2020 年臺灣國際科學展覽會
優勝作品專輯

作品編號　**190013**

參展科別　電腦科學與資訊工程

作品名稱　殊途同歸—無既定模式中英文混合輸入

得獎獎項　大會獎：二等獎

出國正選代表

就讀學校　國立羅東高級中學

指導教師　邱柏翰

作者姓名　林容丞、藍苡芯、王禹博

關鍵詞　多元樹、深度優先搜尋法

作者簡介

我是羅東高中數理資優班的藍苡芯，很高興能跟兩位隊友們一起完成這個研究，也很榮幸能踏上國際科展的舞台，也感謝辛苦指導我們的柏翰老師，雖然有時為了這個報告忙到半夜，但也在高中生活留下了珍貴的回憶。

我是羅高數理資優班的王禹博，從小就喜歡資訊，很幸運能在高中進入資訊專題，跟著柏翰老師研究資訊的奧妙。雖然在研究時真的很辛苦，有時甚至為了趕進度整個組員輪流睡覺，但這些經歷也豐富了原本苦悶的高中生活。

我是羅東高中數理資優班的林容丞，很開心能在入選資訊組後和這兩位夥伴共事。和有相同興趣的朋友為了一個共同目標努力，加上柏翰老師的指點和相互討論，即使往往奮鬥到三更半夜，也無疑是最令人振奮的事！

摘要

本研究旨在設計一個使用模式，以不切換中、英文輸入法打字的原則之下，能夠完整的自動辨識出一個包含中文（注音、嘸蝦米、倉頡）與英文完整句子。經實測結果，正確率達到 94.23%以上。

Abstract

The purpose of the program is to present a model in order not to change the IME(Input Method Editor) between Chinese and English, we can recognize a complete sentence including English, Phonetics, Boshiamy and Cangjie automatically.Since we live in such a convenient world that technology brings, why do we still switch IME by ourselves? As the system is verified,the accuracy of the program is up to 94.23%.

一、前言

【研究動機】

研究動機在使用中文輸入法時，因為忘記切換中打和英打而打出一串亂碼的案例層出不窮，電腦有時甚至發出拼字錯誤的嗶聲，我們不禁想著，既然電腦已能判斷輸入錯誤，為什麼仍要「手動」切換中英文呢？為何不讓電腦自己判斷這串文字是中文或者是英文呢？於是我們便嘗試使用中文和英文的輸入法與鍵盤的對照，並配合字頻統計表，希望能找出最適當的排列組合後得出完整的句子，讓往後在輸入能夠更有效率。

