Climbing towards the NLU of Chinese *wh*-expressions

邁向中文 *wh* 詞的自然語言理解工程

陽明交通大學 外國語文學系

周昭廷 鍾孟軒 陳畯田

**Outline**

1 Introduction & problem description 1

2 Empirical focus: The interpretations of Chinese *wh*-expressions 4

2.1 Three readings of Chinese *wh*-expressions 4

2.1.1Interrogative use 5

2.1.2Existential use 5

2.1.3Universal use 8

2.2 How to handle Chinese *wh*-expressions in data-driven NLP? 11

2.3 How data-driven NLP fails in translation 15

3 Proposal / Design 20

3.1 Articut 斷詞系統 20

3.2 Loki 語意理解引擎 26

3.3 Articut/Loki 的運作細節：以存在語意為例 28

3.4 C-command 關係檢查器的程式實作 33

3.5 Articut/Loki 的效率、效能與準確率 35

4 Tentative plan and possible challenges 37

5 Conclusion & future work 40

6 References 42

# Introduction & problem description

徐嘉慧、何萬順、劉昭麟 (2016) 針對語言學門熱門議題的調查報告中，提及「語言學內部 (intradisciplinary) 的跨次領域整合及語言學與其他學科的跨科際 (interdisciplinary) 研究是兩大重要主題」。雖然沒有語言學家會反對這個結論，但是臺灣國科會於1995 年成立語言學門，至今已接近30年，真正付諸實行跨科際的語言學研究成果並不能算是豐碩[[1]](#footnote-1)。語言學的跨科際對話的貧乏並不只是臺灣語言學門的獨有困難，而是整個語言學領域的共通問題。如 Joe Pater 於 2019 年在期刊 *Language* (美國語言學會出版的指標性期刊) 刊出的文章所述，生成語言學 (Chomsky 1957) 與人工智慧類神經網路 (Rosenblatt 1957，以下簡稱‘類神經網路’)[[2]](#footnote-2) 兩個領域雖是在同一年代誕生，但一直以來的發展態勢經常都是敵對且壁壘分明，Pater提出這兩個領域的衝突與歧見其實並不是本質上的不相容，僅僅只是研究的焦點不同罷了：生成語言學注重語言知識的結構表徵 (structured representation)，而類神經網路著眼於語言知識的學習機制與結果；因此 Pater 認為這兩個領域應該是互補而非互斥，若能互通想法，整合研究，對整個認知科學領域才能產生更豐碩正面的成果。

Pater 試圖彌平跨科際衝突的文章引起廣大迴響，在同一期的 *Language* 中，有兩篇回覆的內容在本計劃的目標下特別重要。首先，Linzen (2019) 陳述生成語言學與類神經網路可以如何達到互惠，生成語言學對於類神經網路的貢獻在於：針對人工智慧運算程式可能呈現的語言能力，提供明確的操作定義以及可驗證的假設與預測，並建構有效的實驗方式，測試特定人工智慧運算程式的學習表現是否明確地符合語言學家界定的標準；而類神經網路對於生成語言學的貢獻在於：提供智慧運算的程式，模擬人類語句處理的結果，且測試生成語言學家假設的語言知識限制，如結構表徵 (structured representation) 與語言規則的結構依存性 (structural dependency)，是否為成功的學習成果之必要條件。

另一篇更深刻的回覆是Berent & Marcus (2019)，他們提出雖然Pater描繪的跨科際研究交流願景的確有益於認知科學的整體發展，但以連結主義 (connectionism；Rumelhart et al. 1986) 為本的類神經網路領域與生成語言學領域有兩個無法妥協的本質歧異[[3]](#footnote-3)。第一個歧見在於語言通則知識 (generalization) 的學習基礎與運用機制，生成語言學家認為語言通則知識的本質是代數性的 (algebraic[[4]](#footnote-4)；見Chomsky & Schützenberger 1963; Fodor & Pylyshyn 1988; Pinker & Prince 1988; Pinker 1991; Marcus 2001)；第二個歧見為結構表徵的需求，生成語言學的理論基石為結構表徵，語言規則運用以及語意解讀的必要條件皆為結構表徵 (即’結構依存性structural dependency’)，這兩個基礎的定義統整如(1)：

1. ) The algebraic hypothesis: (Berent & Marcus 2019: e78)
   1. Structured representations

i(i) Categories (e.g. Noun) form equivalence classes, distinct from their members

(e.g. dog).

(ii) Mental representations are symbols (either simple or complex).

(iii) The meaning of complex representations depends on the syntactic structure of their form and the meaning of their simple constituents.

* 1. Structure-sensitive processes

i(i) Mental processes manipulate the syntactic form of representations in a manner that is blind to their semantic content.

i(ii) Mental processes operate on variables.

另一方面，類神經網路領域則否認 (1) 之中每一條陳述，與 (1) 對比，類神經網路領域在自然語言處理 (Natural Language Processing, NLP) 任務中的基本運作方針可統整如 (2) (Berent & Marcus 2019: e78)：

1. ) a. Mental operations consist of associations between inputs and outputs, induced by

experience.

* 1. There are no abstract categories distinct from their instances.
  2. There are no systematic links between the structure of mental representations and their meaning.

因此，在 (2) 的框架下，語言通則知識的運用基礎並不具備代數性質，僅是特定字詞特徵(features) 之間的相關性連結罷了 (Rumelhart & McClelland 1986; Plunkett & Juola 1999; Ramscar 2002; Bybee & McClelland 2005)；其次，類神經網路領域通常是否認結構表徵在語言知識系統中的真實性或必要性，因此設計出來的智慧運算系統通常不會植入結構表徵知識，而在這樣的自然語言處理系統中，詞彙/語句的結構與其意義並沒有系統性的直接連結關係，語言的理解皆來自字詞之間的相關性連結與類比 (analogy)。

在基本定義如此對立 (diagonal) 的情況下，Berent & Marcus (2019) 認為生成語言學與類神經網路領域若真要進行有建設性的跨科際對話與合作，只有兩條路可以前進：一，(2) 的運作框架是正確的，而生成語言學需要大幅修正其理論基礎 (1)；二，(2) 需要修正為一個接納代數與結構表徵的運作框架[[5]](#footnote-5)。近年來，在類神經網路領域已經有許多研究顯示，第二條路可以大幅優化類神經網路的 NLP 表現；舉例來說，Tran et al. (2018) 發現在判斷一致性協同 (agreement) 的任務上，植入結構表徵知識的類神經網路的表現優於只注意詞頻的類神經網路 (如Vaswani et al. 2017)：後者在雜訊較多的語句中，判斷一致性協同的錯誤比前者多出三倍 (Kuncoro et al. 2018也提出類似的結論)；類似的研究還有 McCoy et al. (2020) 訓練類神經網路學習英語形成是非問句的兩種可能方式：移動主句 (main clause) 中的助動詞 (3b) move-main 或是移動從左至右線性次序上的第一個助動詞 (3c) move-first。

1. ) Subject-auxiliary inversion: move-main or move-first?
   1. Your zebras that don’t dance do chuckle.
   2. Do your zebras that don’t dance chuckle? (move-main)
   3. \*Don’t your zebras that dance do chuckle? (\*move-first)

他們發現只有植入結構表徵的類神經網路的表現能夠穩定地選擇正確的move-main規則來產出正確英語問句[[6]](#footnote-6),[[7]](#footnote-7)。本計劃目標在於延續此發展脈絡，實證焦點為中文生成語言學早期的熱門話題「*wh*詞的解讀多樣性」(如範例 (4)；Huang 1982/1998; Cheng 1991; Li 1992; Tsai 1994; Lin 1996, 1998)，以此經典的語言現象為本，證實 Berent & Marcus (2019) 指出的第二條路是較有競爭力的發展可能性[[8]](#footnote-8)。

1. ) 「什麼」的三種語意：
   1. 疑問 (Interrogative): 約翰吃了什麼? (=What did John eat?)
   2. 存在 (Existential): 約翰可能吃了什麼。(=John might have eaten something.)
   3. 全稱 (Universal): 約翰什麼都吃。(=What did John eats everything.)

本計劃書的結構如下：第 2 節首先簡介中文 *wh* 詞的三種語意解讀及其句法分佈限制，目標在指出現代生成語言學的研究成果並非資訊科學界認為的「龐雜、主觀、無法窮盡或因人而異」；接著討論立基於 (2)，以資料為本 (data-driven)，不植入結構表徵的 NLP 系統如何處理相關中文語句。第 3 節介紹立基於 (1)，植入代數特質與結構表徵的「Articut/Loki」(Wang et al. 2019) 如何以語言規則為本 (rule-driven) 進行 NLP 以及自然語言理解 (NLU)。第 4 節為計劃期程與預期挑戰。第 5 節為本計劃結論與發展願景。

# Empirical focus: The interpretations of Chinese *wh*-expressions

## Three readings of Chinese *wh*-expressions

如上節範例 (4) 所示，中文的 *wh* 詞除了有最基本的疑問語意解讀 (4a)，也可在特定的語境中產生存在 (4b) 或是全稱 (4c) 解讀，而這三種解讀都各有其句法分佈限制；本節目標有二：闡明三種解讀的句法環境分佈以及討論立基於 (2)，以資料為本 (data-driven)，不植入結構表徵的NLP運算系統如何處理隨著句法環境變化，中文 *wh* 詞展現的解讀多樣性。

### Interrogative use

中文能夠在生成語言學的發展歷史中佔有一席之地，Huang (1982/1998) 的論文有決定性的影響，其中的關鍵語料為中文 *wh* 詞的疑問語意分佈限制。Huang (1982) 發現在補語子句的中文 *wh* 詞，其疑問語意範域 (interrogative scope) 取決於主句中的動詞特性，舉例來說，如「問」或「想知道」類別的動詞，其補語子句必須是疑問句(標註為 [+wh] )，因此 (5a) 中的「誰」的疑問語意範域便局限於補語子句，形成了間接問句；而「相信」類的動詞，其補語子句必須是直述句 (即 [-wh] )，而在這種補語子句中的 *wh* 詞疑問範域便延伸至主要子句，形成直接問句如 (5b)；最後，「知道」類的動詞對於其補語子句的疑問語氣沒有特定限制，即其補語子句可以是問句也可以是直述句，因此在這類補語子句中的 *wh* 詞的疑問範域便可侷限於補語子句內於或是延伸至主要子句，如(5c)。

1. ) The [wh] feature of selecting verbs: (Huang 1982:180)
   1. 張三問我 / 想知道 [誰買了書]。 Indirect question
   2. 張三相信 [誰買了書]? Direct question
   3. 張三知道 [誰買了書]。/ ? Ambiguous

根據以上簡略說明，我們期待一個真正能夠「理解」中文的 NLP/NLU 智慧運算程式應該能做到 (6)：

1. ) 本計劃對中文 NLP/NLU 智慧運算程式的期待 (part 1)：

區辨直接與間接 *wh* 問句。

### Existential use

除了疑問語意，許多學者發現中文 *wh* 詞在特定的句法環境中能產生表存在量化的解讀(=existential polarity *wh*-phrase, EPW; Huang 1982/1998; Cheng 1991; Li 1992; Lin 1996, 1998)，例如在否定詞之後(7a)，是非問句中(7b)，或是條件子句中(7c)：

1. ) 核可 EPW 的語境：
   1. 他沒說什麼。 (否定詞之後)
   2. 你吃了什麼嗎? (是非問句中)
   3. 要是/如果誰欺負你… (條件子句中)

Li (1992:134) 提出 EPW 的分佈環境為：“contexts where the truth value of the proposition is not positively fixed in a definite manner”，因此，在 (8) 之中真假值無法明確界定的環境下，「什麼」都呈現存在語意。

1. ) 核可 EPW 的語境：
   1. 他 好像/似乎/應該/可能/大概 喜歡什麼人的樣子。 (uncertainty contexts)
   2. 我以為 [你喜歡什麼]。 (non-factive complements)
   3. 他看到什麼了。 (circumstantial *le*)

Lin (1998) 進一步限定 Li (1992) 定義的 EPW 分佈環境，他的通則概念從 Li (1992) 的命題真假值限縮到了 *wh* 詞指涉人事物的存在必然性如 (9)。

1. ) Non-Entailment-of-Existence Condition on EPWs (NEEC):

The use of an EPW is felicitous iff the proposition in which the EPW appears does not entail

existence of a referent satisfying the description of the EPW. (Lin 1998:230)

除了界定存在語意可以出現的環境，Li (1992) 更重要的發現是核定 EPW 的關鍵字詞(licensor) 必須在結構上成分管轄 (c-command, Reinhart 1976) EPW，c-command是現代生成語言學中最重要的結構幾何概念，其定義為：”a node in a parse tree c-commands its sister node and all of its sister's descendants”，以 (10) 的樹狀圖為例，A 與 B 互為 sisters (因兩者同屬 M 之下的同階層分支)，因此 A c-command B 以及 B 之下所有的成分 (C, D, F, G)。

1. ) C-command

M A 的 c-command 範圍

A B

C D

F G

舉例來說，雖然 (7a) 告訴我們否定詞可以核定 EPW，但 (11) 的例句及其簡易的樹狀結構顯示此核定關係的前提是否定詞 (含有「不」或是「沒」的字詞) 必須在結構上 c-command EPW，因此雖然 (11) 之中有否定詞「沒」，但「誰」並不在其 c-command 範圍內，因此只得到疑問語意解釋。

1. ) **誰** 沒買這本書?

Sentence

誰 Verb Phrase 「沒」的c-command 範圍

沒 Verb Phrase

買 這本書

同樣的道理，雖然 (12a) 中的 A-*not*-A 問句可以核定「什麼人」的存在語意，但是當「什麼人」離開了 A-*not*-A 的 c-command 範圍，(12b) 便成為一個不合法的句子[[9]](#footnote-9)。

1. ) A-*not*-A 審核 EPW 的 c-command 效應：
   1. 他 喜不喜歡 什麼人?
   2. **\***什麼人 喜不喜歡 他?

特別值得強調的是 (7a)/(11) 以及 (12) 的對比無法由 *wh* 詞與核定詞的相互線性前後次序 (linear order) 決定；的確，表面上核定存在語意的核定詞都在 *wh* 詞之前，但是線性次序在 *wh* 詞之前並不必然等同於結構上 c-command *wh* 詞，以 (13) 為例，雖然否定詞「不」在「什麼」之前，但由於鑲嵌在複雜名詞主語中，在結構上並不 c-command 「什麼」，因此(13)句的「什麼」無法表達存在語意，整個句子為一個 *wh* 問句。

1. ) [張三 不 喜歡的那個學生] 說了什麼?

除了核可詞的 c-command 結構要求，EPW 還有其他句法限制。在 (14a/b) 中，雖然否定詞以及 A-*not*-A 疑問詞都在「誰」之前且 c-command 「誰」，但在這兩個句子中的「誰」都只能有疑問語意 (即兩句中的補語子句皆為間接問句)，Li (1992) 提出若是句子中有疑問語意的核可詞 (如「想知道」此類動詞)，那麼即使句中同時有 EPW 的核可詞，那麼核可的效力以疑問語意的核可詞為優先。(11)

1. ) The minimality competition between [wh] and the existential licensor:
   1. 他 不 想知道 [誰買了這本書]。

‘He does not want to know who bought this book.’

* 1. 他 想不想 知道 [誰買了這本書]?

‘Does he want to know who bought this book?’

例句 (15) 也示現相同的核可效力差異，「以為」核可 EPW，而「想知道」核可疑問語意，此句唯一可能的解讀是「誰」與「什麼」都是疑問語意，範域限於補語子句，構成間接問句。

1. ) 他 以為 我 想知道 [誰 喜歡 什麼]。

‘He thinks that I wonder who likes what.’

\*’He thinks that I wonder what someone likes.’

\*’He thinks that I wonder who likes something.’

除此之外，疑問語意核可詞的優先效力甚至強過結構近距效應 (structural minimality)，以 (16) 為例，疑問語意核可詞「呢」位於主句最高的 CP 詞組，結構上高於EPW核可詞「以為」，因此「以為」與「呢」相比，靠補語子句中的 *wh* 詞更近，但有趣的是此句的兩個 *wh* 詞都只能有疑問語意 (=(16a))，其他的語意解讀 (=(16b-d)) 都不存在。

1. ) **他 以為** [**誰** 喜歡 **什麼**] 呢?
   1. ‘Who(x), what(y), he thought x liked y?’
   2. \*’Who(x), he thought x liked something?’
   3. \*’What(y), he thought somebody liked y?’
   4. \*’He thought who liked what.’

根據 (15) 與 (16) 的對比，Li (1992) 提出了通則 (17)。

1. ) A licensor of an existential *wh*-phrase will not block the licensing of a higher constituent

question licensor, but the lower constituent question licensor will block the licensing

potential of a higher existential licensor.

