The 2018 Conference on Computational Linguistics and Speech Processing
ROCLING 2018, pp. 61-75

繁體中文依存句法剖析器
Traditional Chinese Dependency Parser

李彥璇 Yen-Hsuan Lee

國立交通大學電信工程研究所
Department of Communication Engineering
National Chiao Tung University
yhl0305.cm05g@g2.nctu.edu.tw

王逸如 Yih-Ru Wang

國立交通大學電機工程系
Department of Electronic Engineering
National Chiao Tung University
yrwang@cc.nctu.edu.tw

摘要

近年來依存句法分析樹一直是 CoNLL 比賽的重要項目，其重要性可見一班，但其中參賽的隊伍並非以繁體中文為母語，若遇到繁體中文的語料通常會轉為簡體中文再做更進一步處理。因此，本研究以建立繁體中文依存句法分析樹為主要目的，透過目前相當熱門的深度學習模型，多層感知器及遞迴神經網路，與現有的斷詞器結合，得到繁體中文語句之句法分析樹。這樣的模型對於句法分析樹有相當好的成效，並比過去類似 Task 相比有高達 70%的成長。

Abstract

Recently, dependency parser has been paid highly attention in CoNLL conference, in this case we can find how important it is. However, there is no native Taiwanese attend to this competition. Therefore, we aim to build a traditional Chinese dependency parser in this paper. Through different famous neural network structure , such as multilayer perceptron and recurrent neural network along with traditional Chinese segmentation, we are able to construct an efficient parser that could transfer a raw traditional Chinese sentences to a dependency tree. We have much better result than similar task that has been proposed at 2012.

關鍵詞：依存句法分析器，自然語言處理，多層感知器，遞迴神經網路

Keywords: dependency parser, natural language processing, MultiLayer Perceptron , Recurrent Neural Network

一、 緒論

在自然語言處理中，語意瞭解是很重要的一部分，而為了做語意瞭解，就必須做句法分析。早期學者們都使用樹狀結構句法分析(tree parsing)，如中研院的 CKIP Chinese Parser[1]。樹狀結構又稱短語結構(phrase structure)，主要是將句子拆解成不同的片語。近年來，學者們大多使用依存句法分析，依存句法分析中的資訊可由短語結構依照規則法轉換而來，儘管這兩種方式夾帶資訊類似，但在句法分析中可以直接表示詞與詞之間的相依關係。

由於近年來機器學習的技術一日千里，過去曾經做過的自然語言處理方式，又被提出來重新檢視，依存句法分析也不例外，並獲得相當好的成果。首先是 2013 年 Mikolov 提出新的詞向量表示法(Word Embedding)[2]，使句中的各詞可以存在一定的關聯，並能更好的利用每個輸入單詞，接著是不同機器學習架構，例如 MLP (Multilayer Perceptron) 和 LSTM (Long Short Term Memory)，將複雜的系統簡單化，像是 dependency parser 原本透過機率模型，需要設計不同規則來尋找詞與詞之間的可能關聯，在尋找的同時會耗費較多的苦工，然而使用深度學習技術，將選定好的特徵輸入，經過隱藏層後面的激活函數，可以直接學習出整個句子的語法，將規則比對與計算條件機率的部分簡化成學習神經網路中的權重。

依存句法分析深度學習的模型是使用監督式學習，因此正確的人工依存句法分析標示語料，是建立依存句法分析器不可或缺的訓練資料。這幾年，學術界組織了數個組織且訂定了依存句法分析的一套標準，並建立各個語言的依存句法標示語料庫。其中，由 Joakim Nivre 主持的 Universal Dependencies(UD)[3]是一個多人參與並建立資料較完整的組織，包含了全世界絕大數語言，2.1 版中新增了繁體與簡體中文。我們研究室在過去已發展有繁體中文的 word segmenter [4], 、POS tagger 再加上依存句法的訓練工具及訓練語料應該可以達到很好的訓練效果。事實上，還有多問題需要解決，在 UD 繁體中文語料庫中的訓練語料僅有大約十萬詞，且其中使用的詞性與斷詞並沒有依照中研院的標準來建立。對於我們的現況來說 UD 語料庫並不適合建立一套 End-to-End 的

系統。然而中研院也推出了自己的中文結構樹資料庫[1]，並且是以 dependency tree 的方式呈現，斷詞與 POS 標示都與中研院詞庫小組一致，且中研院的詞性標記種類多達 47 類，其中 P（介詞）更可細分成 66 類，而這些能提供句法分析更多資訊，使分析準確大為提升，以下圖一和圖二「與」為例，若沒有標示 POS 在句法判斷上，就無法知道他代表的是伴隨還是對等的意思，如圖一「與」POS 為 P35 代表的是伴隨，前面可以是人或其他事物；而圖二的「與」為 Caa，代表前後必須為對等的名詞片語或是動詞，然而在斷詞上，並無法分出這兩者的不同，故 POS 在訓練依存句法分析器中是個不可缺少的特徵。相較其他語言來說，中文的語法架構較為複雜，UD 語料庫中為了統一各個語言的 dependency type 只使用了 42 個，相較於中研院的 68 類，UD 可能無法涵蓋中文語意更深入的層面。