【研究目的】

（一）能自動判別輸入法類別並顯示。

（二）以 C++設計相關資料結構與能夠自動轉換成中、英文的演算法。

（三）以 Microsoft Visual C# 實作視窗式程式。

二、研究方法或過程

【第一部分：研究設備及器材】

（一）

1.個人電腦，CPU ：Intel i7 7500U、記憶體：8G、硬碟空間：512G

2.程式開發工具（DEV C++、Visual studio community 2017、Python（撰寫網路爬蟲））

3.Microsoft Office（Word、Excel 試算表、Access 資料庫等工具）

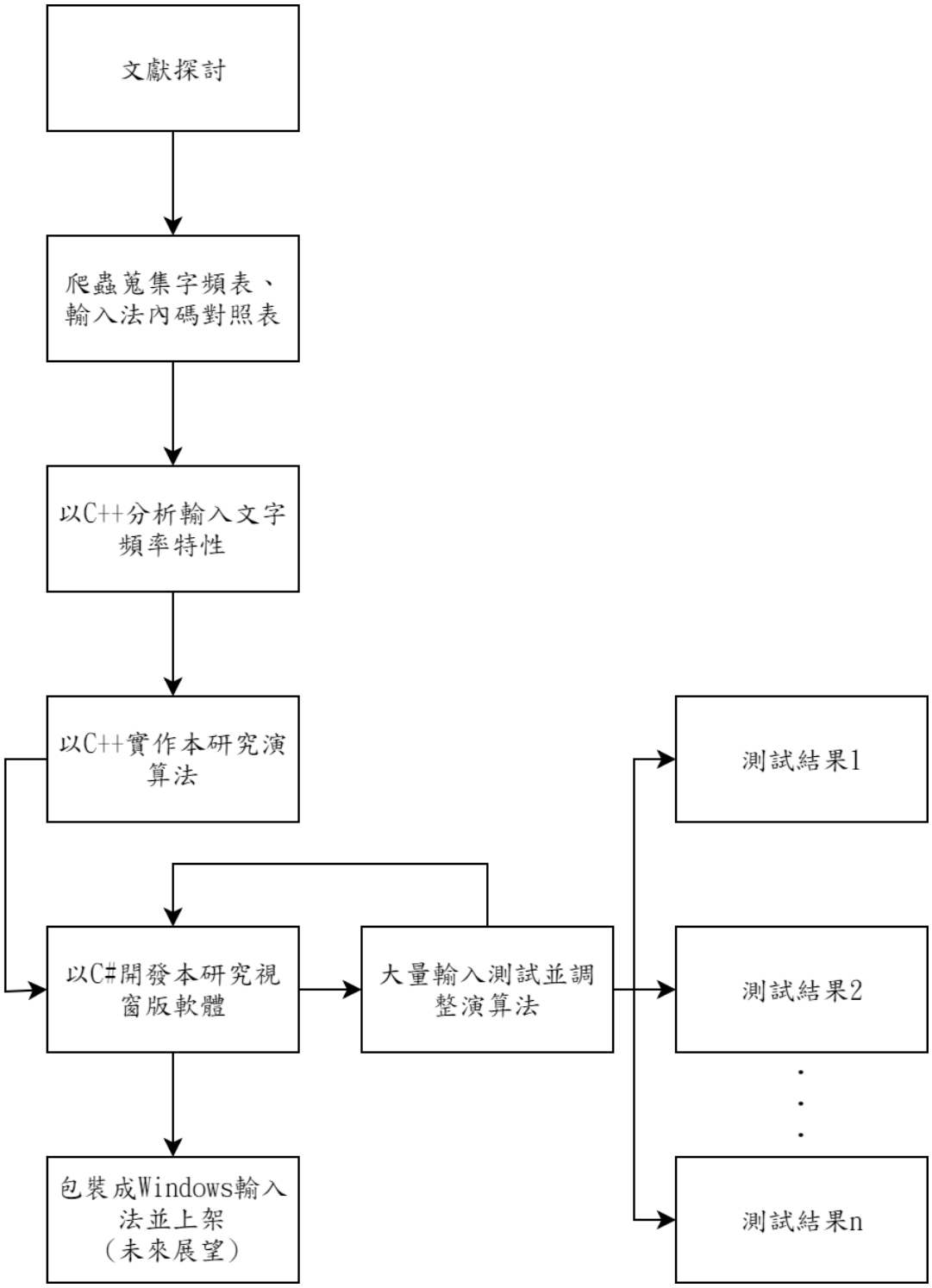
4.TinyTask 鍵盤滑鼠錄製工具

（二）

1.中、英文字頻表（含教育部中文字頻、中研院詞頻及維基百科英文字頻）

2.注音、嘸蝦米與倉頡中文對照表

【第二部分：演算法之構思與實踐】

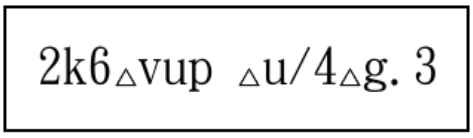


圖（一）：本研究之流程

步驟一、參考文獻

嘗試使用建立邊界來進行不同輸入法的切割

2008 年時，Yo Ehara 與 Tanaka-Ishii 的 Type Any 可以判斷當前輸入的語言為何，但輸入者必須協助程式建立「邊界」，也就是空白。但使用者須配合程式改變輸入習慣，這樣的輸入方式並不人性化，因此我們在程式中改善了此一限制。



圖（二）：Type Any 演算法之示意圖

如圖所示，若欲以 Type Any 之模式輸入「得心應手」，則需在每一個字之後加上空白作為分隔，才能進行輸入法的決定。

嘗試使用人工智慧的支援向量機來進行不同輸入法的切割

我們曾參考暨南大學碩士畢業論文（賴逸駿，2018），發現其使用支援向量機進行實驗。但我們在經過討論後，認為支援向量機的方法是用有限的資料去預測無限的可能性，而此題的按鍵組合乃是一個有限的組合，使用人工智慧模型不符合研究所需，而且文章中並未詳述如何將嘸蝦米「向量化」，因為沒有想出適當的方法將長度不同的拼字組合要轉成同維度的向量，因此最終決定不使用支援向量機。

步驟二、蒐集資料

（一）以爬蟲抓取資料：

由於中研院每次僅限讀取 300 筆資料，因此我們透過 python 撰寫爬蟲程式，並將網站之資料以 html 檔案方式存取。

（二）輸入法內碼表、字頻及詞頻整理：

1.嘸蝦米輸入法內碼表：

由於嘸蝦米一個中文只會對到一個輸入，因此我們將其整理成一行輸入、一行中文的模式，並將其存放於 dictionary。

2.中文字頻：

我們參考教育部 84 年的中文字頻，並將其整理成一行一字的格式，並儲存於陣列中。

3.英文字頻：

我們參考維基百科的英文字頻，並將其整理成一行一字的格式，並儲存於陣列中。

4.倉頡、注音輸入法內碼表：

在倉頡及注音中，同樣的輸入可能對到不同的字，即兩字含有重複的鍵值（key），導致資料輸入時錯誤。因此我們最終決定以換行為資料分界，並將擁有相同鍵值的資料事先整理並以「空格」作為分隔後合併成一行，再將其儲存於 dictionary 中。而在需要運用值（value）中的各筆資料時，再以 Split 將其分割。

5.中文詞頻：

我們參考中研院的現代漢語語料庫詞頻統計，將超過 10 萬筆的資料由爬蟲抓取，並整理為一行一個詞的格式，以便將其儲存於陣列中，在輸入為兩個以上同音的詞時，可用來比對其前後順序。

6.中文詞頻對照：

因為詞有多個以上的字元，必須讓程式個別讀取，因此我們將詞中的每個字分別對照輸入法，並用「Tab 鍵」作為分隔後合併成一行。且如同倉頡及注音，同樣的輸入也可能對到不同的詞，因此我們必須將有相同鍵值的資料事先整理，同樣以「Tab 鍵」作為分隔後合併成一行，並儲存於 dictionary 中。

