結合依存句法分析及圖神經網路的文本分類方法

Combining Dependency Parser and GNN models for Text Classification

周冠勳　Kuan-Hsun Chou
國立臺北科技大學資訊工程系
Department of Computer Science and Information Engineering
National Taipei University of Technology
t107598035@ntut.org.tw

吳炎蒼　Yen-Tsang Wu
國立臺北科技大學資訊工程系
Department of Computer Science and Information Engineering
National Taipei University of Technology
buddyswu@gmail.com

王正豪　Jenq-Haur Wang
國立臺北科技大學資訊工程系
Department of Computer Science and Information Engineering
National Taipei University of Technology
jhwang@csie.ntut.edu.tw

摘要

隨著數據量的擴增，人為地對文本進行分類是相當耗成本的，因此自動化的文本分類變得十分重要，如垃圾郵件偵測、新聞分類、情緒分析等。目前自然語言方面的深度學習模型大致上分為兩類：sequential 和 graph based，sequential 模型通常都是使用 RNN 和 CNN，以及近年來在各方面都很突出的 BERT 模型及其變種；近年來有許多的研究，開始將 graph based 的深度模型應用在 NLP 上，利用文字之間的 co-occurrence 關係，從而學習到文字和文本的特徵，以進行分類。本論文首先使用 RNN 計算出文本中的文字特徵，將所有文字當作 node，並用文字之間的修飾關係建 graph，使用 graph model 重新得到文字特徵，並預測文本類別。

在實驗中，我們使用多種資料集，MR、R8、R52 和 Ohsumed 作為驗證。與多種模型，TF-IDF+LR、CNN、LSTM、PV-DBOW、PV-DM、PTE、fastText、SWEM、LEAM 和 Text GCN 進行比較。在 MR 上獲得較好的結果（Accuracy: 79.42%）。

關鍵詞：依存句法分析，圖神經網路，文本分類

ABSTRACT

As the amount of data increases, manually classifying texts is expensive. Therefore, automated text classification has become important, such as spam detection, news classification, and sentiment analysis. Recently, deep learning models in natural language are roughly divided into two categories: sequential and graph based. The sequential models usually use RNN and CNN, as well as the BERT model and its variants; In recent years, researchers started to apply the graph based deep learning model to NLP, using word co-occurrence and TF-IDF weights to build graphs in order to learn the features of words and documents for classification.

In the experiment, we use different datasets, MR, R8, R52 and Ohsumed for verification. Comparing with sequential and graph-based models, the accuracy of our proposed method on MR can achieve 0.79.

Keywords: dependency parser, graph neural network, text classification

一、緒論

隨著網路的發展，資訊傳播的速度與數量快速的增加，若想搜尋某種資料，沒有特別分類方法，使用者便需要仔細地閱讀整筆資料，從而將不必要的資訊過濾掉。如何減少使用者的負擔，同時輔助查詢的準確度，文本分類可以說是一個基本的技術。

文本分類是各種應用軟體的核心，Email 系統裡面使用分類器來區分是不是垃圾郵件;在電商裡面，可以用來區分商品底下的評論是否為垃圾評論，幫助消費者與賣家更好地從評論中得到回饋，或是從文字中，了解文本所述說的主題。

早期的文本分類方法都是使用 bag-of-word 來代表文本的意思，即計算文本中出現哪些字詞，來作為文本的表示，而這種方法並未考慮字詞的上下文，隨著類神經網路模型的興起，開始出現了 RNN 和 CNN 應用在文本分類上，近年來開始有新的研究，將類神經網路應用在 graph 的資料結構上，常見使用在 knowledge graph、社群網路、文章引用、分子結構……。在文本分類上，由於句子或文章本身沒有這種結構，會需要其他方式進行處理。其中一種方法是將文字和文本作為 graph 的 node ，以他們的 co-occurrence

作為邊，以這種方式學習文本特徵，然而隨著文字和文本的增加，graph 也會變大，需要更多的記憶體來載入整個 graph；另一種方法則使用 denpendency parser 對句子做處理，使其變成 tree 的結構，再進行後續的任務。

根據本論文實驗，使用 dependency parser 的結果結合圖神經網路（Graph Neural Network , GNN）在情緒分類文本 MR 上，Accuracy 可以到達 79.42%。

