, 破蓮八著, 輕重著, 碎骨爪, 碎骨擒拿, 神脈訣, 經脈十二, 神風動, 風雨江湖, 神鷹奪, 鷹揚萬里, 童子金身, 童心訣, 紫氣天羅, 紫氣收, 紫焰七勢, 神鷹版, 絲綿掌, 天蠶吐絲, 羅漢藥, 羅漢卸勁, 聖心訣, 萬刃穿雲, 聖心訣, 萬刃穿雲, 聖火神功, 爆肘撞, 般若神功, 綠菩提, 般若神掌, 金剛印, 芙蓉腿, 蓉桂柳芳, 若腿法, 一念三千, 菩薩滅, 菩薩滅頂, 虎形, 猛虎出射蛇, 蠍尾腿, 倒柳逆施, 蠍鞭腿, 錫子擺尾, 血劫爪, 千劫萬難, 血神經, 血氣方剛, 血道殺拳, 血冲天, 觀音亂, 觀音亂影, 誅天訣, 天誅地滅, 貝葉刀, 慈航普渡, 貝葉掌, 法華渡世, 赤手兇拳, 鐵劃銀, 赤煞掌, 鷹臨天下, 赤蓋四陽功, 首陽, 追命十三腿, 追影, 逆天破, 天王破日, 逍遙令神功, 鍛筋, 逍遙掌, 雙仙拱月, 郭家神功, 紫旋破, 金晨曦, 朝陽耀天, 金甲神功, 山海篇, 銀月掌, 月營流梭, 鐵家雷拳, 雷橫空, 鐵布衫, 銅皮鐵骨, 鐵殺拳, 鐵血長空, 鐵線拳, 三度珠橋, 閃電四式, 幻影式, 閃電掌, 疾電驚虹, 開山掌, 怒劈華山, 閻王掌, 閻王開山, 阿鼻道, 無間旋渦, 陰風爪, 陰差拍門, 雙弓抱月, 童子拜佛, 雙手掌, 苦海無涯, 雙陽劍, 驚虹破日, 雷神疾, 雷神疾閃, 雷電合擊, 電驚雷, 雷霆魔功, 驚雷破, 霹靂神腿, 追風式, 靈蛇七絕, 毒探穴, 青

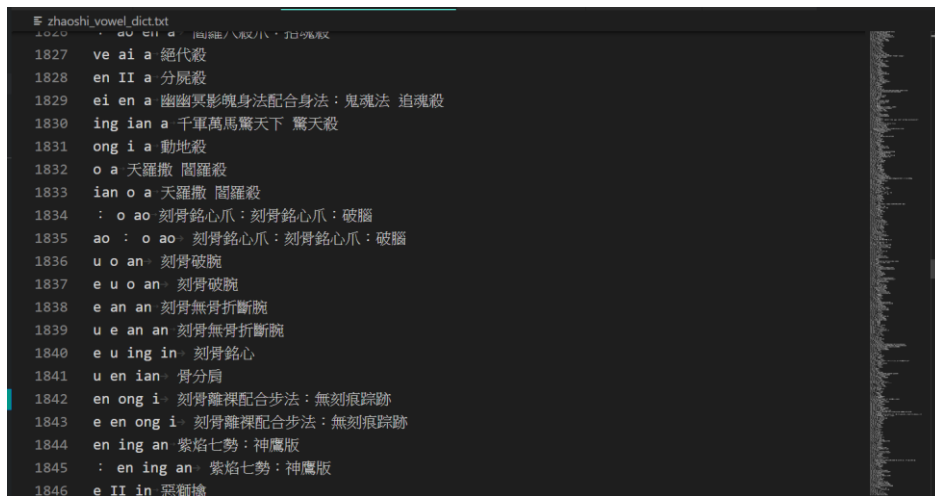
(圖七)