步驟三、利用 C++判斷輸入特性

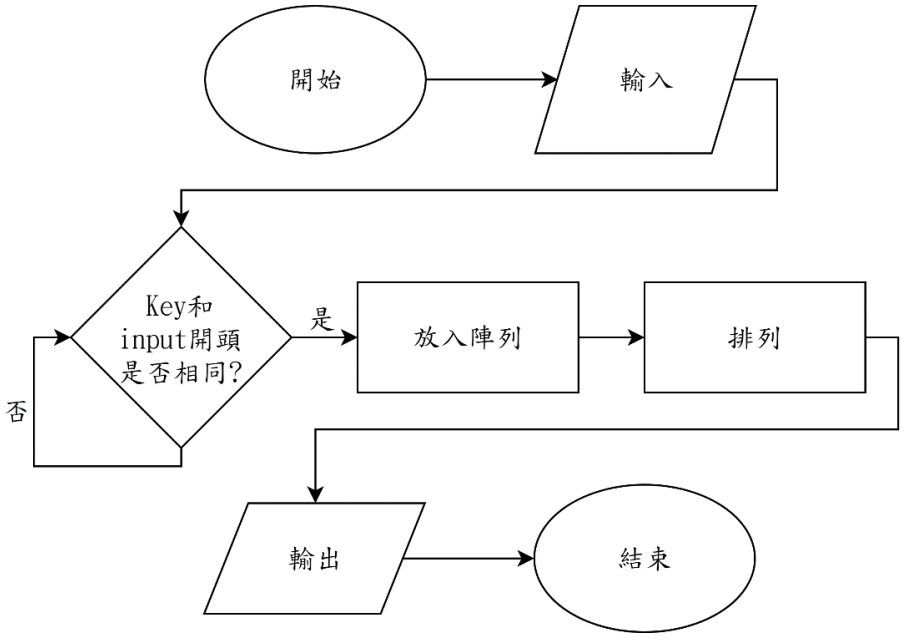
我們利用中文注音和英文單字進行分析，試著分析出中文字和英文字在排序上的機率來進行比較，得到每一個字元最有可能為中、英文，並輸出 C（中文）、E（英文）來方便我們觀察。而斷字的地方我們則運用中文字的聲符（3、4、6、7、空白）來進行斷字，並在斷字處補上「0」。但我們發現，因為還是會有一些少數的例外產生，導致一個字中會同時判斷出兩種不同的情況如圖（三），雖然最後是以整個單字都是中文的才歸類到中文字，因此「斷字判斷」法的穩定度和準確率是不夠高的，所以我們決定換另外一種方法來進行判斷。

圖（三）：以 C++進行中英混合判斷演算法之示意圖

由上圖可知，此程式對中文的判別較佳，但英文如 allow，to 等字因為有部分被判定為英文，雖不影響結果，但可見此演算法之準確度有限。而 go 一字甚至全字皆被判定為中文，更顯驗算法仍有待更正。

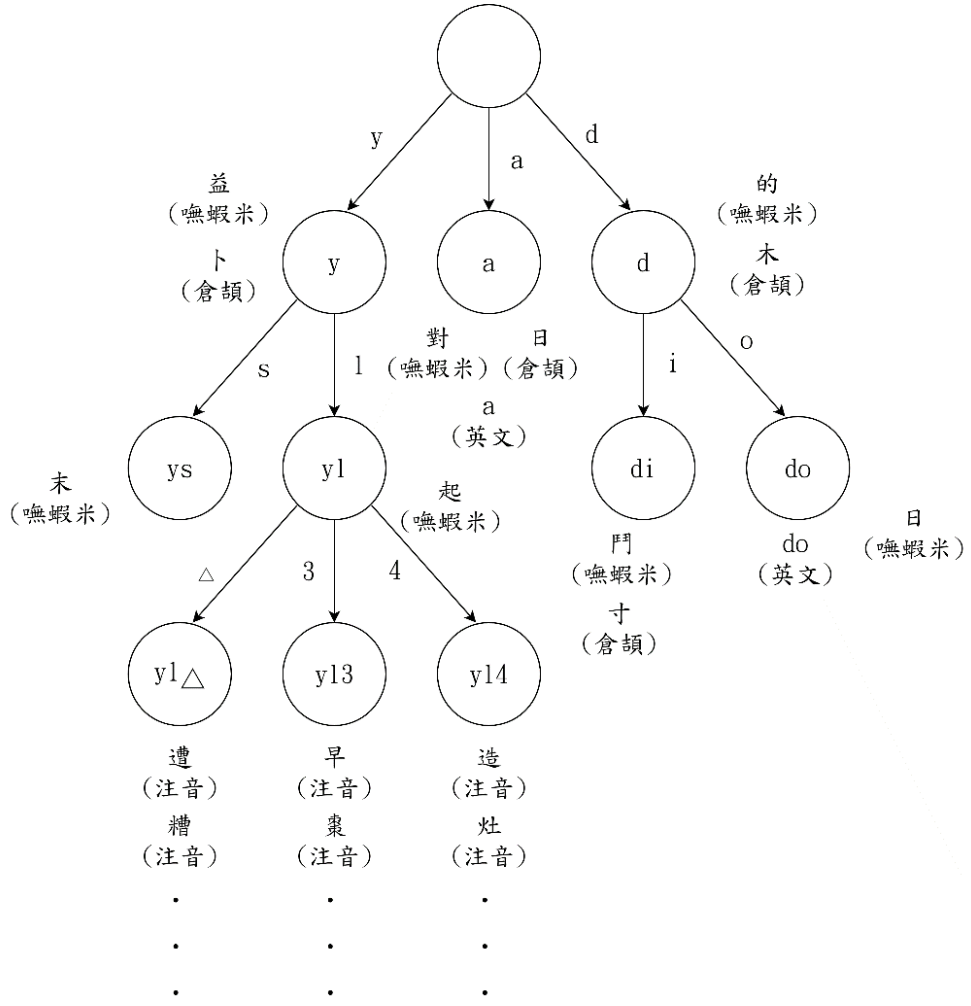
步驟四、以 C++開發演算並驗證成效

考量之後會進行大量實測及對 C++的熟悉程度，因此我們決定以 C++開發演算法。以 file stream 讀入純文字格式的注音中文對照表、嘸蝦米中文對照表、倉頡中文對照表及中、英文字頻等資料，並將對照表放入 C++標準函式庫 map 中以便之後查詢。透過排序等操作，能從對照表中找出開頭前 n 個字和輸入字串完全相同的 key（也就是對照表中的英文）後，再將其 value（即對照表中的中文）所對應的字頻放入陣列中。重複此動作直到前 n 個字與 input 不同為止。最後將陣列進行排序，便可依照字頻進行輸出動作。程式流程圖如下圖：



