語料模板生成

A. 演算法-Learning Neural Templates for Text Generation[1]

雖然端到端（end-to-end）的方法在資料到文字生成上取得了一定的成果，但其不可解釋性和不可控性一直廣為詬病。為了學習抽取和使用模板，該作者採用適合片段建模的隱半馬爾可夫模型（hidden semi-markov model, HSMM）對文字進行建模，並用神經網路實現其中所有概率項的引數化。在完成模型訓練後，可以利用 Viterbi 演算法推斷出隱狀態序列，並將其獲取為模板，因此可以很方便地對模板進行控制並利用模板引導下一步的文字生成。在 E2E NLG Challenge 資料和 WikiBio 資料上的實驗結果表明，該方法可以取得和端到端神經模型可比的效能，但是更具可解釋性和可控性。

該研究考慮構建可解釋性和可控性的神經生成系統，並提出了具體的第一步：建立一種數據驅動的新生成模型，以學習條件文本生成的離散、模版式結構。核心系統使用一種新穎的神經隱藏半馬爾可夫模型（HSMM）解碼器，它為模板式文本生成提供了一種原則性方法。研究人員進一步描述了通過反向傳播推導以完全數據驅動的方式訓練該模型的有效方法。由神經 HSMM 引起的模版式結構生成明確表示了「系統打算說什麼」（以學習到的模板形式）以及「它想如何說」（以實例化模板的形式）。

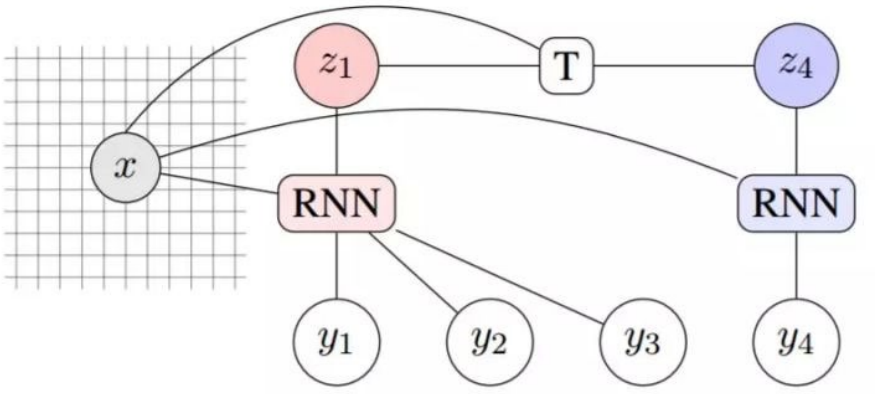

圖 1.HSMM 因子示例圖

如圖 1.所示，我們從紅色狀態 z_1 開始（在 K 個可能性中），然後在發出三個單詞後轉換到藍色狀態 z_4。轉換模型是兩個狀態和神經編碼源 x 的函數。發射模型是生成單詞 1、2 和 3 的紅色 RNN 模型的函數。轉換後，下一個單詞 y_4 由藍色 RNN 生成，重要的是該單詞與前面的單詞無關。

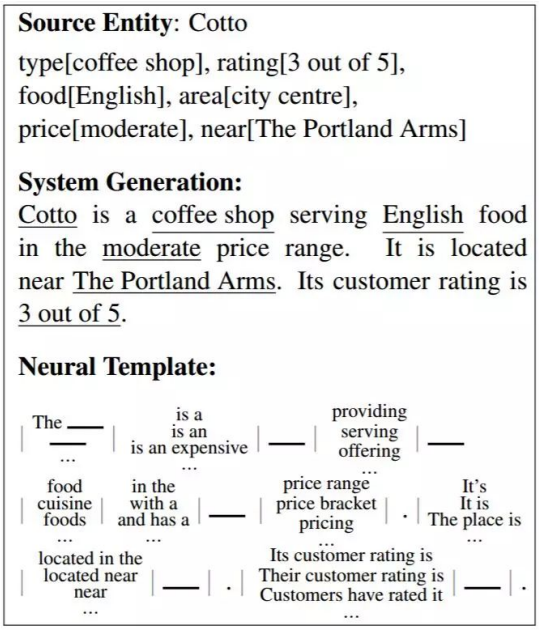

圖 2. 模版式生成示例

如圖 2.所示，為 E2E 生成數據集(Novikova et al., 2017) 中的模版式生成示例。語料知識庫 x（上）包含 6 條記錄，yˆ（中）是系統生成; 記錄顯示為[value] 類型。系統會學習生成的神經模板（下）並將其用於生成 yˆ。每個單元格代表學習段中的一個片段，「空白」表示在生成期間通過複製產生的填空位置。

B. 案例分析

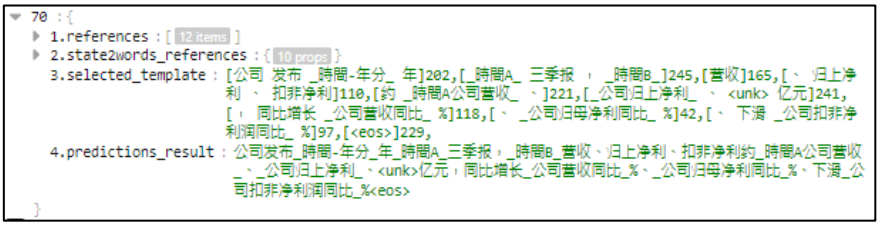

圖 3. 編號 70 神經模板示例

如圖 3.所示，以 ensemble_v2.json 中編號 70 神經模板為例，每個神經模板中包含以下 4 個內容:

1. references 為該神經模板中，模型有參考原數據集的哪些句子。
2. state2words_references 為模型將內些句子排列分成各個 hidden 狀態，每個 hidden 狀態也都會將相似字詞、句構分群在同狀態下。
3. selected_template 表示此樣式模板的狀態序列
4. predictions_result 為從每個 hidden 狀態挑選字詞後，即可組合成一段句子。

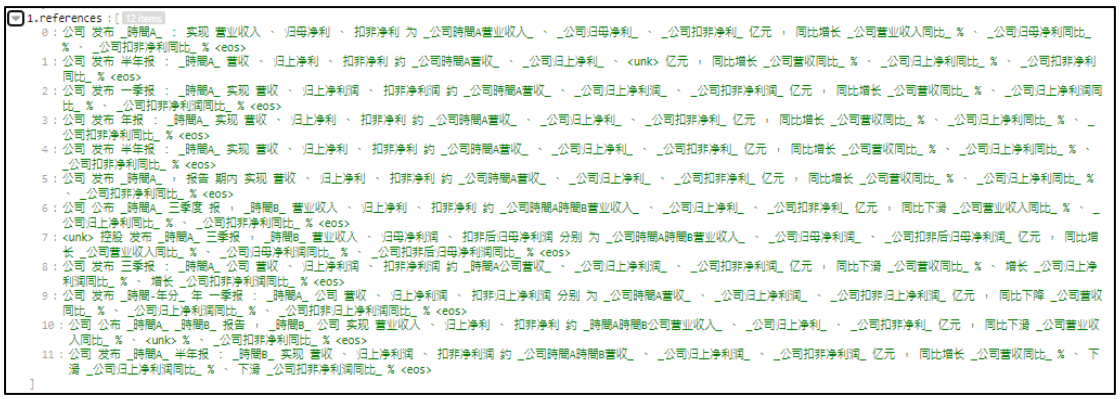

圖 4. 編號 70 神經模板-references

圖 5. 編號 70 神經模板-state2words_references

從圖 4.中可以看出編號 70 神經模板參考了該 11 句句子做後續的模板生成。在圖 5.中可看出模型將每個句中的字詞分在不同 hidden 狀態，如 245 狀態代表季報加上時間等等的內容；而 118 狀態是有關營業收入同比率的升降。從 selected_template 中可得知該神經模板的狀態序列是 202->245->165->110->221->241->118->42->97->229，以此狀態序列做排序並從每個 hidden 狀態中挑選其中一字詞後，即可達到文本生成，也就是 predictions_result 的部分。(perdictions_result 是我自行加上的，挑選的部分先以隨機挑選做顯示)

關於上章節提到可解釋性和可控性的部分，可解釋性是指模板中的填空槽，其代表該句意中因輸入的資訊；可控性是指使用者可根據個人喜好，挑選自身喜歡的敘述方式。

C. 程式說明

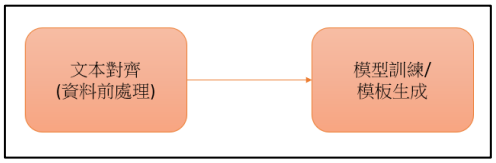

圖 6.程式運作流程

文本對齊

1. 執行 1_preprocess_formal_V2_1.py
   產出./Statistics_word.txt
2. 執行 1_preprocess_formal_V2_2.py
   產出./train_data_prefect.json
3. 執行 2_spilt_to_trainvalid_prefect-data.py
   產出./dataset/train.json、./dataset/valid.json
4. 執行 3_build_chinese_data-V2_prefect-data.py
   產出./ chinese_data/ src_train.txt、./ chinese_data/ src_valid.txt、
   ./ chinese_data/ train.txt、./ chinese_data/ valid.txt、
   ./ chinese_data/ train_tgt_lines.txt、./ chinese_data/ valid_tgt_lines.txt

執行環境

- Ubuntu 18.04
- Python 2.7.18
- Torch 0.3.1

環境安裝

1. conda create --name torch031 python=2.7
2. pip install --default-timeout=6000 https://pypi.tuna.tsinghua.edu.cn/packages/e0/d2/e069611c5e05b4c0d77f841969aa0e6136c43b7a81c0a6cde4910e49a0ce/torch-0.3.1-cp27-cp27mu-manylinux1_x86_64.whl#sha256=db31189d3f4008db7aa988a912cbc358f006151da4f10f1448ab87bcc0477829 (-no-cache-dir)

模型訓練

先將上面產出的./ chinese_data 放置./data/labee2e 內

1. train.cfg-->設定超參數、設定讀存檔(data:、save:、logs:)
2. seg.cfg-->設定讀存檔(load:、seg_file、logs :)
3. python train.py --config train.cfg
4. python segment.py --config seg.cfg

模板生成

1. config.py -->設定讀存檔(load:、tagged_fi:)
2. python write_all_result_test.py
   產出模板 ./user/…….