第二十二屆旺宏科學獎

成果報告書

參賽編號：SA22-160

姓名：王冠侖

作品名稱：修正未切換注音輸入法產生之字元

參賽類別：資訊

關鍵字：注音輸入法、機器學習、維特比演算法

摘要

在電腦相當普及的現在，臺灣的許多人要使用電腦打字時皆會選擇「注音輸入法」作為輸入的方法。然而要使用注音輸入法輸入中文時若無切換輸入法則有可能會誤用到英文輸入法，接著打出一段令人不解的亂碼文字。如要輸入「今天天氣好」五字，使用英文輸入法輸入時會輸出「rup wu0 wu0 fu4cl3」。這種文字幾乎不易理解，易造成閱讀與溝通的障礙。本研究的目的即為研究將未切換到注音輸入法而打出的英數符號混和字元翻譯為漢字的方法。

本研究使用「PTT 中文語料」中的中文句子訓練 GRU、BiGRU、LSTM 和 Transformer、微調 mT5 模型以及計算維特比演算法，並與現有使用 Google 輸入工具的翻譯方法進行比較，在分數的表現上機器學習最好的是 LSTM 模型以及 Transformer 模型。整體來看以維特比演算法的 BLEU4 分數最高，在準確率以及 BLEU4 的評分皆高於 Google 輸入工具，分數分別為 0.94 與 88.3 分。可知本研究使用的維特比演算法為最佳方案。

本研究之成果在應用方面極廣，可應用於線上翻譯或聊天軟體的即時翻譯，只要會經常在英文和注音輸入法之間切換即有可能需要使用本翻譯器。本研究使用之程式碼開源於 GitHub 頁面，除了可讓使用者下載使用外，使用者也可訓練自己的模型，或者調整程式使其更符合使用者的需求。

Abstract

Nowadays, when computers are quite popular, many people in Taiwan choose the "Zhuyin input method" as the input method when typing on computers. However, when using the Zhuyin input method to input Chinese, if you do not switch the input method, you may mistakenly use the English input method, and then type out a confusing garbled text. If you want to input the five words "今天天氣好", using the English input method will output "rup wu0 wu0 fu4cl3". This style of writing is almost difficult to understand and can easily cause confusions to read or to communicate. The purpose of this research is to study the method of translating mixed characters of English and numerical symbols typed without switching to the "Zhuyin input method" into Chinese characters. This research uses Chinese sentences in the "PTT Chinese corpus" to train GRU, BiGRU, LSTM and Transformer, fine-tune the mT5 model and calculate the Viterbi algorithm, and compare it with existing translation methods by using Google input tools, in terms of score performance. The best machine learning models are then the LSTM model and the Transformer model. Overall, the Viterbi algorithm's BLEU4 score is the highest, and its accuracy and BLEU4 score are both higher than the Google input tool, with scores of 0.94 and 88.3 points, respectively. It can illustrate that the Viterbi algorithm used in this research is the best solution. The results of this work have a wide range of applications and can used to online translation or real-time translation of chat software. If you frequently switch between English and "Zhuyin input" methods, you may then need to use this kind of translator. The program code used here is an open source on the GitHub website. In addition to allowing users to download and use it, users can also train their own models or adjust the program to get better acquirement by the user's feedback.

壹、　研究動機

在日常生活中，電腦是不可或缺的電子產品。而使用電腦時最重要的其中一項動作即為「打字」，要打出不同的文字就必須使用不同的輸入法進行輸入。臺灣地區目前最主流的輸入法為注音輸入法，透過注音拼音拼出漢字。不過電腦的注音鍵盤和英文鍵盤大部分都是共用按鍵的，因此需切換模式才可輸入注音。如果要用注音輸入法輸入時模式沒有切換到注音，則會打出一串由英數字元以及符號組合的字串，且無法成功輸入漢字。這會對時常需要切換英文與注音輸入法的人造成極大的不便。若可以將未切換到注音輸入法而打出的英數符號混和字元（以下簡稱錯誤字元）轉為漢字將會提高輸入的效率。因此，本研究的目標是建構一個模型，能夠精準地將錯誤字元轉為漢字。

貳、　研究目的

一、克服中文的「同音字」特性，尋找並研究將錯誤字元轉換為漢字的方法。
二、比較不同方法的成效差異。
三、探討將研究成果實際應用之可行性。

參、　研究過程

一、文獻探討

(一) 注音輸入法

注音輸入法為一種使用注音符號輸入漢字的中文輸入法。注音符號在鍵盤上的排列有許多種，本研究所採用的排列方法為佔有率較高的「大千式」排列法（圖1）。

圖 1、大千式注音符號排列法(Sakurambo, 2007)

電腦的注音輸入法不同於手機的注音輸入法，輸入「你」一字，可先輸入「S」作為「ㄋ」，再輸入「U」作為「ㄧ」，最後輸入「3」作為三聲「ˇ」，即可得到「你」。也可先輸入「U」再輸入「S」最後輸入「3」得到「你」。代表輸入時可以無視聲母、介音以及韻母的順序，只有聲調需最後輸入。

(二) 機器學習

1. 模型架構

(1) Seq2Seq

Seq2Seq(sequence-to-sequence)模型由一個編碼器(Encoder)和一個解碼器(Decoder)組成。輸入一串序列後，會使用編碼器將序列轉換為 context vector，其中使用循環神經網路（如：GRU、LSTM），再使用解碼器轉換為另一串序列。其輸出之 context vector 會作為解碼器的輸入。

解碼器接收編碼器產生之 context vector，並輸出與輸入序列相對應的序列。編碼器與解碼器不同的點為：解碼器會以時步 t 預測時步 t + 1 的輸出。（圖 2）

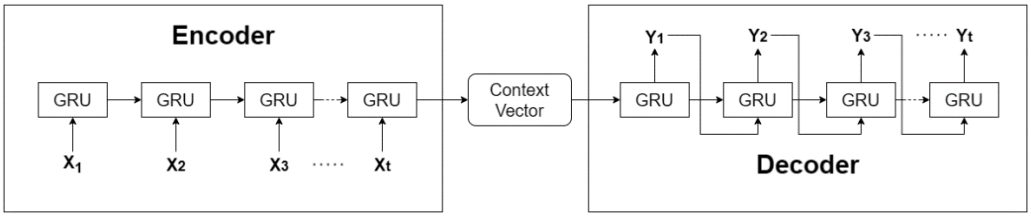


圖 2、Seq2Seq 結構示意圖，以 GRU 為例

(2) LSTM

LSTM(Long Short-Term Memory)改善了 RNN(Recurrent Neural Networks)難以捕捉長期時間關聯的缺點。其主要由四個部分組成：輸入閥(Input Gate)、單元狀態(Cell State)、輸出閥(Output Gate)、遺忘閥(Forget Gate)。

當資料輸入時，輸入閥會控制使否要將這次的值輸入並進行運算，單元狀態會記憶運算得到的數值，輸出閥會控制是否輸出此次運算得到的數值，遺忘閥會控制是否清空單元狀態（圖 3）。

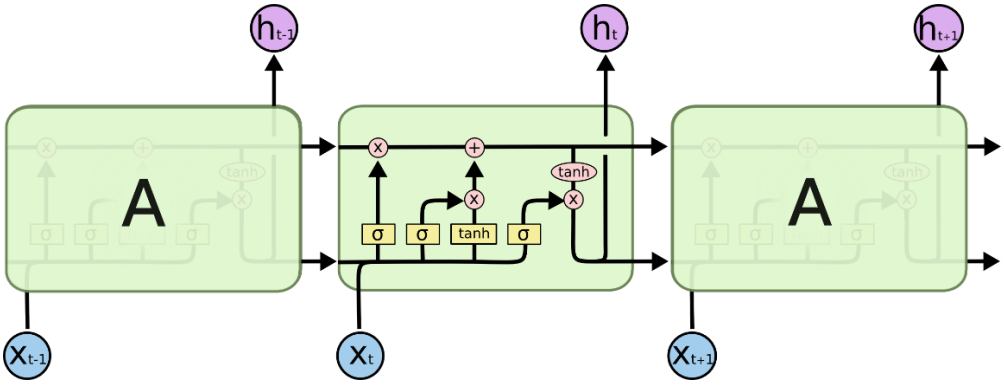


圖 3、LSTM 示意圖 [1]

(3) GRU

GRU(Gated Recurrent Unit)相較於 LSTM 加快了執行速度以及減少記憶體的使用，且將遺忘閥與輸入閥合併改以更新閥(Update Gate)取代、將輸出的單元狀態和隱藏狀態合併，如圖 4 所示。

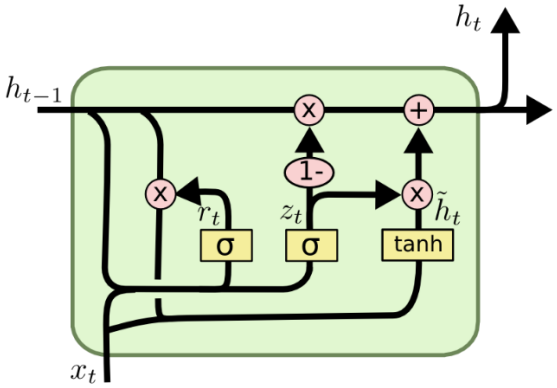


圖 4、GRU 示意圖 [1]

(4) BRNN

BRNN(Bidirectional Recurrent Neural networks)使用兩個獨立的 RNN，對於其中一個 RNN 輸入序列以正常的順序輸入；對於另一個 RNN，序列以相反順序輸入，最後連結到同一個輸出。此種模型可以同時獲得過去以及未來的狀態，改善了標準的 RNN 只能獲得過去狀態的缺點（圖 5）。

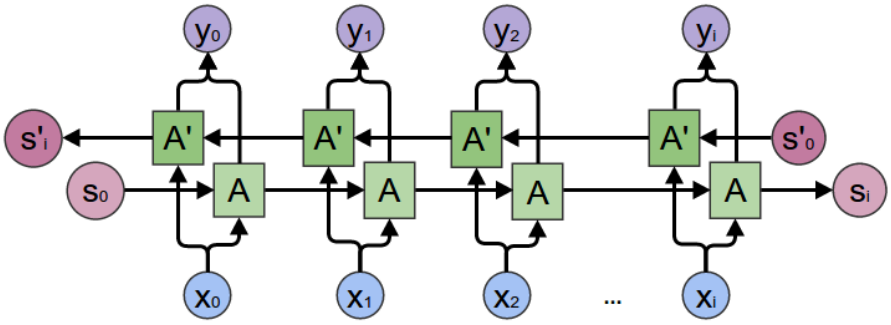


圖 5、BRNN 模型結構示意圖(Christopher Olah, 2015)

本研究所使用的 BiGRU(Bidirectional GRU)與 BRNN 結構類似，差別在於使用 GRU 取代了 RNN。

(5) Transformer

Transformer 是基於注意力機制的模型，其成果較增加了注意力機制的 RNN 模型好。

Transformer 中包括兩種注意力機制：縮放點積注意力(Scaled Dot-Product Attention)和多端口注意力(Multi-Head Attention)。在其注意力機制中分別有三個序列：Query、Key 和 Value。

縮放點積注意力會先計算 Query 和 Key 的點積得到注意力加權。為防止維度太大導致不穩定，之後會將此加權除以向量維度的平方根，再使用 softmax 函式歸一化。最後與 Value 做運算（圖 6，公式 1）。[2]