二、相關研究

Mikolov 等人[1]提出 Word2vec 將文字轉成向量的模型，用大量的文本輸入到模型中，來捕獲文字之間的相關性，同時也避免了 bag-of-word 的缺點；Omer 等人[2]將 dependency parser 與 word2vec 結合，學習到句法（Syntax）特徵；Melamud 等人[3]提出 context2vec，使得文字特徵包含了前後文資訊，隨後 Peters 等人[4]基於 bidirectional language model (biLM)的想法學習文字特徵，在 NLP 各種 task 上得到高分；Kim 等人[5]將 CNN 模型應用在文本分類。

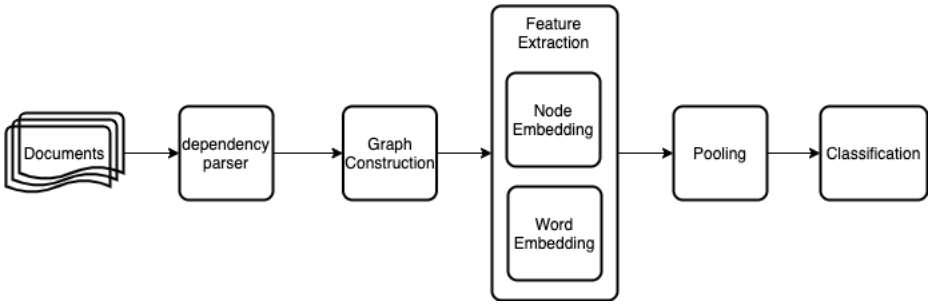
近年來有許多的 GNN 模型應用在 NLP 的任務上。Kipf 等人[6]提出 GCN 模型，GCN 模型為一種多層的架構，藉由 node 與鄰居 node 之間的 aggregate，來更新 node 的特徵，在 knowledge graph 和引文網路上取得了很好的分類效果。Veličković 等人[7]提出 GAT 模型，將 attention 和 multi-head 機制[8]加到 GCN 中，在 aggregate 階段，給予每個 node 不同的重要度，效果超出 GCN。隨後便有人將 GNN 和 dependncy parser 結合，應用在 aspect level sentiment classification[9, 10]和 word embedding[11]。Yao 等人[12]提出 textGCN 則將 GCN 應用於一般的文本分類。

本論文嘗試將 dependency parser 與 GAT 結合，應用在一般的文本分類，並且與 TextGCN 進行比較，期望得到較好的分類效果。

三、研究方法

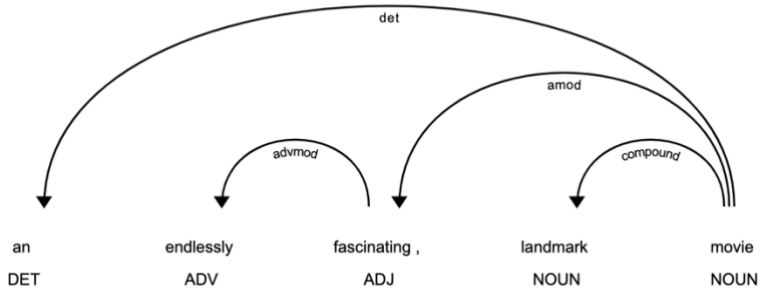
本方法分為五大步驟，如下圖一，首先，使用 dependency parser 取出文本中每個文字的

修飾關係，利用這種關係建 graph 的 edge，而文本中的文字則作為 graph 的 node。在 Feature Extraction 階段，會先使用 RNN 模型得到 word embedding，隨後將此作為初始的 node embedding，與 graph 一同輸入到 GAT 模型中，得到新的 node(word) embedding；在 Pooling 階段，我們對 node embedding 進行 pooling 作為整個 document 的特徵，輸入到分類當中。



圖一、系統架構圖

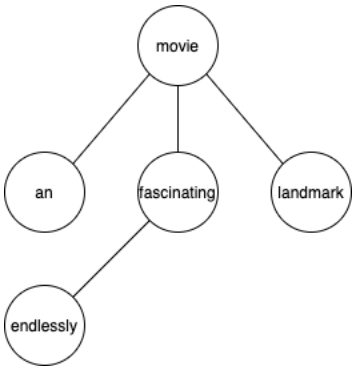
dependency parser 的結果如圖二所示，每個句子都有個 root 文字，不修飾其他文字，如同圖二中的 movie，而其他的文字則會去修飾其他的文字，如 fascinating、a 和 landmark 會去修飾 movie，endlessly 會修飾 fascinating，我們將會在後續利用這種關係建 graph。