以及 zhaoshi\_vowel\_dict.txt

```
27 v with open('zhaoshi_vowel_dict.txt', 'w', encoding="utf-8") as f:
28 v     for k, v in look_up_vowel.items():
29     |     f.write('{}\t{}\n'.format(' '.join(list(k)), ' '.join(sorted(v))))
30     |     print('完成!')
31
```

(圖八)

生成結果為：



```
zhaoshi_vowel_dict.txt
1827 ve ai a 絕代殺
1828 en II a 分屍殺
1829 ei en a 幽幽冥影魂身法配合身法：鬼魂法 追魂殺
1830 ing ian a 千軍萬馬驚天下 驚天殺
1831 ong i a 動地殺
1832 o a 天羅撒 閻羅殺
1833 ian o a 天羅撒 閻羅殺
1834 : o ao 刻骨銘心爪：刻骨銘心爪：破腦
1835 ao : o ao 刻骨銘心爪：刻骨銘心爪：破腦
1836 u o an 刻骨破腕
1837 e u o an 刻骨破腕
1838 e an an 刻骨無骨折斷腕
1839 u e an an 刻骨無骨折斷腕
1840 e u ing in 刻骨銘心
1841 u en ian 骨分肩
1842 en ong i 刻骨離裸配合步法：無刻痕踪跡
1843 e en ong i 刻骨離裸配合步法：無刻痕踪跡
1844 en ing an 紫焰七勢：神鷹版
1845 : en ing an 紫焰七勢：神鷹版
1846 e II in 雲龍槍
```

(圖九)

(四)接下來開始寫生成歌詞的 python 文件\_\_main\_\_.py，先導入剛剛生成的韻母辭典字典文件。

```
6 zhaoshi_vowel_dict = {}
7 v with open('zhaoshi_vowel_dict.txt', 'r', encoding="utf-8") as f:
8 v     for line in f:
9     |     items = line.strip().split('\t')
10    |     zhaoshi_vowel_dict[tuple(items[0].split())] = items[1].split()
11
12 zhaoshi_len_dict = {}
13 v with open('zhaoshi_length_dict.txt', 'r', encoding="utf-8") as f:
14 v     for line in f:
15     |     items = line.strip().split('\t')
16     |     zhaoshi_len_dict[int(items[0])] = items[1].split()
```

(圖十)

(五)再來，根據關鍵詞生成兩個四字短語及一個三字短語。

```
18 ∨ def main():
19     ... word1 = input('■ 請輸入第一個關鍵詞: ') or ''
20     ... word2 = input('■ 請輸入第二個關鍵詞: ') or ''
21     ... word3 = input('■ 請輸入第三個關鍵詞 ') or ''
22     ... word4 = input('■ 請輸入一句詩句 ') or ''
23
24 ∨     ... line1 = '{}，這{}，'.format(word1,
25     ... |         ...''.join(gen_words_with_pattern(word1, [4, 4, 3])))
26 ∨     ... line2 = '{}，我{}，'.format(word2,
27     ... |         ...''.join(gen_words_with_pattern(word1, [4, 4, 3])))
28 ∨     ... line3 = '{}，它{}，'.format(word3,
29     ... |         ...''.join(gen_words_with_pattern(word1, [4, 5])))
30 ∨     ... line4 = '{}，我自手持{}!'.format(word4,
31     ... |         ...choice(get_zhaoshi_by_vowel(word4)[3]))
32
33     ... print('\n...生成中...\n')
```

(圖十一)

(六)再來，創建一個方法，會根據一個詞語生成對應數量長度的押韻短語。

```
35     ... song = '\n'.join([line1, line2, line3, line4])
36     ... print(song);
37
38 ∨ def gen_words_with_pattern(word, breakdowns):
39     ... z = get_zhaoshi_by_vowel(word)
40     ... return [choice(z[b]) for b in breakdowns]
```

(圖十二)