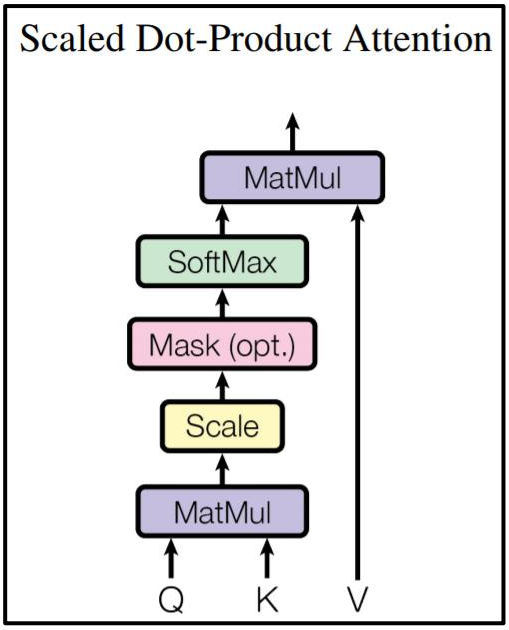


圖 6、縮放點積注意力示意圖 [2]

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{1}$$

多頭注意力的多頭代表 Attention 層的輸出空間會被分成多個獨立的頭(head)，皆為個別學習。每個頭皆可針對不同的「相關性」計算不同的注意力權重。在每個頭 Query、Key 和 Value 會分別經過三組獨立的 Dense 層運算產生三個獨立的向量，每個向量再經過縮放點積注意力處理。最後將每個頭的輸出串接再一起形成最後輸出（圖 7）。[2]

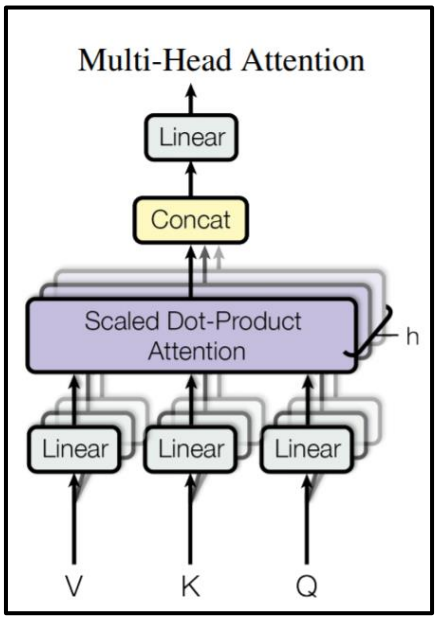


圖 7、多頭注意力示意圖 [2]

(6) T5

T5(Text-to-Text Transfer Transformer)基於 encoder-decoder（圖 8），為一種預訓練模型(pre-trained model)。

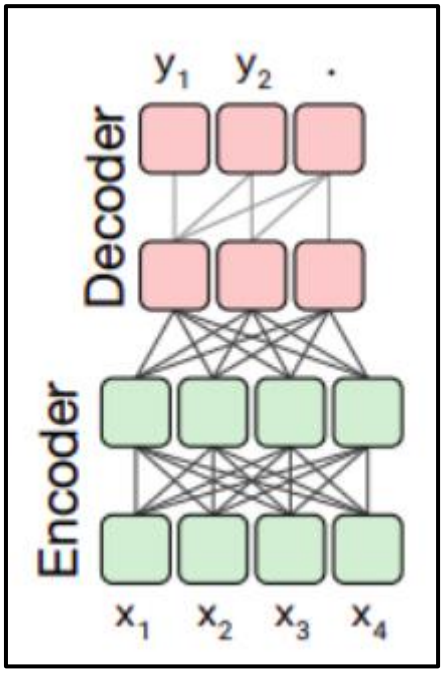


圖 8、Encoder-Decoder 示意圖 [3]

其特點為結合了所有的 NLP(Natural Language Processing)任務，並將其化為 Text-to-Text 的方式。在輸入前都需要加入前綴指定要進行的任務，例如將英文翻譯為德文，則需添加「translate English to German: 」作為前綴。

T5 等預訓練模型在使用前需要先進行微調(fine-tune)，讓已經理解語言架構的預訓練模型可以更準確地進行任務。本研究所採用的模型為 Google 發布的 mT5-small，mT5(Multilingual T5)與 T5 相比支援更多語言。

2. BLEU

BLEU(Bilingual Evaluation Understudy)可比較一個句子對於一個或多個參考句子有多相近，可用於機器翻譯的評分。

本研究所採用之 BLEU-4 分數即逐一計算 Unigram 到 4-gram 對於預測句子以及參考句子的精度分數($P_1 \sim P_4$)，再計算四個分數的幾何平均。

為防止預測句子較短時精度分數較高，會在最後乘上懲罰分數(Brevity Penalty)。公式 2 [4] 中 c 為預測句子中的詞數，r 為參考句子中的詞數。

$$\mathrm{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r/c)} & \text{if } c \leq r \end{cases}$$

$$\text{BLEU} = \text{BP} \cdot \exp(\sum_{n=1}^{N} w_n \log p_n)$$

(2)

計算後得到的分數會介於 0 和 1 之間（本研究之數值已乘以 100），在一般的翻譯任務中，0.6 到 0.7 為可到達的最佳分數。本研究與一般翻譯任務性質較不同，因此分數略高於一般分數。[5]

(三) 維特比演算法

維特比演算法(Viterbi algorithm)為一種動態演算法，用於尋找最有可能發生之觀測事件序列。可搭配隱式馬可夫模型(Hidden Markov Model，HMM)進行計算（圖 9~11）。基本計算如下。[6]

$\delta_{qi}(t)$：對於觀測值(observations)在時間 t 狀態(state)為 q_i 的形況下最大的路徑機率。

$\psi_{qi}(t)$：在時間 t 狀態為 q_i 的情況下，導致具有最大概率的序列。

1. 初始化：創建一個起始狀態 q^*，找到初始狀態的機率以及給定初始狀態的觀測值。

$$\delta_{q_i}(1) = P(q_i|q^*)P(0_i|q_i) \tag{3}$$

2. 遞推：當 t 大於或等於 2 且小於或等於 T 時，執行遞推步驟。T 是觀測值數量+1。

$$\delta_{q_i}(t) = \max_{q_j} P(q_i \,|q_j) P(o_i|q_i)\delta_{q_j}(t-1) \tag{4}$$

$$\psi_{q_i}(t) = \arg\max_{q_j} P(q_i \,|q_j) P(o_i|q_i)\delta_{q_j}(t-1) \tag{5}$$

3. 終止。

$$\text{P(Q, O)} = \max_{q_j} \delta_{q_j}(T) \tag{6}$$

$$\hat{q}_T = \arg\max_{q_j} \delta_{q_j}(T) \tag{7}$$

$$\hat{q}_t = \psi_{\hat{q}_{t+1}}(t);\ \ t = k-1, k-2, k-3, \ldots, 1 \tag{8}$$

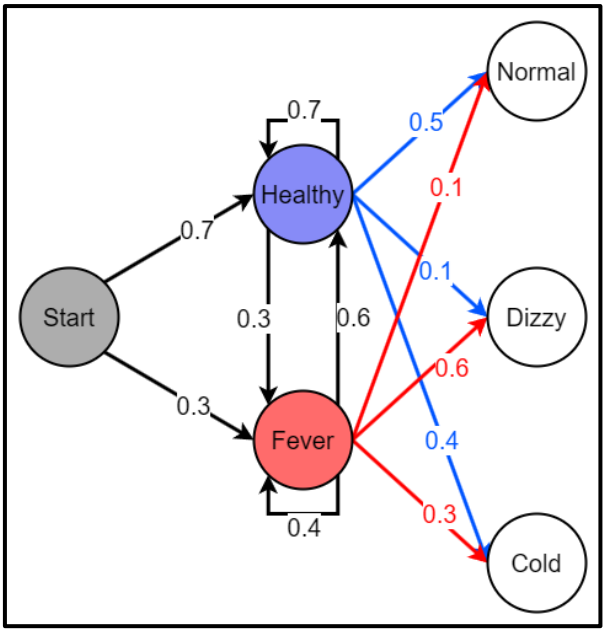


圖 9、HMM 示意圖

```
{
    "states": ["Healthy", "Fever"],
    "observations": ["normal", "cold", "dizzy"],
    "start_probability": {"Healthy": 0.7, "Fever": 0.3},
    "transition_probability": {
        "Healthy": {"Healthy": 0.7, "Fever": 0.3},
        "Fever": {"Healthy": 0.6, "Fever": 0.4}
    },
    "emission_probability": {
        "Healthy": {"normal": 0.5, "cold": 0.4, "dizzy": 0.1},
        "Fever": {"normal": 0.1, "cold": 0.3, "dizzy": 0.6}
    }
}
```

圖 10、HMM 程式示意圖

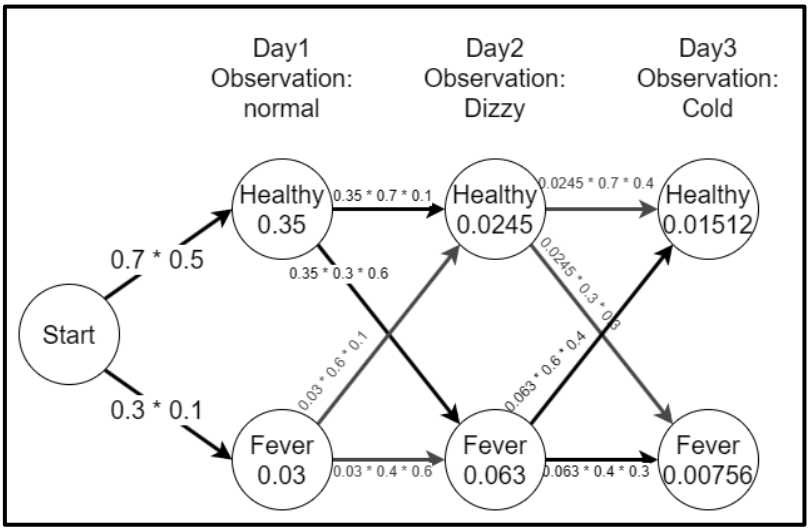


圖 11、維特比演算法示意圖

(四) Google 輸入工具

Google 輸入工具（圖 12）是一個可供使用者輸入字詞的服務。本研究透過「Requests」套件爬取線上試用的 Google 輸入工具進行翻譯，並與自行訓練、微調以及計算的模型進行比較。

圖 12、Google 輸入工具圖標

二、研究設備與器材

(一) 筆記型電腦

1. 硬體設備
   - CPU：AMD Ryzen 7 4800H with Radeon Graphics
   - GPU：NVIDIA GeForce GTX 1650 Ti
2. 軟體套件
   - Windows10 (22H2)
   - CUDA Toolkit (11.6)
   - cuDNN (8.9.1)
   - Python (3.9.12)
   - Tensorflow (2.10.0)
   - Numpy (1.23.2)
   - Requests (2.31.0)
   - sacreBLEU (2.3.1)
   - OpenCC (0.1.7)
   - pypinyin (0.49.0)

(二) Google Colaboratory

1. 硬體設備
   - CPU：Intel Xeon with 2 vCPUs
   - GPU：NVIDIA Tesla V100
2. 軟體套件
   - Ubuntu (22.04.2 LTS)
   - CUDA Toolkit (11.8)
   - Python (3.10.12)
   - Tensorflow (2.12.0)
   - Numpy (1.23.5)
   - sacreBLEU (2.3.1)
   - Transformers (4.31.0)
   - SentencePiece (0.1.99)