圖二、dependency parser 示意圖

建出的 graph 如圖三所示，原本 dependency parser 的修飾關係是有方向的，此處我們將其去掉，改成了無向邊，邊的方向藉由後面的模型學習，若 node 沒有通到另一個 node，則邊的權重視為 0。在文字特徵的結合上，moive 會佔有比較多 an、fascinating 和 landmark 的特徵，而 endlessly 是 movie 的 2-hop neighborhood，兩個文字之間並無直接的關係，得到的特徵會相對的少；對 fascinating 而言，由於無向的關係，可以結合 movie 和

endlessly 的特徵，由此來學習到 node embedding。



圖三、dependency graph 示意圖

整個 node embedding 的更新使用 GAT 來計算，如公式(1)和(2)。公式中 x 為 node embedding，l 為更新的次數，Θ 為線性轉換，α 為 node 之間的權重，$N(i)$為 node i 的鄰居 node，從整個公式來看，所有 node embedding 經過一次線性轉換，然後計算 node i 和他的所有鄰居 node j 之間的權重，計算完權重之後，乘上 node embedding 做加總。以圖三為例，movie 會計算 an、movie、fascinating 和 landmark 之間的權重，若 movie、fascinating 和 landmark 所佔的權重較高，則新的 movie 特徵則包含了 movie、fascinating 和 landmark 的特徵。

權重的計算方法如公式(2)將線性轉換過後的 node 特徵x_i, x_j作串接，隨後乘上一個 weight vector $\vec{a}$，再經過一個 LeakyReLU activation，接下來使用 softmax 求權重。

$$x_i'^{(l+1)} = \alpha_{i,i}\Theta x_i^{(l)} + \sum_{j\in N(i)} \alpha_{i,j}\Theta x_j^{(l)} \tag{1}$$

$$\alpha_{i,j} = \frac{\exp\left(LeakyReLU(\vec{a}^T[\Theta x_i^{(l)} || \Theta x_j^{(l)}]\right)}{\sum_{k\in N_i} \exp\left(LeakyReLU(\vec{a}^T[\Theta x_i^{(l)} || \Theta x_k^{(l)}]\right)} \tag{2}$$

而最初的 node embedding，$x^{(0)}$，我們使用 biRNN[13]的架構來取得各個文字包含上下文的特徵，RNN 的部分則使用 GRU[14]，如公式(3)，$w_0, w_1, \ldots, w_n$為文本中的文字經過 word2vec 轉換過的向量，n為該文本的長度。

$$x^0 = [h_0, h_1, \ldots, h_n] = biRNN(w_0, w_1, \ldots, w_n) \tag{3}$$

在 pooling 階段我們測試兩種做法，1.以 root 文字作為整個句子特徵如公式(4)，如同前面所述，dependency parser 會有一個 root 文字，其會被其他文字修飾或間接的修飾，在

經過 node embedding 的計算過後，root 文字本身就包含了修飾詞的特徵。2.我們將所有的 node embedding 相加取平均，作為整句話的特徵如公式(5)。隨後將句子特徵輸入到一層 NN 做分類，如公式(6)。

$$\overrightarrow{doc} = x_{root}^{l} \tag{4}$$

$$\overrightarrow{doc} = \frac{1}{n}\sum_{i=0}^{n} x_i^l \tag{5}$$

$$class = softmax(\Theta\overrightarrow{doc}) \tag{6}$$

四、實驗結果

本論文參照 TextGCN[12]所使用的資料集，包括：MR、R8、R52 和 Ohsumed。MR 資料集為電影評論，主要是作為情緒分類使用，包括正面情緒 5331 筆，負面情緒 5331 筆。R8 和 R52 為 Reuters-21578 部分資料，分別取其中的 8 種和 52 種類別作為資料集。Ohsumed 為醫學摘要，經過過濾[12]，每篇文章都只描述一種心血管疾病，總共有 23 種類別。

