**DSP Final Project**

B02902015 資工三 梁智泓

B20902019 資工三 陳柏均

**Motivation:**

我們寫了一個錯字糾正系統，現今大多數的語言或文字處理技術都建立在正確的文字上，因此當有錯字產生時許多技術達不到原本的效果。然而，真正的資料來自各種形式，錯字的產生是必然的，藉由這個錯字糾正系統，能將原本無法使用的資料變成有所貢獻。這便是我們希望能做到的目標

**Problem Definition:**

然而錯字的可能性有非常多種，可能是同音不同字，也可能是在拼注音時有其中一個注音拼錯，更可以是完全拼錯。而我們將問題Focus在『同音不同字』的問題上，也就是文件中的錯字，必定是同音，沒有注音拼錯的情況。

**Dataset:**

我們使用了Hw3時所用到的Corpus，裡面有多達100萬句的新聞文章，除此之外我們更跟助教要了另外的一份Data裡面總共5047篇新聞文章，約有4萬句左右的新聞資料。所以我們總共收集了104萬句的新聞文章中句子。而為了要生出test data（有錯字的data），我們用了助教給的5000篇新聞的data在裡面隨機產生同音不同字的錯字，然後用這5000篇有錯字的documents作為testing data。

**Technique:**

* **Dictionary Mapping:** 查找了一下資料，發現中文的常用字大概是5000左右，而在Hw3助教所給的Big5\_to\_ZhuYin有13000多個字，所以我們認為這樣的量已足夠，便直接使用Big5\_to\_ZhuYin mapping作為我們的Dictionary之一，而為了到解決『同音不同字』的問題，我們便必須要建出一份所有注音拼音可能產生文字的對照表，所以透過助教的Mapping，我們更自己生出了一份ZhuYin\_to\_Big5的文件，而透過這兩個Map便可以幫助我們找到一個錯字可能的拼音，以及那些拼音可能對到的文字，再用這些文字作為Candidates。
* **Jieba:** 而為了要讓判斷錯字更為精準，所以我們必須先對文章做一些切割的處理，將文字分割成詞，因為在中文之中，有許多單字往往不具任何意義，而這樣的情況會使我們再判斷錯字時失準，所以我們先透過了Jieba一個NLP的tool去對文章的文字做切詞的動作，以幫助我們判斷出在詞中的錯字，減少在詞與詞中間因出現機率過低而發生的誤判。
* **Total Language Model:** 我們透過所有104萬句的資料，透過SRILM的Tool建立出Total 的Bigram Language Model
* **Viterbi within Viterbi Net:** 有了切出的詞之後，我們便可以在詞裡面跑一個 Bigram 的Viterbi，並在裡面找到最有可能的前20個Term，再用這前20個terms去跑Term與Term之間的bigram Viterbi，形成一個Viterbi in Viterbi的結構，最後選出機率最大的字詞組合作為更正結果。會有這樣的發想，也是來自於Neural Network in Neural Network (NIN) 的啟發，而這樣也的確讓判斷並且更正錯字變得更為精準。
* **LDA (Latent Dirichlet Allocation):** 如果僅僅使用上面的辦法仍然會出現問題，例如：我喜歡“城市” 跟 我喜歡“程式”，兩句話的可能都不低，而且前者的出現機率較大，若僅用Viterbi的話，會很有可能計算出前面的答案，但如果文章的作者原本希望的是後者呢？而為了解決這樣的問題，我們就必須將文章的Topic給考慮進來，也就是這篇文章可能在說什麼，如果這篇文章的Topic是在說“城市”，那結果就應該是前者，如果是在說“程式”，那結果就應該是後者。而為了要能做到這件事，我們使用了Unsupervised的LDA。
* **Topic Language model:** 利用LDA，將一開始收集到的5047文章去做training，得到4種Topic category，並將所有分類好的文章各自連接起來做出各個類別的Language Model，然後再將testing data (有錯字的)根據這個model去預測它的topic分類，最後利用已經算好的Topic bigram Language Model去對詞內與詞與詞之間的機率分部作出調整。
* **User Trust:** 在判斷的依據上除了機率模型外我們增加了另一個考量的因素，那就是『傾向相信』作者本身並沒有寫錯字，即對文章本身的信賴程度，信賴程度會將原本詞彙的組合給予一定(+1~+3)數量的機率加權(在log環境下)，提高該組合最後被選中的機率，信賴程度的大小是在我們知道文章錯誤率的假設下給的，藉此用來降低LM影響原本語意的機會。