(七)在這個方法裡，對應每個詞，去剛剛生成好的字典裡找到與其最後一個韻母相同，但最後一個字不同的詞，將其按長度不同分組後，隨機在對應長度裡挑選我想要數量的詞語。

```
43  def get_zhaoshi_by_vowel(target_word):
44      num_char = len(target_word)
45      pinyins = word_parser(target_word)
46
47      zhaoshi = defaultdict(list)
48
49      target_vowel = pinyins[-1][1][-1]
50      for k, v in zhaoshi_vowel_dict.items():
51          if k[-1] == target_vowel:
52              for word in v:
53                  word_pys = word_parser(word)
54                  if word[-1] != target_word[-1] \
55                      and word not in zhaoshi[len(word)]:
56                      zhaoshi[len(word)].append(word)
57      return zhaoshi
```

(圖十三)

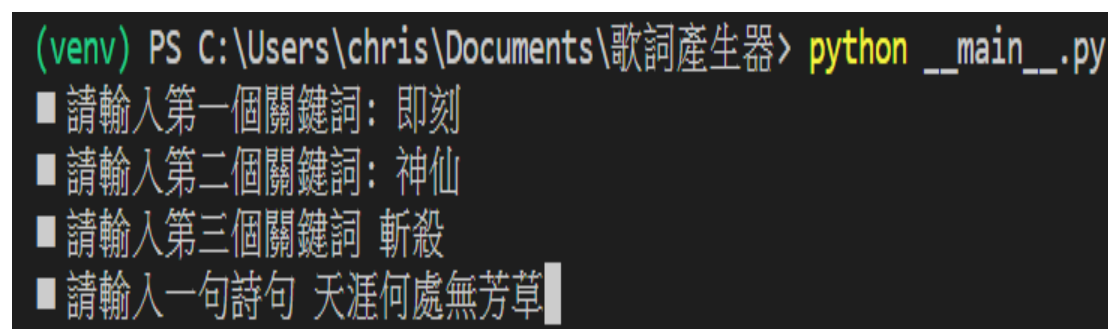
(八)最後將這些詞語變成句子。

```
59  def get_zhaoshi_by_len(len):
60      return choice(zhaoshi_len_dict[len])
61
62  if __name__ == '__main__':
63      main()
```

(圖十四)

## 肆、研究分析與結果

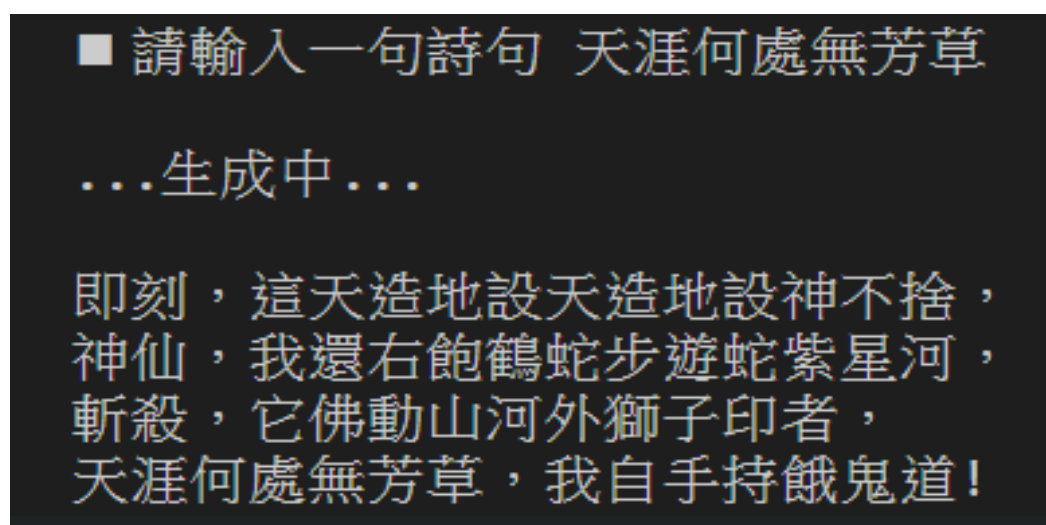
按照關鍵詞依序生成第一句、第二句、第三句以及最後一句的關鍵詞。



```
(venv) PS C:\Users\chris\Documents\歌詞產生器> python __main__.py
■ 請輸入第一個關鍵詞：即刻
■ 請輸入第二個關鍵詞：神仙
■ 請輸入第三個關鍵詞 斬殺
■ 請輸入一句詩句 天涯何處無芳草
```