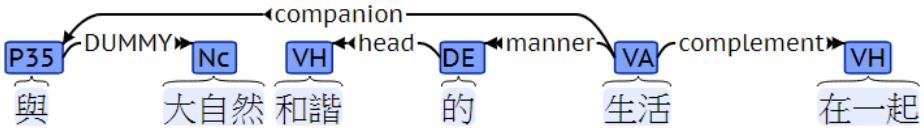

圖 一 、與的 POS 為 P35

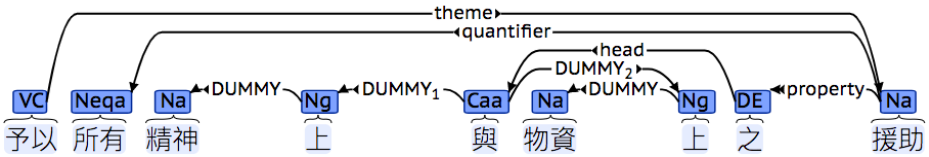

圖 二 、與的 POS 為 Caa

綜觀上面幾種情形，再加上我們實驗室現有的斷詞與 POS 標示都與中研院標準相同，以致於我們能更著重在訓練方法上。本研究提出了端到端的依存句法分析器，來提供未來自然語言處理發展上一項有利的工具。

後面小節中會提到依存句法分析器的做法，使用了 Joakim Nivre 提出的 Transition-based algorithm[5]，並結合 Stanford 提出的高效率剖析器[6]中的 Multilayer perceptron 架構，除了 Feedforward 外，也使用 Recursive Neural Networks 中的 Gated Recurrent Unit 來建立我們的依存句法分析器[7]，來彌補 MLP 中缺失的順序問題。

二、 方法

[1] http://www.aclclp.org.tw/use_conll_c.php

（一） 系統介紹

本研究中使用了交通大學語音處理實驗室的斷詞器與 POS 標示，斷詞器是繁體中文自然語言處理上很重要的一個元件，並不像其他語系，中文句子中沒有明確的詞語界線，需要有個明確的標準才能讓後續的使用更為靈活，而 POS 標示也十分依賴斷詞器的結果。透過高效能的斷詞器與 POS 標示，我們在多次 share task 中獲得亮眼的成績，因此我們有十分堅固的基礎來建立依存句法分析器。

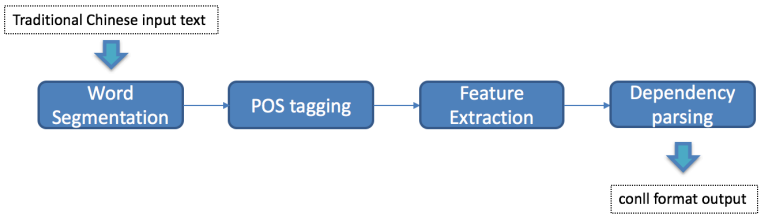

圖 三、End-to-End 架構中所包含的元件

在本篇論文，我們著重於圖三中的 Dependency Parsing，即為依存句法分析，在進入模型建立的方法前，先簡單介紹它的表示方式及代表意義：

在 dependency parser 輸出中(如圖四之範例)，會以弧 (arc) 來表示詞與詞間的相依性，圖中 arc 會從中心語(Head)指向修飾語，如：”書架” 一詞會指向”一座座”。並且每一個 arc 會標示 dependency relation (如：quantifier(數量詞修飾語))。而”一座座書架”就組成一個詞組或片語。而再下一層，”一座座書架” 詞組修飾 ”上” 。如此，一層一層下去，最後，句中未修飾他詞的中心語(Head)則稱為 root (如句中”放”一詞)。

如此，dependency parser 以 head 和 dependent 來表示兩個詞之間的關聯，以 head 代表一個 phrase 的中心詞（名詞片語中的名詞，動詞片語中的動詞），剩下的詞可能與 head 有 direct、indirect 或 dependent 的關係。我們可以透過這樣的關聯，將 head-dependent 的配對做更進一步的分類，稱作 dependency relation 或 grammatical function。

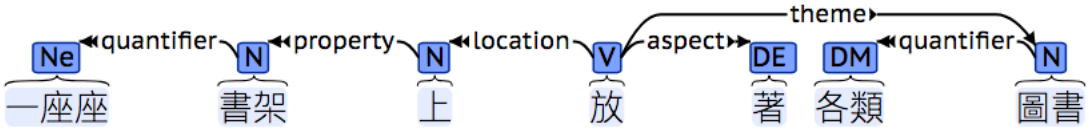

圖 四 、依存句法範例

而建構依存句法分析器尚需滿足三項限制，(1) 外加的 root，並沒有任何 arc 指向它，(2) root 以外的 node(詞或標點符號)，都會被一個 arc 指向，(3) root 到所有其他 vertex 只有一條唯一路徑。上述的 vertex 指的是在短語結構中（如圖五），詞組的中心語。短語結構樹也可稱為 constituency tree，並無上下文順序關係，僅以片語為分析重心。