**Probability Model:**

* **Term Internal Probability Model:**
* **Between Terms Probability Model:**

**Experiment & Observation:**

**How to Evaluation:**

因為以詞為單位的準確率相當高，所以看不太出結果的好與壞。所以我們決定以一句話為單位，判斷其中是否所有的錯字皆被更正作為Evaluation。然後我們會分別看testing data預測出來的Topic內的文章與整體的準確度。

**Terms Accuracy**

|  |  |  |  |
| --- | --- | --- | --- |
| Doc error rate | 0.2 | 0.5 | 0.8 |
| Topic0 | 0.96461 | 0.960446 | 0.957413 |
| Topic1 | 0.618837 | 0.605051 | 0.59237 |
| Topic2 | 0.967283 | 0.96446 | 0.962913 |
| Topic3 | 0.965655 | 0.960715 | 0.956995 |
| Total | 0.935373 | 0.929918 | 0.925649 |

**Sentence Accuracy**

|  |  |  |  |
| --- | --- | --- | --- |
| Doc error rate | 0.2 | 0.5 | 0.8 |
| Topic0 | 0.403106 | 0.366786 | 0.341776 |
| Topic1 | 0.023968 | 0.022239 | 0.019768 |
| Topic2 | 0.586585 | 0.542683 | 0.526829 |
| Topic3 | 0.392887 | 0.347657 | 0.318868 |
| Total | 0.366538 | 0.327623 | 0.30243 |

可以看出，當原始文章的錯字率越高時，更正錯字的Performance就會越差，除此之外，我們更發現了Topic內部文章的好壞也影響很大，從面的數據可以清楚的看到，Topic1的Performance不管是在Term還是在Sentence上都很差，而經由我們後仔細的將Topic1中的文章看過之後，發現Topic1內的文章常常是『一行文』或是沒什麼重點的文章，而且Topic1內的文章數相較於其他的Topic少非常多，所以我們認為Topic1其實就是經由LDA計算後，所謂的『其他文章』應該是無法分類的文章的集合。這便是topic1的結果會如此糟的原因，因為我們將Topic的Language model考慮進去，而裡面的文章分布過於偏差，導致產生大量誤差。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Lamda | 1 | 0.5 | 0.8 | 0.2 |
| In-term Lamda | 0.5 | 0.5 | 0.2 | 0.8 |
| Topic0 | 0.401593 | 0.403106 | 0.403106 | 0.403266 |
| Topic1 | 0.024463 | 0.023968 | 0.023968 | 0.023988 |
| Topic2 | 0.586585 | 0.586585 | 0.586585 | 0.586585 |
| Topic3 | 0.392923 | 0.392887 | 0.393029 | 0.392958 |
| Total | 0.366189 | 0.366538 | 0.366626 | 0.366626 |

從這個實驗可以看出某些Topic比較特別，與Corpus的Language Model中的分布不同，如果用比較多比例的Topic Language Model，Performance會變得比較好，而有些Topic比較General，Language Model得分佈差不多，所以並沒有相對大的Improvement。

|  |  |  |  |
| --- | --- | --- | --- |
|  | User Trust | 0 | 3 |
| Error Rate |  |
| 0.2 | | 0.354455 | 0.30243 |
| 0.5 | | 0.355395 | 0.327623 |
| 0.8 | | 0.355832 | 0.366538 |

從結果可以看到當在文章本身的Error Rate很低時，User Trust給的越高，Performance越好，若文章本身的Error Rate很高時，User Trust則會造成反效果