表一顯示本論文提出的模型和其他模型的分類結果比較[12]，我們對每個資料集跑 10 次，來計算平均值和標準差。

表一、各個模型與對應資料集的 accuracy

Model	R8	R52	Ohsumed	MR
TF-IDF+LR	0.9347	0.8695	0.5466	0.7459
CNN-non-static	0.9571±0.0052	0.8759±0.0048	0.5844±0.0106	0.7775±0.0072
Bi-LSTM	0.9631±0.0033	0.9054±0.0091	0.4927±0.0107	0.7768±0.0086
PV-DBOW	0.8587±0.0010	0.7829±0.0011	0.4665±0.0019	0.6109±0.0010
PV-DM	0.5207±0.0004	0.4492±0.0005	0.2950±0.0007	0.5947±0.0038
PTE	0.9669±0.0013	0.9071±0.0014	0.5358±0.0029	0.7023±0.0036
fastText	0.9613±0.0021	0.9281±0.0009	0.5770±0.0049	0.7514±0.0020
SWEM	0.9532±0.0026	0.9294±0.0024	0.6312±0.0055	0.7665±0.0063
LEAM	0.9331±0.0024	0.9184±0.0023	0.5858±0.0079	0.7695±0.0045
Text GCN	**0.9707±0.0010**	**0.9356±0.0018**	**0.6836±0.0056**	0.7674±0.0020
Dep-GAT-root	0.9654±0.0025	0.9263±0.0062	0.6194±0.0118	**0.7942±0.0059**
Dep-GAT-avg	0.9611±0.0075	0.9229±0.0066	0.5630±0.0146	0.7839±0.0029

TF-IDF+LR 為傳統模型，使用 TF-IDF 作為文本特徵，使用 Logistic Regression 分類，LDA+LR 使用文本的主題分佈作為特徵，輸入到 Logistic Regression 分類；sequential deep learning 的模型有 CNN[5]和 bi-LSTM[15]，PV-DBOW 和 PV-DM 為 doc2vec[16]，將文本作為一組向量表示，使用 Logistic Regression 進行分類； PTE[17]、fastText[18]、SWEM[19]和 LEAM[20]使用不同方式學習 word embedding，並對文本所有文字的 word embedding 相加取平均或用其他 pooling 的方式，來算出文本的向量；graph deep learning 的模型為 TextGCN[12]。Dep-GAT-root 和 Dep-GAT-avg 為本論文所提出的方法。

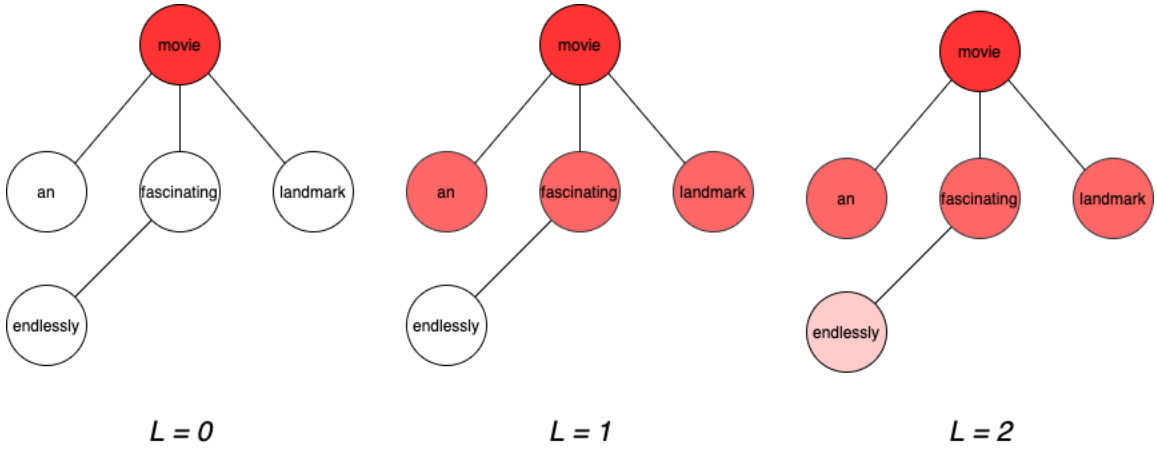
結果顯示在 MR 資料集上取得了較好的效果，accuracy 為 0.7942，BiRNN 可以在較短的文本上得到好的結果，同時 graph 學出來的 embedding 也能很好的輔助分類；然而在 R8、R52 和 Ohsumed 的分數反而不如 TextGCN，其原因可能在於單純 LSTM 和 GRU 在長文本摘要上面效果不是很好，且 dependency parser 的結構對於普通分類文本不太有幫助，在特徵擷取上，dependency parser 找出 root 文字，可將其視為句子的主要文字 (被其他詞修飾)，使得分類效果雖然不是在所有模型中表現最好，但也不會太差。