三、研究方法

(一) 訓練資料準備

不管是機器學習或是維特比演算法，皆需要大量的訓練資料才可正確轉換。預期的訓練資料應有數對，每一對包含兩行句子：分別為錯誤字元以及相對應正確的中文句子（圖 13）。

```
[
    ("jo4gp6ak7g/4ru,6g6cjo41o4nj0 -65k4fmp6bp61j6cjo4？", "為什麼聖結石會被酸而這群人不會？"),
    ("u.3ao6u.3504gp68 bjo4n 2k718 ej84", "有沒有戰神阿瑞斯的八卦"),
    ("jo4gp6ak75k4ak72ji bp6d041;4fu.6", "為什麼這麼多人看棒球"),
    ...
]
```

圖 13、預期的訓練資料格式示意圖

注音輸入法只在臺灣地區普及，且較少人上傳大量錯誤字元以及其對應之漢字供他人下載使用，不易尋找現成的訓練資料。因此本研究尋找大量的中文句子，經過整理後再將中文句子逐字轉換為注音並以對照表將注音轉換為大千式鍵盤上所對應的英數符號。

本研究使用「PTT 中文語料」(Justin Yang, 2021)中的「Gossiping-QA-Dataset-2_0.csv」（圖 14）中文句子作為訓練資料，此檔案中的句子皆從批踢踢實業坊（PTT）抓取而來。檔案中分為一個問句與一個對應的答句，共有 774115 對問句與答句。本研究只需中文句子，不須問與答的關係，因此捨棄問答關係共得到 1548230 行中文句子。

```
question,answer
為什麼 聖結石 會被酸而 這群人 不會？,質感 劇本 成員 都差很多好嗎 不要拿腎結石來污辱這群人
為什麼慶祝228會被罵可是慶端午不會？,因為屈原不是台灣人，是楚國人。
有沒有戰神阿瑞斯的八卦?,爵士就是阿瑞斯 男主角最後死了
理論與實務最脫節的系,哪個系不脫節...你問最不脫節的簡單多了...
為什麼PTT這麼多人看棒球,肥宅才看棒球 系壘一堆胖子
為什麼達摩祖師傳那麼好看?,達摩從頭到尾都是被動（別人問他問題
```

圖 14、Gossiping-QA-Dataset-2_0.csv 檔案部分截圖

原始的句子中不只有中文字，還有英文、數字以及符號。為了得到沒有參雜非中文字元的句子同時盡量保持句子語意暢通，將句中字元分為漢字、英文、數字以及符號。漢字不進行任何處理，英文則將其從句中刪除。若是數字則將其轉換為漢字，符號則保留中文常用標點符號（、，。？！：；），其餘則刪除。如「20 年前補過 很受雄女學生歡迎」一句，保留所有漢字，無英文故不進行刪除，將阿拉伯數字「20」轉換為漢字「二十」，最後刪除「過」與「很」之間的空白符號，處理後的句子為「二十年前補過很受雄女學生歡迎」。

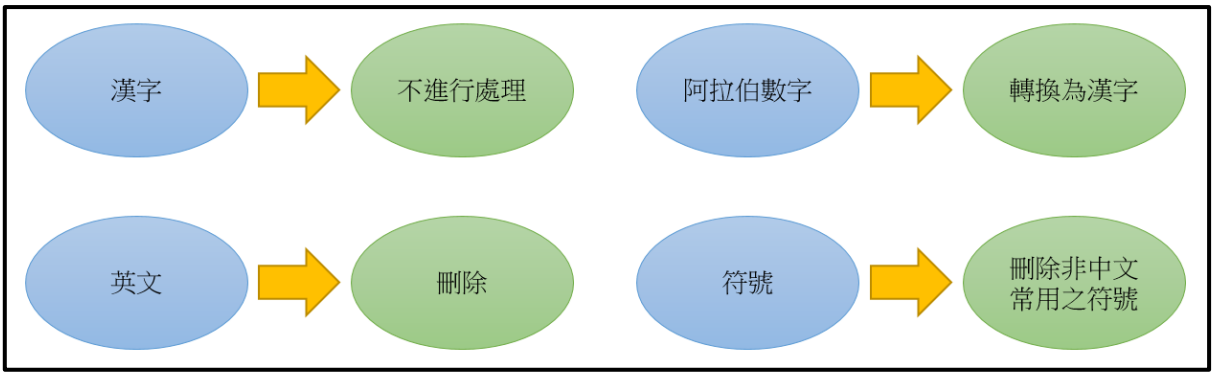


圖 15、去除非中文字元方法示意圖

本研究參考「number2hanzi」(xiaoyvning, 2018)中的「index.js」，將其中的簡體字改為繁體字、增加數字最大可轉換位數，修正「二」以及「兩」的使用，並利用相同原理改寫成 Python 函式。例如原程式對於「22222」的轉換為「二万二千二百二十」，經過修改後的程式轉換之結果為「兩萬兩千兩百二十」。

取得整理過的中文句子後，本研究使用套件「pypinyin」將句中漢字逐一轉換為注音。然而 pypinyin 的轉換效果並不理想（如：將「噱」翻為「ㄐㄩㄝˊ」，而正確發音應為「ㄒㄩㄝ」)，破音字的翻譯也不符合日常使用（如：將「一個」的「個」翻為「ㄍㄜˋ」，而較常使用的發音為「ㄍㄜ˙」)。研究此套件的轉換方法後，得知此套件會將輸入的句子進行分詞，並且於對照表(phrases_dict.py)中尋找與單詞最符合的拼音，再將拼音轉換為注音輸出。但對照表中的單詞皆為簡體字，使用繁體字會配對不到任何單詞而逐字對照發音。逐字對照發音會失去前後字元的資訊導致轉換準確度下降，因此本研究使用套件「OpenCC」編寫一個函式將對照表中的簡體字翻譯為繁體字（圖 16)，解決多數破音字問題。

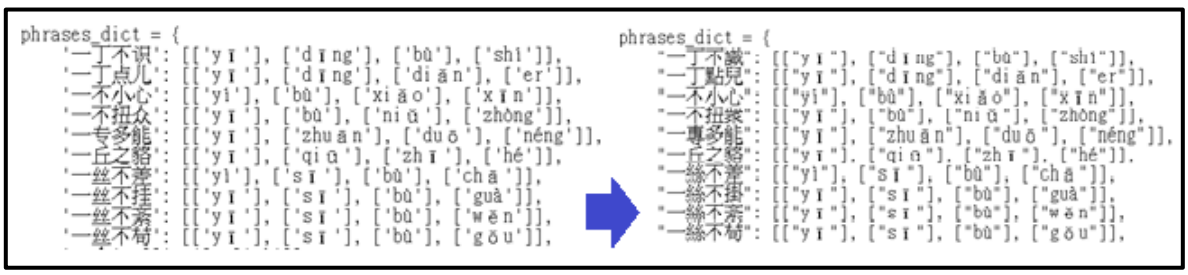


圖 16、phrases_dict.py 部分截圖，左為原 phrases_dict.py，右為修改過後之 phrases_dict.py

其餘轉換錯誤的問題可透過「pypinyin」中的「load_phrases_dict」載入自定義的拼音來解決。

將整理過後的句子轉為注音後須再透過對照表（圖 17）將注音轉換為大千式鍵盤上對應的英數符號。因 pypinyin 轉換的注音一聲皆無特別標示聲調，如此無法轉換出空白符號，所以先進行逐字檢查，若字尾不為聲調符號則加入一聲聲調符號。

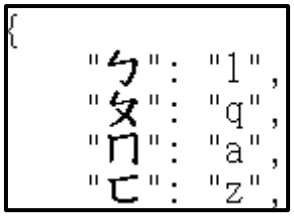


圖 17、大千式鍵盤注音與英數符號對照表部分截圖

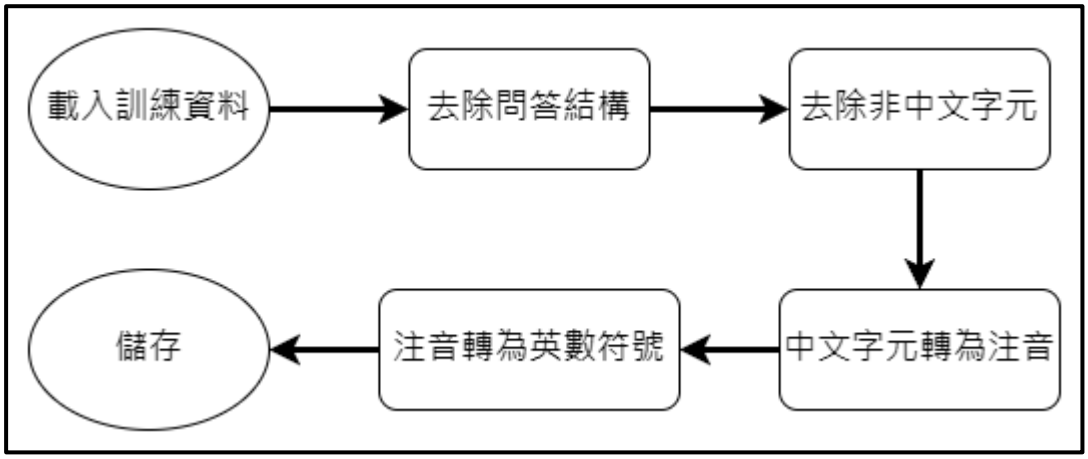


圖 18、訓練資料預處理流程圖

(二) 機器學習訓練

本研究將錯誤字元以及漢字視為兩種不同的語言，並分別使用 GRU、BiGRU、LSTM、Transformer 以及 mT5-small 等模型架構進行機器翻譯任務的訓練與微調。

在使用以上提及的模型時皆須先從訓練資料中分出驗證資料以及測試資料，本研究將訓練資料、驗證資料以及測試資料以 14:3:3 的比例分配。

資料輸入模型前，須先將句子中的每個字詞轉換為數字再輸入模型，而中文不像大部分語言會使用空格將字詞分開。為了使模型知道一個句子中哪些部分是一個字，因此本研究使用不常見的符號「ᛯ」作為分隔符號。使用此符號而不使用空白的原因為錯誤字元的一聲聲調會使用空白符號，且此符號較不常見，不會與字詞混淆。

GRU、BiGRU、LSTM 以及 Transformer 的中文句子皆需要開始與結束的標記，因此在句首以及句尾增加起始標記「[start]」以及結尾標記「[end]」(圖 19)。

```
[
    ("jo4ᛏgp6ᛏak7ᛏg/4ᛏru,6ᛏg6ᛏcjo4ᛏ1o4ᛏnj0 ᛏ-6ᛏ5k4ᛏfmp6ᛏbp6ᛏ1j6ᛏcjo4ᛏ？", "[start]ᛏ為ᛏ什ᛏ麼ᛏ聖ᛏ結ᛏ石ᛏ會ᛏ被ᛏ酸ᛏ而ᛏ這ᛏ群ᛏ人ᛏ不ᛏ會ᛏ？ᛏ[end]"),
    ("u.3ᛏao6ᛏu.3ᛏ504ᛏgp6ᛏ8 ᛏbjo4ᛏn ᛏ2k7ᛏ18 ᛏej84", "[start]ᛏ有ᛏ沒ᛏ有ᛏ戰ᛏ神ᛏ阿ᛏ瑞ᛏ斯ᛏ的ᛏ八ᛏ卦ᛏ[end]"),
    ("jo4ᛏgp6ᛏak7ᛏ5k4ᛏak7ᛏ2ji ᛏbp6ᛏd04ᛏ1;4ᛏfu.6", "[start]ᛏ為ᛏ什ᛏ麼ᛏ這ᛏ麼ᛏ多ᛏ人ᛏ看ᛏ棒ᛏ球ᛏ[end]"),
    ...
]
```