統整此節對於 EPW 分佈限制的討論，我們期待一個真正能夠「理解」中文的 NLP/NLU 智慧運算程式應該能做到以下三件事：

1. ) 本計劃對中文 NLP/NLU 智慧運算程式的期待 (part 2)：
   1. 在不同的語境中，區辨 *wh* 詞的疑問與存在語意。
   2. 判斷核可詞與 EPW 之間的結構 c-command 關係，而不只是線性前後次序。
   3. 判定疑問核可詞與 EPW 核可詞的優先順序。

### Universal use

最後一項的可能語意解讀為全稱量化語意，如 (19) 中的「什麼、誰」都沒有疑問語氣，也不具備存在的預設 (existential presupposition)，表達的是所有任意的物件、個體皆能符合命題需求。

1. ) 全稱量化的 *wh* 詞：
   1. 約翰 什麼 都吃。
   2. 約翰 誰 都喜歡。

針對此類語意，有三點特質值得一提。首先，全稱量化的解讀與「都」出現在同一個句子中，且得到全稱量化解讀的 *wh* 詞必須在「都」的左側 (the Leftness Condition, Lee 1986; Cheng 1995; Lin 1998)[[10]](#footnote-10)，因此從 (19) 變化為 (20)，就喪失了全稱語意，句子的解讀皆為疑問句。

1. ) 「都」審核全稱量化語意及其方向性限制：
   1. 約翰都吃什麼?
   2. 約翰都喜歡誰?

第二， *wh* 詞的全稱量化語意不受限於孤島環境，在 (21) 中，雖然「什麼」出現在一個複雜名詞子句中，其全稱量化語意仍然被核可。

1. ) [李四吃 什麼] 都跟我無關。

‘Whatever Lisi eats is irrelevant to me.’

(Cheng 1995: 203)

第三，當「都」的左側出現一個以上的 *wh* 詞時 (如 (22) )，較鄰近「都」的 *wh* 詞才會得到全稱量化語意：雖然「誰」與「什麼」皆位於「都」的左側，只有「什麼」可以取得全稱量化語意，而「誰」只有疑問語意。

1. ) 誰 什麼 都吃?
   1. ‘Who eats everything?’
   2. ‘\*What does everyone eat?’
   3. ‘\*Everyone eats everything.’

(Cheng 1995: 203)

有趣的是，(23) 的語意解讀似乎構成了此規則的反例：「誰」與「什麼」皆位於「都」左側，而且與「誰」比起來，「什麼」比較靠近「都」，但「誰」與「什麼」都得到了全稱量化語意。

1. ) [誰吃什麼]都跟我無關。

Lit. ‘Whoever eats whatever is irrelevant to me.’

(Cheng 1995: 222)

(22) 與 (23) 的對比進一步證明我們在上一節提出的重點：線性前後次序 (linear order) 並無法準確描繪語言運作的原則。(22) 與 (23) 的差異是結構上的：在 (22) 中，兩個 *wh* 詞都在結構上 c-command 「都」，但 (23) 中的兩個 *wh* 詞皆鑲坎在主詞中，因此與「都」無法建立 c-command 關係。

最後，上一節的 (14)-(16) 例句揭示疑問語意的審核順序高於存在語意，延伸此通則，我們提出中文 *wh* 詞的三類語意的審核優先順序通則如 (24)。在 (24a) 中，動詞「想知道」與副詞「都」審核不同的語意，而此句中的「誰」為疑問語意。(24b) 中的否定詞「不是」以及副詞「都」審核不同的語意，而此句中的「誰」為全稱語意。

1. ) 疑問語意 > 全稱語意 > 存在語意：
   1. 疑問 > 全稱： 我想知道 誰 都 考第一名
   2. 全稱 > 存在： 不是 誰 都 對我這麼好
   3. 疑問 > 存在：見 (14)-(16)

總結本小節討論，一個中文 NLP/NLU 智慧運算程式若要計算 *wh* 詞的全稱量化語意，必須達到以下四項要求：

1. ) 本計劃對中文 NLP/NLU 智慧運算程式的期待 (part 3)：
   1. 可判別全稱量化語意的 *wh* 詞需與「都」出現在同個句子中，且需要出現在「都」的左側 (the Leftness Condition)。
   2. 允許 *wh* 詞的全稱量化語意出現在孤島中。
   3. 判別以下兩種狀況：當兩個 *wh* 詞與「都」在同個句子中時，且都 c-command 「都」，則只有結構距離較近 (structural minimality) 的 *wh* 詞可到全稱量化語意；若兩個 *wh* 詞皆沒有 c-command 「都」，那兩者皆可獲得全稱量化語意。
   4. 判定疑問、全稱、存在語意的優先順序。

下一節我們將介紹以資料為本的 NLP 運算程式的基礎邏輯，並討論在這個架構下設計出來的運算程式如何處理中文詞 *wh* 詞的解讀多樣性。[[11]](#footnote-11)

## How to handle Chinese *wh*-expressions in data-driven NLP?

若將人類的語言視為一個模組化認知系統 (modularized cognitive system, Fodor 1983)，則其主要子模組為「語意」、「句法」以及「音韻」[[12]](#footnote-12)三個層面，因此一個資訊運算程式要能妥善處理自然語言，勢必在這三個領域中需有對應的處理設計。中文 NLP 工具依運作原理可分成四大類型整理如 (26a-d)，分別代表 NLP 設計概念與工具技術的演進，本節討論(26a-c)，而(26d)的細節、運作邏輯以及產學價值留待第 3 節詳述[[13]](#footnote-13)。

1. ) 四種中文 NLP 工具：
   1. 依字典運算：以 mmseg 為代表

(<http://technology.chtsai.org/mmseg/>)

* 1. 依資料統計機率運作：以 Jieba 為代表

(<https://github.com/fxsjy/jieba>)

* 1. 依機器學習模型運作：以中研院 CKIP CoreNLP 以及 OpenAI 的 ChatGPT 為代表

(<https://ckip.iis.sinica.edu.tw/service/corenlp/>)

(<https://openai.com/blog/chatgpt>)

* 1. 依生成語言學原理運作：以 Articut/Loki 兩套彼此相依的系統為代表

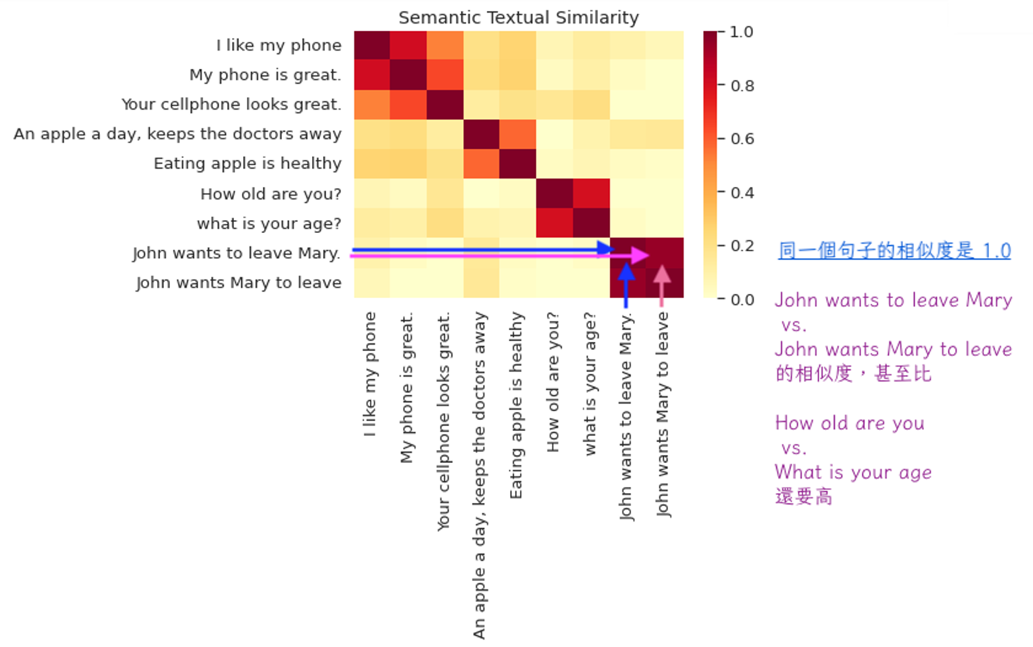
([https://api.droidtown.co](https://api.droidtown.co/))

類型 (26a) 代表資訊領域早期 NLP 程式設計者對語言的想像：「只要編寫完整的字典，就能處理自然語言理解的需求」；但此類型 NLP 程式的困難在於語言本身為一動態平衡系統，新詞不斷產生，且舊詞也會產生新解，在這個不穩定的字典資料基礎下，此類型 NLP 程式無法處理自然語言語句的結構與語意，因此被第二代資料統計機率方式類型 (26b) 取代。

第二代系統類型 (26b) 依資料統計機率方式運作，好處是對於新詞提出了一個機率解算法，用以推算詞彙邊界 (人類嬰兒運用機率推算詞彙邊界的能力，可參閱 Saffran et al. 1996 的經典研究)。但其限制有二：一是統計機率方法受母體資料本身的影響極大，例如最流行的 Jieba 版本採用中國 1998 年人民日報的人工標記內容為母體，故其處理台灣的城市名稱如「台中」時，永遠無法學會「台中」為一個獨立詞彙，而以「台/中」為處理結果；二是 Jieba 的詞性標記系統 (Part-of-Speech, POS) 效果不佳，而大量基於 POS 的命名實體辨識 (Named Entity Recognition, NER)[[14]](#footnote-14)效果更是差至無法使用的程度。由於 (26b) 的設計目標在於提供詞彙邊界，因此其應用範圍自然就侷限於重視詞頻的場景，當然無法滿足更高層次的語意/句法運算需求。

其後，隨著 2012 年大數據 (Big Data) 興起，依第三代機器學習/深度學習演算法 (26c) 實作而成的中研院 CKIP Tagger 和 CKIP CoreNLP 雖然一舉成功處理了詞彙邊界和 POS 及部份 NER 的困難，但其基於數據的本質仍使其受訓練資料的影響極大 (與上述 (26b) 的第一個問題相同)。以上三種方法都是延續了資訊領域面對NLP任務時，無法確認「何為詞彙」的困難而實作出的解決方案，但是光是找出詞彙邊界並為詞彙標上 POS/NER 標記，仍然無法適切處理語言系統運算過程中所需的「語意」與「句法」層面的問題。

在資料為本的 NLP 程式架構下，詞彙的語意解讀，來自模型訓練資料中，該詞彙次序前後的字符分佈以及向量化資訊：詞彙次序前後的字符分佈即為「資料中的分佈」[[15]](#footnote-15)；一個詞彙的向量化資訊為一數字序列，為一組的數字，作為電腦計算詞彙語意的資訊基礎 (即 word-to-vector)，如此的詞彙資訊表徵可以讓相似語意或是相似分佈特質的詞彙呈現相似的向量資訊[[16]](#footnote-16)。比詞彙語意解讀再高一階層的語句語意的運算基礎有二：詞彙語意的理解以及句法結構表徵提供的計算框架；雖然 word-to-vector 向量模型的語意類比結果可以有一些成功表現，缺乏句法結構表徵解算能力的向量模型，無法準確及穩定地處理語句的語意解讀。舉例來說，在[Google Colab](https://colab.research.google.com/github/tensorflow/hub/blob/master/examples/colab/semantic_similarity_with_tf_hub_universal_encoder.ipynb)，我們可以產出圖一，顯示在machine learning /data-driven NLP/NLU 模型下，英文語句之間的文意相似性；當圖形能如圖一形成一個左上到右下的粗線，那表示「模型已經收斂了，該學的都學完了 (但學完不表示學會)」[[17]](#footnote-17)。當模型收斂了，我們可以用這個圖來了解系統如何判斷兩個英文句子 a 與 b 之間的文意相似性 (=similarity (a,b))[[18]](#footnote-18)；當 similarity (a,b) 為 1.0 的時候 (在圖中以深紅色顯示)，這表示模型認為 a 和 b 的文意是一模一樣。



圖一：向量模型下的文意相似性測試

根據圖一，我們可以看到雖然系統正確判斷 x軸 ”John wants to leave Mary” 與 y軸 “John wants to leave Mary” 兩個一模一樣的的文意相似性為 1.0 (100%，以深紅色顯示)，但更重要的是，”John wants to leave Mary” 與 “John wants Mary to leave” 的文意相似度竟落在 0.9，但 “How old are you?” 與 “What is your age?” 的文意相似度卻只在 0.7 左右。根據此測試結果，我們可以知道在前後文相似的情況下 (即具有相近的向量值)，向量模型甚至無法區辨 "John wants to leave Mary" 和 "John wants Mary to leave" 之間明顯而巨大的語意差異。因此，仰賴前後文相似性的向量模型無法掌握語句文意任務。同樣的，若將詞彙向量化模型應用至中文的 *wh* 詞，取得的資料就是「*wh* 詞前後出現的字元」，而不是從語境或是結構來思考。然而，(27)和(28)中的 *wh* 詞「什麼」的前後文都是 {你, 都, 不, 看, 電影} 等字元的集合。換言之，這兩句 *wh* 詞的向量差會非常小，從向量語意學的角度來看也會是「非常接近」到分不出來的程度。

1. ) 你都不看什麼電影 (?/。) (疑問/存在)
2. ) 你什麼電影都不看。 (全稱)

接著我們示範以資料為本的 NLP 程式架構如何處理中文的「什麼」在 (29a-c) 呈現的三種可能語意：

1. ) 「什麼」的三種語意：
   1. 疑問 (Interrogative): 你看見了什麼?
   2. 存在 (Existential): 我想我看見了什麼。
   3. 全稱 (Universal): 你什麼都沒看見。

基於「詞彙」思維的機器學習解決方案 (以 CKIP CoreNLP 為例) 應用至(29a-c)會產生(30a-c)的詞彙詞性處理結果[[19]](#footnote-19)：

1. ) CKIP CoreNLP 詞彙詞性處理結果:
   1. Interrogative: 你(Nh)　看見(VE) 了(Di) **什麼(Nep)**
   2. Existential: 我(Nh)　想(VE)　 我(Nh) 看見(VE) 了(Di)　**什麼(Nep)**
   3. Universal: 你(Nh)　**什麼(Nep)** 都(D) 沒(D)　 看見(VE)

三個句子的「什麼」的詞性都被標記為指定代詞 (某種代名詞)，由於只有疑問以及存在語意的*wh* 詞可(勉強)分類為指定代詞，因此針對 (29a-c)，CKIP CoreNLP 詞彙詞性處理的正確率為 2/3 (=0.66)。接著以 (30) 為基礎，CKIP CoreNLP 會產生 (31a-c) 的詞彙語意處理結果：

1. ) CKIP CoreNLP 詞彙語意標記結果:
   1. Interrogative:

你(listener) 看見(look) 了(Vachieve) **什麼(Ques)**

* 1. Existential:

我(speaker) 想(think) 我(speaker) 看見(look) 了(Vachieve) **什麼(Ques)**

* 1. Universal:

你(listener) **什麼(Ques)** 都(all) 沒(Vachieve) 看見(look)

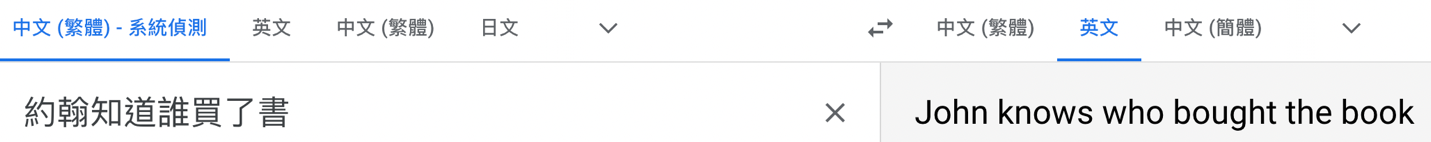
在 CKIP CoreNLP 框架下，詞彙語意的分析都表達相同的疑問語意 (即 (31) 中的 Ques 標記)，這個解算結果不只無法完整展現 *wh* 詞的語意多樣性，更是錯解了 (29b-c) ，正確率只有 1/3 (0.33)。總體而言，針對 (29a-c) 句的句法和語意分析，CKIP CoreNLP 正確的機率為 0.66\*0.33 = 0.2178。因此，基於字典、基於資料統計機率模型以及基於機器學習方法的三種中文 NLP 運算系統無法處理中文 *wh* 詞依句型和上下文各有不同解讀的研究成果。

## How data-driven NLP fails in translation

本節我們用另一種 NLP 任務「機器翻譯」來示現純然以文字資料 (text data) 與詞序為本去處理人類語言會遇到的盲點 (Sutskever et al. 2014)。首先，附屬子句中的 *wh* 詞的疑問語意範域受主句動詞的特質影響 (見(5))，(32) 的結果顯示 Google 翻譯無法正確處理「約翰相信誰買了書」是一個直接問句，這顯示 Google 翻譯重視語言之間顯性的 (overt) 線性詞序對應，但遇到需要遠距結構依存性 (long-distance structural dependency) 才能解讀 (decode) 的語意時就會出現盲點。有趣的是，當我們將「約翰相信誰買了書」加上問號時，Google 翻譯便可正確地將此句翻譯為英文直接問句，這顯示 Google 翻譯在抓取 (encode) 文字資料時，著眼點在每一個字符及其上下文線性排列，包括問號符號，而非詞彙本身帶有的隱性 (covert) 特質 (如「相信」是 (-wh)，其補語子句不是間接問句) 或是字詞之間有規律的組合順序與結構，也就是說 Google 翻譯進行 encode-decode 的標的物僅是線性字符排列而非具有高低深淺結構的語言組合[[20]](#footnote-20)。