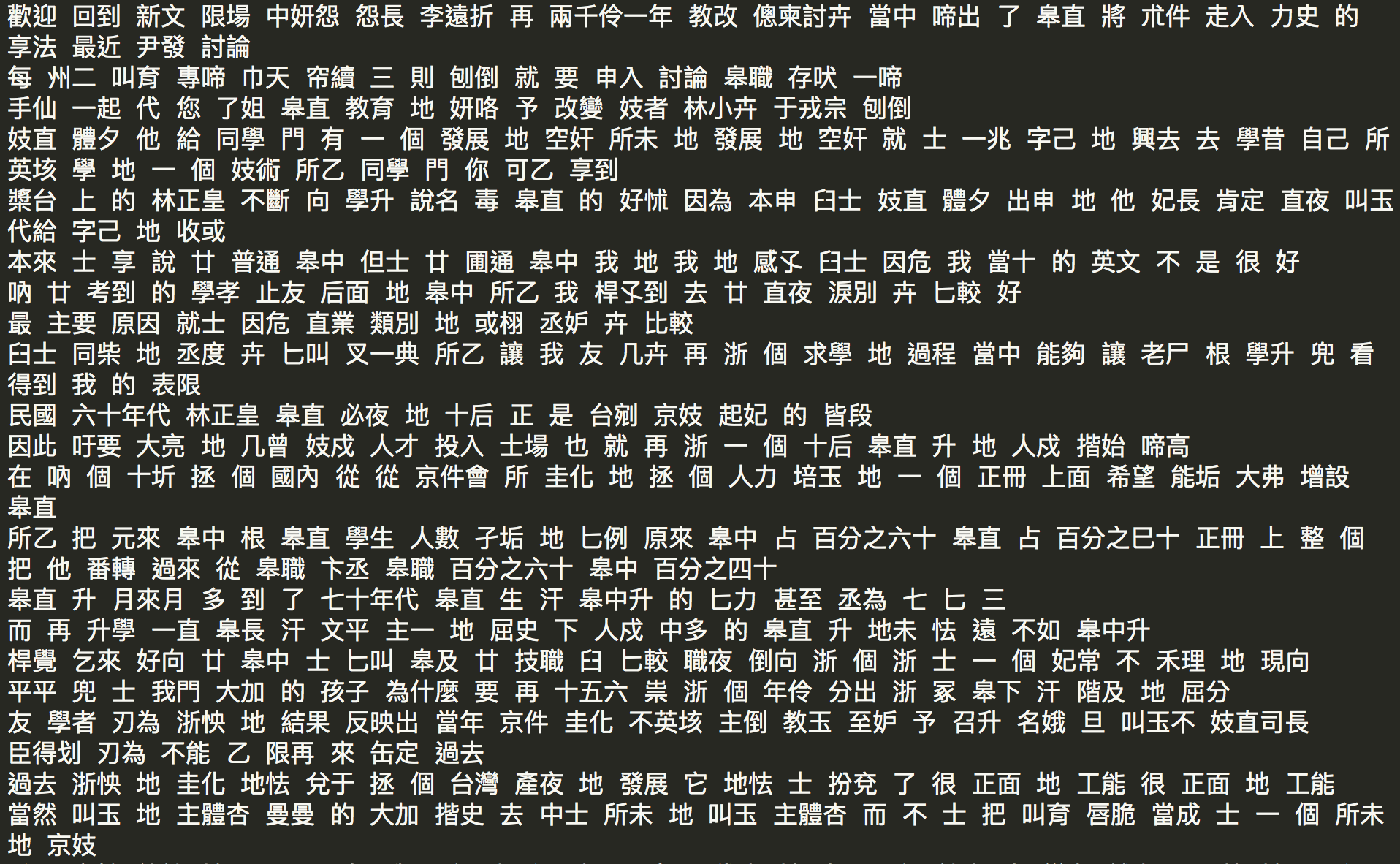
**Discussion:**

這次的測試資料限於新聞，所以在辨識文字上不論是LDA或者LM都有非常好的成效，但實際碰到不同句子時LDA仍會受限於Training Corpus涵蓋的範圍與大小，而且我們在做LDA時設立每個文章最後能夠收斂到某個主題的假設在現實生活中可以說是有漏洞的，每篇文章中都可能是不同主題混合出來的LM，而混合的比例並不是我們能夠單獨從文章中推測出來的，這是尚未解決的問題。