圖 19、插入分隔符號以及開始、結束標示後的訓練資料示意圖

mT5 模型在進行任務時都須加上前綴指定要執行的任務，本研究使用「translate

engTyping to Traditional Chinese:」作為翻譯任務的前綴，並將前綴置於每個錯誤字元組成的句子前面。

最後 GRU、BiGRU、LSTM 以及 Transformer 所需之訓練資料以「Tensorflow」中的「TextVectorization」物件將字詞轉為數字，mT5 所需之訓練資料則是用「transformers」載入「tokenizer」進行。

GRU、BiGRU、LSTM 以及 Transformer 等模型皆使用筆記型電腦進行訓練，mT5-small 模型則使用 Google Colaboratory 進行微調。

1. GRU

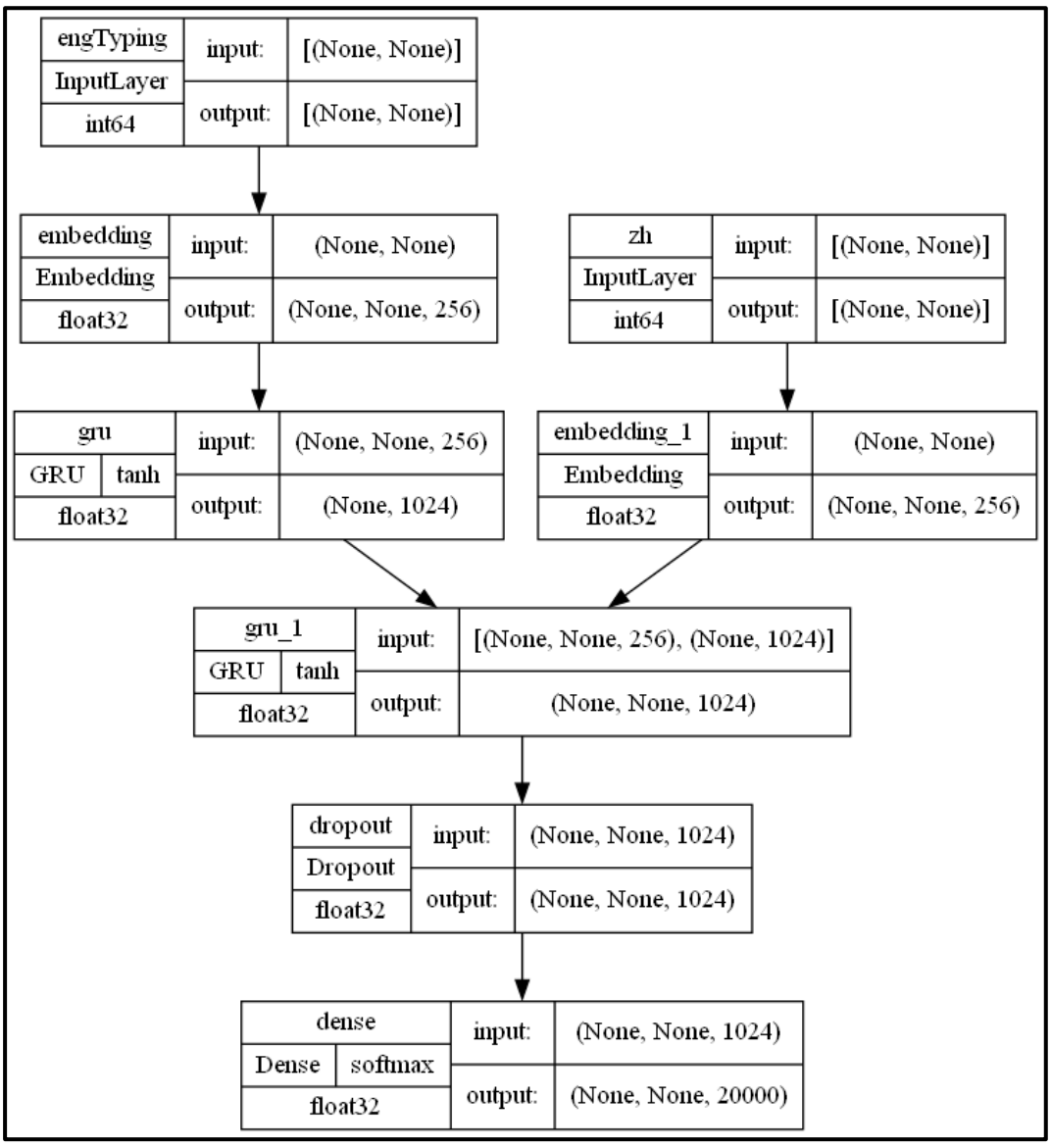


圖 20、本研究之 GRU 模型結構圖

2. BiGRU

與 GRU 不同的點為，此模型之編碼器為雙向 GRU。

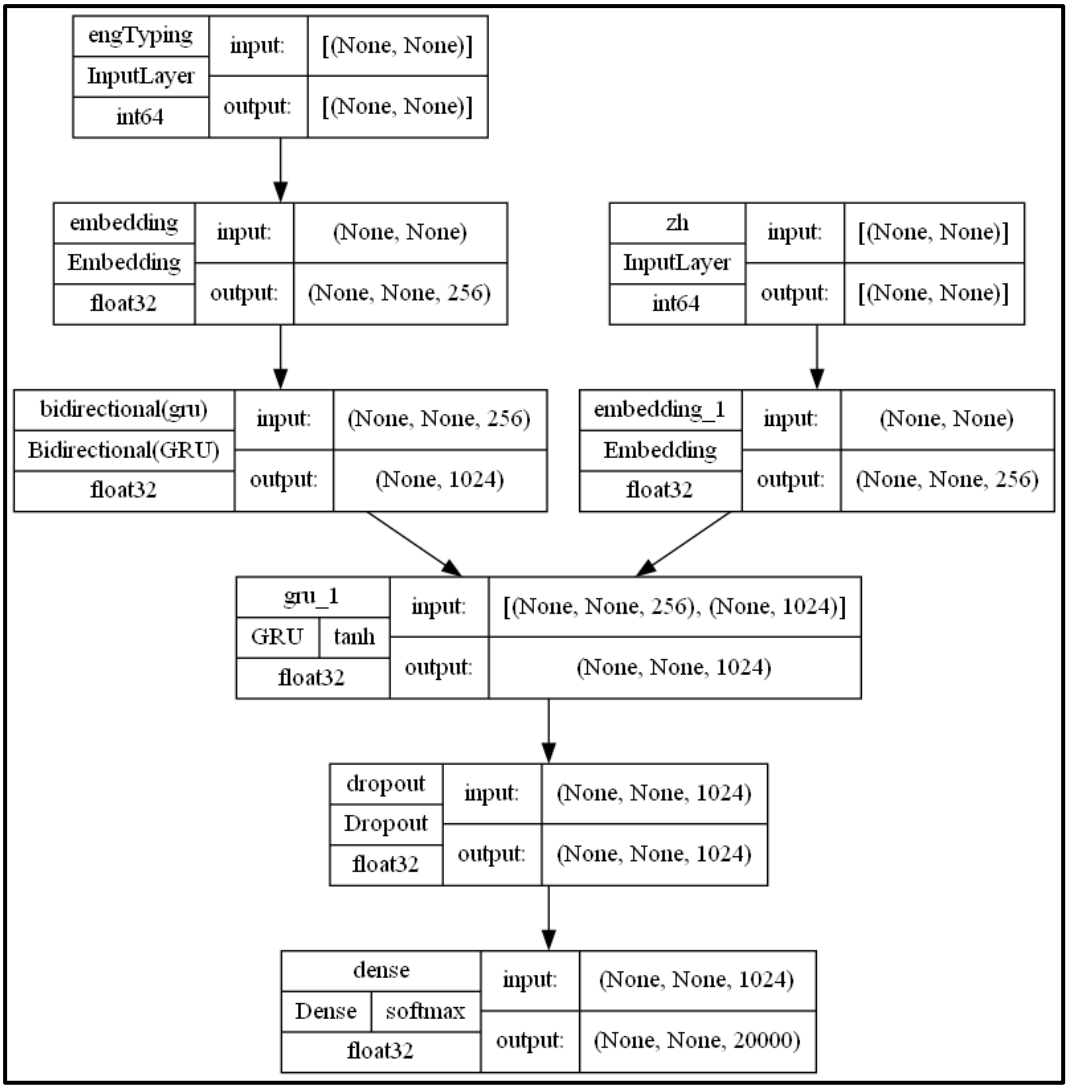


圖 21、本研究之 BiGRU 模型結構圖

3. LSTM

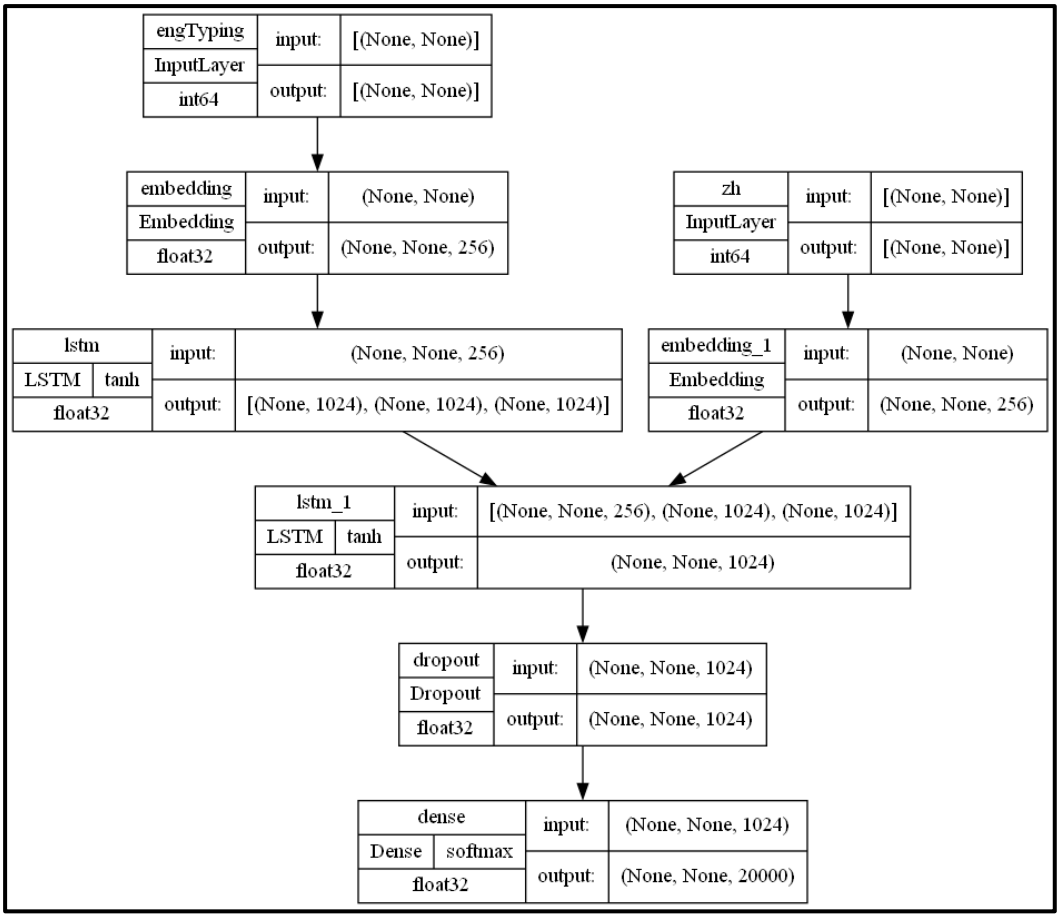


圖 22、本研究之 LSTM 模型結構圖

4. Transformer

在 Transformer 模型中，注意力機制會忽略序列中元素的順序，如此可更容易找出每個元素間的關聯性。但在使用機器翻譯時，句子中的文字順序十分重要，改變順序甚至會影響語意。因此這個在模型中使用位置嵌入法(Positional Embedding)在詞嵌入向量加入位置資訊：將元素與位置資訊分別輸入嵌入層得到嵌入向量後相加。