1. ) *Wh* 詞的疑問語氣範域：Google 翻譯結果 (測試日期2022/09/30)





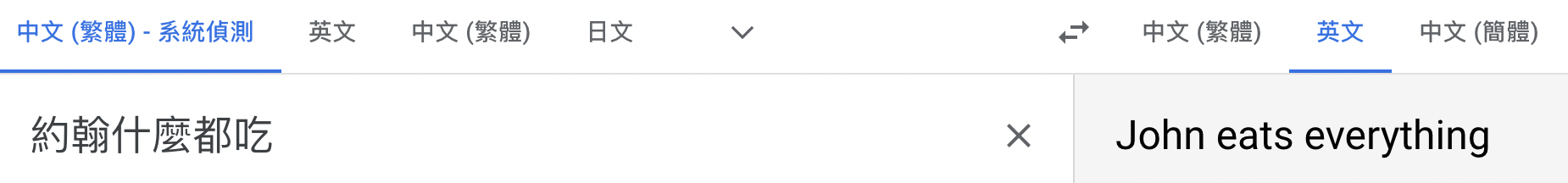


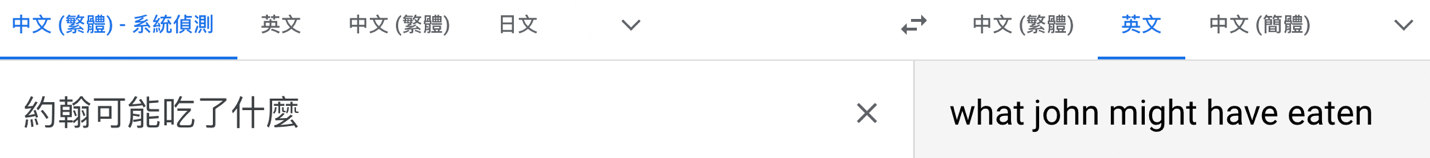


第二，我們檢驗 Google 翻譯是否能正確捕捉 *wh* 詞的另外兩種語意，從 (33) 的結果可以看到，Google 翻譯正確地處理了疑問與全稱量化語意，但「約翰可能吃了什麼」的存在語意無法正確翻譯出來。這代表以資料為本的 NLP 無法處理 NEEC (=(9)) 這種需要綜觀全句語意的原則。

1. ) 三種 *wh* 詞的語意：Google 翻譯結果 (測試日期2022/09/30)

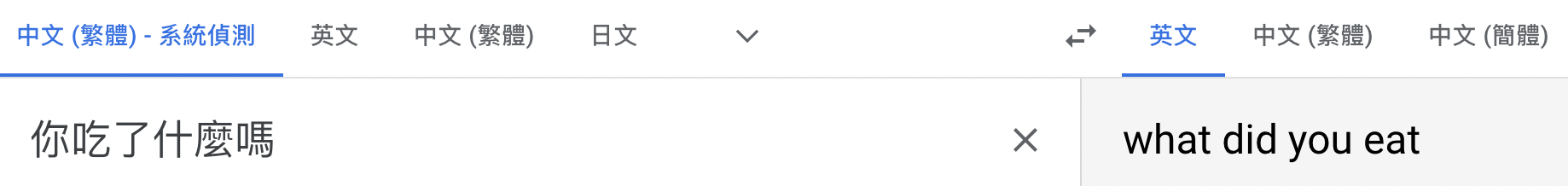


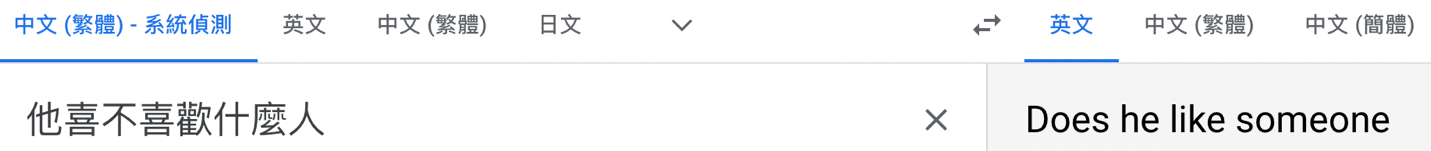


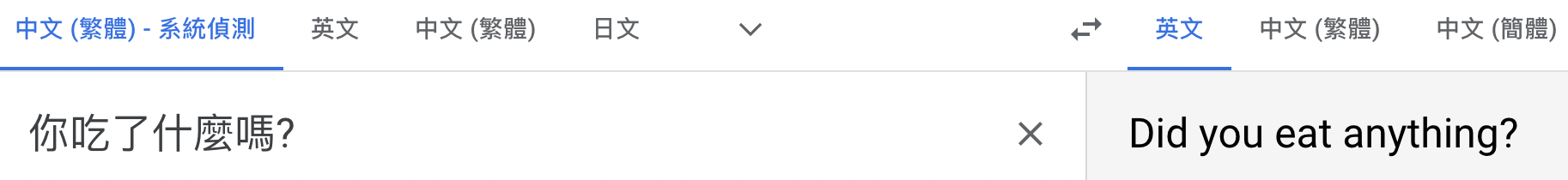


我們進一步測試其他核可存在語意的語境，如 (34) 中帶「嗎」字的 *yes*-*no* 問句以及 A-*not*-A 問句；有趣的是，帶「嗎」字的 *yes*-*no* 問句中的「什麼」無法被正確 decode 為存在語意，但是 A-*not*-A 問句中的「什麼人」就是正確的翻譯；除此之外，如同例 (32)，當帶「嗎」字的 *yes*-*no* 問句加上問號時，其中的「什麼」就可以成功的被 decode 為存在語意。

1. ) 「什麼」的存在語義：Google 翻譯結果 (測試日期2022/09/30)







這種詭異的翻譯表現差異的背後道理在於 Google 處理輸入的線性文字資料 (text data) 並不是按照規則去劃分為有結構的語言資訊 (linguistic information)，而是用 P(A|B) 條件機率的排序邏輯 (=在事件 B 發生的條件下，事件 A 發生的條件機率為何) 去處理線性文字資料。根據此邏輯，「你吃了什麼嗎」以及「你是不是吃了什麼」的處理歷程如 (35)-(36) 所示。

1. ) Google翻譯處理「你吃了什麼嗎」的歷程：

P(end|你吃了什麼嗎) \* P(嗎|你吃了什麼) \* P(麼|你吃了什) \* P(什|你吃了) \* P(了|你吃)

\* P(吃|你) \* P(你|bgn)

1. ) Google翻譯處理「你是不是吃了什麼」的歷程：

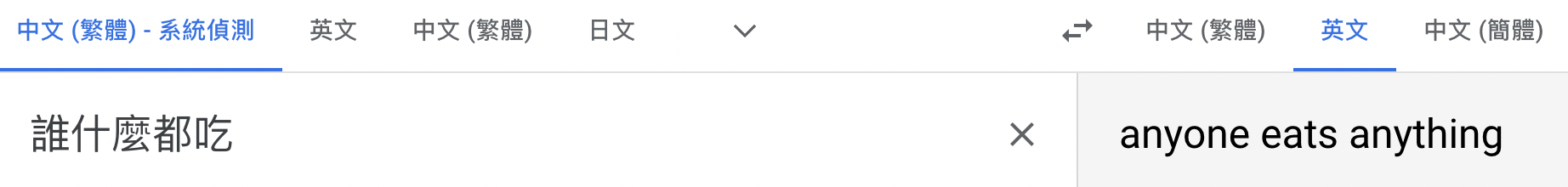
P(end|你是不是吃了什麼) \* P(麼|你是不是吃了什) \* P(什|你是不是吃了)

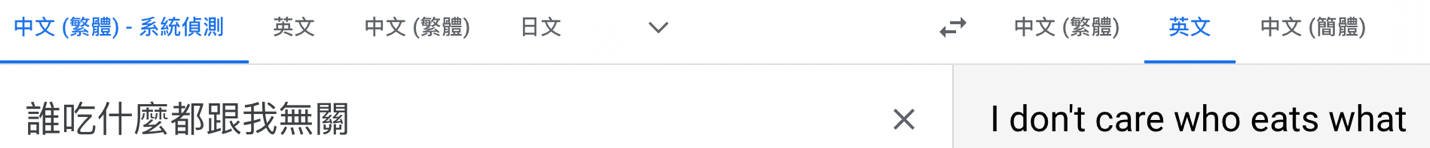
\* P(了|你是不是吃) \* P(吃|你是不是) \* P(是|你是不) \* P(不|你是) \* P(是|你) \* P(你|bgn)

從 (35) 我們可以看到，會影響整個句子是偏向 *wh* 問句或是 *yes*-*no* 問句的「嗎」，只佔了一個位子；但是 (36) 的「你是不是吃了什麼」中有三個變數 (是，不，是) 會影響整個句子的解讀方向，因此在資料分佈上的比重以及翻譯 encode-decode 的任務中，A-*not*-A 的影響比較大。即使從語言學的角度來看，A-*not*-A 是個有其特殊規律的問句結構 (Hagstrom 2006)，「嗎」是一個有特殊功能的功能核心詞 (functional head)，但只看文字線性排序資料的 data-driven NLP 運算程式不知道 A-*not*-A 問句無法與 *wh* 問句取得相同的疑問範域，也不知道「嗎」有其特定的句法/語意/語用功能存在，對此類 data-driven NLP 運算程式來說，問句符號的影響甚至大於「嗎」這個語言功能核心詞。

在2.1.3節我們討論過「都」與其核可的全稱量化*wh* 詞之間的 c-command 結構關係 (見 (22) 與 (23) 的對比)，根據 (37) 的測試結果，我們發現 Google 翻譯無法捕捉此結構上的差異，「誰什麼都吃」中的「都」左側的「誰」被錯譯為全稱量化詞，只有在加上問句符號的時候，「誰」才能被正確翻譯為疑問語意。這個翻譯表現的差異再次顯示只看文字線性排序資料的 data-driven NLP 運算程式在面對需要結構關係 (如 c-command) 才能解讀的語言資料時的手足無措。

1. ) 「都」核可全稱量化*wh* 詞的結構依存性：Google 翻譯結果 (測試日期2022/09/30)

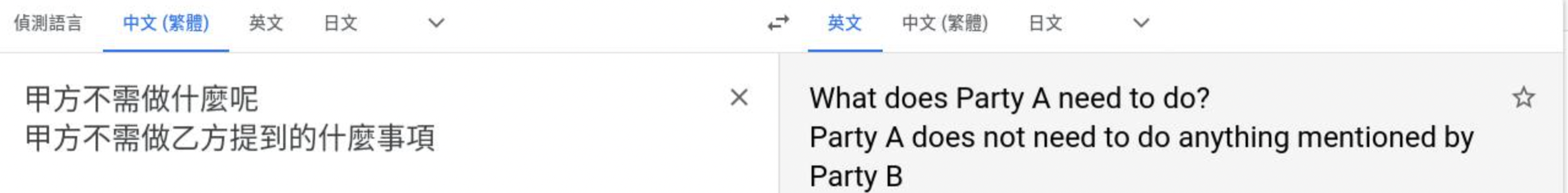






在業界實務層面，若依賴此類以線性字符排列為本的NLP 運算程式進行商業合約條文翻譯，會產出如 (38) 的結果，很清楚的可以看到兩句條文中的 *wh* 詞都被錯解，若未來跨國商務使用的 NLP/NLU 服務亦以線性字符排列為設計核心，如 (38) 的結果就可能會造成相當程度的損失。

1. ) 類商業合約條文測試：Google 翻譯結果 (測試日期2022/09/30)



最後，我也測試了最先進的 ChatGPT 3.5 判讀中文 *wh* 詞的表現。首先，針對下圖中 7 個中文句子的翻譯任務，ChatGPT 犯了三個錯誤[[21]](#footnote-21)：句 5 應該是直接問句，句 6 中的 *wh* 詞應為存在語意，句 7 中的 「誰」應為疑問語意。當然，翻譯有對有錯並不是唯一或是最好的評估方式。另一個方式是區辨 minimal pair，因此我接著詢問 ChatGPT 圖三中的兩個語句差異，ChatGPT 並無法正確辨認兩句的語意差異，它認為兩句意義相同。此外，ChatGPT 的表現有一個特殊之處，就是它面臨重複的問題時，會自動修正改變原答案 (因為假定你不滿意原本答案，才會重複提問)；因此當我如圖四重複提問時，即使 ChatGPT 改變了答案，陳述兩句的語意是不同的，但它還是錯解了「你都不吃什麼」的語意。

一張含有 文字, 螢幕擷取畫面, 軟體, 網站 的圖片

自動產生的描述

圖二：ChatGPT 的中文*wh* 詞理解任務#1

一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

自動產生的描述

圖三：ChatGPT 的中文*wh* 詞理解任務#2

一張含有 文字, 螢幕擷取畫面, 字型, 軟體 的圖片

自動產生的描述

圖四：ChatGPT 的中文*wh* 詞理解任務#3

# Proposal / Design

Articut/Loki (Wang et al. 2019)[[22]](#footnote-22) 為根據生成語言學原理建構的一套彼此相依的 NLP/NLU 智慧運算系統，本節目標是介紹這兩項智慧運算軟體的設計邏輯以及示範如何運用Articut/Loki 處理中文 *wh* 詞的解讀多樣性，本節結構如下：3.1 節先介紹 Articut 斷詞系統的設計邏輯與表現；3.2 節介紹以 Articut 斷詞結果為基礎，近一步解算句意的 Loki 語意理解引擎的設計邏輯與表現；3.3 節以存在語意為例，示範如何運用Articut/Loki 處理中文 *wh* 詞的解讀多樣性；3.4 節展示以 Articut 斷詞結果為基礎建立的 c-command 關係檢查器。3.5 節簡略說明 Articut/Loki 在其他 NLP/NLU 任務上如何表現優於傳統的 data-driven NLP 運算系統。

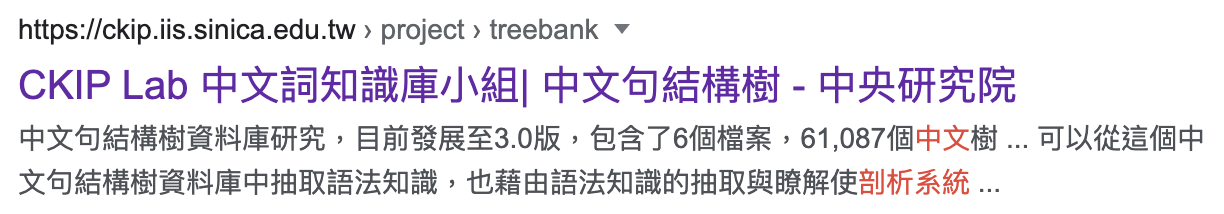
## Articut 斷詞系統

斷詞 (segmentation) 是 NLP 的基本操作第一步，若斷詞錯誤遑論後續的其他語言科技應用。舉例來說，我們希望電腦可以解算出 (39a) 的斷詞結果，但不會產出 (39b)。此外，雖然在 (39a)中，「餃子」與「包」被正確斷開，但是我們不希望電腦將此成功經驗類比延續至 (40)，因為此處的「麵包」不應該如 (40b) 被斷開。

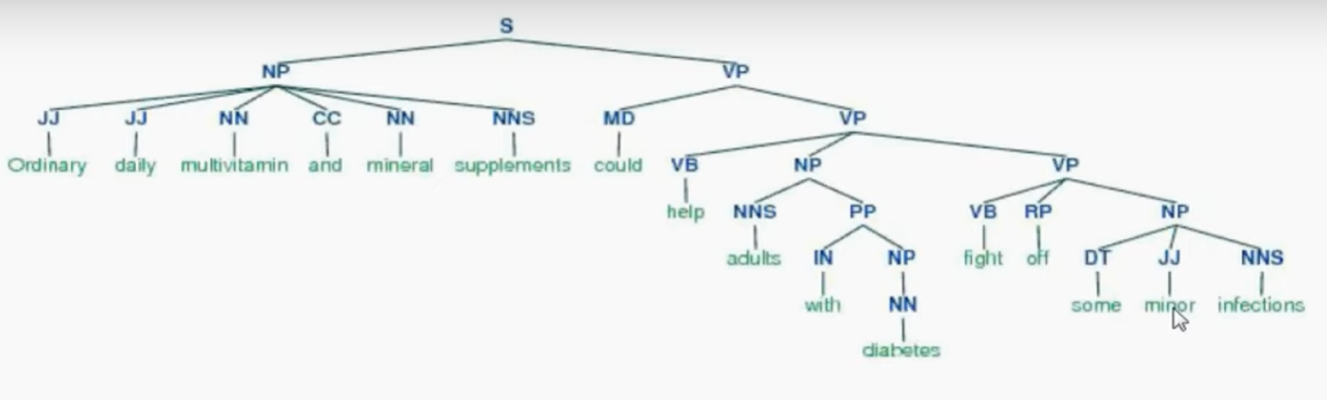
1. ) 餃子包高麗菜：
   1. 餃子 / 包 / 高麗菜
   2. # 餃子包 / 高麗菜
2. ) 麵包牛奶：
   1. 麵包 / 牛奶
   2. # 麵 / 包 / 牛奶