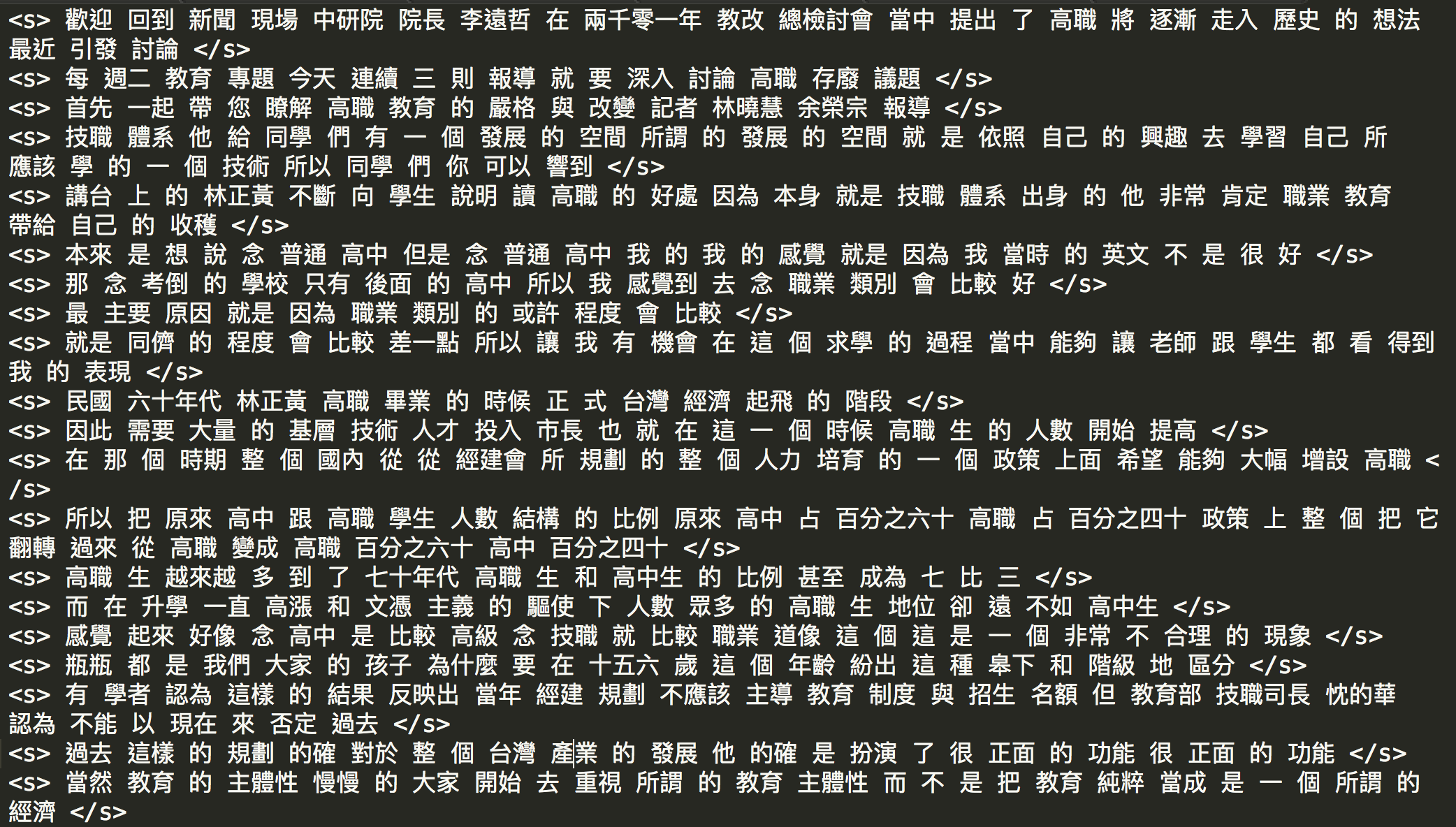
這次我們發現最有效的就是User Trust的發想，相信User在一定程度上沒有打錯字能夠有效降低“城市”與“程式”間的辨識的錯誤。在這個系統下解決詞彙的辨識中還是它的漏洞，因為我們只利用bigram LM的緣故，在多於兩個詞彙的處理上就常常會有問題，沒有辦法正確的辨識出多字詞原本的面貌，引入N-gram LM雖然可能有助於解決這樣的問題，但同時它提高的複雜度並不是我們可以接受的，如果加入我們的User Trust能夠有效的降低需要計算的計算量，達到長字詞辨認的效果。

**Demo:**

**Before**



**After**



**Work division:**

|  |  |
| --- | --- |
| B02902015 梁智泓 | Construct Total, Topic Language Model  Construct Words and ZhuYin`s two-way Mapping  Construct Viterbi in Viterbi Net  Evaluation and Experiment  Report |
| B02902019 陳柏均 | Data Correction  Data Preprocessing  LDA training and Topic classification  Report |