圖（四）：以 C++開發演算之流程圖

在比對尋找的過程，即在前綴樹（prefix tree）當中巡訪（traversal），若單純在前綴樹中巡訪，效能低落，而且有很多可能的巡訪結果，因此必須加入「切割」、「頻率分析」等技巧，才能提升準確率。本研究經過數據實測，提出「前端切齊分割」法，經實測達到最高準確率。如下圖（五），可以由圖中發現這是一個相當巨大且複雜的多元樹，甚至在相同的葉可能還會對應到不同的值，更顯其資料的難以分析。若在此中一一搜尋其是否為我們所要的資料會使得時間複雜度提升，耗費許多不必要的時間。



圖（五）：以前綴樹表示輸入資料之示意圖

RE

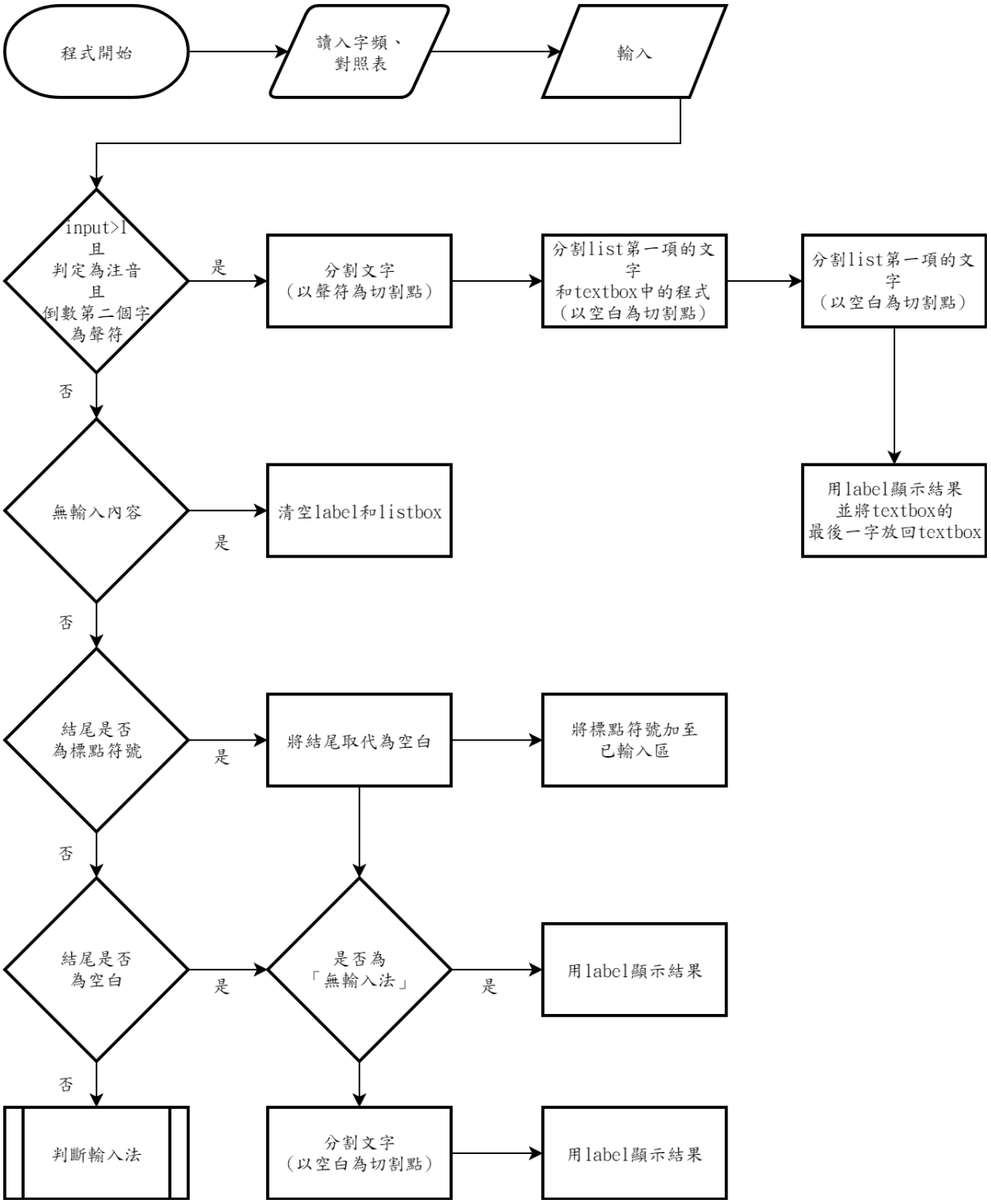
出 23
READ 316
RECEIVED 319
RETURN 336
REASON 369
REPLIED 370
師 372
REALLY 397

圖（六）：以頻率排序符合字彙之示意圖

舉例來說，如圖（六）所示，在所有資料庫中尋找開頭包含 RE 的資料，並將其頻率（以順序做表示）放入陣列。待所有頻率都放入後便將數字由小到大做排序。最後依照頻率將中英輸出。

【第三部分：以 Visual studio 寫出 C#視窗式程式】

呈上節所述，由於以 C++開發演算法正確率達一定程度，表示本研究所提出的方法可行，因此改以 C#開發視窗程式，改善人機介面並邁向真正的輸入法應用程式。



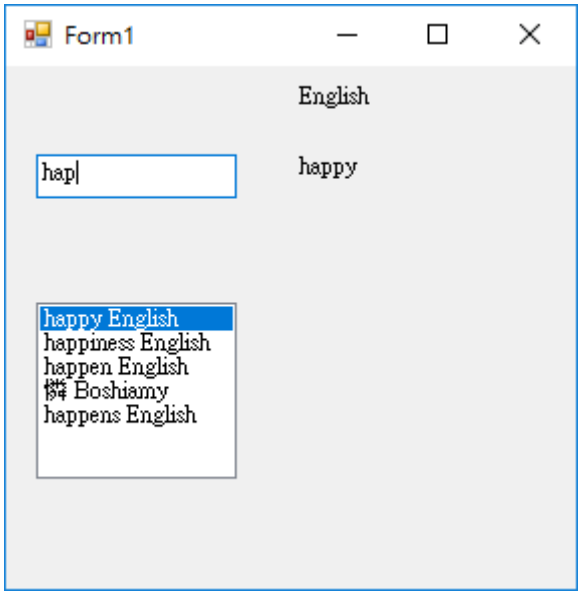
圖（七）：本研究視窗版軟體演算之流程圖

開發過程中發現，C#和 C++函式庫存在一些差異，例如 C#的 dictionary 並如 C++的 map 資料容器能自動排序，因此透過最後根據字頻排列顯示之結果來彌補此缺點。另外，C#的 dictionary 採 1 對 1 設計，因此資料的輸入也做了些許調整。透過 C#程式的編寫，不僅能夠在輸入的同步顯示結果，也增加了選字以及選擇輸入法的可能性，更接近真實輸入法的使用習慣。補足 C++ console 模式的限制。