面對這個基礎的斷詞任務，Articut 與其他市面上的斷詞系統 (如 Jieba 與 CKIP tagger) 最大的差異在於後者是用大量的字典/語料庫資料去訓練模型且結合 1950 年代生成語言學剛起步的結構樹概念，導致此類的 data-driven NLP 資料庫中承載大量的斷詞結構樹，舉例來說，CKIP Lab 的中文結構樹就高達 61,087 棵：

1. ) CKIP Lab 的 61,087 棵結構樹 (treebank.sinica.edu.tw)：(查詢日期2022/09/30)



在資工 NLP 領域，也經常看到如同圖五，一個節點多分岔的語言結構樹圖。



圖五：資工領域的結構樹圖範例 (from 史丹佛 CoreNLP)

事實上，在 1960~1970 年代，就是這種無章法，一個節點多個分支的結構以及其他類似的語言現象描述規則，讓語言學很難進入「專家系統」的實作裡，讓資訊科學學者認為運用「(語言學)專家系統」設計的類神經網路 NLP 計算程式在維護上有很大的困難，而且不同的專家繪製的結構樹圖以及撰寫的主觀描述規則又可能會互相衝突，因此在人力與時間成本考量下，語言學的領域專業知識就被資訊科學棄用至今日。

然而，現代的生成語言學，在X-bar 詞組結構 (42) (Jackendoff 1974, 1977) 提出以後就已棄用了繁瑣的詞組語法 (Phrase Structure Grammar) 以及變形律 (transformational rules)，且不用二元分叉樹以外的結構，不會有一個結點下轄 6 個分支這樣的結構 (如圖五中的主語名詞組)。

1. ) 生成句法學家的 X-bar 結構樹藍圖：

XP

X

X

根據 X-bar 詞組結構設計的Articut 斷詞系統有三大亮點，第一，不需要在資料庫中儲存大量的結構樹去進行斷詞對比[[23]](#footnote-23)，只需要一棵結構樹藍圖即可處理文字斷詞 (=(43a))，由於沒有統計機率模型，就沒有標註資料的人力/時間成本需求。第二，由於 X-bar 詞組結構以一貫之處理所有種類詞組的精神，在使用 X-bar 藍圖去進行斷詞的同時，詞性 (part of speech, POS) 也可同步判定 (=(43b))，而data-driven NLP 的斷詞與 POS 判定需要分開兩次訓練。第三， data-driven 的斷詞系統在蒐集完資料以後的一個關鍵預先處理為「去除停用詞 (stop words)」，停用詞主要項目為高頻的功能詞 (functional words)，如*the*/*a*/*an* , *is*/*am*/*are*, *in*/*on*/*at* 等等，data-driven NLP 工程師認為此類功能詞出現頻率極高但又沒有什麼實際含義，因此在斷詞預先處理步驟中就會選擇將此類高頻功能詞列為停用詞進行刪除後，再執行斷詞[[24]](#footnote-24)。反之，此類高頻功能詞在Articut斷詞系統扮演著關鍵的角色，由於 Articut 並不使用 data-driven NLP 的字典、語料庫或語言模型，斷詞時依 X-bar 句法樹的藍圖以 top-down 的方式先尋找句中的功能詞，將其視為功能核心語 (functional heads)，並在每個功能核心語節點做上下推演 (=(43c))。推演時，依現代漢語傾向雙音節結構以及構詞中心語的考量，將上下一定範圍內之詞彙字串送入不同功能詞函式處理。函式內帶有該層功能詞列表，若該輸入詞彙字串不含有該層功能詞列表中的元素，則略過該層計算。依此分層執行斷詞的同時，也能依其在句法樹藍圖上的節點位置，加上可能 POS 標記[[25]](#footnote-25)。

1. ) Articut 斷詞系統的亮點：
   1. 使用 X-bar 結構判定，因此不需要大數據做模型訓練。
   2. 根據 X-bar 結構，斷詞與詞性 (POS) 可同步判定。
   3. 根據 X-bar 結構，功能詞在斷詞程序上有關鍵作用，不再需要被列為停用詞。

除了運用 X-bar 藍圖高效節能正確斷詞與詞性判定，Articut 為了流線性處理 (streamline) 後續 NLU 任務，在斷詞結果中反映了 Pustejovsky (1991) 關於事件結構語意研究的洞見[[26]](#footnote-26)。以下簡略說明其學理根據。語意研究裡，四種事件 (state, activity, achievement, accomplishment) 可被分成 initial, internal，和 final 三個內部結構；Pustejovsky (1991) 的研究裡說明，事件結構同時需要考察 lexicon 和 syntax 兩層的資訊，若要在語言科技裡滿足這個需求，就表示最基本的中文斷詞處理裡，就至少需要能提供兩種不同，但又彼此以「構詞-句法」關係互為因果的輸出，即「若一詞組 X 為 YZ 兩個詞彙元素 (字/詞/詞綴) 結成而言，我們亦可推論 YZ 兩個詞彙元素 (字/詞/詞綴) 可結合成詞組 X」(即XYZ)。此一要求顯然和中國流行的 NLP 斷詞工具中的「粗粒度 (輸出詞較長，例如「機器學習」) 和「細粒度 (輸出詞較短)，例如「機器/學習」) 的工作原理不同。粗粒度與細粒度的差別，乃在統計、機率模型中「兩個詞彙共現的頻率閥值」有兩個。當閥值較高時，容易產生粗粒度的結果，例如「機器學習」；而閥值較低時，容易產生細粒度的結果，例如「機器/學習」。然而，這仍然是理論上的推估結果，實際應用時更常出現的處理結果反而是「機器/學/習」這種過度細粒的結果。

由此可見，雖然表面上似乎「詞組 (word phrase)」與「詞彙 (word)」的斷詞需求，和「粗粒度 (詞長度較長)」與「細粒度 (詞長度較短)」很類似，但因其運作原理的差異，使得中國諸多流行的 NLP 斷詞工具無法滿足計算事件結構所需的「同時考察 lexicon 和 syntax 兩層資訊」的條件。

此外，我國中研院的詞庫小組依 Google 提供的工具及 Stanford 大學 CoreNLP 的研究方法所設計的 CKIP 中文斷詞系統，「僅有一種」斷詞輸出。考察其結果，很難一言以蔽之該處理結果究竟屬於詞組或是詞彙。例如，在 CKIP CoreNLP 中文斷詞系統中，我們可以得到 (44) 的處理結果：

1. ) CKIP CoreNLP 中文斷詞處理結果無法反映 activity 與 accomplishment 的差異：
   1. 張三(Nb) **愛(VL) 著(Di)** 那(Nep) 個(Nf) 女生(Na)
   2. 張三(Nb) **愛(VL) 過(Di)** 那(Nep) 個(Nf) 女生(Na)

(44) 對於後續 NLU 任務的問題在於只取其動詞 (VL)「愛」，並不足以區辨 (44a) 為一進行中 (activity) 的事件，而 (44b) 則為具有結束時間點的 (accomplishment) 的事件。再者，我們在 CKIP CoreNLP 中文斷詞系統中還可得到 (45) 的處理結果：

1. ) CKIP CoreNLP 中文斷詞處理結果無法反映 activity 與 achievement 的差異：
   1. 張三(Nb) 剛剛(D) **走路(VA) 去(VCL)** 公司(Nc)
   2. 張三(Nb) 剛剛(D) **走路到(VCL)** 公司(Nc)

其中 (45a) 為一動態事件 (activity)，(45b) 為一具有結束點 (achievement) 的事件。但在兩個句子裡，關鍵的「去」和「到」，卻似乎分別呈現了細粒度的「詞彙」(45a) 和粗粒度的「詞組」(45b) 處理。如此不一致的結果，使得自動化程式的撰寫成為不可能，也因此阻斷了大規模考察的可能性。

而本研究計劃採用的 Articut，本身即具有詞彙結果與詞組結果兩種選項。前述 (44) 中的兩個句子，將分別處理為 (46) 與 (47)：

1. ) Articut詞彙斷詞結果：
   1. 張三(ENTITY\_person) 愛(ACTION\_verb) 著(ASPECT)

那(FUNC\_determiner) 個(ENTITY\_classifier) 女生(ENTITY\_noun)

* 1. 張三(ENTITY\_person) 愛(ACTION\_verb) 過(ASPECT)

那(FUNC\_determiner) 個(ENTITY\_classifier) 女生(ENTITY\_noun)

1. ) Articut詞組斷詞結果：
   1. 張三(ENTITY\_person) **愛著(VerbP)** 那個(ENTITY\_DetPhrase) 女生(ENTITY\_noun)
   2. 張三(ENTITY\_person) **愛過(VerbP)** 那個(ENTITY\_DetPhrase) 女生(ENTITY\_noun)

透過比對詞彙斷詞結果 (46) 和詞組斷詞結果 (47)，即可確認本句涉及一事件結構。再接著取詞彙斷詞結果 (46) 中的 ASPECT 標記，即可進行事件分類與 internal 和 final 的內部事件結構之分析。

此外，(45) 的句子，透過 Articut 的處理，可得以下結構一致之輸出：

1. ) 詞組斷詞結果：
   1. 張三(ENTITY\_person) 剛剛(MODIFIER) **走路(ACTION\_verb) 去(ACTION\_verb)**

公司(ENTITY\_noun)

* 1. 張三(ENTITY\_person) 剛剛(MODIFIER) **走路(ACTION\_verb) 到(ACTION\_verb)**

公司(ENTITY\_noun)

(48a/b) 和(45a/b)相比，明顯具有輸出結果的一致性。如此一來，透過比對「走路(ACTION\_verb)」此一無界動詞 (atelic) 其後的搭配詞為「去(ACTION\_verb)」或「到 (ACTION\_verb)」，即能進行事件結構的分類。得到 (48a) 具有 initial 結構 的動態事件 (activity)，而 (48b) 為具有 final 結構的 Achievement 事件。

因此，就「大規模考察」的目標而言，一個無法提供穩定且一致的「構詞-句法」結構對比的系統，是無法程式化進行自動處理的。以粗粒度/細粒度為輸出結果的中國各流行 NLP 中文斷詞工具以及中研院的 CKIP CoreNLP 中文斷詞系統皆無法滿足前述要求。因此本研究採用 Articut 這套可滿足前述需求以實現自動化大規模考察目標的系統。[[27]](#footnote-27),[[28]](#footnote-28)

## Loki 語意理解引擎

延續 Articut 的斷詞結果，後續的 NLU 任務就轉接到 Loki 語意理解引擎。然而，如 2.2 節中所述，不論是基於字典、基於統計資料模型或是基於機器學習模型，皆可視為是「資料驅動 (data-driven)」的方法。而資料驅動的運算系統，在 NLU 的任務處理上，頂多只解決了詞彙語意層面的問題，無法繼續向上堆疊至句法甚至語意層面以解決實際的需求；諸多研究也往往只能計算各種詞頻，而無法呈現語言上下文中的不同事件之間的因果關係。相反地，透過語言學原理驅動的 (rule-driven) 的 Articut/Loki 系統[[29]](#footnote-29)，從生成語言學的角度面對自然語言，何為詞彙不是一個問題，「結構與語意何為」才是重點；根據此核心精神 (即 Berent & Marcus 2019 提及的假設 (1)) 設計的 rule-driven Articut/Loki，為自底層即基於生成語言學原理運作的兩套彼此相依的系統。因為考量了言談分析 (Nunan 1993) 以及形式語用學 (Potts 2010) 裡的語境變數，因此 Loki 的系統架構具備「語境 (context)」、「意圖 (intent)」和「語句 (utterance)」三個層次的語意參數，對應在某場合/語境 (context) 下，為了表達某個意圖 (intent)，會說什麼話 (utterance)。以下示範 Loki 如何運用此三層次語意解算系統，分析中文含有 *wh* 詞的語句，運用 Loki 處理中文語句的系統擷圖如圖六。Loki 可偵測語句內容決定是否要在含有 *wh* 詞的上下文語境 (context)中進行該句的語意解算，而在此語境詮釋條件下的語句中的 *wh* 詞有三種詮釋可能 (intent)，系統中也註記三種詮釋各自有數種可能句型 (utterance)；如此一來，即可透過 Loki 系統基於句型進行分類的系統架構，將語言學理論實際套用於 NLP/NLU 的應用場景中，並幫助語言科技發展各種細膩的人機互動技術。

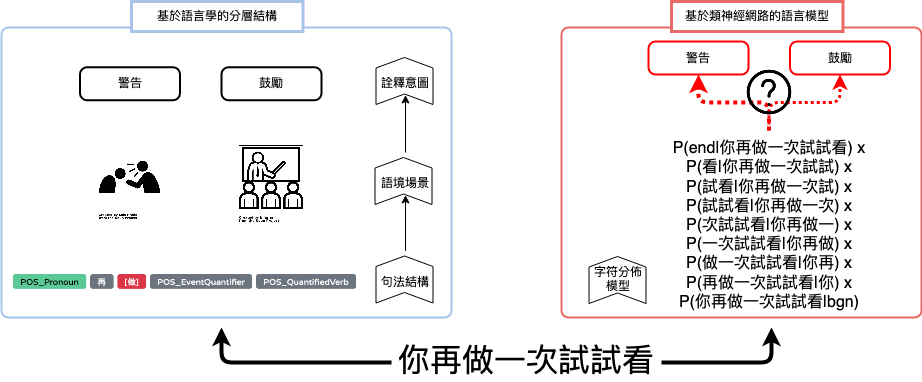
一張含有 文字 的圖片

自動產生的描述

圖六：Loki 的語意理解引擎

根據以上的邏輯與流程說明，本計劃想強調的是，由於 Articut/Loki 脫離了「詞彙標記」的思維，而改以句法結構、語意/語用場景做為語言系統的運算操作符號，故能提供句法層甚至語意層面的解算空間，以滿足實際社會的自然語言處理應用需求 (NLP Applications)，可從根本上避免了諸如聊天機器人等人機互動自然語言科技一收到句子中含有「什麼」即誤以為「這是一個疑問句」(參見 CKIP CoreNLP 處理的結果 (30)/(31))，而開始答非所問的落地應用問題。

除了句法層和語意層以外，Loki 亦是目前唯一提供「語境/場景」操作空間的 NLP 系統。以語句「你再做一次試試看」為例，這個句子既可能表示「鼓勵」意圖，也可能表示「警告」意圖。究竟該採用哪一種語意，完全視語境而定。假設這個語句發生於「家教老師對學生」，那麼偏向「鼓勵」的可能性較高。相對地，如果是「家長對子女」，則「警告」的意味就相當濃厚了。這兩種語意之間的差異，不在語句「你再做一次試試看」的任何一個字符、詞彙甚至是句法結構裡，而是由說話方與受話方彼此的社會關係決定 (見圖七的對比)[[30]](#footnote-30)，Loki 系統內為滿足此一「語境會影響語意詮釋」的計算需求，除將「親屬」和「非親屬」的人物標記分為 pronoun 和 noun 以外，更有讓開發者可自行設定與操作的空間[[31]](#footnote-31)。



圖七：基於語言學的分層結構和基於類神經網路以字符分佈做為模型依據的結果

綜而言之，Articut/Loki 的表現代表 NLP/NLU 研究工具與學理上的突破，藉由語言學原理驅動的系統，本研究首先能突破 2.2 節中提及前三種方法皆受制於「詞彙資料」，而只能計算詞頻做為分類依據的限制；再者，學理研究上，本研究藉由導入新的研究工具 (rule-driven NLP/NLU)，將成為近年來第一個考察語言中大量事件結構之間的因果關係之嘗試。

## Articut/Loki 的運作細節：以全稱語意為例

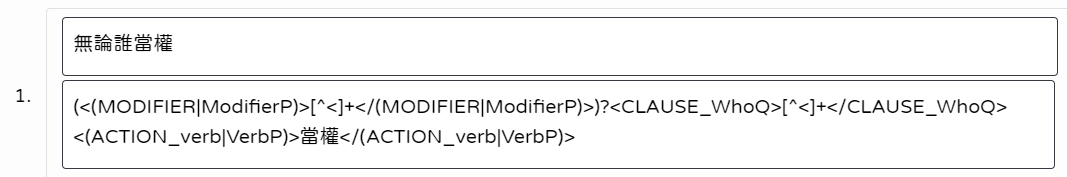
以下用全稱語意 (程式中註記為universal\_use) 為例說明運用 Articut/Loki 處理中文 *wh* 詞詮釋的流程。首先，我們收集中研院平衡語料庫中 (<https://asbc.iis.sinica.edu.tw/>) 所有包含「誰」的語料，總計2979 筆。經語料預處理過後抽取出含有「誰」的語料為 2695 筆。此處給定之例句為「無論誰當權」。系統在Articut將其表層解析為 (49)，其結構在(50) 以 regex 表示式呈現。而在系統後台，則將該例句儲存為 (51) 以符合 lambda 邏輯式呈現的真值條件。