以上面的邏輯，我們使用 Transition Based Parsing 的方式，來一一找出詞與詞的關聯，這個方法是來自於 shift-reduced parsing，使用了 stack 和即將被剖析的 word list。在 transition-based parsing 中有個重要的特徵組態（configuration）概念，其中有三個重要成分，stack、buffer 和 dependency relations 的集合。最初的特徵組態中，stack 只包含 ROOT，buffer 則是句子中的所有詞，而 dependency relations 一開始是空的。終止狀態中 stack 中僅剩下 ROOT 和 buffer 必須是空的，dependency relations 的集合代表最終的剖析結果。為了產生新的特徵組態，我們會使用三個 operator：

LeftArc : 在最上層的詞和他下一層的詞中加入新的 head-dependent relations，並 pop 第二層的詞。

RightArc : 在第二層的詞和最上層的詞中加入新的 head-dependent relations，並 pop 最上層的詞。

Shift :將 buffer 中最前面的詞 push 進 stack 後，將其移除。

在圖六中，我們可以將 Check dependency 分成不只 Left-arc 跟 Right-arc，加上 dependency relation，變成一個多類別的分類器，在判斷左右的同時也能判斷類別就能得到圖七帶有標籤的結果。

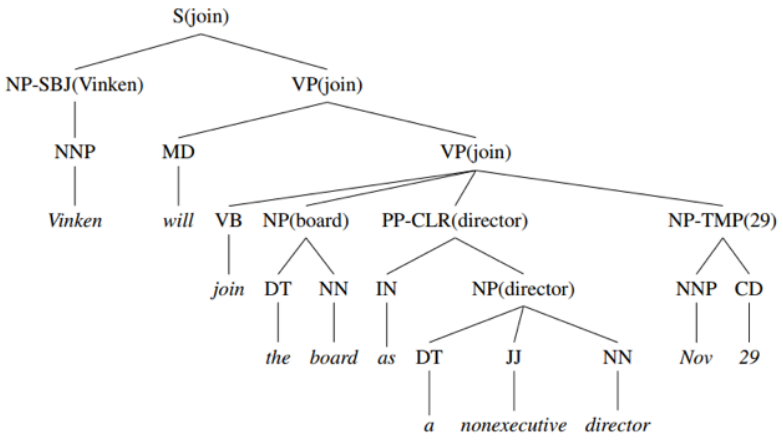

圖 五、Constituency Tree

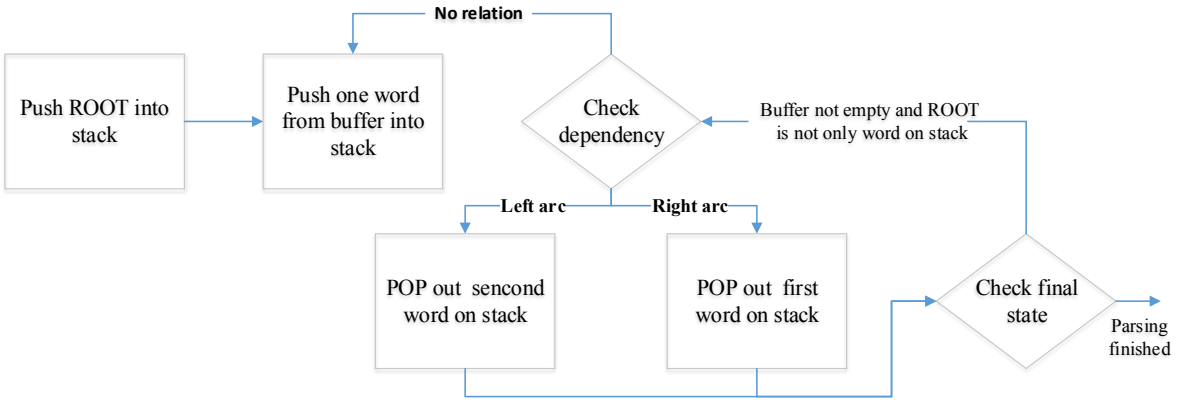

圖 六、transition-based 流程圖

將上述的流程套用在一個實際例子，由於句子的長度與 transition 為線性關係，我們選擇較短的句子「由四樓走上五樓」，斷詞完為四個詞，每個詞皆會被 pop 出 Stack 和 push 進 Stack 一次，因此只需要 4*2 步[2]就可以剖析完一句話。從表一我們可以知道，初始狀態從 ROOT 開始也到 ROOT 結束，每個詞也都會輪流進入 stack 檢視，直到所有關聯被建立。當執行完表一的所有步驟，將關聯圖像化即可得到如圖七

表 一 、Transition based parsing 的過程

Step	*Stack*	*Word List*	*Action*	*Relation*
0	[root]	[由,四樓,走上,五樓]	Shift	
1	[root,由]	[四樓,走上,五樓]	Shift	
2	[root,由,四樓]	[走上,五樓]	RightArc	(由→四樓)
3	[root,由]	[走上,五樓]	Shift	
4	[root,由,走上]	[五樓]	LeftArc	(由←走上)
5	[root,走上]	[五樓]	Shift	
6	[root,走上,五樓]	[]	RightArc	(走上→五樓)
7	[root,走上]	[]	RightArc	(root→走上)
8	[root]	[]	Done	