(圖十五)

最後將得到依據你編排的結果，生成對應數量且押韻的一段歌詞



```
■ 請輸入一句詩句 天涯何處無芳草

...生成中...

即刻，這天造地設天造地設神不捨，
神仙，我還右飽鶴蛇步遊蛇紫星河，
斬殺，它佛動山河外獅子印者，
天涯何處無芳草，我自手持餓鬼道！
```

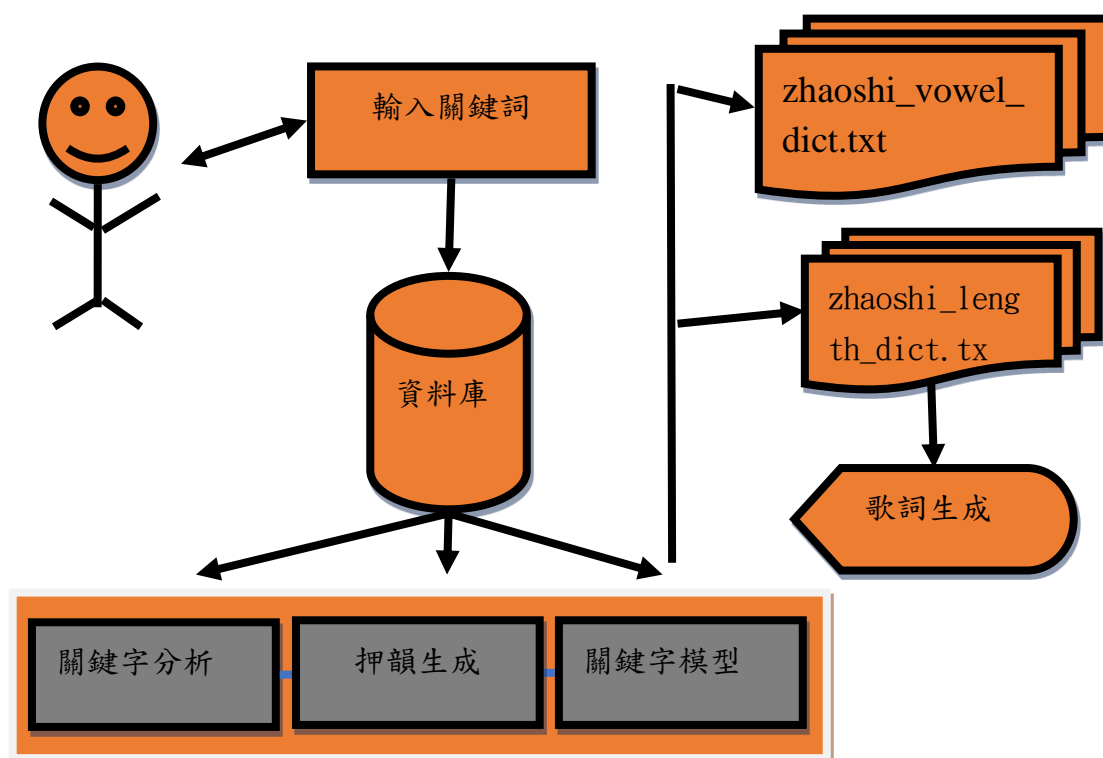
(圖十六)