1. ) 無論/誰/當權
2. ) <MODIFIER>無論</MODIFIER><CLAUSE\_WhoQ>誰</CLAUSE\_WhoQ>

<ACTION\_verb> 當權</ACTION\_verb>

1. ) ∀x (無論 (CLAUSE\_WhoQ(x)🡪 當權(x))

而Loki同樣使用 regex 表示式的概念，而使同一結構的句型能夠被判讀。系統初始判斷此句的結構如圖八所示:



圖八：「無論誰當權」的結構呈現與語意解式邏輯式

其中「|」符號意指 "or", 以圖八出現的 "<(MODIFIER|ModifierP)>" 為例，表句中可出現MODIFIER 或是 ModifierP。

而 「[^<]+」 符號表任意字詞，以 <(MODIFIER|ModifierP)>[^<]+

</(MODIFIER|ModifierP)> 為例，「[^<]+」 置於詞性標記中間，意即只要句中出現符合MODIFIER 或是 ModifierP 的任意字詞，此句型都會將其判讀。

再來是將帶有詞性標記的字串置於括弧中，並於後括弧處附加「?」符號，以 (<(MODIFIER|ModifierP)>[^<]+</(MODIFIER|ModifierP)>)? 為例，意指整段帶有詞性標記的字串為可有可無。以此句為例，也就是句子中的MODIFIER或是ModifierP不一定需要出現。

同時，為使「無論誰當權」此句結構能夠符合本研究主題，需要限縮此 regex 表示式，並同時達到正確判讀同一類的句型之目的。需要修改 regex 表示式的內容，加上原句「無論誰當權」用詞的同義詞，並且刪減可有可無的結構，形成最終圖九的 regex 表示式



圖九：限縮「無論誰當權」的結構呈現與語意解式邏輯式

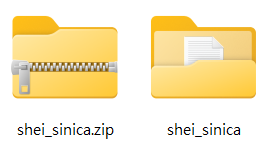
其中 (<FUNC\_inter>不論</FUNC\_inter>|<MODIFIER>無論</MODIFIER>)，「不論」和「無論」兩個帶詞性標記字串，中間的 「|」 符號意指 "or"，也就是句中可以出現「不論」或者「無論」，而此段 regex 表示式中加上的一段 (<AUX>是</AUX>)?，意指助詞「是」不一定需要出現。而原本的 <CLAUSE\_WhoQ>[^<]+</CLAUSE\_WhoQ> 意指所有符合 <CLAUSE\_WhoQ> 詞性的任意字詞都會被判讀，在這裡為符合本研究關於中文 *wh* 詞「誰」的主題，將詞性 <CLAUSE\_WhoQ> 之間由任意字詞 「[^<]+」 改為僅限字串「誰」。最後，動詞「當權」並不是本句型必要出現的動詞，故在此regex表示式將其刪減。

給定例句以後，系統自動設定完成表層的結構呈現以及深層的語意解釋邏輯式。此時點擊頁面左上角的 Home 圖示，回到最上層，選擇下載系統自動生成以 Python 程式語言執行的範本。



圖十：下載「無論誰當權」Python程式語言執行範本

範本下載後為一 zip 格式的檔案如圖十一，將之解壓縮得到一與專案名稱同名的 shei\_sinica目錄：



圖十一：解壓縮「無論誰當權」Python程式語言執行範本

進入shei\_sinica目錄，可見一意圖設定目錄、一帳號資訊檔及一主程式。

以句子「無論誰當權」為例，對於此句的處理機制，系統在兩種模式下分別執行。聊天機器人模式 (60 行的 CHATBOT\_MODE) 和非聊天機器人模式 (62 行的 else:)，系統更進一步處理該句子。此研究非重視回覆多樣性之聊天機器人，故使用非聊天機器人模式 (CHATBOT\_MODE = False)

首先，以句子「無論誰當權」為例，在 63 行將此句子加入通用意圖列表 "universal" 中，以表達系統辨識到該句符合此語意的結構。其次，系統以前文示例之修正過的結構 (<FUNC\_inter>不論</FUNC\_inter>|<MODIFIER>無論</MODIFIER>)(<AUX>是</AUX>)?<CLAUSE\_WhoQ>誰</CLAUSE\_WhoQ> 為依據，更新 "intent" 列表，將 "c3" 加入其中。這裡 "c3" 為自定義編號，"c" 表徵全稱語意，"3" 表示全稱語義中的其中一個句子的結構 (pattern)。並且 64 行出現的 "a1", "a5" 和 "a9" 也是自定義編號，"a" 表示疑問語意，"a1", "a5" 和 "a9" 各對應到不同包含強疑問詞的句子結構，分別為以下所示:

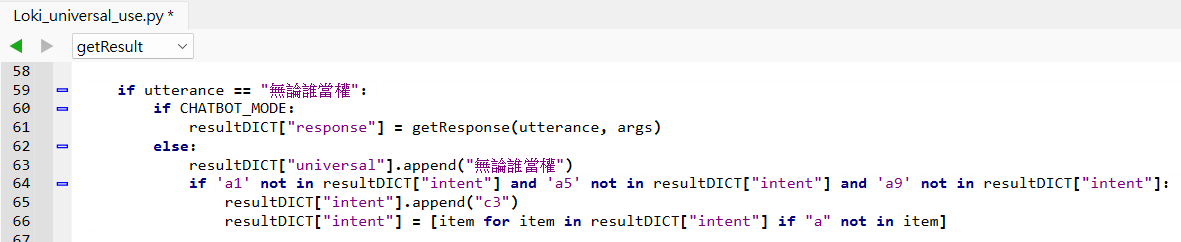
1. ) 強疑問詞結構：

a1: 誰+sentence-final interrogative particle

a5: 還有+誰

a9: 誰+ conjunction + noun or 誰

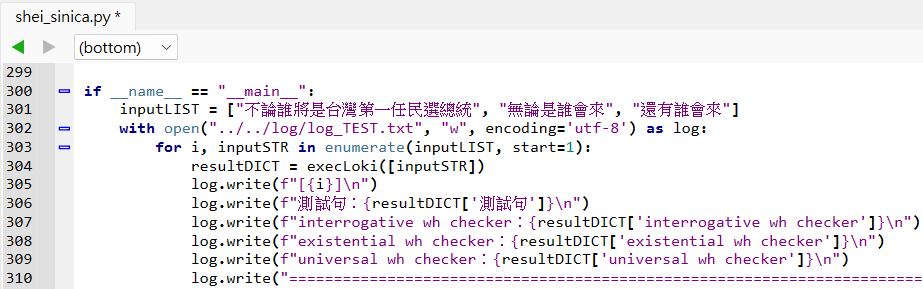
(52) 的結構被列為強疑問詞，也就是當以上結構和全稱語義的核可詞同時出現在句子時，檢查器會將其判斷為疑問語意，如圖十二的第 64 和 66 行所示。本研究預計將在此區塊將生成語言學理論實作為程式邏輯以進行大量測試理論完備性。



圖十二：「無論誰當權」Python程式語言執行範本

接著在主程式中填入待測試用的句子，此處以 (53) 三句為例做說明，測試結果如圖十三所示[[32]](#footnote-32)。系統執行後，如 304 行起所示，在 301 行將每個測試的句子逐一交由 execLoki() 執行，並在302行中名為log\_TEST.txt的檔案中寫入進行測驗結果，計算以判讀是否符合前述的已預先設定之結構。

1. ) 測試句：
   1. 不論誰將是台灣第一任民選總統
   2. 無論是誰會來
   3. 還有誰會來



圖十二：(53) 測試句檢驗截圖

再以「不論誰將是台灣第一任民選總統」為例，其結構在圖十三以regex表示式呈現。該句表層結構包含給定例句「無論誰當權」的表層結構。圖十三中的「將」、「台灣」、「第一任」、「民選」和「總統」以句子結構的判讀需求來說，皆為可有可無之「修飾詞」和可替換而不影響結構之「名詞」，故不影響表層結構之判斷。



圖十三：「不論誰將是台灣第一任民選總統」的結構呈現

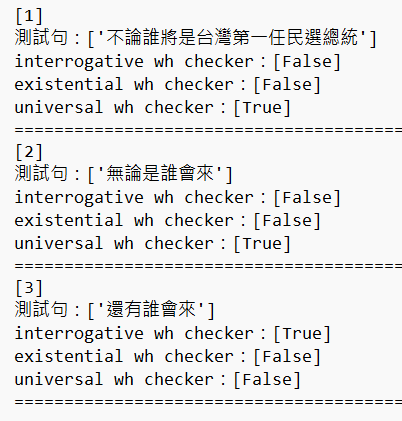
當表層結構中含有預給定的例句結構後，系統接著進入深層結構之轉譯。以「不論誰將是台灣第一任民選總統」為例，將其表層結構相符指定例句之部份轉為深層結構如(54)，由於系統將「修飾詞」和可替換而不影響結構之「名詞」判讀為皆為可有可無的函式，故可將其簡化為(55)。

1. ) ∀x∃y 不論(CLAUSE\_Who(x)) & 總統(y) -> 將(是(y, x) & 民選(y) & 台灣第一任(y))
2. ) ∀x (不論(CLAUSE\_WhoQ(x)🡪 是(x))

(55) 除了助動詞「是」之外，與前述之(51)相容。加上在圖九做的更動調整，在「不論」後加入可有可無的助動詞「是」，並且不一定需要動詞的出現，以符合更多相似結構，成為 (56)，以表示該輸入句「不論誰將是台灣第一任民選總統」結構與指定的 utterance「無論誰當權」相符。

1. ) if utterance == "無論誰當權":

如此反覆將每個要測驗的句子輸入程式中，即可如圖十四所示得出 (53) 之中每個句子依理論邏輯的計算結果是否相符於人類直觀的判斷結果 (如「是否呈現全稱語意」)，並依此檢驗理論邏輯上之完備性。



圖十四：log\_TEST.txt檔案內(53) 測試句的檢驗結果

## C-command 關係檢查器的程式實作

根據第 2 節的觀察，c-command關係的檢查對於中文 *wh* 詞的語意理解工程是一個關鍵的功能。然而，c-command 從 Reinhart (1976) 即提出，但似乎並沒有成為任何標準 NLP 模組或工具中的固定功能，亞歷桑那大學 Sandiway Fong 教授的[課堂簡報](https://sandiway.arizona.edu/ling581-22/lecture18.pdf) ([lecture18 (arizona.edu)](https://sandiway.arizona.edu/ling581-22/lecture18.pdf)) 裡利用 NLTK 這個略顯古老但廣為流傳的 NLP 工具實作了一個示範的簡化版本，但其中的句法樹是手動鍵入的教學用範例，而非取用語料自動生成的真實語料。因此無法直接套用於本研究所蒐集的語料。

要程式化檢驗 c-command 關係需要先有採用語法結構理論以及句法剖析器 (syntactic parser)，然而也正因為各機構採用的語法結構理論不同，故一直缺乏一可嚴格遵守 X-bar 語法結構理論的剖析器。以來自史丹佛 CoreNLP的圖二為例，其單一節點多分支的結構，即無法呈現 X-bar 語法結構理論的上下層關係。再者，根據中研院 CKIP CoreNLP 的呈現的圖十五，即可看出無法妥善處理反身代名詞「自己」依據 c-command 幾何關係[[33]](#footnote-33)，處理其指代消解的問題。因此，本計劃無法採用現有產出如圖二/圖十五的句法剖析器。

一張含有 黑色, 黑暗 的圖片

自動產生的描述

圖十五：CKIP CoreNLP 的結構圖

以 Articut 斷詞系統為基礎，本計劃無需自行開發新的句法剖析器，而是利用自 Jackendoff (1976) 提出的X-bar 結構原型，不斷修正迭代至今的詞組結構原則，搭配 regex 以實現 c-command 的檢驗。在本計劃草擬階段即已進行初步的測試，確認其可行性。開發中的 c-command 關係檢驗程式執行畫面如圖十六，在程式中編寫的 regex 如圖十七，其運作原理，以圖像化顯示其基於 X-bar 的剖析器結果如圖十八。

一張含有 文字, 軟體, 多媒體軟體, 字型 的圖片

自動產生的描述

圖十六：C-command 關係檢驗程式執行劃面

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

自動產生的描述

圖十七：C-command 關係檢驗程式 regex

一張含有 文字, 圖表, 行, 繪圖 的圖片

自動產生的描述

圖十八：C-command 關係檢驗程式之圖像化顯示

## Articut/Loki 的效率、效能與準確率

在前面的小節中，我說明了Articut/Loki 具備語句斷詞/詮釋的設計架構與擴充潛力，本節我簡略說明 Articut/Loki 在其他 NLP/NLU 任務上如何表現優於傳統的 data-driven NLP 運算系統。第一點突破為 NLP/NLU 工具效率與效能的突破，基於資料的三種方法的本質導致需要海量訓練資料集[[34]](#footnote-34)，才能得到較佳的處理結果。但資料愈大，運作所需的時間與效能成本越高，如此則難以在面臨實際社會產業需求時，提供不具 Google 運作規模的企業公司一個符合成本效益的解決方案，因任何業界領域的資料相較於模型本身，都是「小數據」而非大數據。以目前已上線的大數據機器學習解決方案 GPT-3 為例，其訓練資料集的規模為 45TB 不同來源的文字資料，相較之下，國內排名前三的媒體巨人聯合報，一年能產生的文字資料量只有 1GB。換言之，GPT-3 需要的訓練資料量約為 5 萬年份的聯合報資料量。依此數量為參考點，沒有任何一家產業內的頂尖企業能將自己的資料套用在機器學習方法中，並期待能取得 GPT-3 的成果。頂多只能以 GPT-3 做為預訓練模型 (pretrained model)，再加上自己的資料做為細調校訓練資料 (fine-tune) 以便應用機器學習方法在實際社會產業需求上。然而，基於語言學原理驅動的 Articut/Loki 系統不需要這麼多訓練資料，以玉山銀行 2020 年舉辦的 [NLP 黑客松挑戰](https://tbrain.trendmicro.com.tw/Competitions/Details/11)為例，相較於其它隊伍苦於 5000 篇訓練文本仍無法產出有效模型的時候，基於 Articut/Loki 系統的方案只使用了 146 篇便能產出 F1-Score 為 92.84% 的驚人成效[[35]](#footnote-35)。相較之下，底層即基於生成語言學原理運作的 Articut/Loki 更能讓語言學研究獲得轉化 (spin-off) 為業界實際落地應用的機會，並在極低的資料成本條件下，解決實際社會產業需求[[36]](#footnote-36)。

第二，有效驗證 NLP/NLU 程式理解語言意義能力的一個方式為測試程式是否能夠正確解答如 (56a) 的Math Word Problem (MWP，數學應用問題，又稱 Quantitative reasoning problem)：

1. ) Math Word Problem:
   1. 數學應用問題：火車 **48** 小時行駛 **5920** 千米，汽車 **25** 小時行駛 **2250** 千米，

汽車平均每小時比火車每小時慢多少千米?