步驟五、以嘸蝦米輸入法實作演算法

因為嘸蝦米屬於一對一輸入，及一個輸入只對到一個中文字，所以我們選擇以嘸蝦米作為實驗並實作演算法的輸入法。

演算法概述：仿造前部分的 C++，以 C#語法將嘸蝦米與英文字庫進行分析、按照頻率排列，並將其放入列表框以供使用者進行候選。



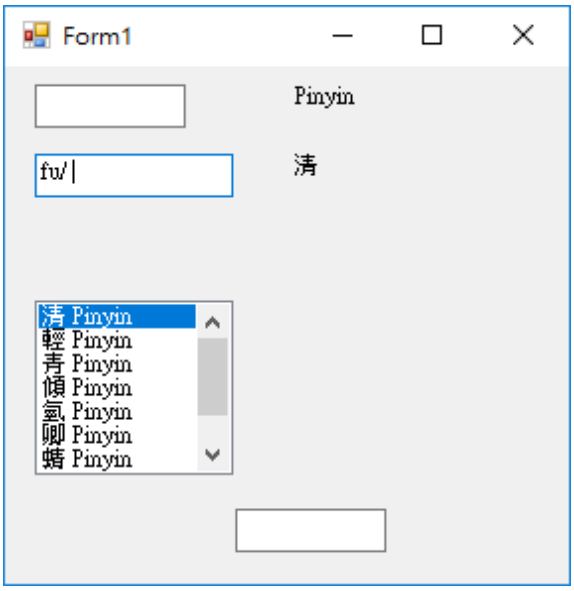
圖（八）：以嘸蝦米實作演算之視窗

步驟六、加入注音輸入法和倉頡輸入法

成功實作嘸蝦米輸入法之後，我們嘗試加入注音輸入法和倉頡輸入法。

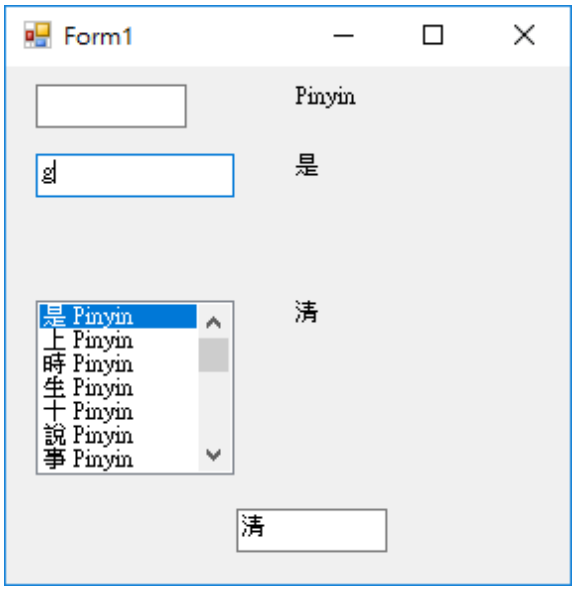
問題一：為了實現無須由使用者建立「邊界」的概念，必須將注音輸入法的聲符當作一個字的結尾來進行判斷。但如果使用和空白一樣的手法，將會造成無法對候選字進行選擇的困境。

解決方式：經過不斷測試，決定將之視為另一個情況：若目前列為第一選項之字為注音輸入法，則待下一字元輸入（也就是開始做下一字的輸入）時才將候選字確定，並將最後一字元當作輸入放入程式。如此便留下得以選字的空檔，也無須透過建立邊界來判斷注音。



圖（九）：以注音輸入後進行候選字判斷之視窗

已在字串後方輸入空白字元，但因為其為注音輸入法，因此被視為是「一聲」而非結束字元。



圖（十）：在空白字元後輸入下一字並輸出前一字之視窗

在輸入下一字元 g 之後，程式將「清」字選定，並將 g 字放回輸入框再次進行判斷。

步驟七、處理未知的詞彙

問題二：有時使用者會輸入字庫中未有的英文字，或甚至是縮寫。這時程式將無法判斷此為何種輸入法。

解決方式：因為無論嘸蝦米、倉頡，或注音輸入法，所形成的組合都是有限且與英文相比較為少量的。因此若遇到字庫中未有之字時，程式將自動判定為無輸入法，程式不予處理，即按照其原輸入字串輸出。