[2] 視情況也可以為 n*2+1，若一開始將 ROOT 放在 word list，而非 stack，則需要將 ROOT push 至 stack

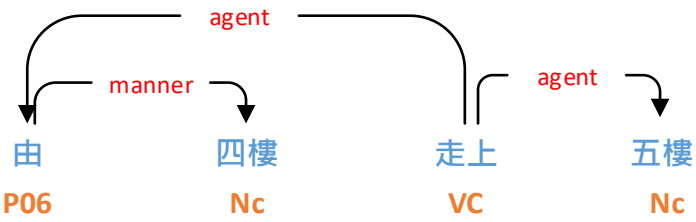

圖 五、表一的依存句法分析樹

（二） 模型設計

自從 Mikolov 提出新的詞表達方式後，各式各樣的神經網路訓練模型，又成為自然語言處理界的寵兒，依存句法分析器當然也不例外，透過預先訓練好的詞向量，神經網路可以更有效地找出詞和詞之間的模式（pattern）。我們透過現在最常見的兩種模型——遞迴神經網路與多層感知器來實現依存句法分析器，由於 transition-based 是一種貪婪演算法，分類出當前的動作後，並無法使用回溯法更改前面判斷的動作。因此在這兩種深度模型中加入了定向搜索（Beam Search），產生多個句法分析樹，最後再以分數高的作為最終答案。

1. 多層感知器 Multilayer Perceptron

2014 年 Chen & Manning，提出了一種快速的 transition-based parsing 方法，保留了 stack 和 buffer 當作 input layer 的 feature，除了 word 以外也包含了 POS tags 和 arc label。在這當中如何選取 feature 是很重要的一個步驟，由於句子的長短並不一致，在 C&M 的論文中，選擇了 stack 和 buffer 最上層的 3 個 element、stack 上兩層個別的左一和一右修飾詞，以及最左關係詞下的最左修飾詞，最右修飾詞的最右修飾詞。總共會有 18 個 features，這些 words 的 POS tags 也有 18 個，還有 arc label 有 12 個（不包含 stack 和 buffer 中的六個詞），以上這些 features 皆拿來作為 dependency parsing 的輸入。以上介紹很可會過於抽象，因此以圖八為例，剖析完成的依存句法樹，transition-based 中每一個動作都是透過特徵組態(如表一的 step)來預測下一個動作。但輸入的特徵就像上面說的不僅僅只是拿當前狀態的 stack 和 buffer，還包括了已知的修飾關係組合。如圖九，「他」和「了」都在前面步驟被移出 stack，因為這兩者都修飾了「喝」這個詞，然而這項關係在下一步驟並沒有因此被忽略，進而成為我們的輸入特徵。

在圖九的架構中，我們將 features 分成三部分，詞、詞性和 Label，在輸入詞向量的時候，可能會出現 OOV (Out of Vocabulary)，因此將 POS 標記與詞向量合併做為新的詞向量來作訓練，而 POS 獨立出來訓練的目的，則是為了更有效的學習它在依存句法架構的重要性，在後面的一些實驗中，我們也發現 POS 在此模型中，有著舉足輕重的地位。

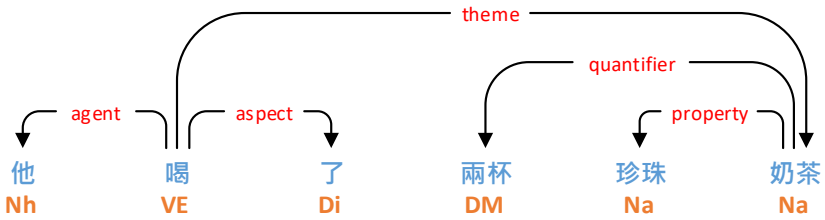

圖 六 、剖析後之句法樹

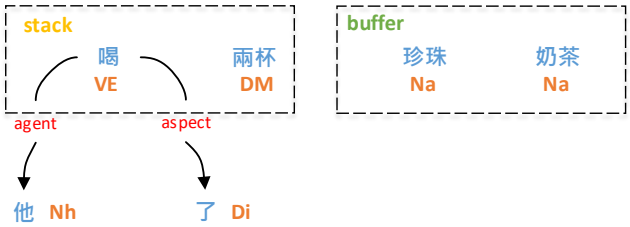

圖 七、預測下一個動作之特徵組態

2. 遞迴神經網路 Recurrent Neural Network

接著，在如此依賴字詞順序的架構上，2016 年，Marianas Labs 提出了一個基於 transition-based 的遞迴神經網路，又稱 Stack Long Short Term Memory 結合了 Stack、buffer 與 Action，擁有各自的 RNN 架構，Stack 的 RNN，夾帶了部分 parsing tree 的資訊，能更全面的預測當前的 action， 在 S-LSTM 中，多了 Action 這部份的資訊，將每個狀態預測出的 Action 變成時間序列，加以訓練。在部分 parsing tree 中，延用了 Stanford Parser 的 feature，將部分剖析樹合併到 Stack 的最上面兩層，由於剖析樹不只有詞其中還有 action 的 label，如此性質不同的情況下，會透過 composition 的方式各別訓練 label 和詞，再以線性轉換的方式，得到新的 stack 表達式。