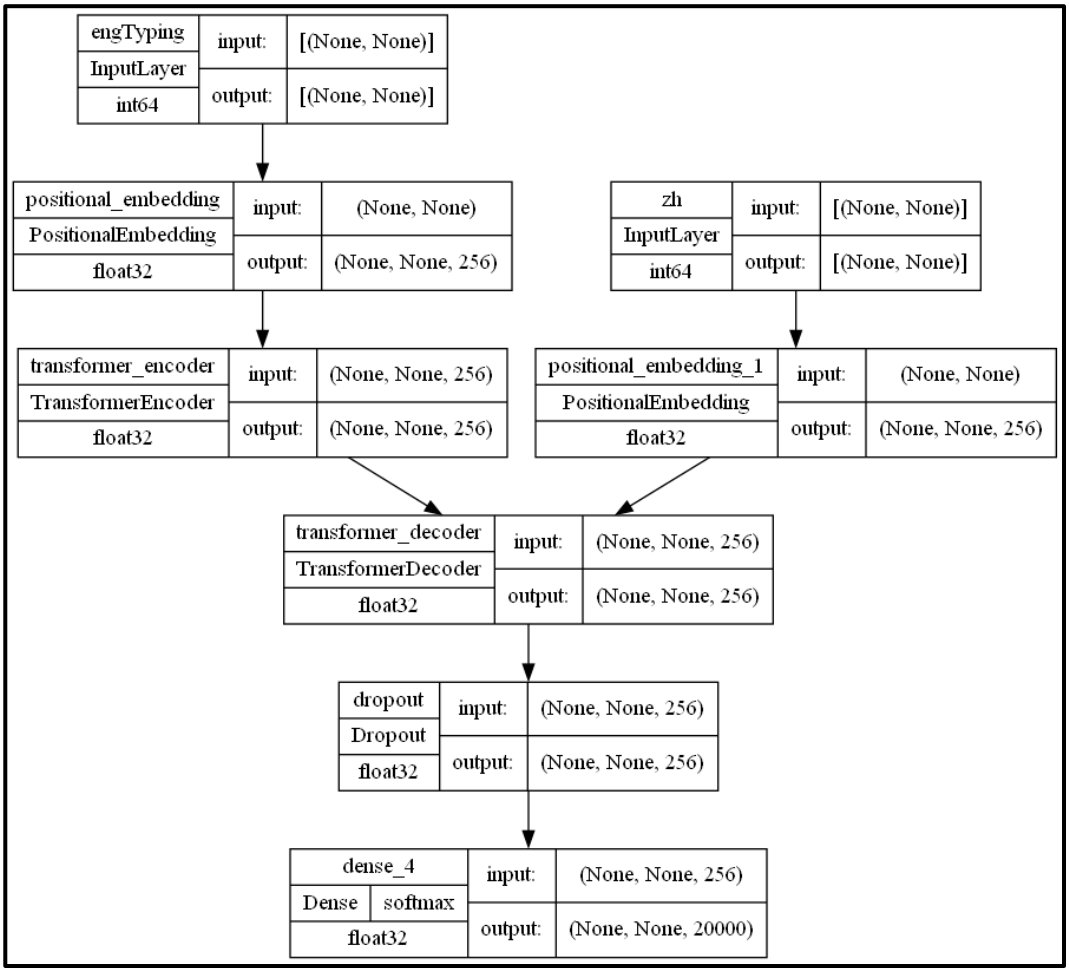


圖 23、本研究之 Transformer 模型結構圖

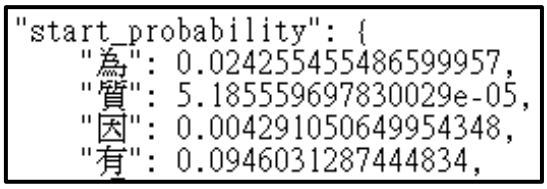
5. mT5-small

本處使用的 mT5-small 模型層數為 8 層，且有 6 個注意力頭。

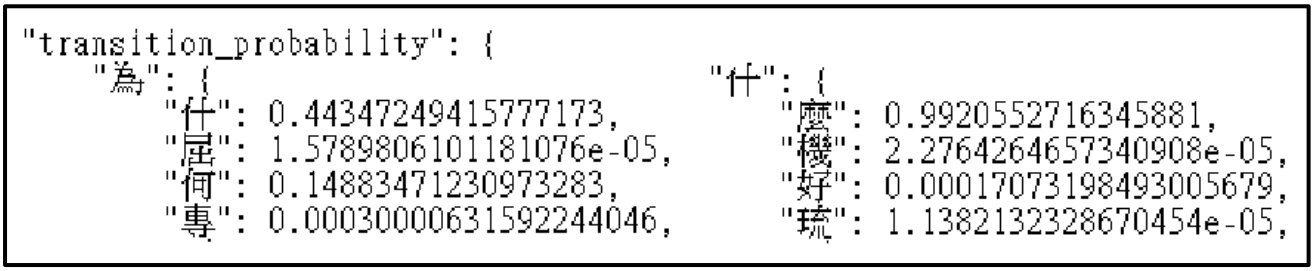
(三) 維特比演算法計算

要使用維特比演算法進行任務，需要給定一個 HMM 模型。其中需包含起始機率(start probability)、轉移機率(transition probability)、發射機率(emission probability)以及隱含狀態(state)，可觀察的觀測值(observations)作為輸入。本研究為確保訓練資料與機器學習之訓練資料一致，僅使用資料的前 70%進行訓練。接著將漢字作為隱含狀態，使用錯誤字元作為觀測值，並使用訓練資料計算每個漢字作為句子開頭的起

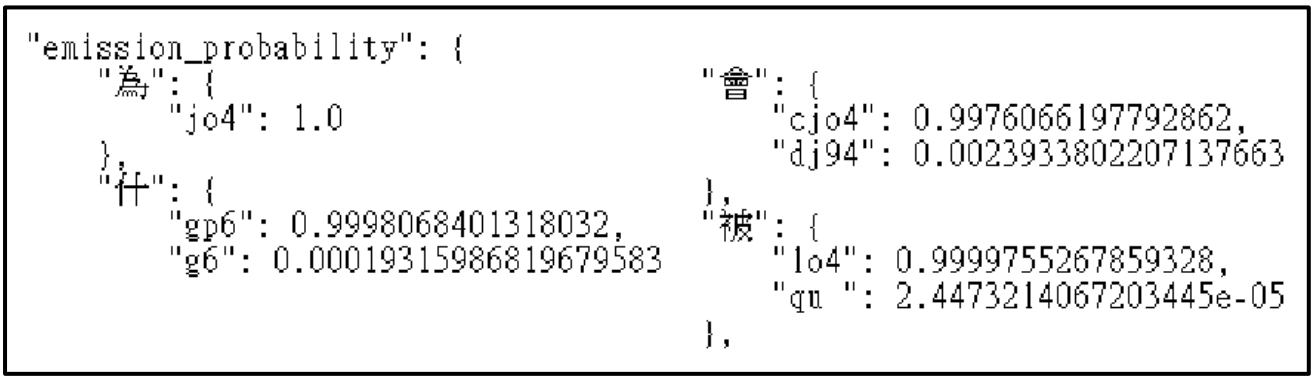
始機率、漢字連接另一漢字的轉移機率以及每個漢字對於觀測值的發射機率，並用 json 格式儲存（圖 24）。給定計算得出的 HMM 模型後即可使用維特比演算法透過觀測值找到最有可能發生之觀測事件序列（即為翻譯結果）。



(a)



(b)



(c)

圖 24、 HMM 模型之機率部分截圖

(a) 起始機率 (b)轉移機率 (c)發射機率

在計算時，若將所有漢字帶入隱含狀態進行運算，計算時間會增加很多。經過研究後發現，隱含狀態不必帶入所有漢字，只需帶入可能會產生的漢字即可。例如輸入「su3cl3」(你好)，可能產生的漢字只有發音為「ㄋㄧˇ」以及「ㄏㄠˇ」的漢字。因此在訓練 HMM 模型時會同時紀錄每個音可能代表的漢字（圖 25）。透過只帶入可能產生的漢字至隱含狀態，可大幅減少運算時間以及運算資源的消耗。

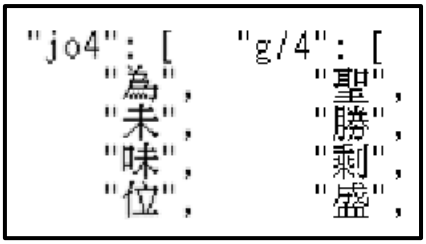


圖 25、紀錄每個發音可能代表的漢字（部分截圖）

(四) 現有的方法

經過查詢發現網路上現有的翻譯器大部分使用了 Google 輸入工具進行翻譯工作，於是透過瀏覽器的開發者工具（圖 26）研究 Google 輸入工具後發現，可透過爬蟲使用線上試用的 Google 輸入工具進行翻譯。本研究透過「Requests」套件進行爬蟲，對其網站發送請求後即可得到一個串列，裡面包含翻譯結果與其他資訊（圖 27）。

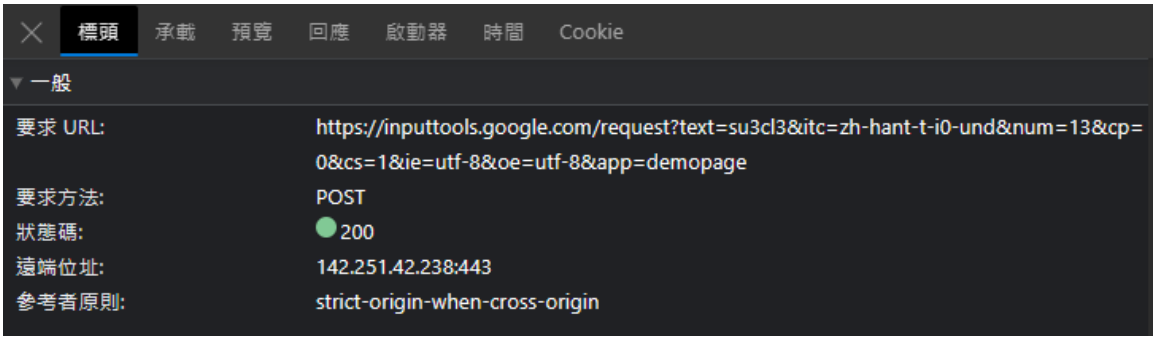


圖 26、使用瀏覽器開發者工具查看網路請求

["SUCCESS",[["su3cl3",["你好","妳好","你","妳","擬","禰","儗","旎","靡","鈮","堄","掜","隬"],[],{"annotation":["su3 cl3","su3 cl3","su3","su3","su3","su3","su3","su3","su3","su3","su3","su3","su3"],"candidate_type":[0,0,0,0,0,0,0,0,0,0,0,0,0],"lc":["69 69","69 69","69","69","69","69","69","69","69","69","69","69","69"],"matched_length":[6,6,3,3,3,3,3,3,3,3,3,3,3]}]]]