* 1. 正解數學式：2250 ÷ 25 − 5920 ÷ 48

一方面，解題器需要可以正確理解應用問題中每個句子所描述的語意為何，重點是什麼，可計算的數量各自是哪個而題目的求解目標又是什麼；另一方面，解題器也需要有適當的數學求解能力，將取得的數字以加、減、乘或除等方式加以處理。Wang et al. (2020) (<https://aclanthology.org/2020.rocling-1.21.pdf>) 運用Articut/Loki 系統展示以「語言學理論」為基礎用「非機率模型」建立的數學應用問題作答系統只用了 464 個句子 (約 150 題)，即能達到 98.57% 的解題正確率；這個解題正確率的表現優於中國騰訊公司在 2017 使用 的 LSTM 架構[[37]](#footnote-37) (Wang et al. 2017)，該研究使用 23,000 題數學應用問題訓練中文的數學題庫的 MWP 模型後，達到 58% 的解題正確率；此表現亦高於 Chiang & Chen (2019) 所提出的機器學習方法成果的 65.8% 解題正確率 (<https://arxiv.org/pdf/1811.00720.pdf>)。綜合前述，我們可以觀察到從 2012 年大數據 (Big Data) 興起以後，隨之而起的機器學習在面對需要精細理解語意的數學應用問題時，雖 Wang et al. (2017) 與Chiang & Chen (2019) 皆嘗試以機率模型挑戰此困難，但仍無法妥善處理，直至 2020 年國內出現第一次以語言學原理設計的 NLP/NLU 系統才開始有所突破。

# Tentative plan and possible challenges

本計劃將運用 Articut/Loki 智慧運算系統處理中文 *wh* 詞的解讀多樣性。根據第 2 節的討論，我們期待 Articut/Loki 可以處理以下實際語言現象[[38]](#footnote-38)：

1. ) 本計劃對中文NLP/NLU智慧運算程式的期待 (summary)：
   1. 區辨直接與間接 *wh* 問句
   2. 在不同的語境中，區辨 *wh* 詞的疑問與存在語意
   3. 判斷核可詞與 EPW 之間的結構 c-command 關係，而不只是線性前後次序
   4. 判定疑問核可詞與 EPW 核可詞的優先順序
   5. 可判別全稱量化語意的 *wh* 詞需與「都」出現在同個句子中，且需要出現在「都」的左側 (the Leftness Condition)。
   6. 允許 *wh* 詞的全稱量化語意出現在孤島中。
   7. 判別以下兩種狀況：當兩個 *wh* 詞與「都」在同個句子中時，且都 c-command 「都」，則只有結構距離較近 (structural minimality) 的 *wh* 詞可到全稱量化語意；若兩個 *wh* 詞皆沒有 c-command 「都」，那兩者皆可獲得全稱量化語意。
   8. 判定疑問、全稱、存在語意的優先順序。

在進入本計劃期程細節前，我必須說明本計劃中「語言理解」的含義。一個 NLP 智慧運算軟體的「語言理解」有兩個層次的定義以及評估方式，要建構一個真正理解人類語言的 NLP 模型，兩個層次都對，也都是必須的。第一個層次是「top-down 的對話能力」，就是在給定的問答系統中，該 NLP 程式可以給出適當的回應或是通過題庫的抽樣測試 (達到一定分數)，放在日常生活的情境中，就是一個服務性機器人會接受基本指令，可對談互動，會發問或提出建議。此理解能力的表現仰賴於統計性知識與訓練。而本計劃定義中的「理解」是「bottom-up 根據詞組結構合成的語意」 (可參考 2020美國計算語言學會年會最佳主題論文 Bender & Koller 2020 對於 NLU 的闡釋)，指透過表層結構與深層結構之間的真值條件，藉由逐步消除歧義的可解釋性邏輯，使系統不僅是判斷結果和人類一致，而其推算過程亦逐步可考，此時方為知其然，亦知其所以然的理解。此理解能力的表現仰賴於模組化的結構知識 (構詞、句法、語意真值邏輯式)。根據以上兩個必要層次的定義，本研究計劃的目標並不是推翻現行主流的作法與判斷標準，而是要邁向混成式的 NLP 建構方式 (hybrid approach)，並針對 bottom-up 作法的概念、流程與實證應用提出更精確的細節。藉由本計劃的執行，可以產出 bottom-up作法的成果與概念性驗證 (Proof of Concept, POC)，方可與已有豐碩成果的 top-down作法銜接，邁向真正理解人類語言的 hybrid NLP model。

為逐步打造可以捕捉 (57) 的NLP/NLU 系統，本計劃的三年期程細節如下表：

* Tentative plan (from 08/2024-07/2027)

|  |  |
| --- | --- |
| Time | Plan |
| * 1. – 12.2024 | 1. 訓練研究助理熟悉計劃內容與研究問題。 2. 撰寫語料格式預處理與資料清洗程式。 3. 撰寫語料蒐集的爬蟲程式。 |
| * 1. – 07.2025 | 1. 針對*wh*詞三種語意出現的環境，進行初步分類與 regex 建模。 2. 測試初版 regex 模型是否能正確區辨*wh*詞三種語意。 3. 運用 Articut/Loki 建模，撰寫「誰」全稱語意的判斷程式。 |
| 08.2025 – 12.2025 | 1. 持續優化regex 模型。 2. 運用 Articut/Loki 建模，撰寫「什麼」全稱語意的判斷程式。 |
| 01.2026 – 07.2026 | 1. 優化 c-command 關係檢查器，利用本研究至今累積的語料和程式、語言模型與主流 NLP 只採計線性前後次序而建出的各種大型語言模型進行比較。 2. 優化 c-command關係檢查器並利用爬蟲程式蒐集的資料驗證程式與假設。 3. 考察所蒐集之語料和設計之程式、語言模型是否皆可正確解算多個 *wh* 詞搭配同一個被 c-command 的全稱量化詞 (都) 時，唯有維持最小結構距離 (structural minimality) 的 *wh* 詞可獲得全稱量化語意。且若兩者皆不 c-command 該全稱量化詞，則兩者皆可獲得預設的全稱量化語意。即和傳統理論分析看法不同的是，工程實作的切入點為「*wh* 詞的全稱量化語意不是被核可而來，而是被取消的」。 |
| 08.2026 – 12.2026 | 1. 運用 Articut/Loki 建模，撰寫「誰」存在語意的判斷程式。 2. 運用 Articut/Loki 建模，撰寫「什麼」存在語意的判斷程式。 |
| 01.2027 – 07.2027 | 1. 執行及維護爬蟲程式運作，蒐集直接與間接 *wh* 問句的語料。 2. 運用 Articut/Loki 建模，撰寫「誰」疑問語意的判斷程式。 3. 運用 Articut/Loki 建模，撰寫「什麼」疑問語意的判斷程式。 4. 確認三種語意判斷程式可以判定疑問、全稱、存在語意的優先順序。 5. 將累積至今的研究成果、所蒐集的語料整理為線上開放原始碼的資源，除做為跨域整合的成果，亦提供各界 NLP 對細膩 NLP 語意解算需求一個基於語言學原理，且經過長時間，大量語料驗証的資訊工具。 6. 將累積至今的研究成果，撰寫並打包為符合 pipy (Python 第三方軟體套件的軟體存儲庫) 規格的模組工具包以發佈予全球 Python 程式語言使用者用於處理中文 NLP 之語意計算。 |

本計劃以明確且可逐步拆解的的詞組結構與 lambda 邏輯式語意為底，處理中文 *wh* 詞的詮釋問題；而採用此方式的誘因之一為詞組結構一旦建立完成 (明確的詞組結構寫得完，而不會像海量資料沒有上限)，則不需大量訓練語料，如此可以降低瀏覽大量訓練語料所需的人力與時間成本。此外，根據[中研院許聞廉特聘研究員針對 NLP 的演講影片](https://www.youtube.com/watch?v=lhxWE6b0Cpo&t=4105s)，「不可解釋性」不只是深度學習模型的唯一問題，由於其非線性、多層次數列交錯影響的黑盒子學習歷程，一但遇到問題時，往往無法 pinpoint 問題來源進行精確直接的修正，導致其建構出來的語言模型還會有「難以預測」的錯誤以及「知錯難改」兩大問題；而本計劃運用Articut/Loki 建立 NLP 智慧運算程式的方式立基於明確且可逐步拆解的結構/語意規則運作，因此遇到問題時，可以精準地找到問題所在，進行修正，朝向「可解釋性 AI (explainable AI, XAI)」邁進 (cf. Samek et al. 2017)。

在打造能捕捉 (57) 的 NLP/NLU 系統過程中，我們預期遇到的兩大挑戰如下。首先是**爬蟲程式的撰寫與長期維護**是一項不得不為之的苦工。爬蟲程式的任務是持續從不同的公開網站平台收集含有研究目標語料的句子做為研究材料。而一方面網站平台本就改版頻繁，每一次網站改版，爬蟲程式就需要做出相對應的程式碼修改，否則就失效了。其次是各網站伺服器為了將頻寬保留給真人用戶，通常會有各種防止爬蟲程式的設計。因此在本計劃一開始，就要先召募相關人力，投入爬蟲程式的設計，並在整個計劃期間持續維護。第二，透過和其它基於機器學習的 NLP 方案的比較，並以累積三年的大量語料做為驗證，取得較佳的語意詮釋與語意計算能力的證明後。**將成果打包為符合 pipy (Python 第三方軟體套件的軟體存儲庫) 規格的 Python 程式語言模組工具包以發佈予全球 Python 程式語言使用者**用於處理中文 *wh* 詞在 NLP 應用中之語意計算，亦有一定的技術門檻需要克服。以上兩個問題的核心在於維護爬蟲程式以及程式撰寫與發佈的時間與人力成本，相應解決方案為聘請兩位碩士兼任助理以及兩位大專生兼任助理，逐步分工處理本研究計劃的分段期程任務。

# Conclusion & future work

科學領域裡的每一個基礎學科，諸如物理、化學、生命科學等，隨著知識的累積與發現，在技術成熟時，都能產生工程實作，進而產生科技應用，解決人類文明進程中的實際需求。生成語言學的研究與辯證方法皆為科學方法，亦為認知科學的一個核心分支 (如圖十九所示)，現今的資訊技術已成熟至可以支援驗證生成語言學領域過去六十年積累的研究與發現。

一張含有 圖表, 行, 三角形, 折紙 的圖片

自動產生的描述

圖十九：認知科學核心領域

Adapted from Gardner, Howard 1985. *The Mind’s New Science: A History of Cognitive Revolution*.

New York: Basic Books, Inc.

本計劃一開始提到，人工智慧類神經網路與生成語言學的誕生皆為 1957 年，前者自 1957 年至今的發展如圖二十所示，可分為三波，第一波為 1950-1960 年的符號邏輯時代，第二波為 1980-1990 年的專家系統年代，在第二波發展中，當時的生成語言學之所以被資訊科學領域判斷為不堪使用的主因在於當時的詞組語法 (Phrase Structure Grammar) 與變形規則 (transformational rules) 被認定為龐雜寫不完，耗費人力，且不同人的主觀意識差異很大，導致不同工程師寫的規則甚至會互相衝突 (請參照註腳 29 與 38)[[39]](#footnote-39)。

一張含有 文字 的圖片

自動產生的描述

圖二十：人工智慧發展簡史與對應關係

(credit: [我讀《人工智慧在台灣》搞懂的四件事｜經理人學習人工智慧的第一本書（內有全息圖） - 職人簡報與商業思維 - Medium](https://medium.com/%E8%81%B7%E4%BA%BA%E7%B0%A1%E5%A0%B1%E8%88%87%E5%95%86%E6%A5%AD%E6%80%9D%E7%B6%AD/%E6%88%91%E8%AE%80-%E4%BA%BA%E5%B7%A5%E6%99%BA%E6%85%A7%E5%9C%A8%E5%8F%B0%E7%81%A3-%E6%90%9E%E6%87%82%E7%9A%84%E5%9B%9B%E4%BB%B6%E4%BA%8B-%E7%B6%93%E7%90%86%E4%BA%BA%E5%AD%B8%E7%BF%92%E4%BA%BA%E5%B7%A5%E6%99%BA%E6%85%A7%E7%9A%84%E7%AC%AC%E4%B8%80%E6%9C%AC%E6%9B%B8-f69eb8b81358))

本計劃的重點之一在於點出生成語言學自 Jackendoff (1974, 1977) 之後，便淘汰了詞組語法與變形規則，改採 X-bar theory 這個有普世應用性的詞組結構藍圖至今；另，中文的 *wh* 詞解讀規則 (=(57)) 也是客觀歸納的通則，不論是 X-bar theory 或是 (57) 皆非「龐雜」或「因人而異」，但此類生成語言學的研究成果在資工與人工智慧領域並未廣傳，導致生成語言學在資工與人工智慧領域背負著不必要的負面標籤 (詳見 Pater 2019 的討論)。本計畫運用以 X-bar theory 為底層邏輯設計的 Articut 針對含有 *wh* 詞的中文語句進行準確的斷詞，斷詞結果不只可以送至 Loki 進行後續語意理解參數的設計 (詳細過程如節3.3所示)，亦可作為 c-command 關係檢查器的撰寫基礎 (詳細過程如節3.4所示)。

深度學習教父級人物 Yann LeCun (卷積神經網路之父，2018 年的圖靈獎獲獎人之一，Meta 首席科學家) 一直以來都反對 Gary Marcus 支持生成語言學的論點，堅定抱持假設 (2)；有趣的是，[LeCun (2022)](https://www.technologyreview.com/2022/06/24/1054817/yann-lecun-bold-new-vision-future-ai-deep-learning-meta/) 改變了一直以來堅持的立場 ([Yann LeCun’s big bet for building intelligent machines | MIT Technology Review](https://www.technologyreview.com/2022/06/24/1054817/yann-lecun-bold-new-vision-future-ai-deep-learning-meta/))，提出概率論 (probability theory，即假設 (2)) 不應該是解釋機器學習的唯一框架，且產學界更需要重新思考目前 AI 技術面臨的問題，不能只是把海量數據丟進大型類神經網路裡，就希望系統能學會所有東西；他表示類神經網路若要克服莫拉維克悖論 (Moravec’s paradox，即人類認為困難的事，電腦可以解決，但人類視為理所當然的事 (如語言能力)，電腦卻很難克服) 的 Artificial General Intelligence (AGI)，必須相當程度地接受假設 (1) (即 Berent & Marcus (2019) 提出的第二條路)。這個震撼性言論象徵連 AI 領域都已經準備好進行相當程度的典範轉移 (paradigm shift) 了，生成語言學界也該應勢而起，給予友善的建設性回應，付諸實行 Linzen (2019) 陳述的生成語言學與類神經網路互惠願景。

以上闡述的學界/產業界現況驅動著本計劃的初衷、實證構想與執行細節。之所以選擇中文 wh 詞的解讀多樣性為計劃實證焦點是因為此題目象徵著中文生成語言學的起始 (Huang 1982; Cheng 1991; Li 1992; Tsai 1994; Lin 1996, 1998)，中文生成語言學踏入跨科際研究時代也可以此經典題目為關鍵支點展示生成語言學的研究成果，可以實作落地成為可運作、可驗證的程式碼。在以大量語料考察驗證程式碼的過程中若發現特例或是以往未曾注意過的現象，可進一步豐富生成語言學基礎研究的主題，而實作成果亦可成為實際產業需求的工具。

# References

徐嘉慧、何萬順、劉昭麟 2016，「語言學門熱門及前瞻學術研究議題調查」結案報告，科技部人文社會科學研究中心 (MOST 104-2420-H-002-016-MY3-PH10406)。

Bender, E. M. & A. Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 5185–5198, Online. Association for Computational Linguistics.

Berent, I., & G. Marcus. 2019. No integration without structured representations: Response to Pater. *Language* 95(1): e75-e86.

Berent, I., S. Pinker & J. Shimron. 1999. Default nominal inflection in Hebrew: Evidence for mental variables. *Cognition* 72: 1–44.

Bloomfield, L. 1939. *Linguistic Aspects of Science*. Chicago: University of Chicago Press.

Bybee, J & James L. McClelland. 2005. Alternatives to the combinatorial paradigm of linguistic theory based on domain general principles of human cognition. *The Linguistic Review* 22: 381–410.

Cheng, L. L.-S. 1991. *On the Typology of Wh-questions*. Doctoral Dissertation. MIT, Cambridge, MA (published by Garland, New York, 1997).

Cheng, L. L.-S. 1995. On *dou*-quantification. *Journal of East Asian Linguistics* 4: 197–234.

Cheng, L.L.S. & C.-T. J. Huang. 1996. Two types of donkey sentences. *Natural Language Semantics* 4: 121–163.

Chiang, T.-R. & Y.-N. Chen. 2019. Semantically-Aligned Equation Generation for Solving and Reasoning Math Word Problems. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2656–2668, Minneapolis, Minnesota. Association for Computational Linguistics.

Chomsky, N. 1957. *Syntactic Structures*. The Hague: Mouton.

Chomsky, N. 1993. A Minimalist Program for Linguistic Theory, MIT Occasional Papers in Linguistics, 1-67. Reprinted in: Chomsky (1995).

Chomsky, N. 1995. *The Minimalist Program*, MIT Press, Cambridge, Massachusetts/London.

Chomsky, N. & M. P. Schützenberger. 1963. The algebraic theory of context-free languages. *Studies in logic and the foundations of mathematics, vol. 35: Computer programming and formal systems*, ed. by P. Braffort and D. Hirschberg, 118–61. Amsterdam: Elsevier.

Elman, J. L., E. A. Bates, M. H. Johnson, A. Karmiloff- Smith, D. Parisi & K. Plunkett. 1996. *Rethinking Innateness: A Connectionist Perspective on Development*. Cambridge, MA: MIT Press.

Fodor, J. A. 1983. *The Modularity of Mind: An Essay on Faculty Psychology*. Cambridge, MA: MIT Press.

Fodor, J. A. & Z. W. Pylyshyn. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition* 28: 3–71.

Frank, R., D. Mathis & W. Badecker. 2013. The acquisition of anaphora by simple recurrent networks. *Language Acquisition* 20: 181–227.

Gardner, H. 1985. *The Mind’s New Science: A History of Cognitive Revolution*. New York: Basic Books, Inc.

Hagstom, P. 2006. A-*not*-A questions. In: Martin Everaert & Henk van Riemsdijk (eds.). *The Blackwell Companion to Syntax, Volume I*, 173-213..

Hart. B & T.R. Risley. 1995. *Meaningful Differences in the Everyday Experience of Young American Children*. Baltimore, MD: Brookes.

Hestvik A, Maxfield N, Schwartz RG, Shafer V. 2007. Brain responses to filled gaps. *Brain and Language*. 100(3):301-16.

Hockett C. F. 1968. *The State of the Art.* The Hague, Neth.: Mouton.

Hochreiter S. & J. Schmidhuber. 1997. Long short-term memory. *Neural Comput*. 9:1735–80.

Huang, C.-T. J. 1982/1998. *Logical Relations in Chinese and the Theory of Grammar*. Doctoral Dissertation. MIT, Cambridge, MA (published by Garland, New York, 1998).