在我們研究中，將上述的 LSTM 取代成 RNN 的另一種變種 GRU(Gated Recurrent Unit)將 LSTM 中的 forget gate 與 input gate 合成一個 update gate，另一個 gate 則與 cell state 合併成為 reset gate，依然能保存長短距離的資訊，在架構上 LSTM 與 GRU 的差異並不明顯，但在本研究中，會使用比英文更大維度的資料，希望降低權重數量，提

升收斂速度。且在資料量不夠的狀態下，LSTM 學習效果並無法展現出來。在後面章節會直接稱我們的模型為 stack-GRU。

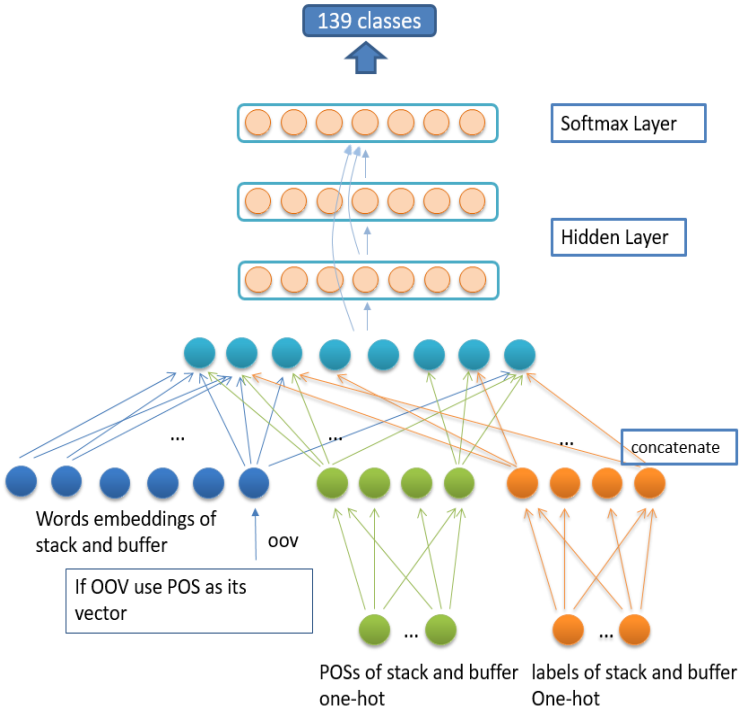

圖 八、MLP 架構

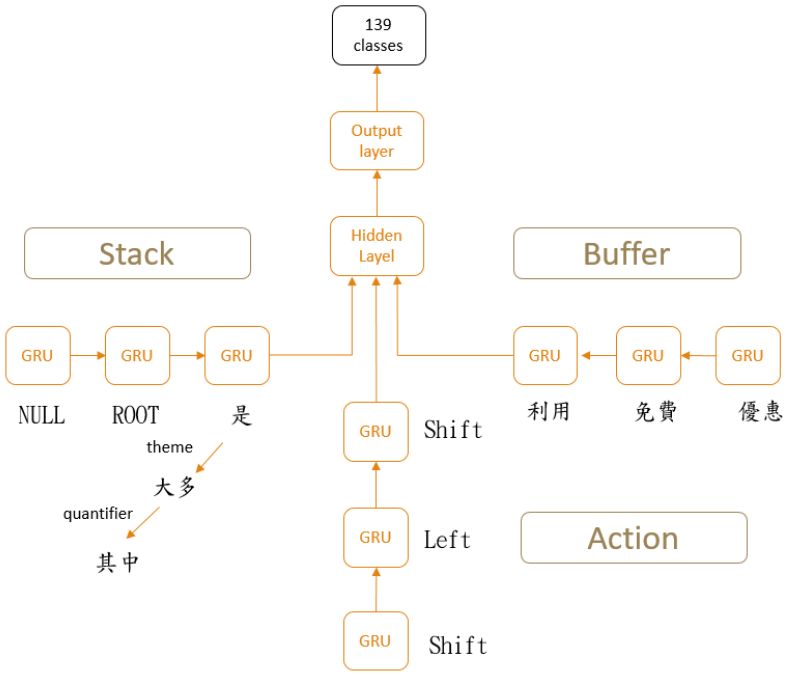

圖 九、Stack LSTM 架構

3. 定向搜索 Beam search

Transition-based 在前面提到是個計算上非常有效率的演算法，只要依據 stack 與 buffer 的狀態提供選擇，並一步一步剖析完整句話，然而這樣的做法，有個極大的缺點，一但做了決定，便無法回頭，即使後來發生難以判斷的狀態也無能為力。因此，我們希望能有系統地多看其他選項，並從中找出最適合的句法。而定向搜索則使用了橫向優先搜尋的策略以階層式的方法篩選使每一層只需要處理固定寬度（beam width）的資料。