圖 27、發送請求後得到之回傳訊息

(五) 研究成果

1. 機器學習

本研究訓練得出的模型並非最後一次迭代產出的模型，而是選出在所有迭代產出的模型中表現最佳的模型進行儲存。

(1) GRU

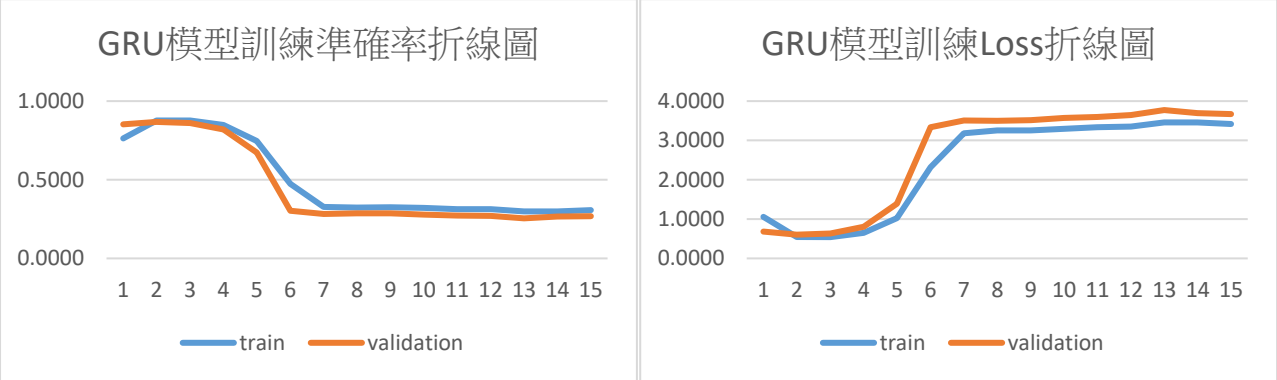


圖 28、GRU 模型訓練準確率折線圖（左）與 Loss 折線圖（右）

表 1、GRU 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.8797	65.27	80.1	70.8	63.3	56.6
Loss	BP	ratio	hyp_len	ref_len	
0.5242	0.972	0.973	2830323	2909826	

(2) BiGRU

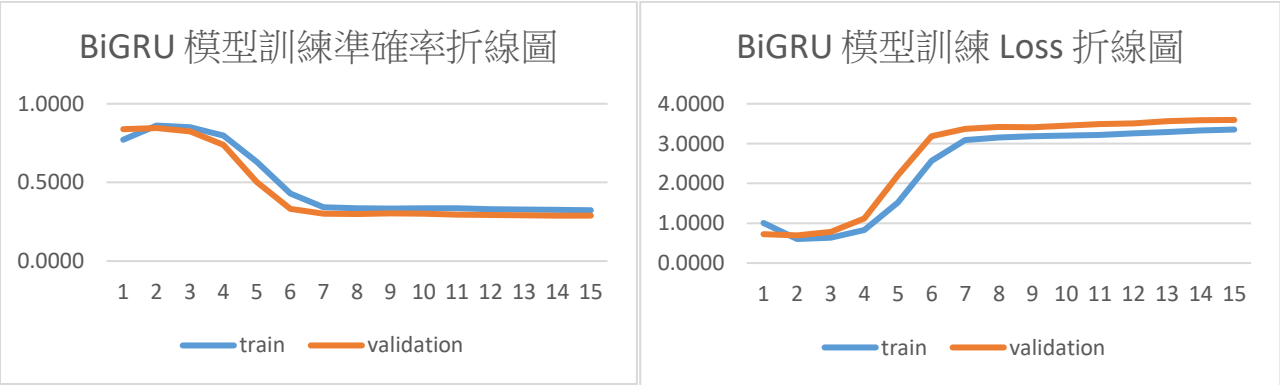


圖 29、BiGRU 模型訓練準確率折線圖（左）與 Loss 折線圖（右）

表 2、BiGRU 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.8581	56.39	75.2	61.9	52.4	44.4
Loss	BP	ratio	hyp_len	ref_len	
0.6118	0.983	0.983	2860500	2909826	

(3) LSTM

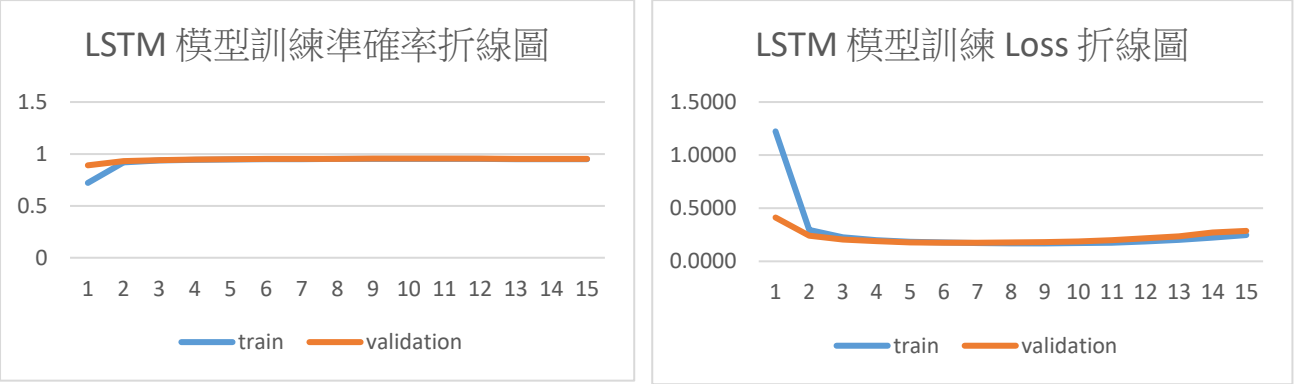


圖 30、LSTM 模型訓練準確率折線圖（左）與 Loss 折線圖（右）

表 3、LSTM 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.9553	78.56	86.6	81.8	77.3	73
Loss	BP	ratio	hyp_len	ref_len	
0.1555	0.988	0.988	2874949	2909826	

(4) Transformer

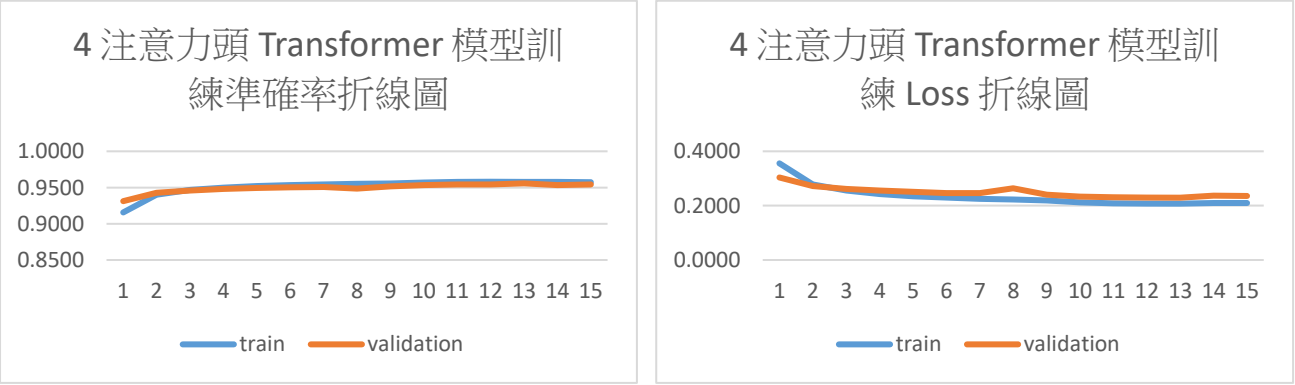
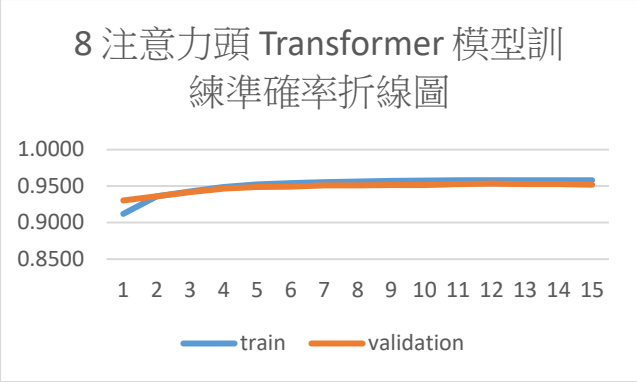


圖 31、4 注意力頭 Transformer 模型訓練準確率折線圖（左）與 Loss 折線圖（右）

表 4、4 注意力頭 Transformer 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.9577	79.56	87	82.6	78.4	74.4
Loss	BP	ratio	hyp_len	ref_len	
0.208	0.989	0.989	2877760	2909826	



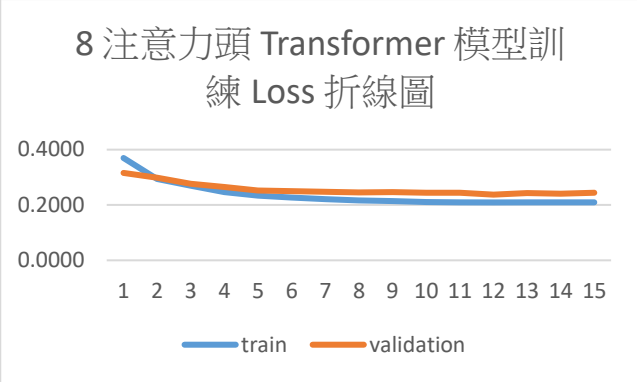
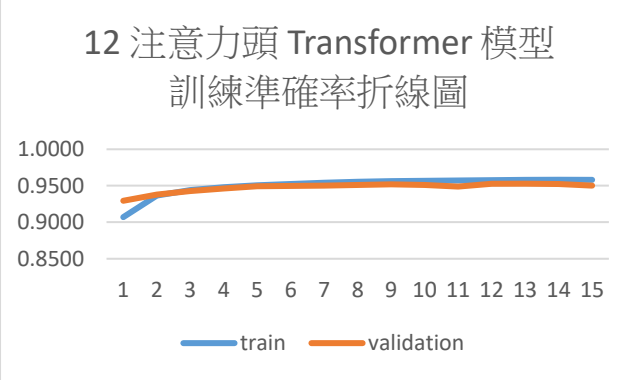


圖 32、8 注意力頭 Transformer 模型訓練準確率折線圖（左）與 Loss 折線圖（右）

表 5、8 注意力頭 Transformer 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.9557	79.25	86.8	82.3	78	73.9
Loss	BP	ratio	hyp_len	ref_len	
0.215	0.989	0.989	2878068	2909826	