Jackendoff, R. 1974. *Introduction to the X-bar Convention*. Indiana University Linguistics Club.

Jackendoff, R. 1977. Constraints on Phrase Structure Rules. In: P. W. Culicover, T. Wasow & A. Akmajian (eds.), *Formal Syntax*, Academic Press, New York, pp. 249–83.

Joos M. 1995. Preface to Volume I. In *Readings in Linguistics I & II, Abridged Edition*, ed. E Hamp, M Joos, FW Householder, R Austerlitz, pp. ix–x. Chicago: Univ. Chicago Press

Kam, C. X.‐N. & J. D. Fodor. 2012. Children's acquisition of syntax: Simple models are too simple. In: Massimo Piattelli-Palmarini & Robert C. Berwick (eds). Rich Languages From Poor Inputs, 43-60.

Kim, John J., G. F. Marcus, S. Pinker, M. Hollander & M. Coppola. 1994. Sensitivity of children’s inflection to grammatical structure. *Journal of Child Language* 21: 173–209.

Koepke, P. 2007. Gödel’s completeness theorem with natural language formulas. *Logik, Begriffe, Prinzipien des Handelns*. Brill mentism, 49-63.

Kuncoro, A. C. D, J. Hale, D. Yogatama, S. Clark & P. Blunsom. 2018. LSTMs can learn syntax-sensitive dependencies well, but modeling structure makes them better. *Proceedings of the 56th annual meeting of the Association for Computational Linguistics (long papers)*, 1426–36. Online: http://aclweb .org/anthology/P18-1132.

Lee, H.-T. 1986. *Studies on Quantification in Chinese*. Doctoral Dissertation. University of California, Los Angeles.

Li, Y. A. 1992. Indefinite *wh* in Mandarin Chinese. *Journal of East Asian Linguistics* 1: 125–155.

Lin, J.-W. 1996. *Polarity Licensing and Wh-phrase Quantification in Chinese*. Doctoral Dissertation. University of Massachusetts at Amherst, Massachusetts.

Lin, J.-W. 1998. On existential polarity *wh*-phrases in Chinese. *Journal of East Asian Linguistics* 7: 219–255.

Linzen, T. 2019. What can linguistics and deep learning contribute to each other? Response to Pater. *Language* 95(1): e99-e108.

Linzen, T. & M. Baroni. 2021. Syntactic structure from deep learning. *Annual Review of Linguistics* 7: 195-212.

Linzen T., E. Dupoux & Y. Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Trans. Assoc. Comput. Linguist.* 4: 521–35.

Marcus, G. F. 2001. *The Algebraic Mind: Integrating Connectionism and Cognitive Science*. Cambridge, MA: MIT Press.

Marcus, G. F. 2018. Deep learning: A critical appraisal. arXiv:1801.00631 [cs.AI]. On-line: https://arxiv.org/abs/1801.00631.

Marcus, G. F., U. Brinkmann, H. Clahsen, R. Wiese & S. Pinker. 1995. German inflection: The exception that proves the rule. *Cognitive Psychology* 29: 189–256.

McCoy T., Frank R. & T. Linzen. 2020. Does syntax need to grow on trees? Sources of hierarchical inductive bias in sequence-to-sequence networks. *Trans. Assoc. Comput. Linguist.* 8: 125–40.

Nunan, D. 1993. *Introducing Discourse Analysis*. London: Penguin English.

Pater, J. 2019. Generative linguistics and neural networks at 60: Foundation, friction, and fusion. *Language* 95(1): e41–e74.

Pinker, S. 1999. Words and Rules: The Ingredients of Language. New York: Basic Books.

Pinker, S. & A. Prince. 1988. On language and connectionism: Analysis of a parallel distributed processing model of language acquisition. *Cognition* 28: 73–193.

Pinker, S. & M. T. Ullman. The past and future of the past tense. *Trends in Cognitive Sciences* 6: 456–63.

Plunkett, K. & P. Juola. 1999. A connectionist model of English past tense and plural morphology. *Cognitive Science* 23: 463–90.

Potts, C. 2010. Formal pragmatics. In *The Routledge Pragmatics Encyclopedia*, ed. by Louise Cummings, pp.167-170.

Prasada, S. & S. Pinker. 1993. Generalisation of regular and irregular morphological patterns. *Language and Cognitive Processes* 8: 1–56.

Raffel, C., N. Shazeer, A. Roberts, K. Lee, S. Narang S. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. arXiv:1910.10683 [cs.LG]

Ramscar, M. 2002. The role of meaning in inflection: Why the past tense does not require a rule. *Cognitive Psychology* 45: 45–94.

Reali, F. & Christiansen, M. H. 2005. Uncovering the richness of the stimulus: Structure dependence and indirect statistical evidence. *Cognitive* *Science* 29: 1007-28.

Reinhart, T. 1976. *The Syntactic Domain of Anaphora*. Doctoral dissertation, Massachusetts Institute of Technology.

Rosenblatt, Frank. 1957. The perceptron: A perceiving and recognizing automaton (Project PARA). Report 85-460-1. Ithaca, NY: Cornell Aeronautical Laboratory.

Rumelhart, D. E. & J. L. McClelland. 1986. On learning the past tenses of English verbs. In Rumelhart et al., 216–71.

Rumelhart, D. E. & J. L. McClelland & The PDP Research Group (eds.) 1986. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, vol. 2. Cambridge, MA: MIT Press.

Saffran J.R., R. N. Aslin & E. L. Newport. 1996. Statistical learning by 8-month-old infants. Science. 1996 Dec 13;274(5294): 1926-8.

Samek, Wojciech, Thomas Wiegand, Klaus-Robert Müller. 2017. Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models. arXiv:1708.08296 [cs.AI]

Saussure, F. 1916. *Course in General Linguistics*. London: Duckworth.

Sutskever I, Vinyals O, Le QV. 2014. Sequence to sequence learning with neural networks. In *NIPS’14: Proceedings of the 27th International Conference on Neural Information Processing Systems*, Vol. 2, ed. Z Ghahramani, M Welling, C Cortes, ND Lawrence, KQ Weinberger, pp. 3104–12. Cambridge, MA: MIT Press.

Tran, K., A. Bisazza & C. Monz. 2018. The importance of being recurrent for modeling hierarchical structure. *Proceedings of the 2018 conference on Empirical Methods in Natural Language Processing*: 4731–36.

Tsai, W.-T. D. 1994. On nominal islands and LF extraction in Chinese. *Natural Language & Linguistic Theory* 12: 121–175.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, K. Lukasz & I. Polosukhin. 2017. Attention is all you need. *Advances in Neural Information Processing Systems (NIPS)* 30.5998–6008. Online: <https://papers.nips.cc/paper/7181-attention-is-all-you-need>.

Wang, Y., X. Liu & S. Shi. 2017. Deep Neural Solver for Math Word Problems. Paper presented at the Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, Copenhagen, Denmark.

Yang, C. 2004. Universal Grammar, Statistics, or both. *Trends in Cognitive Science* 8(10): 451-456.

Wang, Wen-jet, Chen, Chia-jung, Lee, Chia-ming, Lai, Chien-yu, Lin, Hsin-hung. 2019. Articut: Chinese Word Segmentation and POS Tagging System [Computer program]. Version v.258, retrieved from [https://api.droidtown.co](https://api.droidtown.co/).

# LOKI建模流程

以我現在常用的 advance 來說，流程如下：

1. 人觀察語料，做出歸納，找出簡短且能代表該 pattern 的句子。

* e.g., pattern：差一點+ telic verb

句子：差一點跌倒

1. 把它以上句子手動/自動的送上Loki 網頁工具
2. Loki 會自動把它變成regex 的形式來表示 Articut 斷詞結果

* e.g., 差一點跌倒

(<modifier>任意詞彙<modifier>)?<verb or verbP>跌倒<verb or verbP>

此處 modifier 被()? 即 regex 表現 adjunct 可有可無的方式，Loki 預設會把 modifier 視為可有可無。

1. 在網頁工具上編輯 regex

* e.g.,

原：(<modifier>任意詞彙<modifier>)?<verb or verbP>跌倒<verb or verbP>

編輯後：

<modifier>差一點<modifier><verb or verbP>［任意字］+倒 or 死<verb or verbP>

此步驟就是在依照句法上的 cue 描述

1. 重複以上步驟到一定程度後在網頁上部署，Loki 就把模型建好了，使用者在網頁上把模型下載下來，此時該模型內涵的東西就有
   1. 上面編輯過的 pattern
   2. execLoki() ：一個呼叫會把 input 句子拿去比對所有 pattern，然後回傳比對到哪個 intent 的函式。
2. 下載下來的原始程式碼只負責比對 pattern 跟 intent，至於比對到之後要做什麼處理，由使用者自己寫程式處理。

* e.g.,
  1. 比對到有強疑問詞的 intent 為 interrogative 的 pattern 時，就不考慮另外兩個 intent
  2. 抓出句中的強疑問詞，輸出：「句中的強疑問詞XX」，讓此句中的「誰」產生疑問語意。

這樣基本就是一個完整的模型了。

1. 少數的例外為蔡維天老師與張俊盛老師等人共同主持的「運用資料科學與深度學習之語言學、語言工程、語言教學之整合研究I-IV」(MOST106-2633-M007-003，MOST107-2633-M007-001，MOST108-2633-M007-001，MOST109-2639-M007-001) 以及「運用知識超越資料之人工智慧研究 — 語意、篇章、數位人文」(MOST110-2221-E007-099)。 [↑](#footnote-ref-1)
2. 類神經網路的本質是數字序列 (sequence of real numbers)，運作的基本操作是一個數字序列映射 (map) 至另一數字序列的函數運算。類神經網路數列的函數運算可以表現出學習現象的原因在於輸入 (input) 函數的每一個數字是帶有比重的 (weighted)，而數字比重也可視訓練資料與訓練者的監控反饋 (supervision) 進行彈性調整，進而影響每次運算輸出結果 (output)，因此類神經網路可以表現出隨著資料訓練與監控反饋產生逐步接近研究者期望的學習表徵。更重要的是，雖然每一次的數列函數運算都是簡單直觀的 (simple)，但由於類神經網路的結構中可以有成千上萬的數列層次彼此產生交錯網絡影響 (deep networks)，每一層的運算輸出結果亦為許多其他層次的運算輸入，如此不同層次數列比重彼此交錯影響堆疊起來產生的整體改變即為類神經網路深度學習 (Deep Learning)。由於類神經網路操作的著力點為數字序列函數運算，將此數列函數運算的學習方式應用在自然語言處理上就需要將每個詞彙轉譯為向量 (vector)，運用向量模型理解詞彙語意的例子請參見註腳16。對於類神經網路的運作原理，[Josh Starmer的教學影片](https://www.youtube.com/watch?v=CqOfi41LfDw&ab_channel=StatQuestwithJoshStarmer)有相當深入淺出的介紹。 [↑](#footnote-ref-2)
3. 相關爭議的討論文獻相當豐富：Fodor & Pylyshyn 1988; Pinker & Prince 1988; Prasada & Pinker 1993; Kim et al. 1994; Marcus et al. 1995; Elman et al. 1996; Berent et al. 1999; Pinker 1999; Marcus 2001, 2018; Pinker & Ullman 2002; Frank et al. 2013。 [↑](#footnote-ref-3)
4. 舉例來說，一個任意英語動詞 (如 *blix*) 可加上 -*ed* 表達過去式，即為一個通則；生成語言學家認為這個通則的運用知識具備代數性質，即「動詞」這個統稱的代數符號為這個通則的應用對象，而不只是一個特定的動詞 *blix* 作為通則知識運用標的物。 [↑](#footnote-ref-4)
5. 根據 Berent & Marcus (2019)，針對自然語言處理與學習的任務，(1) 與 (2) 優劣的試金石在於其發展出的語言通則廣度 (across-the-board generalization)，而此討論細節與本計劃內容無直接關係，有興趣的讀者可參閱原文。 [↑](#footnote-ref-5)
6. Kam & Fodor (2012) 的實驗結果指出Reali & Christiansen (2005) 訓練的 n 元序列模型 (n-gram model) 可以正確判斷運用move-main 形成的問句為合法句並不是因為他們的 n 元序列模型成功學會了抽象結構表徵知識，而是特定的 bi-gram 字串 (如 *that/who is*) 固定出現在合法的測試語句中，提供模型一個合法句判斷的局部性字串線索 (local word-level heuristics)。 [↑](#footnote-ref-6)
7. 即使有些研究發現沒有植入結構表徵的類神經網路有能力判定如例句 (i) 中的一致性協同 (如 Linzen et al. 2016)，Linzen & Baroni (2019:198-199) 指出這並不代表該類神經網路的正確判斷是因為找到了決定一致性協同的主語名詞核心字 (=length)，因為該類神經網路有可能是運用其他的非因果相關線索 (heuristics) 來協助判別，如詞頻與字詞之間的相鄰機率 (transitional probabilities)。

   1. [The length of the forewings] (is/\*are) …

   訓練語料庫中的資料分佈頻率對類神經網路的一致性協同判別有很大的影響，舉例來說，Marvin & Linzen (2018) 發現 (ii) 之中的 nested agreement dependencies 在訓練語料庫中的出現頻率很低，想當然爾，其訓練出來的類神經網路對於 (ii) 的判別表現就很差。此研究結果呼應了註腳 5 中提到的 Berent & Marcus (2019) 評論，測試一個類神經網路 NLP 能力的試金石在於其發展出來的語言通則廣度是否能超越訓練語料庫中的熟悉結構。

   1. [The farmer [that the parents love]] swims/\*swim.

   [↑](#footnote-ref-7)
8. 針對智慧計算領域排斥理論語言學的研究文化 (即 Berent & Marcus 提出的第一條路)，徐嘉慧、何萬順、劉昭麟 (2016:28) 提出有趣的意見：「從語言學門的角度看，那一些智慧計算學門的同好所做的研究或許都屬於應用語言學，而少進入理論語言學的研究範疇 […] 這一點顯然有許多可以再多思考的地方，好的研究不太可能只看膚淺的資訊，而能達成深度的效果的」。 [↑](#footnote-ref-8)
9. (11) 的「誰」雖然失去存在語意，但其疑問語意仍可以合法存在，因此 (11) 是個合法句；(12b) 不合法是因

   為「什麼人」的疑問語意與 A-*not*-A 疑問語意相互競爭，導致整個句子無法產生一個一致同質的疑問語意。 [↑](#footnote-ref-9)
10. 核可全稱量化語意的詞/語境不侷限於副詞「都」左側，舉例來說，Lin (1996) 提出 *wh* 詞得到全稱量化語意的句子其實是刪略了「無論」的條件句，因此「無論」條件句也可以核可 *wh* 詞的全稱量化語意，如 (i)，雖然「誰」與「都」不在同一個句子中，仍可以得到全稱量化語意。除此之外，即使沒有「都/無論」，(ii) 此類驢子句 (donkey sentences, Cheng & Huang 1996) 也可以核可「什麼」的全稱量化語意。考量篇幅限制，計劃本文並沒有討論此類語境，但本計劃建構中的 NLP/NLU 智慧運算程式會將此類分佈納入考量。