套用定向搜索至 transition-based 剖析上，我們需要稍微修改我們原本的演算法，原本僅需預測當前特徵組態的最適動作，現在則挑出機率前N高的動作，每個動作都會產生新的特徵組態，這些組態我們會列入 agenda（可能動作的集合），只要沒超過我們設定的寬度（beam width），就可以持續加入新的組態。當 agenda 達到預設的大小時，我們僅會加入比 agenda 中最差值機率高的組態。為了找到最好的句法樹，以迴圈的方式進行全局搜索直到終止狀態。其中機率為原本深度學習模型下，將最終輸出改成選擇的 N 個機率，而不使用單一的分類結果，下圖為本研究使用的定向搜索示意圖，每個節點都代表一個特徵組態，每個組態會輸入模型內並回傳前三高的機率，當達到終止狀態時，則計算機率總和最高作為最終答案。這樣一來我們便能較全面篩選每一個可能的句法。

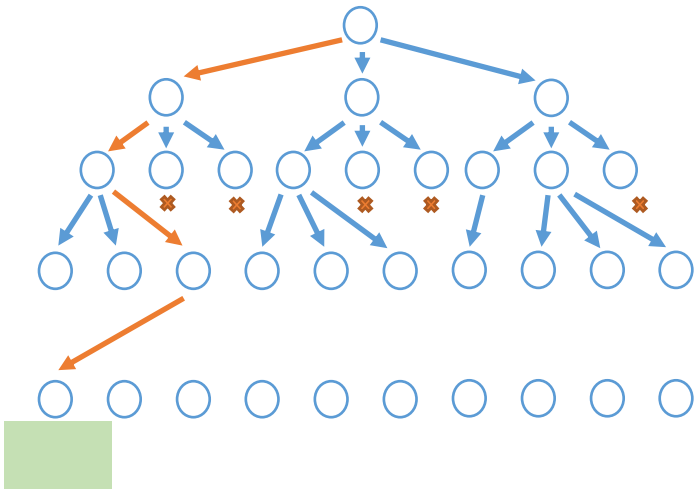

圖 十、Beam-search

三、 結果與討論

（一） 實驗配置

在評估標準上，依存句法較常使用 UAS(Unlabeled Attachment Score)及 LAS (Labeled Attachment Score)，來確認 head-dependent 是否正確配對，Labeled 則更進一步地表示 head-dependent 的關聯。如圖十三與十四，只看箭頭不看標籤可以看出 7 條關聯中只有 4 條正確，而帶有標籤的情況則只有 3 條正確，因此這個範例中，UAS 為 4/7 而 LAS 為 3/7，儘管我們可以看到修飾「這場」的關聯是「quantifier」但修飾它的詞不正確，故不能計算 LAS。