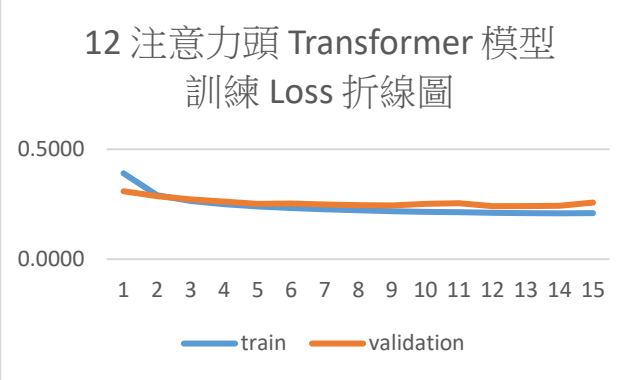
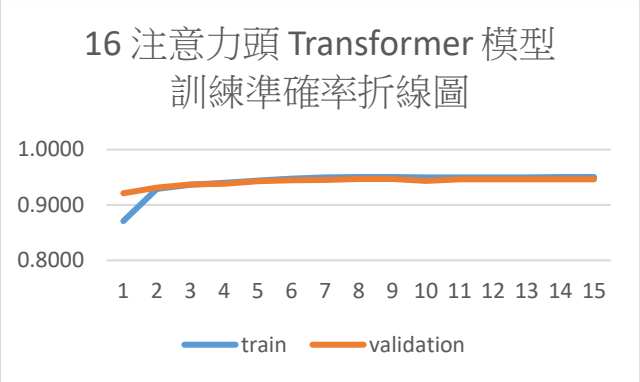


圖 33、12 注意力頭 Transformer 模型訓練準確率折線圖（左）與 Loss 折線圖（右）

表 6、12 注意力頭 Transformer 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.9552	79.04	86.8	82.1	77.8	73.6
Loss	BP	ratio	hyp_len	ref_len	
0.2178	0.989	0.989	2878061	2909826	



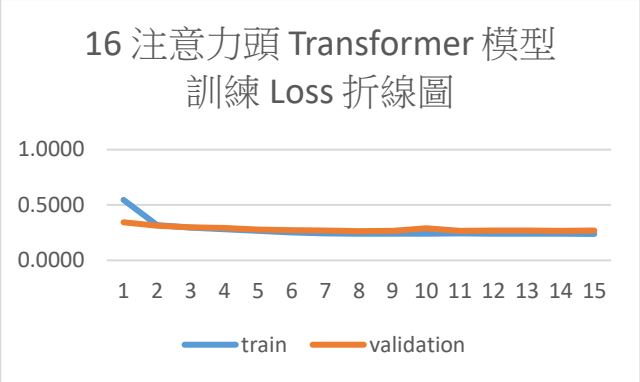
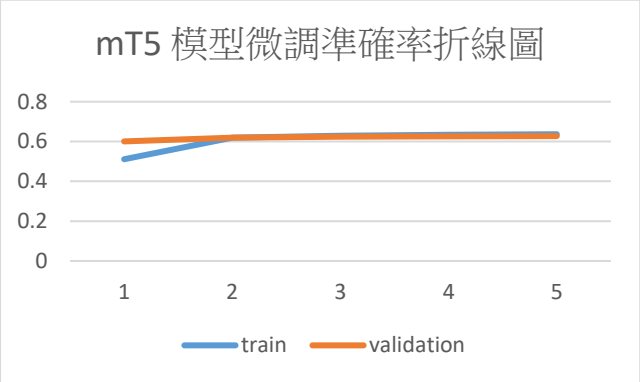


圖 34、16 注意力頭 Transformer 模型訓練準確率折線圖（左）與 Loss 折線圖（右）

表 7、16 注意力頭 Transformer 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.9492	77.8	86.2	81.1	76.3	71.8
Loss	BP	ratio	hyp_len	ref_len	
0.2381	0.989	0.989	2878065	2909826	

(5) mT5



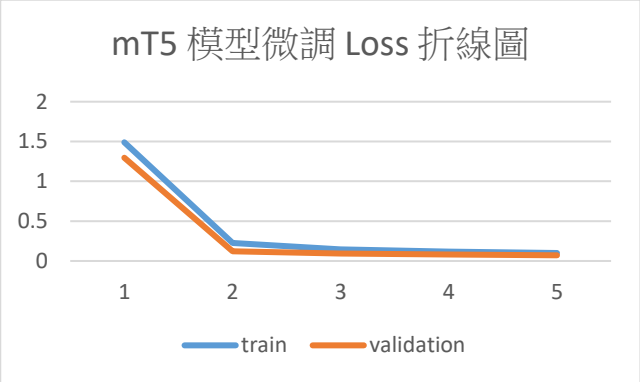


圖 35、mT5 模型微調準確率折線圖（左）與 Loss 折線圖（右）

表 8、mT5 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.6076	83.17	92.7	86.9	81.4	76.2
Loss	BP	ratio	hyp_len	ref_len	
0.07	0.989	0.989	2878258	2909826	

在機器學習中，最好的為 LSTM 模型以及 Transformer 模型，兩者的準確率、Loss 以及 BLEU4 分數相差並不大，準確率約在 0.94~0.96，Loss 約在 0.15~0.24，BLEU4 分數約在 77~80。

GRU 模型在準確率、Loss 以及 BLEU4 皆略好於 BiGRU，推測其原因為此種翻譯並不是常規的翻譯，使用雙向模型結構不一定有助於轉換。

mT5 模型的準確率遠低於其他模型，僅有約 0.61。但其 Loss 卻較其他機器學習模型低，約為 0.07，且 BLEU4 分數高達 83.17，高於所有機器學習模型。實際測試後發現多數字發音以及字形正確，少部分字字形錯誤，大部分全形標點符號會轉為半形，且有時翻譯的句子會出現重複的字詞，如翻譯「su3cl3」會得到「你好好」，然而正確的翻譯為「你好」。推測因為會出現重複字詞導致準確率較低，但對於 BLEU4 分數影響較小。

本研究分別訓練了注意力頭數為 4、8、12 和 16 的 Transformer 模型，前三者的準確率皆約為 0.95~0.96，Loss 約為 0.20~0.22，BLEU4 分數約為 79~80，表現差異不大，唯獨 16 注意力頭的模型準確率只有 0.9492，Loss 高達 0.2381，BLEU4 分數只有 77.8。可見注意力頭數不一定愈多愈好。

2. 維特比演算法

為確保評估維特比演算法之分數時使用之資料與機器學習相同，使用資料集的後 15%進行評估。

表 9、維特比演算法搭配 HMM 模型各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.9448	88.3	94.5	90.3	86.3	82.6
	BP	ratio	hyp_len	ref_len	
	1	1	2909767	2909826	

維特比演算法不同於機器學習可能會輸出長短不同的字句，只要輸入的觀測值中無 HMM 模型沒有記錄到機率的漢字，其長度必定等於輸出的序列長度，且可確保翻譯得出的字發音一定正確，會錯誤的只有可能為字形，因此得出的 BP 分數為 1，同時 BLEU4 分數較高。

3. Google 輸入工具

為確保評估 Google 輸入工具之分數時使用之資料與機器學習相同，使用資料集的後 15%進行評估。

表 10、Google 輸入工具各項分數

Accuracy	BLEU4	1gram	2gram	3garm	4garm
0.9337	87.17	93.9	89.6	85.7	81.9
	BP	ratio	hyp_len	ref_len	
	0.994	0.994	2893645	2909826	

使用 Google 輸入工具進行翻譯字詞的準確率以及 BLEU4 分數極高，尤其 BP 分數來到第二名的 0.994，只比維特比演算法低 0.006，相較其他模型更不易漏字。

4. 綜合比較

表 11、各項方法數值對照表

方法 / 項目	準確率	Loss	BLEU4
GRU	0.8797	0.5242	65.27
BiGRU	0.8581	0.6118	56.39
LSTM	0.9553	0.1555	78.56
4 注意力頭 Transformer	0.9577	0.2080	79.56
8 注意力頭 Transformer	0.9557	0.2150	79.25
12 注意力頭 Transformer	0.9552	0.2178	79.04
16 注意力頭 Transformer	0.9492	0.2381	77.80
mT5	0.6076	0.0700	83.17
維特比演算法	0.9448	N/A	88.30
Google 輸入工具	0.9337	N/A	87.17

從表 11 可知，成效最好的為使用維特比演算法搭配 HMM 模型的方法，準確率約為 0.94，而 BLEU4 分數高達 88.3，為所有方法中的最高得分。

(六) 實際測試

要將以上成果投入使用，需先對使用者輸入的文字進行處理。本研究參考 Input_Method_auto-Modifier (voidism, 2018, ModifyInputType.py)中的「IsZhInput」和「IsZhInputs」函式判斷一串字串是否為錯誤字元。

從使用者輸入的文字中提取出錯誤字元後，需要將每個字分開，本研究注意到在使用注音輸入法時，按下聲調才可完整地打出一個字，因此聲調會最後輸入。而聲調一聲、二聲、三聲、四聲以及輕聲分別對應大千式鍵盤上的「 」(空白符號)、「6」、「3」、「4」以及「7」，所以尋找以上字元的位置即可分割錯誤字元（圖 36）。

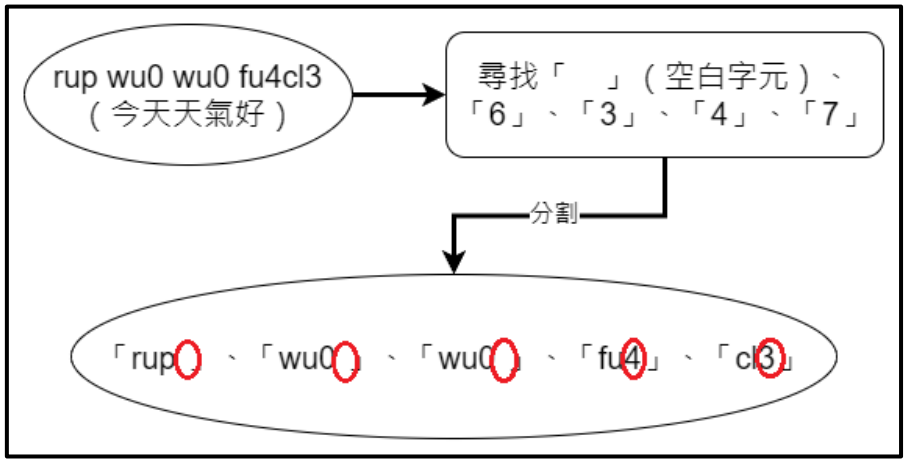


圖 36、透過聲調分割錯誤字元示意圖

表 12 至表 14 為各種方法對於六句不同的句子之表現，其中紅字代表與正確翻譯不符合。

表 12、各項方法翻譯五個字的句子之結果

方法 / 輸入	vup 5j6vu84284m3	u.3w96j0 ru8456
正確翻譯	新竹下大雨	有台灣價值
GRU	新竹下大雨	有台灣價值
BiGRU	新竹下大雨	有台灣價值
LSTM	新竹下大雨	有台灣價值
4 注意力頭 Transformer	新竹下大雨	有台灣價值
8 注意力頭 Transformer	新竹下大雨	有台灣價值
12 注意力頭 Transformer	新竹下大雨	有台灣價值
16 注意力頭 Transformer	新竹下大雨	有台灣價值
mT5	新竹下大雨	有台灣價值值
維特比演算法	新竹下大雨	有台灣價值
Google 輸入工具	新竹下大雨	有台灣價值

表 13、各項方法翻譯十個字的句子之結果

方法 / 輸入	aup3s06cj84ep w96m3u u;4a87？	vul3m4vu ej8 yp3ai7s84ai7cl3t
正確翻譯	閩南話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
GRU	仇南話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
BiGRU	若男話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
LSTM	閩南話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
4 注意力頭 Transformer	閩南話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
8 注意力頭 Transformer	閩南話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
12 注意力頭 Transformer	敏難話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
16 注意力頭 Transformer	閩南話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
mT5	閩南話跟台語一樣嗎?	小玉西瓜怎麼那麼好吃
維特比演算法	閩南話跟台語一樣嗎？	小玉西瓜怎麼那麼好吃
Google 輸入工具	閩南話跟台語一樣嗎	小玉西瓜怎麼納麼好吃