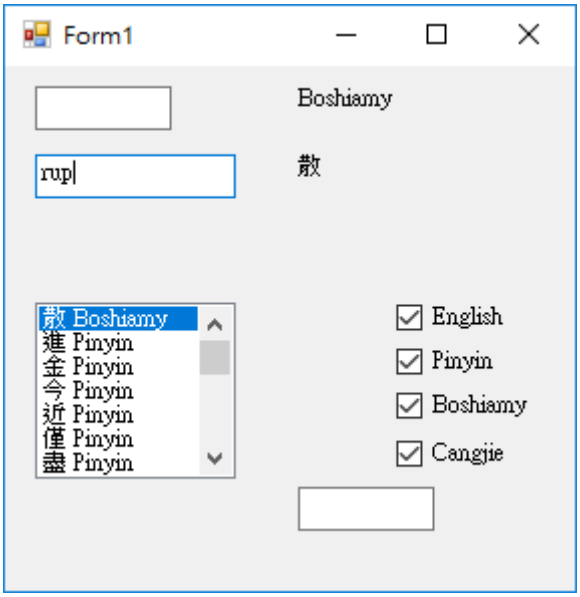
步驟八、調整候選字標準與順序

問題三：發現完全符合輸入字串的候選字會因為頻率較低而未出現在最上層。

解決方式：考慮到大部分人在輸入時會將字輸入完整，因此將與輸入字串完全相同之候選項目移至列表框最上方。若有多個完全相同的候選字，則依然以字頻排序之。

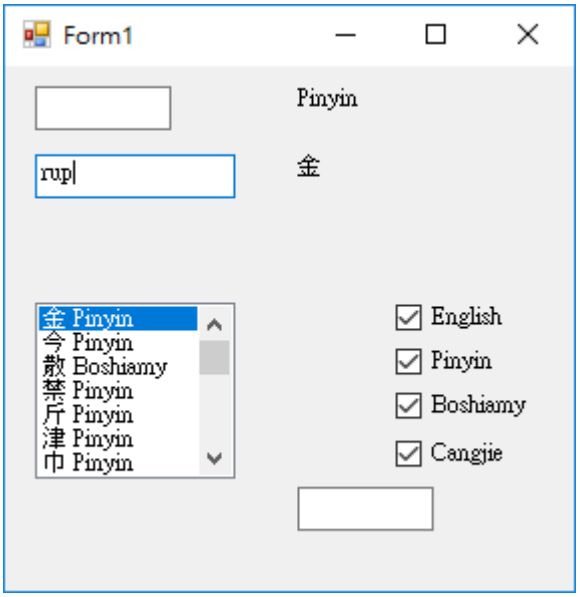
問題四：由於注音中的一聲和嘸蝦米、倉頡的結束字元相同，也就是空白字元。所以一聲之注音字無法因為與輸入字串相同而移至最上方。

解決方式：若注音輸入法且結尾為空白字元之字，當使用者已輸入結尾以外之所有字元時，將此字視為「完全相同」。



圖（十一）：未調整候選字之視窗

如圖可見，金、今等字因輸入為”rup ”而無法位於最上層（因為輸入為”rup”）。



圖（十二）：經修正後已調整之視窗

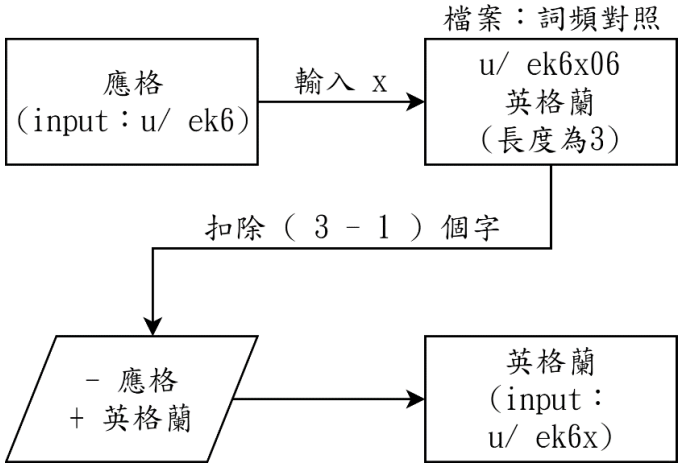
經過調整，將結尾為空白之注音字（即一聲字）視為無空白。

步驟九、使用詞頻協助進行輸入法判斷

問題五：字的候選排列若僅依靠字頻仍舊難以達到高準確率。

解決方式：建立「詞」之概念，以陣列存取已輸入過的字後，與資料庫進行比對，將符合的詞依頻率排列後輸出。在該字被選擇之後，須將先前已輸入之字進行刪除，再加上新的詞彙。

以下圖為例：將輸入至「u/ ek7x」時，程式在詞頻對照中尋得「u/ ek6x06」→英格蘭一詞。由於此時輸出處已有「應格」二字，因此必須移除此二字後再將「英格蘭」加至輸入處。

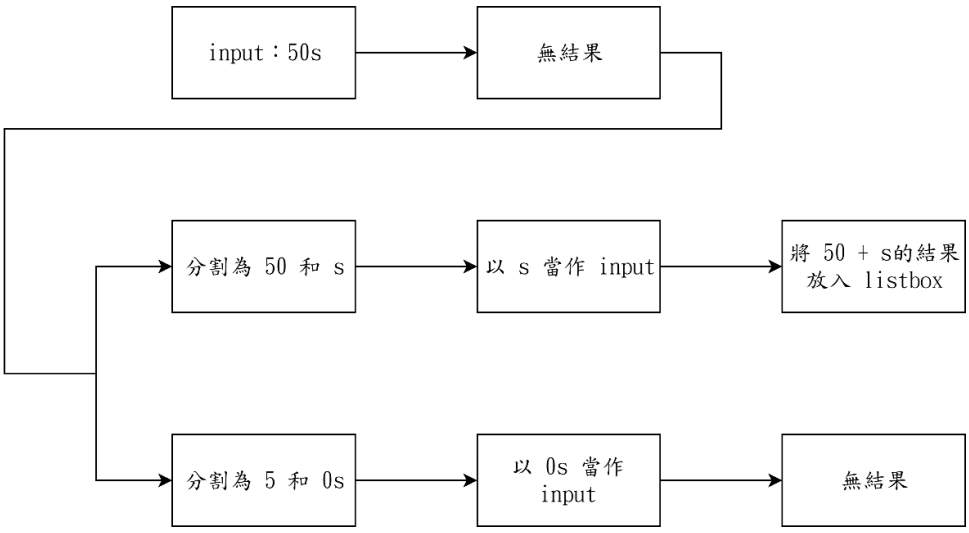


圖（十三）：由詞頻判斷詞彙之示意圖

步驟十、增加「數字加中文」之判斷

問題六：數字後方常未加「空白字元」而是直接加上中文，但先前之演算法並未考慮此問題，導致無法對後方之注音進行判斷。

解決方式：當上述所有方法皆未有結果時，使用「後端數字分離」法，將輸入字串從後方依序向前做分割，並檢視其前段是否全為數字。若是則將後方的字串再次進行驗算法判斷，最後加上前方的數字，作為候選詞。