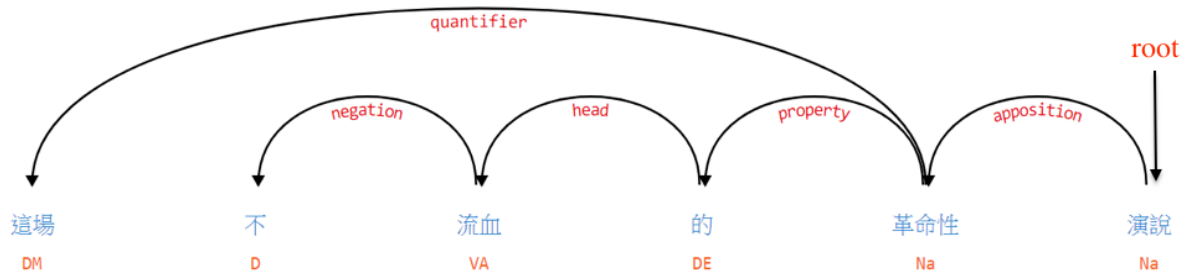

圖 十一、系統答案

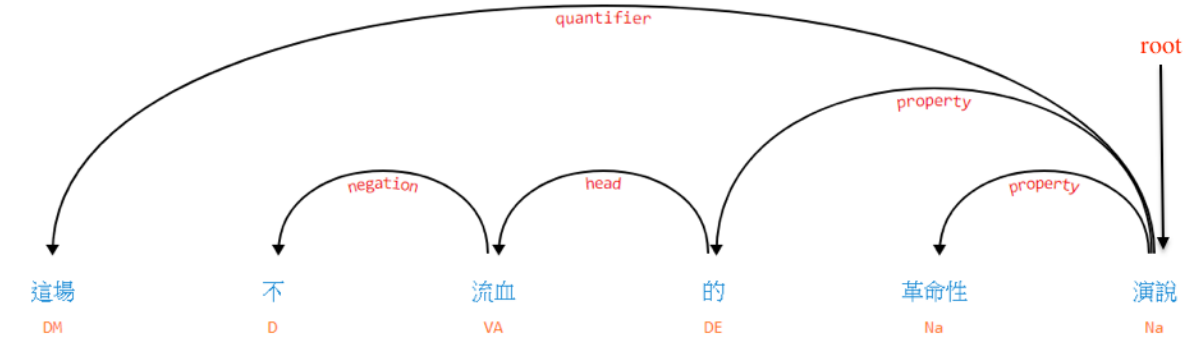

圖 十二、正確答案

除了基本的 UAS 與 LAS 外，中文多了一項評估標準以 f1-score 作為測試系統效能的依據。f1-socre 的公式為

$$\mathrm{f1} = \frac{2 * precision * recall}{precision + recall}$$

而這裡的 precision（準確率）是以標準答案當作分母來計算，如下圖正確指向的箭頭為 4 個，而標準答案共有 5 個，因此準確率為 4/5，然而 recall（召回率)是以我們的系統輸出為分母計算，下圖系統輸出共有 8 個箭頭，召回率為 4/8，最後計算 f1-score 為 0.615。

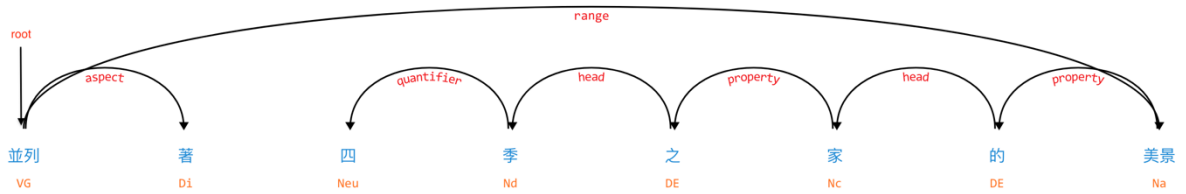

圖 十三、系統答案

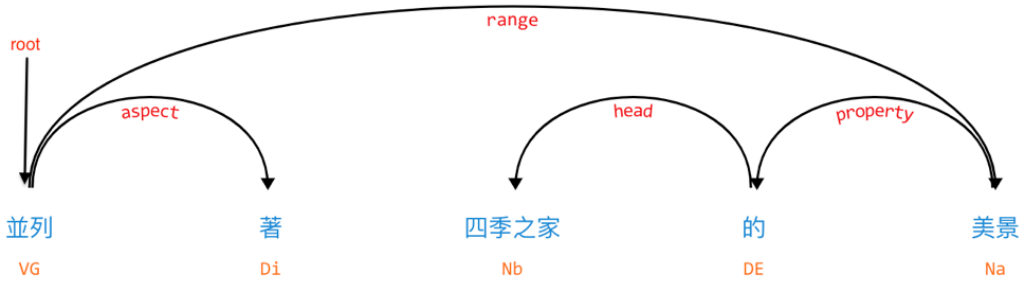

圖 十四、正確答案

在本研究討論了四種配置，Stack-GRU 與 MLP，以及各別加上 beam search 後的結果。其中包含中研院提供的測試語料經過模型的準確度，與本實驗使用的斷詞器斷詞後的結果，而我們使用的斷詞器與中研院斷詞有些許出入，因此使用 f1-score 作為分析結果。

（二） 實驗結果

目前繁體中文並沒有任何已知的基準提供我們做參考，因此以 2012 年 bake-offs 的 Traditional Chinese Parsing task[7]的評估結果來當我們的基準。此繁體中文剖析使用的是短語的樹狀結構，如上面所說，短語句法樹可以轉換成依存句法結構，因此使用此基準我們認為是可信的。如表二之一，2012 年所使用的資料集，與本次實驗訓練與測試語料比率均為 1%左右。表二之一的測試語料佔 1.5%，表二之二則佔 1.2%。

表 二之一、2012 Traditional Chinese Parsing task 資料集

Data Set	#Word	#Sent	Avg. Length
Training	391,505	65,243	6
Test	8,565	1,000	8.57
Validation	341	37	9.2

表 三之二、中研院提供的資料集

Data Set	#Word	#Sent	Avg. Length
Training	337,174	56,957	5.92
Test	5,160	690	7.47

我們以當時各隊比賽的結果當作比較，由於斷詞可能與測試語料不一致，在此也看 F1 做為參考依據，其中分為微平均與宏平均，而宏平均與我們實驗的算法一致。表三中是以帶語意角色的短語結構剖析後的結果，相當於實驗中 LAS 的評估方式，表中最好的結果為 0.4287，故以此為基準。

表 四、帶語意角色的短語結構分析結果

Submitted Runs	Micro-averaging			Macro-averaging		
	Precision	Recall	F1	Precision	Recall	F1
NCU-Run1	0.3755	0.3429	0.3585	0.3506	0.3538	0.3522
NEU-Run1	0.4358	0.4328	0.4343	0.4192	0.416	0.4176
NEU-Run2	0.4409	0.4379	0.4394	0.4239	0.4209	0.4224
CRF (Baseline)	0.4382	0.4216	0.4297	0.4347	0.4229	0.4287