表 14、各項方法翻譯二十個字的句子之結果

方法 / 輸入	cp3u065j/42k7g4a87？xu/4j94su35 2l4b41p3m061p3ru.4u.3 ao3rmp	yjo4cl3su3g;4j;3s/65 2l4j;3xj4xu/4u 2j0 2k7bp6y942u/ su3x87！
正確翻譯	很嚴重的事嗎？另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在盯你啦！
GRU	很嚴重的是嗎？另外你知道日本原味道，問題	最好你上網能知道網路創一種的人在個你不啦
BiGRU	很嚴重的是嗎？以外還知道日本原本就有美軍	最好你上網能知道網路連一百的人在還唱啦！
LSTM	很嚴重的事嗎？另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在盯你啦！
4 注意力頭 Transformer	很嚴重的事嗎？另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在釘你啦！
8 注意力頭 Transformer	很嚴重的事嗎？另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在叮你啦！
12 注意力頭 Transformer	很嚴重的是嗎？另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在盯你啦！
16 注意力頭 Transformer	很嚴重的事嗎？另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在蹤你啦！
mT5	很嚴重的事嗎?另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在盯你啦!
維特比演算法	很嚴重的事嗎？另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在盯你啦！
Google 輸入工具	很嚴重的事嗎？另外你知道日本原本就有美軍	最好你上網能知道網路另一端的人在盯你啦

由表 12 至表 14 可知，所有方法在翻譯只有五個字的短句子時，皆完美翻譯。翻譯十個字的句子時，成果較差的 GRU 和 BiGRU 模型出現字音錯誤，12 注意力頭 Transformer 模型出現字形錯誤，mT5 模型翻譯出半形符號，Google 輸入工具則是無法翻譯位於結尾的標點符號，且出現字形錯誤。翻譯二十個字的長句子時，模型普遍出現錯誤，尤其以 GRU 和 BiGRU 的錯誤最為嚴重。

整體來看，只有 LSTM 模型以及維特比演算法完全正確的翻譯這六句句子。

肆、 討論及應用

一、模型的可修改性

隨時代變動，語言也會改變，若要使本研究之成果在今後也適用，必須可隨時修改

模型。而重新計算 HMM 模型所消耗的運算資源遠低於機器學習，更適合更動以及修改。

二、開放原始碼

為使所有人皆可使用本研究之研究結果，本研究將程式碼公開於 GitHub 頁面。使用者可使用此程式碼訓練符合自己需求的模型。（附錄 1）

三、用於聊天軟體

為使大眾能夠更方便的使用本研究之訓練成果，將翻譯模型部署到 Discord Bot（附錄 2）上。在使用 Discord 這個聊天軟體時，只要將本研究建立之機器人邀請至欲使用服務的伺服器，當有人使用英文輸入法輸入中文時，此機器人可即時翻譯錯誤字元至漢字。（附錄 8）

四、修正同音字

本研究使用維特比演算法搭配 HMM 模型進行計算的成果十分良好，因此將發音正確但用字錯誤的中文句子去除字形的資訊後再輸入進行計算即可得出發音正確字形也正確的中文句子（圖 37，附錄 7）。

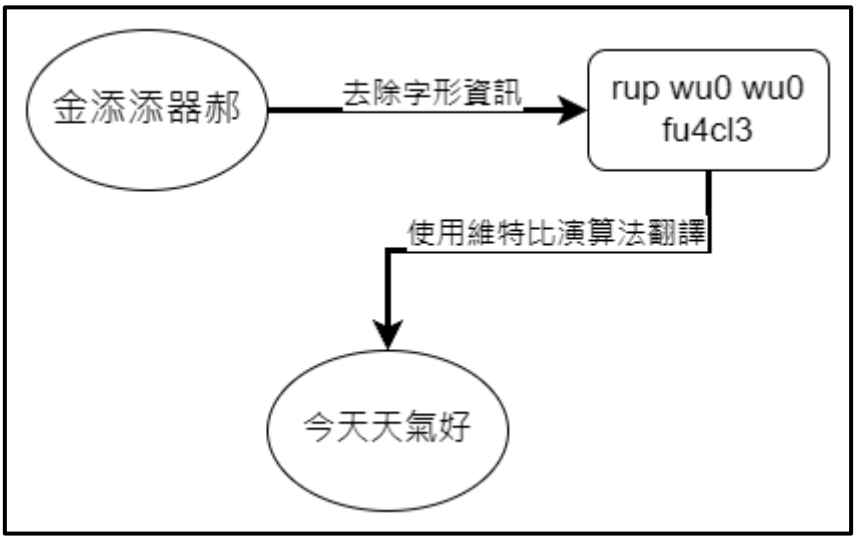


圖 37、透過維特比演算法修正破音字流程圖

伍、 結論

本研究針對注音輸入法在切換上的問題，以網路論壇的文字作為訓練資料，訓練多種機器學習模型和使用演算法，成功做出翻譯準確的翻譯器。最好的方法為使用維特比演算法搭配 HMM 模型，其準確率約為 0.94，且 BLEU4 分數高達 88.3，為最高分，比第二高的 Google 輸入工具高出約 1.13 分。GRU 模型略好於 BiGRU 模型，得知雙向結構不一定適用於此。LSTM 模型以及 Transformer 成效不相上下，而 Transformer 的注意力頭數對於成效並無太大影響，只可注意到 16 注意力頭的 Transformer 模型準確率和 BLEU4 分數稍微較低，Loss 分數稍高。對於 mT5 模型，其準確率相較其他方法有較大的落差，推測原因為少數翻譯字詞會重複以及全形符號會被翻譯為半形符號。本研究的成果對比現有的 Google 輸入工具，只有

維特比演算法所有分數皆高於 Google 輸入工具。LSTM 以及各 Transformer 模型準確率皆高於 Google 輸入工具，BLEU4 分數卻較低。其餘模型如 GRU、BiGRU 以及 mT5 的分數皆低於 Google 輸入工具。

綜觀整體，維特比演算法搭配 HMM 模型為最好的方法，略勝於現存的 Google 輸入工具，可精準的將錯誤字元翻譯為中文。

若希望提高各方法的翻譯能力，需增加訓練資料的數量，讓模型或演算法學會更多字詞的組合。同時訓練資料也須更精準，包括句中英文和數字的處理，以及如何正確地將漢字轉換為注音。

在應用方面，可用於線上翻譯或聊天軟體的即時翻譯，甚至可修正中文句子中錯誤的同音字。此成果增加了注音輸入法打字的便利性，使大眾能夠輕鬆翻譯錯誤字元。

陸、 參考資料

[1] Christopher Olah. (2015, August 27). Understanding LSTM Networks. Retrieved from https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, & Illia Polosukhin. (2017). Attention Is All You Need. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

[3] Colin Raffel, Noam, Adam Roberts, Katherine Lee, Sharan, Narang, Michael Matena, Yanqi Zhou, Wei Li, & Peter J. Liu. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. Google, Mountain View, CA 94043, USA.

[4] Kishore Papineni, Salim Roukos, Todd Ward, & Wei-Jing Zhu. (2002). BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

[5] Ketan Doshi. (2021, May 10). Foundations of NLP Explained — Bleu Score and WER Metrics. Retrieved from https://towardsdatascience.com/foundations-of-nlp-explained-bleu-score-and-wer-metrics-1a5ba06d812b

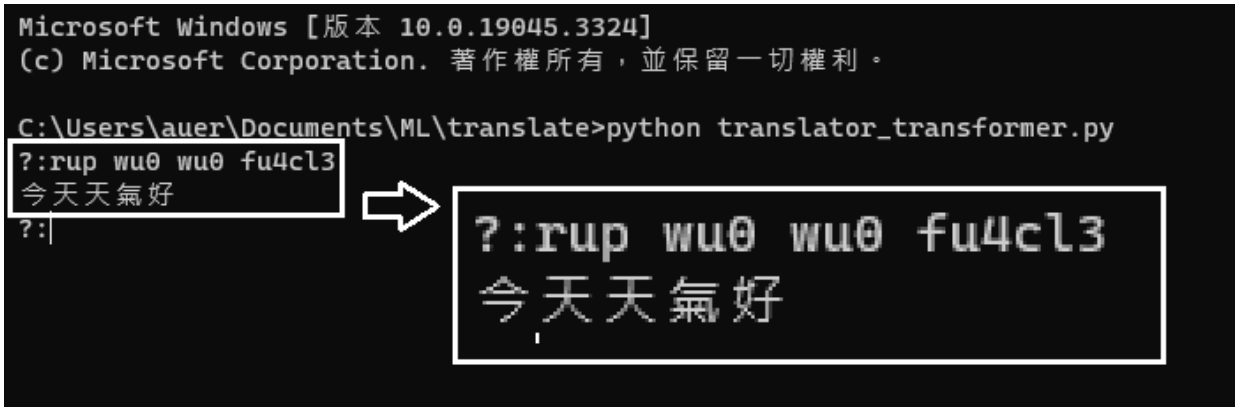
[6] Paul Butler. (2021, March 3). Intro to the Viterbi Algorithm. Retrieved from https://medium.com/mlearning-ai/intro-to-the-viterbi-algorithm-8f41c3f43cf3

[7] 黃逸華、林采薇 (譯) (2022)。Keras 大神歸位：深度學習全面進化！用 Python 實作 CNN、RNN、GRU、LSTM、GAN、VAE、Transformer。臺灣：旗標科技股份有限公司。(François Chollet, 2021)

附錄

1. GitHub 開源頁面連結（報告書定稿時版本）
   https://github.com/gallen881/bopomofo2hanzi/tree/c806ffce1ad2ce1d5776b729c01bf13559185cd0
2. 聊天軟體翻譯機器人邀請連結
   https://discord.com/api/oauth2/authorize?client_id=1102165886766415932&permissions=274872748779&scope=bot

3. 在終端機中執行 Transformer 模型翻譯



4. 在終端機中執行 LSTM 模型翻譯

5. 在終端機中執行維特比演算法翻譯

```
Microsoft Windows [版本 10.0.19045.3324]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。

C:\Users\auer\Documents\ML\translate>python translator_viterbi.py
?:j;4cj/6
(2.4397423422810877e-07, ['旺', '宏'])
total time: 0.0
?:
```

6. 在 Google Colaboratory 執行 mT5 模型翻譯

```
✓ 7 秒
prefix = "translate engTyping to Traditional Chinese:".split()
split_char = '?'
punctuations = '、，。？！：；'

def engTyping_insert_split_char(sentence: str, split_char: str) -> str:
    insert_times = 0
    sentence_list = list(sentence)
    for i, char in enumerate(sentence):
        if char in ' 6347' + punctuations:
            sentence_list.insert(i + insert_times + 1, split_char)
            insert_times += 1
    return ''.join(sentence_list[:-1])

inputs = tokenizer(prefix +                                    split_c
outputs = model.generate(inp                                   top_k=30
print(tokenizer.decode(outpu

?:ru fu4vm,6vu6
機 器 學 習
```

7. 利用維特比演算法修正錯誤同音字

```
Microsoft Windows [版本 10.0.19045.3324]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。

C:\Users\auer\Documents\ML\translate>python fix_hanzi.py
?:金添添器郝
今天天氣好
?:
```

8. 研究成果應用於線上聊天軟體