IRIE project 2—Group 1

GitHub: https://github.com/tsaihsingyu/IRIE_project2

一、組員及分工

學號	姓名	實作	報告
B04902046	廖莉祺	Task 2 BiLSTM	Task 2 BiLSTM + Q + 結論
B04902055	陳巧蓁	Task 1 BiLSTM / Task 2 Majority+Unifact	
B04902063	陳昱儒	Task 1 RandomForest	Task 1 RandomForest
B04902104	蔡欣妤	Task 1 SVM	Task 1 SVM

二、方法

(一)TASK 1 (Given Tokens)

A. Bidirectional LSTM (tokens)

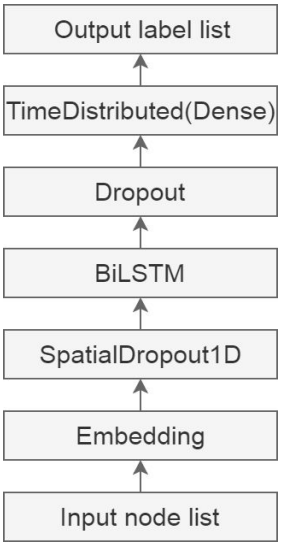
由於Task 2已可以做到正確率100%，因此對於Task 1，我們專注在如何預測每個node的label。由於Node的label會受到前後文影響甚大，我們把句子中的每個node轉為word vector後接在一起丟進BiLSTM，並直接預測句子中每個node的label。模型架構如下方左圖。

此方法在預測Node label得到81.73%的正確率，而加上Task 2的B方法後在預測edge上的F1 score可以達到90.18%。

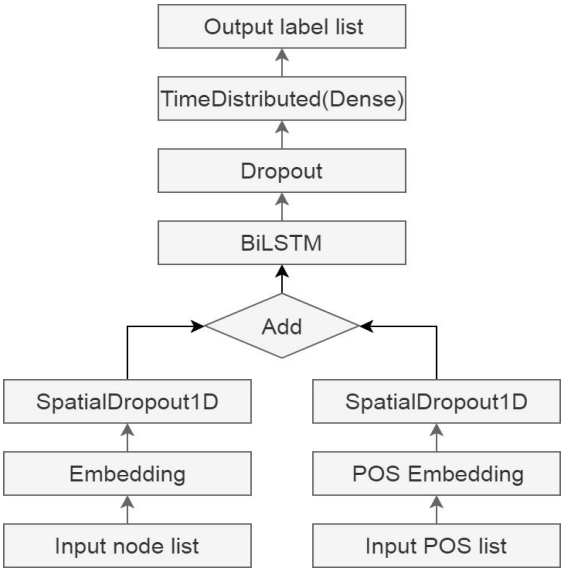
B. Bidirectional LSTM (tokens+pos)

由於node的label跟POS tag有很大的關連性，因此我們嘗試將node的POS資訊一起加入模型，希望可以進一步提升A方法的效能。模型架構如下方右圖。

此方法在預測Node label得到84.43%的正確率，而加上Task 2的B方法後在預測edge上的F1 score可以達到97.21%。



(Model A)



(Model B)

C. SVM (pos)

在用token預測node的方面，我們嘗試了將token轉成vector以及加入pos的資訊放入SVM進行訓練。但經過實驗，我們發現加入word vector會造成SVM預測準確度大幅下降，而且只用pos的資訊時表現為最佳。因此，我們統計了各個token中各種pos出現的數量，將統計出的vector放入SVM中得到node預測的結果為F1-score是0.7，再將node的預測結果放入task 2的模型中預測edge，得到超過0.9的F1-score。

D. RandomForest(pos)

因task2在給訂node label預測edge的情況下已能達到100%的準確率，在task1中我們就只專注在如何使用token預測node label，我直接使用了token的pos_tag，並沒有將token做字詞的embedding，並以node為單位依照node中token的pos_tag做one-hot encoding，預測的label種類則只分為manner,value或是剩下其他(others)，這是依據後面task2的所需資訊做的預測分類，最後使用sklearn中的RandomForestClassifier隨機森林演算法做分類，在這次對個別node預測其label的sub-task中得到了92%的準確率，而後續再搭配task2的Majority+uniFacts方法則得到了93.94的F1 score。

(二)TASK 2 (Given Tokens and Nodes)

A. Bidirectional LSTM + Node information

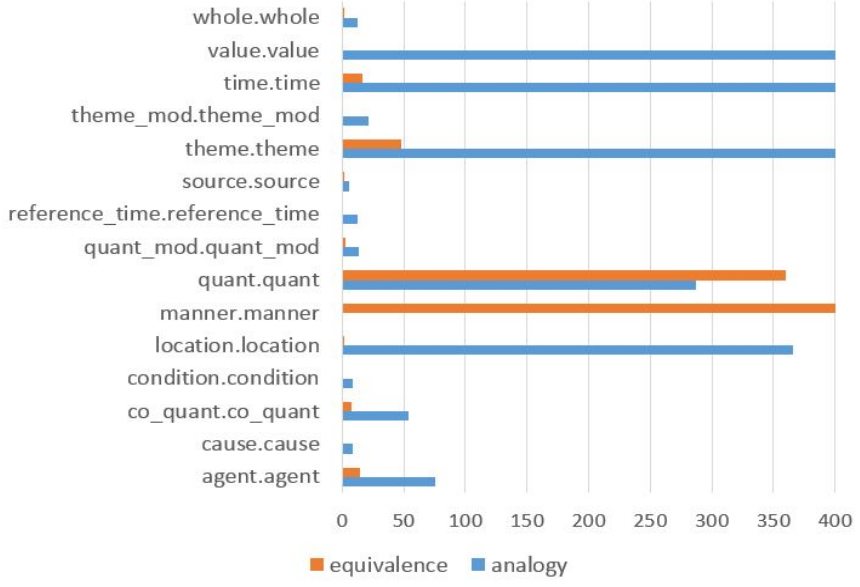
因為這個project的目標是找出edge的種類，因此，有一個很直覺的方法就是架一個 deep learning model，然後把要判斷的兩個node information拿進去做training，就可以得出 edge的種類。