    1. 無論誰去台北，李四都不在乎。
    2. 李四買什麼，我就買什麼。

    [↑](#footnote-ref-10)
11. 站在認知科學的角度，*wh* 詞的非疑問語意與結構複雜的問句即使出現比例相對低，不代表可以在 NLP/NLU 的研究中忽略這些現象。 [↑](#footnote-ref-11)
12. 音韻層面的問題不是本計劃重點，因此以下省略不談。 [↑](#footnote-ref-12)
13. 另亦有結合語料庫和機器學習模型運作的應用框架。以 NLTK 、spaCy 和 HanLP 為代表，但這三者本質上更接近語料庫和應用框架結合後，為自然語言處理工程師設計的 SDK/API 系統，其效果依所載入的語料庫和機器學習模型而定。在語料庫技術的限制下，其受好評處為開發者友善 (developer friendly) 的設計目標，而非處理效果本身的優劣，因此與本計劃目標無直接相關，故本文省略不談。 [↑](#footnote-ref-13)
14. NER 指的是從一個句子裡辨識出來某些特定的實體 (Entity)。比如，在句子「他在聯合國大規模毀滅性武器檢查小組裡工作」中，雖然斷詞結果可以是「他/在/聯合國/大規模/毀滅性/武器/檢查/小組/裡/工作」，但是這個句子裡有一個「組織名稱」，那麼 NER 就需要辨識出「他/在/聯合國大規模毀滅性武器檢查小組/裡/工作」，且辨識出「聯合國大規模毀滅性武器檢查小組」是一個組織名稱，這才是合理的處理結果。NER 對任何 NLP 系統都是一個挑戰。ML 的做法有兩種：Google/Microsoft 採用「直接用字符計算」，而中研院採用「字符加上 POS」來計算。Google/Microsoft 的做法，在中文裡的效果很差。中研院則是因為加上了 POS 的資訊，所以會受到中研院的 POS 結果的正確率的影響而下降。 [↑](#footnote-ref-14)
15. 資料本身是否足以呈現語言結構表徵和世界百科知識 (encyclopediac knowledge) 的重要性顯而易見；舉例來說，中研院詞庫小組的主持人，馬偉雲教授曾說「這條河很難過」(hard to cross) 和「我很難過」(sad) 對電腦來說很難區分。這是由於他們採用的機器學習方法只取「表層形式的字符」(一個字符一個字符依序計算)，因此算不出來這兩個「難過」應該是「難/過」(hard to corss) 和「難過」(sad)。除非，有一個世界百科知識系統可以告訴這個 NLP 系統說「人，才有能力感受情緒」或是「某些水體，才會有難以跨過的問題」。 [↑](#footnote-ref-15)
16. 舉例來說， *table* 的向量資訊可為 (1, 2.5, 3)，相似的 *desk* 為 (1.2, 2.5, 2.8)，較不相似的 *dog* 則為 (2.1, 0.3, 4)。 [↑](#footnote-ref-16)
17. 收斂的意思，是指「用來分群的那個邊界己經劃定清楚」，但不表示它是正確的解答。因為它也不知道正確的解答是什麼。 [↑](#footnote-ref-17)
18. 圖一的讀法是先看 x 軸與 y 軸兩個句子的交會處方塊是什麼顏色，然後去右邊的圖例條上找最接近的顏色。找到顏色後，再往右去比對數字，就知道在系統判斷兩個句子的相似值大概是多少。舉例來說，x 軸與 y 軸相同的兩個句子 ”I like my phone” 交會方塊為深紅色，代表系統正確判斷兩個一模一樣的句子相似值為1.0 ( 即 100%)。 [↑](#footnote-ref-18)
19. 本計劃書中詞彙詞性/語意處理結果中的縮寫意義說明如下： D: 副詞; Di: 時態標記; FUNC: 功能詞; Na: 普通名詞; Nb: 專有名詞; Nc: 地方詞; Nep: 指代定詞; Nh:代名詞; VA: 動作不及物動詞; Vachieve: 表{達成}; VCL: 動作接地方賓語動詞; VE: 動作句賓動詞; VL: 狀態謂賓動詞; Ques: 疑問詞。CKIP CoreNLP之所以會如此龐雜，根據中研院的報告原文是因為「…中研究院語料庫看到的詞類標記似乎是語法標記，然而這些標記在辭典中代表的涵意是語意加上語法功能的標記」。中研院企圖用一個標記系統，同時處理「語法」和「語意」還有「語法-語意介面功能」的需求之餘，還有修辭的需求。所以它的標記彼此之間的上下層關係不明確，卻又是標出了一個「一維線性的輸出結果」。細節請參見[報告原文](https://ckip.iis.sinica.edu.tw/CKIP/tr/9804_2013.pdf) (<https://ckip.iis.sinica.edu.tw/CKIP/tr/9804_2013.pdf>)。 [↑](#footnote-ref-19)
20. 只看文字符號系統產生的 NLP 模型是文字符號模型，它呈現的是「文字符號的分佈模型」，缺乏語言視野，更不該被稱為語言模型。此外，根據Ethnologue ([How many languages in the world are unwritten? | Ethnologue](https://www.ethnologue.com/enterprise-faq/how-many-languages-world-are-unwritten-0))，世界上仍在流通的 7151 種語言裡，至少有 2982 種語言沒有在使用文字系統 (查詢日期2022/12/12)。 [↑](#footnote-ref-20)
21. 站在認知科學的角度下，在此類翻譯任務中，為確認機器是否真正理解中文 *wh* 詞在該句的語意，不應加入如句點或問號等符號來暗示或協助運算軟體排除特定語意，因為這不是人類理解語句意義的方式：人腦理解語句的關鍵不在這些符號，而是語句本身的組合成分與結構。 [↑](#footnote-ref-21)
22. 兩項軟體皆已通過智慧財產權審核 (Articut: I750567 中文斷詞方法及系統；Loki: I799822 自然語言對話意圖分析方法)。根據專利法第 21 條，這代表運用生成語言學建構的 NLP 軟體發明是「利用自然法則之技術思想之創作」。 [↑](#footnote-ref-22)
23. 理想的斷詞邏輯應該是電腦把詞彙以「意義」為單位切割出來，但基於統計方法做出來的斷詞結果，給的是「資料分佈」和「符號相連機率」，而不是「意義」。 [↑](#footnote-ref-23)
24. 中文斷詞系統中常用的停用詞表可參見：[GitHub - goto456/stopwords: 中文常用停用詞表（哈工大停用詞表、百度停用詞表等）](https://github.com/goto456/stopwords)以及[最全中文停用詞表整理（1893個） - 程式人生 (796t.com)](https://www.796t.com/content/1494661154.html)。 [↑](#footnote-ref-24)
25. 相較之下，以字典為本的 data-driven NLP 工具，就必須以 bottom-up 的方式，由左至右逐字比對字典或模型資料進行斷詞。此一關鍵差異不只讓Articut 的斷詞表現高效節能，開發者在系統維護上的流程也比 data-driven NLP 簡潔。由於篇幅的限制，使用 Articut 的詳細說明請參見 (<https://api.droidtown.co/document/#Articut_2>)。近來以機器學習為本的 NLP 宣稱斷詞並不是 NLP 必要的程序並推出受大眾矚目的 ChatGPT，關於不斷詞產生的問題以及 ChatGPT 的盲點，請參考政大應數系蔡炎龍老師的[【人工智慧應用專題】語言學導向的 NLP (之三) - YouTube](https://www.youtube.com/watch?v=nJIOPfxlKco&ab_channel=Yen-LungTsai)。 [↑](#footnote-ref-25)
26. 為節省篇幅，Pustejovsky (1991)的重點洞見原文節錄在此：”grammatical phenomena do in fact make references to the internal structure of events, and that a subeventual analysis for predicates is able to systematically capture these effects”。 [↑](#footnote-ref-26)
27. 有些學者會認為只因為現今 AI/NLP 無法解決中文*wh* 詞解讀，就直接採納 X-bar 理論為基底的 Articut 過於武斷，畢竟尚有 LFG、HPSG、GPSG可以選擇。針對此問題，本計劃有三點回覆。第一，中國科學院計算所研究院白碩博士的文章 (<https://read01.com/AJGzNQN.html>) 針對後者這些語法理論提出意見，節錄如下：「当前技术能够大规模处理的，仍然只是具有 “浅层句法” 或者 “简单标记” 的 NLP 任务。更复杂语言现象的理解、更复杂语义关系的抽取，仍然任重道远」。由白碩博士的意見可見，前述 LFG、HPSG 及 GPSG 等方法，雖然在資訊工程領域已具有長時間的實作可行性，但仍只具有淺層句法的處理能力。X-bar 理論做為生成語言學方法論中橋接句法表層結構 (surface structure) 與語意深層結構 (deep structure) 的結構框架，尤其在深層結構中即可利用形式語意學的 lambda 邏輯式將真值條件轉寫為程式邏輯，用以呈現何為「語意理解」。第二，LFG, HPSG, GPSG等方法的「編寫字典」的工作性質，亦是本研究想避免的研究方法，而正如 Pustejovsky 於「生成詞庫理論」(聯經出版社 2020，謝舒凱譯) 中提到的「…沒有深刻考慮句法結構的詞彙語意學註定要失敗」。綜合考量白碩與 Pustejovsky 的意見，本計劃需採用一能考察深層句法結構的方法論，同時避免陷入為每一個詞彙編寫其詞彙語意結構的方法進行。第三，語法理論在資訊工程中的「實作可行性高低」之比較基準並無客觀測量方式亦無相關文獻佐證，且實作性高低不能只看使用者數量，現今 data-driven NLP 研究者因學門發展脈絡不關注生成語言學框架中的假設 (詳見Pater 2019 以及相關回應文章)，因此也不知生成語言學框架中的假設已可轉化為程式產品且通過智慧財產權審核，導致使用 X-bar 理論處理 NLP 資訊工程任務的人數相對較少，此一結果並不是一個科學上的對錯價值判斷，而是一個學門發展歷史的人為結果。 [↑](#footnote-ref-27)
28. 本計劃所採用的 Articut 剖析器處理空成分 (gap)的方式可自定義空成份 (如以 "e" 作為空成份的代號) 或透過後處理來安插空成份 (待全部計算完以後，再寫段程式插入空成份)。以後處理方式為例，不在計算 (computation) 階段允許空成份的存在，實務上即可避免無限迴圈的問題。以 (i) 之中的空賓語 (牛肉) 為例，可在加入自定義成份後，才把句子由剖析計算結束後的結果投入檢驗計算中 (如 (ii)-(iii))。

    1. 問：你吃牛肉嗎?

    答：我吃(牛肉)呀!

    1. 剖析：<ENTITY\_pronoun>我</ENTITY\_pronoun><ACTION\_verb>吃

    </ACTION\_verb><CLAUSE\_particle>呀</CLAUSE\_particle>

    1. 後處理：<ENTITY\_pronoun>我</ENTITY\_pronoun><ACTION\_verb>吃

    </ACTION\_verb>**<**UserDefined>pro</UserDefined**>**<CLAUSE\_particle>呀

    </CLAUSE\_particle> [↑](#footnote-ref-28)
29. 需注意的是，本研究計劃提出之「語言學原理驅動 (rule-driven) 」和 1950-60 年代的「基於規則 (rule-based)」在本質上有所不同；生成語言學自 1957 年發展至今，已具備哥德爾完備性 (即在一階謂詞演算中所有形式語意邏輯式上的有效公式都可被證明，見Koepke 2007)；相較之下，1950 年以前流傳至今的「基於規則 (rule-based)」的語言學方法仍然處於見招拆招，頭痛醫頭，腳痛醫腳的狀態，故常被資訊領域的NLP專家學者視為過於繁冗複雜而放棄 (見本文圖五之下的討論)。舉例來說，針對 (i) 句中的空格，可以填入的字詞，rule-based NLP會採取窮舉式的方式來解決。反之，rule-driven的方式就會採取 Berent & Marcus (2019) 提到的「代數式」(algebraic) 解法，只需要陳述空格可以填入的字詞為「動詞」這個詞類即可，如(ii)。

    1. 我昨天買了一隻很會 {跳舞、玩球、看家、睡覺、仔細照顧小孩 ……} 的小狗。(rule-based)
    2. 我昨天買了一隻很會 VP 的小狗。(rule-driven)

    唯本研究所採用的 "rule-driven" 方法和傳統 "rule-based" 在名稱上都有 "rule" 一字，且因 Articut/Loki 等工具為較近期才完備之工具，故 "rule-driven" 和 "rule-based" 之間之差異仍未於資訊領域廣傳，特於此說明。 [↑](#footnote-ref-29)
30. 雖然「不同的場景使用不同的語句表達不同意圖」是一種機率式的描述，但這種機率式描述與 driven-NLP 使用機率的方式是不一樣的：其機率的計算在於這個句子使用的「場景」，而非在「語句」本身上。若只能從語句本身的前後文來推測究竟是「鼓勵」或「警告」的語意解讀，則將落入「前後文要看多遠才能做出正確的決定？」的瓶頸。相對地，藉由將「鼓勵」或是「警告」的解讀傾向提升到「句子使用的場景」的層次，則既可透過串接其它領域研究的成果 (例如但不限於「機電感應器」、「多模態資料模型」…) 來進一步確認究竟是哪一個意義的解讀才是正確的。本計劃的重心放在「發現某一語句具有一種以上的解讀語意」上，需要串接其它領域的研究成果或許可做為後續的延伸研究。 [↑](#footnote-ref-30)
31. Loki系統的語境、意圖和語句的設定方法非常多樣。可以透過串接 LLM 直接取得 LLM 的輸出結果，亦可由語言學家手動指定符合特定研究目標或可描述特定語法結構的例句，再交由該系統進行表層形式及其深層形式的轉譯，最後並有一程式介面可供操作者將語法理論實作成程式邏輯，再予以執行。 [↑](#footnote-ref-31)
32. 目前進行的小規模實驗中，皆可正確判讀「疑問、存在、全稱」三種語意，如圖十二所示，即已正確判讀三個句子裡的前兩個具有存在語意，而第三個沒有。針對中研院平衡語料庫中2979 筆含有「誰」的語料，本計畫已經建立了共 42 種語句模式 (疑問 24 種、存在 7 種、全稱 11 種)，正確的語意判斷比例為：疑問語意98%、存在語意 99%、全稱語義 99%；錯誤案例包括：某些不合法句、古文語句、句尾有「呀」的語句，本計劃將繼續針對這三類錯誤案例撰寫程式並擴大測驗範圍，以提供整體量化的正確率資料。 [↑](#footnote-ref-32)
33. 反身代名詞的先行詞在結構上必須 c-command 反身代名詞方可建立語意依存關係。舉例來說，句 (i) 之中的「老張」沒有 c-command 「自己」，因此無法作為「自己」的先行詞。

    1. 老張的爸爸批評了自己。

    [↑](#footnote-ref-33)
34. Google 的 BERT 需要 40 億字的資料庫，GPT-2 需要 80 億字，T5 更是超過了 1000 億字 (Raffel et al. 2019)。一個關鍵的人腦-機器對比是成功學習母語 (以英語為例)的健康人類孩童一年能夠接觸到資料量頂多只有 1000 萬字 (Hart & Risley 2019)；根據 Linzen & Baroni (2019: 207)，這個巨幅資料量差異導致只要 data-driven NLP 高度依賴海量資料，那麼無論 data-driven NLP 的表現如何，對於整體認知科學的根本理解與實際貢獻都有侷限性。 [↑](#footnote-ref-34)
35. F-score 指的是一個 recall (覆蓋率) 和 precision (精確率) 之間的平衡性 (皆為資料擷取模型的效能衡量指標，可參見 Yang 2004 運用在英語斷詞系統的討論)。一個資料擷取模型的 recall 高，表示「資料覆蓋率很好」，precision 高，表示「正確判斷力很好」。舉例來說，若資料庫中共有 100 篇已知的洗錢主題文本，而一個模型越能完整抓完這 100 篇，那麼它的覆蓋率就越高。但只看覆蓋率是不夠的，因為如果一個模型從上述的資料庫中抓了 120 篇文本，即使完整正確地抓到 100 篇的已知洗錢主題文本，整體表現仍然錯判了 20 篇；然而，這個覆蓋率表現卻和只抓到完整 100 篇已知洗錢主題文本是一樣高的，兩個模型的效能顯然不同，所以此時就需要帶入精確率的概念來區辨這兩個模型的效能，一個模型正確判斷一篇文章在講洗錢的主題，且「不誤抓」非洗錢主題的文本，就是精確率的表現。綜合來看，模型的 recall 高 (接近 100)，而且又能準確地判斷這篇文本「是」或「不是」洗錢的文本，做出正確的分類，那麼這個模型的 F-score 就會很高 (最高是 100%)，其定義為 (i)，當 (i) 之中的 β 為 1 的時候 (β為使用者任意指定的值)，稱為 F1-Score。

    1. ((1+β^2)\*precision \* recall ) / β^2\*precision + recall

    [↑](#footnote-ref-35)
36. 在更宏觀的視野下，由於不需要海量訓練資料與相應而起的高速運算硬體與電費成本，本計劃建構與運作 NLP 的方式小而美，在永續性以及成本考量上，都較為符合臺灣發展的國情與國力；跟據[此新聞報導](https://ec.ltn.com.tw/article/paper/1575789?fbclid=IwAR2p4qyuK11JxypXo_eLxEUEKCXZbIyyHe4q_e8NMyu5b23DHOjaKUidhRI)，台大電機工程學系李宏毅教授坦言台灣的 AI 發展跟國際巨頭相差甚遠，算力 (計算能力與速度) 資源受限是主要關鍵因素，讓軟體廠 AI 技術、應用發展空間、大專院校人才培育都受限，拖累產業前進腳步。李教授舉例台灣擁有算力資源的僅國網中心 (國家高速網路與計算中心) 與華碩旗下的台灣智慧雲端服務 (台智雲)，但國網中心的 GPU (圖形處理器) 僅兩千多片，與 OpenAI 的一萬片相距甚遠；以台灣產業整體發展狀況，目前應連 OpenAI 於 2020 年推出 GPT-3 都做不出來、遑論 2023 年三月中才剛推出的 GPT-4 模型。 [↑](#footnote-ref-36)
37. LTSM 是 long-term short memory 的縮寫，允許類神經網路擴張字元之間的依附距離 (dependencies that span a large number of words)，此技術細節非本計劃重點，讀者可參考 Hochreiter & Schmidhuber (1997) 的討論。 [↑](#footnote-ref-37)
38. 值得一提的是 (57) 是根據中文母語者穩定的語句判斷結果彙整出來的客觀結果，不再是資工學者一直以來詬病的「繁冗複雜」或是「因人而異」的主觀語言描述。 [↑](#footnote-ref-38)
39. 第一波符號邏輯的發展失敗與生成語言學無關，當時盛行的語言學理論框架為歐洲傳至北美的結構語言學 (Saussure 1916; Bloomfield 1939; Hockett 1968)。根據 Joos (1995) 的描述，結構語言學在 50 年代似乎沒有可以與之抗衡的對手 (原文：”seemed to be without a serious competitor”)。 [↑](#footnote-ref-39)