Schema Matching

資工三 楊佳峻

A. Introduction

分組資料並 train Bert 模型，使用模型時相似度高的較有可能為相同項，並以此做 matching，不存在不進行配對的策略

B. Related work

常用方法: 根據維基百科

Pre-integration – 對當前模板進行分析，決定整合策略，策略可以是全部或是部分、先後順序、及模式的選擇

Comparison of the Schemas – 跟去兩模板的內容去判斷屬性，可能會發現衝突點，也會有配對成功之處

Conforming the Schemas – 試圖解決衝突點，努力使兩者適應

Merging and Restructuring – 依據上面方法，這時候應該可找到一些方法使兩模板合併，有時還會進行重組

C. Method

- Data Pre-processing

將助教配對好的歸為一組，總共就會有 70 組，這些資料將會丟入 Bert 模型做表層 weight 的 training。

- Matching Algorithm

Step1: 求得相似度

Table1.csv 與 Table2.csv 每個 column 都取前 200 項(更好是用 random 取 200 項)，內容帶入 train 好的 Bert 模型，每個 column 會得到一組向量，最後兩向量做內積即為兩 column 相似的程度。

Step2: 進行配對

將得出來的分數以 column 和 row2 對應做成表格，從最大值由上而下、由左而右，依序查找，且查過的 column 及 row 不重複，必配對出 min(len(column), len(row))個數量，因次不存在不進行配對這種策略。

D. Result

以 pair_1 為例，效果頗低且分數接近

最佳配對	pair_1	pair_2	pair_3	pair_4	pair_5	pair_6	pair_7	pair_8
Table1	建物名稱	地址	總價	格局	樓層	管理費	url	單價
Table2	名稱	價格	每坪單價	格局	總坪數	屋齡	url	車位
successful pair	pair_1	pair_2	pair_3					
Table1	建物名稱	格局	url					
Table2	名稱	格局	url					
failed pair	pair_1	pair_2	pair_3	pair_4	pair_5	pair_6	pair_7	pair_8
Table1	地址	總價	樓層	管理費	單價	類型	建物朝向	建坪
Table2	價格	每坪單價	總坪數	屋齡	車位	電梯	管理費	地址

	建物名稱	地址	總價	格局
名稱	0.021029774	0.021029774	0.021029774	0.021029774
價格	0.021029772	0.021029774	0.021029772	0.021029772
每坪單價	0.021029774	0.021029774	0.021029774	0.021029774
總坪數	0.021029772	0.021029774	0.021029772	0.021029772
屋齡	0.021029772	0.021029774	0.021029772	0.021029772
車位	0.021029772	0.021029772	0.021029772	0.021029772
格局	0.021029774	0.021029774	0.021029774	0.021029774
類型	0.02102977	0.02102977	0.02102977	0.02102977
電梯	0.021029772	0.021029774	0.021029772	0.021029772
管理費	0.021029772	0.021029774	0.021029772	0.021029772
地址	0.021029772	0.021029774	0.021029772	0.021029772
樓層	0.021029772	0.021029774	0.021029772	0.021029772
座向	0.02102977	0.02102977	0.02102977	0.02102977
url	0.021029774	0.021029774	0.021029774	0.021029774

E. Conclusion

- Problem encountered & Solving

Problem 1: 使用 word2vec 但詞語庫不夠及缺乏彈性字詞

本來想進行 word2vec 的 tag grouping，讓相似的 tag 能夠做為同一類進行 train，不過實際分類仍分成 70 類，也就是完全沒有效果，於是便不使用。

Problem 2: Bert 得出來的分數極低，且差異不大，並無法準確判斷

如題，有試圖提高岔具，但想不出有什麼方法，所以效益仍是不大，無法對分數判別起大作用。

- In conclusion

Schema Matching 需要花不少時間在 tagging 與研究版型上，我認為使用 ML 利用關鍵字學

習類型，可以更快速達到這個效果，不過還是需要做人工教調等的後處理。目前最不可能達成的原因可能會出在訓練時間及語言資料庫上，我自己這部分是沒有做的很好。

F. Reference

My colab: https://www.kaggle.com/code/staler2019/wimu-hw2b/settings
My github: https://github.com/Staler2019/NCU-Projects/tree/master/Web-Intelligence-And-Message-Understanding/Schema%20matching