在本實驗中，多層感知器使用了 250 維的詞向量，詞性為 121 類的獨熱編碼（one-hot encoding）和依存關係 69 類的獨熱編碼作為輸入，各自乘上權重矩陣後再合併，並在後面使用兩層 RELU 隱藏層，如圖十，詞向量、詞性和依存關係的隱藏層神經元為 200 個，合併後則為 600 個，通過每一層隱藏層都逐漸減少 200 維，最後一層 Softmax 則輸出 139 類（transition-based action）。其中 Batch size 為 100，Epoch 為 5 就能收斂，因此在訓練上，MLP 十分有效率。

遞迴神經網路在架構上就與多層感知器有相當大的差別，所以在參數的設置也會有所不同，GRU 分成 stack 跟 buffer 兩部份，時間步數（time step）設為 3，由於我們的句長並不長，步數設少一點可以使整體模型執行更快。不同於 MLP，GRU 不是暴力地將特徵展開再餵進模型，stack 中，如我們在方法中所介紹的，會帶有部分剖析樹的資訊，分別有 stack 最上面兩層修飾的詞與依存關係，為了讓詞和依存關係可以有更好的學習效果，便各自加入隱藏層學習特定的權重矩陣，如此一來重要的依存關係特

徵可以被擷取出來。各自隱藏層的輸出會與 stack 前兩層的詞向量合併，再經過一層隱藏層得到該 stack 詞的新詞向量。這樣進入 GRU 的每個時間步數都為同樣維度。在此模型中，GRU 設置兩層隱藏曾，輸出為 200 維，stack 和 buffer 的 GRU 設置完全一樣，最後再將 stack 和 buffer 的輸出合併一起輸入 softmax 層，並得到與 MLP 一樣的輸出形式。

接著探討加入定向搜索的情況，在這裡為了減少運算量我們設置的定向寬度（Beam width）為 10，而每次輸出則取前 4 高，來分析可能的句法樹，因為定向寬度為 10 且取 4 個可能的動作，我們可以看 40 個不同動作造成的不同結果並得到對應的機率，再將 30 個超過定向寬度的節點刪除，以此循環直到終止狀態。

表 五 、不同配置的表現

	UAS	**LAS**	**UAS f1-score**	**LAS f1-score**
Stack-GRU	88.5%	83.1%	80.66%	74.17%
+beam	89.8%	84.5%	81.05%	74.54%
MLP	87.5%	82.7%	79.28%	73.24%
+beam	88.3%	83.5%	80.45%	74.71%

表四為個別看 MLP 與 GRU 模型的結果，可以看出在多數情況 GRU 表現比 MLP 好，還沒加入定向搜索時，LAS 相差 0.4，但在加入定向搜索時則差距 1，可見遞迴神經網路較能找出過去的詞語當前詞的關聯。然而使用本研究中的斷詞器的結果，則是 MLP 加入定向搜索表現最好，原因是當斷詞錯誤或是詞性標示不一致，GRU 對於過去的詞性較為敏感，以至於過去的錯誤會隨著遞迴神經元中傳播至當前的預測結果。

四、 結論

在本研究中實現了完整的句法分析系統，結合交通大學語音實驗室的斷詞器以及詞性標示，跟過去的 share task 比從 0.42 至 0.74 有將近 70%的進步幅度，顯示出在深度學習模型對於自然語言處理的功勞不可小覷。目前在此領域中尚未有人完整的整理出一套輸入句子（未斷詞）即可得到依存句法分析樹的系統，並且已建構線上系統提供學術使用。由於繁體中文的自然處理領域，才剛起步，對於這樣的系統上有十分多的改善空間，如使用 graph-based 方法，可獲得更好的長距離修飾關係，希望以此論文作為開頭，讓更多學者加入研究。

五、 致謝

感謝科技部計畫編號 MOST 107-2221-E-009-MY2 的支持與協助。

六、 參考文獻

[1] You, Jia-Ming, Keh-Jiann Chen, 2004, "Automatic Semantic Role Assignment for a Tree Structure", Proceedings of SIGHAN workshop.

[2] Mikolov, Tomas, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. "Efficient Estimation of Word Representations in Vector Space ", In Proceedings of Workshop at ICLR 2013

[3] Joakim Nivre, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty, Daniel Zeman, 2016, "Universal Dependencies v1: A Multilingual Treebank Collection". In Proceedings of LREC.

[4] Yih-Ru Wang, 2003, "An Improvement on Chinese Parser"

[5] Joakim Nivre , 2003, "An Efficient Algorithm for Projective Dependency Parsing" In Proceedings of the 8th International Workshop on Parsing Technologies

[6] Danqi Chen, Christopher D. Manning , 2014, "A Fast and Accurate Dependency Parser using Neural Networks" in EMNLP

[7] Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, Noah A. ,Smith, 2016, "Transition-Based Dependency Parsing with Stack Long Short-Term Memory"

[8] Yuen-Hsien Tseng, Lung-Hao Lee and Liang-Chih Yu, 2012, "Traditional Chinese Parsing Evaluation at SIGHAN Bake-offs 2012"