圖（十四）：數字加中文演算法之示意圖

【第四部分：分析輸入準確率與分析】

步驟十一、定義準確率與檢測方式

（一）準確率計算方式

1.定義各個字是否正確

（1）正確字：依照測試資料輸入相同的字視為正確字，每輸入正確一字計 1 字。

（2）同音字：輸入與測試資料相同的字音，但不同的字形（如：成和誠），每輸入一同音字計 0.5 字。

（3）錯打字：輸入與測試資料不同音、字形的字視為錯打字，每錯打一字計 0 字。

表（一）：

原始資料：「The United States」這個名字是對美國最傳統和正式的名稱		
「The United States」這個名字是對美國最傳統和正式的名稱	「The United States」這個名字是對每國最船統和正是的名稱	「The United Stage」這個名字是對美國最傳統和正式對名稱
上述與原始資料完全相同，因此記 22 正確字。	上述與原始資料有「船」、「是」、「每」三個同音字，因此記 19 正確字，1.5 次同音字，共 20.5 字。	上述與原始資料有”Stage”、「對」兩個錯打字，因此記 20 正確字。

2.準確率的計算公式

（1）將計得之字數除以輸入的總字數，即得此程式的準確率。其公式如下：

$$準確率 = \frac{計得之字數}{總字數}$$

3. 測資的產生方式

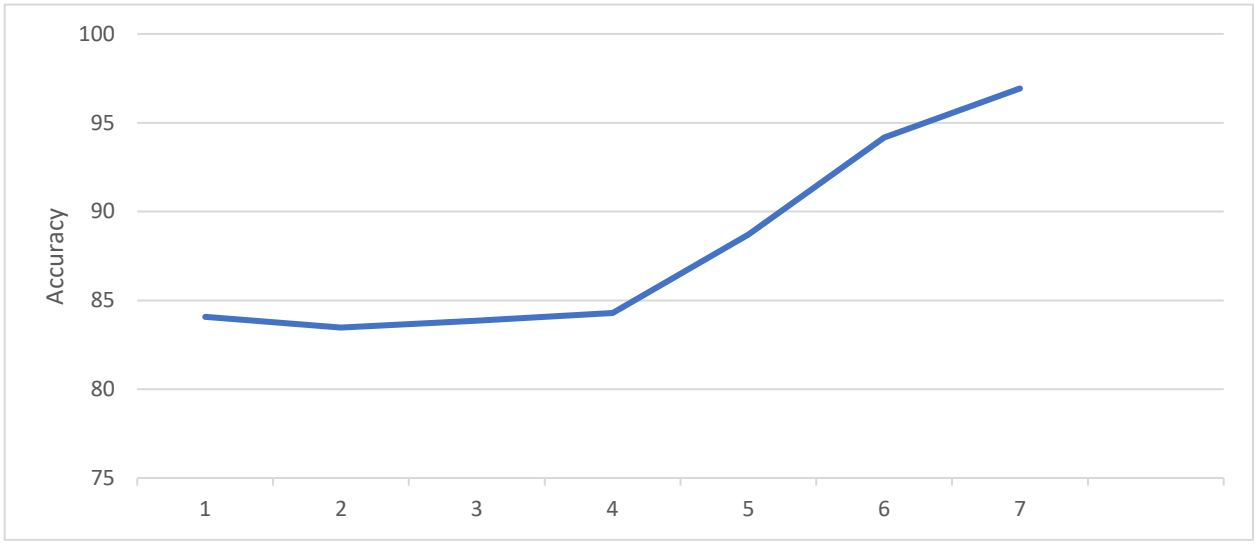
（1）透過 C++程式將原資料轉變成其輸入法（各輸入法依照相同比例）。

（2）原先透過鍵盤滑鼠錄製工具模擬輸入並讀取其值，後為求速度而改成將欲輸入的資料全數放在一個新的文字盒裡，而當此文字盒的內容被更改且長度不為 0 時，便會將第一個字元加至輸入格內。如此一來便能成功模擬真人輸入的情形。