原本的計畫是測試單純使用node 種類當training data時可以得到怎麼樣的performance，沒想到這個方法得出的F1 score 就已經達到 94.26。

不過後來更改了 Neural Network 架構後也很難再有提升了。因此我們研究了一下 training data，發現其實training data很偏向某一個類別，導致我們的模型很難去學習一些其他的資訊。

B. Majority

統計training data edge中的node pair類別會發現它只有兩種情況：一是一個類別為value的node配另一個非value的node，這種情況下的所有edge都對應到fact；另一種情況是兩個node屬於同一個類別，統計數量如下圖。



由上圖可以看出除了在兩個node都是quant的這種狀況下analogy與equivalence的數量差不多之外，其他情況edge都極端偏向analogy或equivalence其中一邊。因此對於每種node pair，我們就直接猜測它在training

data中edge的majority，如此的F1 score可達到94.11%，且fact這個類別的F1 score已達到100%。

C. Majority + UniFacts

B方法最大的缺點是無法準確分辨analogy跟equivalence兩種類別，但由於資料的偏頗，很難藉由單純更改模型提升準確率，因此我們決定更進一步研究資料的特性。

稍微瀏覽過EMNLP那篇原始論文後，發現edge與node之間存在一些天然的限制。其中Unique facts這條規則，也就是"如果兩相異node跟同一個類別為value的node各存在一條類別為fact的edge，則這兩個相異的node必定存在一條類別為equivalence的edge"，恰好可以解決只猜Majority無法完全分辨analogy與equivalence此兩類別的問題。

藉由Majority的統計與Unique facts這個事實，我們訂出以下兩步驟來決定edge的類別：

(1) 若兩node皆為quant，則猜測edge為analogy；否則，直接猜測它在training data中的majority為其類別。

(2) 判斷(1)中的猜測是否符合unique facts，若不符合，則改正之。

最後根據上述的規則，F1 score可以達到100%。

三、實驗結果

		Precision	Recall	F1
TASK 1	**BiLSTM_tok**	88.38	92.57	90.18
	BiLSTM_tok+pos	96.27	98.23	97.21
	SVM_pos	91.51	94.56	92.85
	RF_pos	92.97	95.18	93.94
TASK 2	**BiLSTM**	93.28	95.39	94.26
	Majority	93.19	95.16	94.11
	Majority+UniFacts	100	100	100

四、問題討論

Q1:為什麼Task 1中加了POS tag 可以讓整體score提升?

由於我們是先作出Task 2的，因此我們發現在資料中，其實edge的類別跟對應的node是很有關聯性的。因此在Task 1中，因為不能使用node information，所以只用了tokens放入bidirectional LSTM 訓練，得到的 F1 score 只有90.18%。基於Task 2的經驗，我們想如果能知道更多node的資訊，或許可以得到更好的performance，在只能使用tokens的狀況下，我們決定使用POS tag 讓node 給出多一些資訊。現在的NLTK 或是其他的套件，在判斷POS tag時，準確率都已經高達90%以上，因此，我們相信用tokens去得出的POS tag 可以給出許多有用的資訊。

再加入了POS tag 之後，果然 F1 score達到了 97.21%。我們探討了一下其中的原因，如Task 2中方法B的圖表，我們可以發現真正有使用的node類別並不是很多，而他們相對應的詞性也幾乎都不一樣，因此加入了POS 之後對Task 1 很有幫助。

Q2: Task 2 中為什麼只使用兩個node的類別就可以得到很好的效果？

我們發現edge所分的三類中，fact連接的兩個nodes 其中一定是一個value配上一個 non value。而其餘兩個 equivalence以及 analogy，則是在node pair分類時，就極度偏向其中一個類別。如Task 2 方法B中的統計圖表所示，我們可以發現多數的node pair都極度偏向其中一邊，且多數都偏向analogy。只有在manner,manner時，極度偏向equivalence。而在quant,quant時，兩者的表現卻差不多。因此，再不加入其他features 放入 neural network訓練時，就可以得到很好的performance。

五、結論

在Task 1 以及 Task 2 的實作過程中，我們發現在沒有node information 的時候，如果加入 POS tag ， performance 會提升很多。而在Task 2 中，我們發現資料中有一些規律性，導致我們就算只猜某一種類別作為edge 種類也能達到 90% 以上的效能。也就是說，在這次的project中，其實我們並不太需要去判斷我們所要判斷的edge所連接的兩個nodes跟句子的關係。而是只要能從node中找出對應的node label就可以達到100%的performance。

在Task 2 release的時候，因為擁有node label的資訊，所以我們沒有花太多的心思就可以達到94% 以上的performance。不過因為Task 1中並沒有提供node label的資訊，我們就必須花一些心思在考量要怎麼做才能讓我們猜出的node label能夠成功地讓我們的方法判斷出edge label。

六、參考資料

https://nlp.stanford.edu/pubs/lamm2018analogies.pdf

https://nlpforhackers.io/lstm-pos-tagger-keras/

https://keras.io/layers/wrappers/

https://www.nltk.org/book/ch05.html