Final Report

在這次的 NER 任務中，我們利用深度學習模型來對文本資料進行標註。

我們分成兩個方向，一是確保資料的乾淨程度，二是調校模型的參數以及結構，以下我們會分別對資料以及模型做詳細的說明，並總結我們所觀察到的現象，還有可改進的方向。

該報告使用筆記軟體 Notion 所完成，可點擊下方連結前往閱讀。

https://www.notion.so/Final-Report-f7cb57f3efd34c339d40635ae135f2e4

Members

- **Sharpkoi** 交通大學應數系大四學生
- **JiaLing_Zheng** 成功大學統計系大三學生
- **crowrowrow** 交通大學應數系大四學生
- **tukennytw** 清華大學電機系大四學生
- **RedF** 暨南大學資工系大四學生

Hardware Environment

我們使用Google Colab所提供的免費運算環境來進行這次的比賽。

- OS: Ubuntu 18.04.3 LTS
- GPU: Randomly chosen from Nvidia K80s, T4s, P4s and P100s

📄Dataset

從比賽開始到最後，官方總共只釋出了兩個訓練集，以下我們以 ***Train 1*** *以及* ***Train 2*** *分別表示第一個訓練集和第二個訓練集，並互相比較不同之處。*

Train 1

- 總共120篇對話
- 不傾向將標點符號算進 `entity_text`
- 不將Line歸類為 `contact`
- 不將泰國歸類為 `location`
- `time` 標籤的數量比**Train 2**前120篇多一些

Train 2

- 總共200篇對話，其中前120篇跟 **Train 1** 一樣
- 傾向將標點符號算進 `entity_text`
- 將Line歸類為 `contact` ，此外還有電話號碼以及網址
- 將泰國歸類為 `location`

> https://plotly.com/~SharpKoi/28/?share_key=P34QFTNA2IBYq8O1xqyN6Z

總結來說，如果目標只是為了爬榜，只要針對time label以及med_exam label的資料加強訓練，就可以拿到不錯的分數；倘若要實際運用在生活中的本議題上，則需要增加其他標籤的數量，並針對較稀少的標籤加強訓練，才能保護其他稀少但重要的隱私。

最後我們將 **Train 2** 前120篇的time labels 替換成 **Train 1** 的time labels，使用替換後的 **Train 2** 餵給我們的模型做訓練。

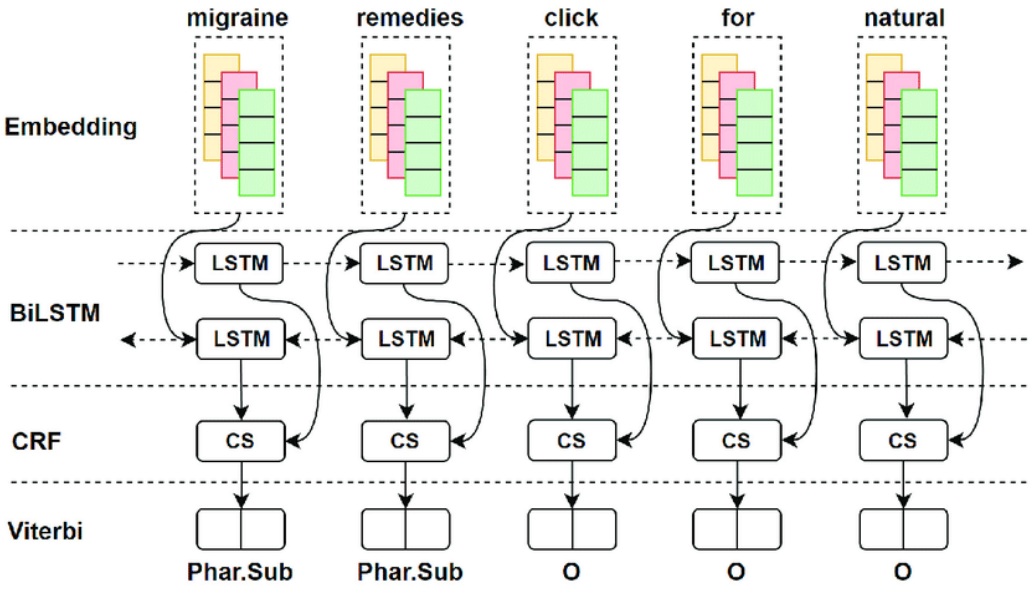
🧠Model

我們訓練 ***Bert+BiLSTM-CRF*** *模型來標註資料，以下我們會說明最終模型的結構、訓練參數，以及表現。*

Implementation

首先要感謝 Kashgari 這個非常方便且有力的套件，我們利用這個套件嘗試許多預訓練模型和模型結構，詳細的實作代碼已附上，在 `AICUP2020/` 下。

Structure



上圖是我們使用的模型結構示意圖，該模型依序分成 Input Layer, Embedding Layer, BiLSTM Layer, CRF Layer and final Output Layer。

首先文字分詞後會先進入Input Layer，接著到Embedding Layer做詞向量的嵌入，這裡我們利用Google的預訓練模型 "BERT" 來負責嵌入，嵌入向量後會進入雙向的長短期記憶層(BiLSTM)，這層會根據上文下文來判斷該詞的意義，最後再把運算的結果丟進CRF Layer，由此層使用CRF演算法來判斷該詞的標籤。

Model Parameters

▼ **For model**

- Bert type = BERT-base, Chinese Google
- LSTM Units = 256
- Dropout = 0.4

▼ **For training**

- Epochs = 100
- Batch size = 16

Model Performation

下圖是我們訓練模型100個epochs後所記錄下來的學習歷程，我們將總共200個對話以 8:2 的比例切分成 訓練集(160筆) 以及 驗證集(40筆)。

F1 Score是模型在每個epoch結束時驗證的結果，而其中分數最高的是 0.75214，我們將其模型參數儲存下來，對官方給的最終測試資料做預測，得到的系統分數分別是 0.7511335(Public) 以及 0.7622392(Private)。

> https://plotly.com/~RedF/8/?share_key=mb3hzK7FjldqqNiH8XusaG

Conclusion

從報告中可以看出我們模型的 validation loss 還是很高，一直處於 **overfitting** 的狀態，我們嘗試進行近千次的訓練以試圖降低 validation loss，最後只能降至300多，上傳分數也沒有提升，這意味著即使模型**在驗證集上表現良好，也未必適應測試集**。

我們應該從中選出幾個較好的模型參數進行預測，應試圖找到最適應**測試集**的模型參數，而非最適應**驗證集**的模型參數。

另外 Bert model 可以在下游任務中進行Fine tuning，也就是說他可以調整到最適應本次任務的詞向量，但苦於硬體方面的限制，我們未能找到詞向量的最優解，只能以 Bert 預訓練的模型參數來嵌入詞向量。

而我們也並未對訓練資料有足夠的了解，倘若我們能得到更多有關訓練資料的資訊，也或許能找到更多值得嘗試的不同優化方向。

Reference

- Google Colab
- 【Github】Kashgari
- 【DOC】Kashgari document
- 【Github】預訓練中文嵌入模型
- 【Paper】BERT
- 【Paper】BiLSTM-CRF