使用 open source 方式呈現：

<https://github.com/chris911024/Implementing-lyrics-generator-with-pypinyin>

對於此項專題，我覺得是一個大大的進步，我認為未來可以運用 LSTM 模型以及深度學習去做出更有深度的作品，這項專題主要是運用押韻技術去組成句子，可以利用此項方式讓自己找出靈感，大大幫助了作詞家進行靈感產生以及運用生成的歌詞編成一首曲子，運用到未來可以讓對歌詞有興趣的歌手利用程式語言去生成一首歌曲，或許未來可以專研技術去針對 AI 訓練旋律搭配歌詞去生成一首歌，或許也會造成不同凡響。

## 一、系統流程





## 伍、研究結論與建議

### 優點：

這個產生器是針對如果對於歌詞不熟悉，或者暫時沒有靈感的時候可以使用這個產生器，產生一些新的發想，當然資料庫越多就可以生成更多不同風格的歌詞組合，可以有更多選擇的方法，這項技術對於未來我覺得可以製作出更多需要靈感、以及設計的一些工作上面，幫助這些設計者以及創作者有多一點的發想，進而創造出更多未知的創新力以及組合力。

### 缺點：

針對此項專題，必須具有龐大資料庫，將此技術運用到雲端做全自動化的方式才能達到商機，不然預估約 10000 筆歌詞將有詞語重複出現的話，就會使歌曲沒有獨特力。

### 建議改進：

我認為可以針對 RNN 模型中，最著名的 Long short-term memory (LSTM) 進行深度學習，做出全自動化一鍵生成產生器、以及給予更多資料庫讓他有選擇不同類型進而創造不同類型的歌詞能力。

## 陸、參考文獻

### 1. 離島指出：

Python 抓取歌词自制 FreeStyle

<https://segmentfault.com/a/1190000015932069>

### 2. python 那些事指出：

Python 如何抓取歌词完成個人創作？

<https://cloud.tencent.com/developer/news/298476>

### 3. 51CTO 博客指出：

Python 有嘻哈：Crossin 教你用代码写出押韵的 verse

[https://blog.51cto.com/u\\_15127553/2707212](https://blog.51cto.com/u_15127553/2707212)

### 4. 巴哈姆特\_茶米指出：

[達人專欄] 【歌曲填詞淺談】第一章-音、韻的心得參考

<https://home.gamer.com.tw/creationDetail.php?sn=2072775>

### 5. Translated from Brandon Rohrer's Blog by Jimmy Lin 指出：

遞歸神經網路（RNN）和長短期記憶模型（LSTM）的運作原理

[https://brohrer.mcknote.com/zh-Hant/how\\_machine\\_learning\\_works/how\\_rnn\\_lstm\\_work.html](https://brohrer.mcknote.com/zh-Hant/how_machine_learning_works/how_rnn_lstm_work.html)

### 6. Dan 指出：

RNN - LSTM 介紹

<https://ithelp.ithome.com.tw/articles/10223055>

### 7. eating 指出：

NLP 會用到的模型(三)-RNN 應用

<https://ithelp.ithome.com.tw/articles/10267429>

8. I code so am I 指出：

循環神經網路(Recurrent Neural Network, RNN)

<https://ithelp.ithome.com.tw/articles/10193469>

9. Shusen Wang 指出：

RNN 模型与 NLP 应用(4/9)：LSTM 模型

<https://www.youtube.com/watch?v=vTouAvxlphc>

10. 容璞玩 Data 指出：

python-深度學習 6.3- LSTM 神經網路-建模

<https://www.youtube.com/watch?v=EDHpfSSD6ZI>

11. 容璞玩 Data 指出：

python-深度學習 6.2- LSTM、GRU 神經網路-簡介

<https://www.youtube.com/watch?v=oBdcTW5TnsM>

12. The A.I. Hacker – Michael Phi 指出：

Illustrated Guide to LSTM's and GRU's: A step by step explanation

<https://www.youtube.com/watch?v=8HyCNIVRbSU>

13. 容璞玩 Data 指出：

python-深度學習 1-甚麼是神經網路 ( what's neural network )

<https://www.youtube.com/watch?v=q4Uow17eb1Q>