（3）與原資料比對並計算其準確率。

步驟十二、準確率的分析與討論

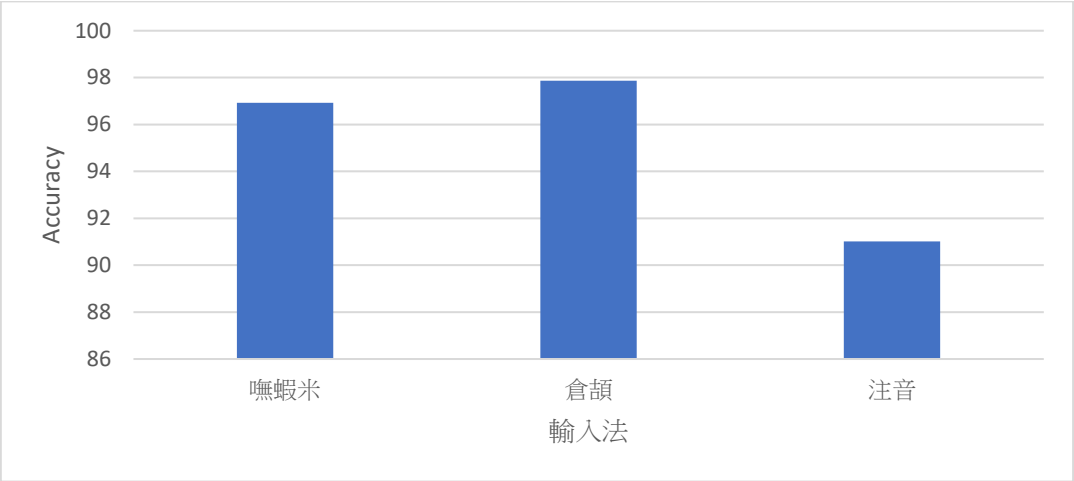
（一）不同版本之間的準確率比較（以英文與嘸蝦米混合為例）



圖（十五）：不同版本之間的準確率比較

從此折線圖發現，我們能夠發現透過不斷的修正與新增判斷機制，確實能夠提升準確率至 96.9%。其中前面幾個版本的準確率有不穩定的趨勢是因為在此程式做了選字排序上的修正，即將完全一樣的字放於最上方，但因為並未對其排列而造成正確率反而下降。

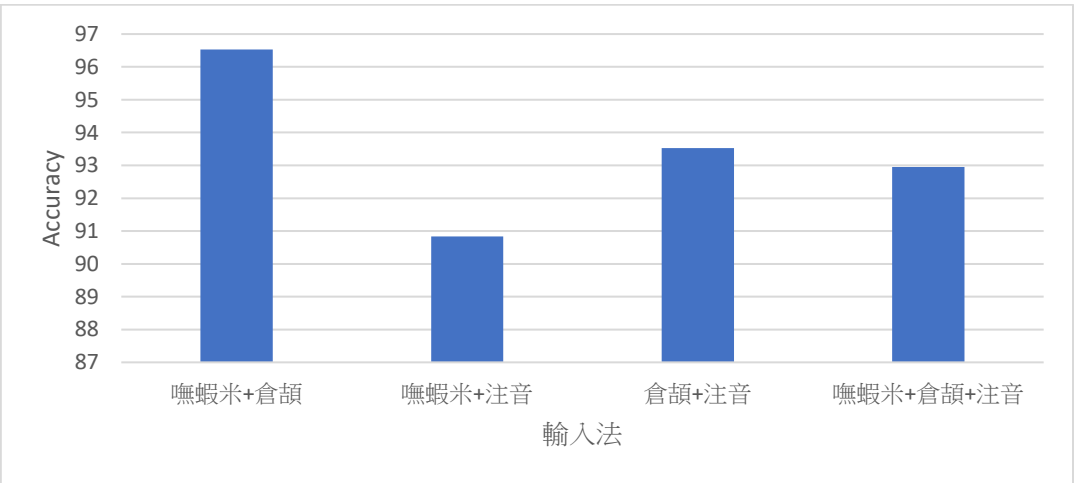
（二）不同輸入法之間準確率的比較（以下皆有判定英文）



圖（十六）：不同輸入法之間準確率的比較

從上表我們可以發現：注音的錯誤率非常的高，是因為注音有很多同音字，只要是輸入同音但不是頻率最高的字時，就會造成與對照資料不同。至於嘸蝦米與倉頡相比，嘸蝦米的錯誤率高出許多，推測原因是在常用的中文字中，嘸蝦米的打法與英文重複較多，所以最後才會得出錯誤率是倉頡 < 嘸蝦米 < 注音。

（三）混合 2 種以上輸入法準確率的比較



圖（十七）：混合 2 種以上輸入法準確率的比較

從上表看得出沒有注音的「嘸蝦米+倉頡」組合一樣維持了非常低的錯誤率，而其他三組資料因為含有注音的部分，所以錯誤率較高。而又如上一點所敘述，含有嘸蝦米的錯誤會較倉頡多。至於「嘸蝦米+倉頡+注音」則是因為注音所占的比例較低所以連帶其錯誤率就會稍稍的下降。

三、研究結果與結論

做出能夠將已輸入字串進行分析，進而將所有輸入法納入考量並篩選出最適當的字。因為輸入法程式涉及視窗焦點擷取且 Windows 7 至 Windows 10 輸入法架構有大幅度改變，因此暫以視窗程式模擬，以驗證此演算法確實可以由電腦自身判斷該用何種輸入法將看似亂碼的文字正確的轉成中、英文，且其能夠以讓人在感覺不到延遲的狀態下運行，同時達到方便與快速的目的。另外藉由我們撰寫出的程式，目前對文章的判讀已可達到 93%以上的正確率。和暨南大學碩士畢業論文相比，我們增加了兩個額外的輸入法。而和 Yo Ehara 與 Tanaka-Ishii 的 Type Any 相比，我們簡化了建立「邊界」的步驟，讓使用者的輸入更加人性化。

四、結論與應用

（一）持續提高正確率。

（二）套用至其他國家語系。

（三）包裝成 Windows 輸入法並上架。

五、參考資料及其他

（一）K. T.-I. Yo Ehara, "Multilingual text entry using automatic language detection," Proceedings of IJCNLP, pp. 441-448, 2008

（二）賴逸駿（2018）。無既定模式中英混合輸入之研究－以嘸蝦米輸入法為例。國立暨南國際大學資訊工程學系碩士論文，南投縣。 取自 https://hdl.handle.net/11296/fu7mm7

（三）英文字頻：https://en.m.wiktionary.org/wiki/Wiktionary:Frequency_lists

（四）84 年教育部字頻統計：https://language.moe.gov.tw/001/Upload/files/SITE_CONTENT/M0001/PIN/biau1.htm?open

（五）漢字構型資料庫：https://cdp.sinica.edu.tw/cdphanzi/

（六）測試資料來源：https://zh.wikipedia.org/wiki/Wikipedia:%E9%A6%96%E9%A1%B5

【評語】190013

- 本作品的主題取材自實際問題，實用性高，且作品完整度高。

- 研究方法偏重資料的整理分析與應用，但在科學意涵上的展現有繼續發展的空間。