表二、各個資料集和卷積次數(L)對應的 accuracy

L / Dataset	1	2	3	4	5
MR	0.784±0.006	0.786±0.009	0.790±0.078	**0.794±0.005**	0.788±0.006
R8	**0.965±0.002**	0.965±0.004	0.964±0.003	0.960±0.004	-
R52	**0.926±0.008**	0.926±0.006	0.922±0.012	0.911±0.127	-
Ohsumed	**0.619±0.011**	0.611±0.015	0.591±0.012	0.572±0.014	-

表二為 Dep-GAT-root 的結果，不同的卷積次數代表著每個 node 的更新次數，同時也代表著每個 node 結合了多少 hop neighborhood 的特徵，如下圖四所示，當 L=0 時，movie node 尚未得到其他 node 特徵，當 L=1 時，則得到了 1-hop neighborhood 的特徵，以此類推，而過多的卷積會帶來更多 noise。在 R8、R52 和 Ohsumed，取得 root 和修飾 root

的文字特徵，在分類上就已達極限；在 MR 上或許會需要更多的修飾詞特徵，而這些修飾詞也許跟情緒很有相關，所以需要更多的卷積次數。



圖四、node embedding 結合示意圖

五、結論

本論文嘗試將 dependency parser 與 GAT 的結合應用在文本分類中，在情緒分類的資料集上，dependency parser 能提升分類效果達到 0.794 accuracy，但是在一般文本分類上，這種修飾的關係沒有太大的幫助。在建 graph 過程中我們並無利用 dependency relation，而是單純使用 attention 來結合 node embedding，未來或許可用別種方式將 relation 的特徵加入計算中，並且將其用在情緒分類的相關任務中。

參考文獻

[1] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781,* 2013.

[2] O. Levy and Y. Goldberg, "Dependency-based word embeddings," in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2014, pp. 302-308.

[3] O. Melamud, J. Goldberger, and I. Dagan, "context2vec: Learning generic context embedding with bidirectional lstm," in *Proceedings of the 20th SIGNLL conference on computational natural language learning*, 2016, pp. 51-61.

[4] M. E. Peters *et al.*, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365,* 2018.

[5] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882,* 2014.

[6] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional

networks," *arXiv preprint arXiv:1609.02907,* 2016.

[7] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903,* 2017.

[8] A. Vaswani *et al.*, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998-6008.

[9] C. Zhang, Q. Li, and D. Song, "Aspect-based sentiment classification with aspect-specific graph convolutional networks," *arXiv preprint arXiv:1909.03477,* 2019.

[10] B. Huang and K. M. Carley, "Syntax-aware aspect level sentiment classification with graph attention networks," *arXiv preprint arXiv:1909.02606,* 2019.

[11] S. Vashishth, M. Bhandari, P. Yadav, P. Rai, C. Bhattacharyya, and P. Talukdar, "Incorporating syntactic and semantic information in word embeddings using graph convolutional networks," *arXiv preprint arXiv:1809.04283,* 2018.

[12] L. Yao, C. Mao, and Y. Luo, "Graph convolutional networks for text classification," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 7370-7377.

[13] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing,* vol. 45, no. 11, pp. 2673-2681, 1997.

[14] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078,* 2014.

[15] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning," *arXiv preprint arXiv:1605.05101,* 2016.

[16] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, 2014, pp. 1188-1196.

[17] J. Tang, M. Qu, and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2015, pp. 1165-1174.

[18] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," *arXiv preprint arXiv:1607.01759,* 2016.

[19] D. Shen *et al.*, "Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms," *arXiv preprint arXiv:1805.09843,* 2018.

[20] G. Wang *et al.*, "Joint embedding of words and labels for text classification," *arXiv preprint arXiv:1805.04174,* 2018.