國立臺灣大學電機資訊學院電信工程學研究所

碩士論文

Graduate Institute of Communication Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

# 基於元學習的資料不足依存句法剖析

# Meta-Learning for Low-resource Dependency Parsing

李仲翊

Chung-yi Li

指導教授：李宏毅 教授

Advisor: Hung-yi Lee, Ph.D.

中華民國一百零九年十月

October, 2020

# 摘要

依存句法分析為自然語言處理系統中非常基礎卻也非常重要的元件之一。然而現今地球上只有大約不到 2% 的語言具有依存句法剖析所需要的語料。現今幫助資料不足語言句法剖析的方法主要利用資料充足語言進行多語言訓練，再將參數轉移到資料不足語言上。這些方法在訓練時對資料充足語言進行優化，測試時的目標卻是在未見過的資料不足語言精細校正後有好表現，造成訓練與測試目標不一致的情況。本論文提出使用模型無關元學習方法改進資料充足語言多語言訓練的演算法，不同於現有方法優化參數在各個語言的語言剖析準確率，而是優化該參數在各個語言上精細校正後的語言剖析準確率，有效解決訓練與測試目標不一致的問題。本研究將模型無關元學習方法實驗在去詞化依存句法剖析，分析不同模型無關元學習演算法的變形其在依存句法剖析的效果優劣，與不同的超參數設置對剖析準確率的影響，發現爬蟲類元學習既適合在訓練語言上訓練完成後直接剖析未見過的資料不足語言，也適合利用資料不足語言的少量語料繼續精進準確率；模型無關元學習與其一階近似則具有接觸資料不足語言語料後快速適應的能力。最後將模型無關元學習推廣到實際的應用場景–詞化的依存句法剖析，發現傳統的多語言協同訓練的基準模型就足夠應付大部分的需求，而模型無關元學習相關方法則有改進的餘地。我們也觀察了這些多語言預訓練方法在精細校正過程中掌握目標語言特性的樣態，為往後改良模型無關元學習演算法提供了有益的觀察。

# Abstract

Dependency parsing is one of the fundamental yet essential components in natural language processing pipelines. However, Only less than 2% of languages in the world have dependency tree data available for parsing. Existing methods of improving low-resource dependency parsing usually employ multilingual training on high-resource languages, then transfer its parameters to low-resource dependency parsing systems. These methods optimize for parsing accuracies on high-resource languages, yet are asked to perform well on low-resource languages after fine-tuning on each of them, which results in a mismatch between training- and testing-time objectives. In this thesis, we apply model-agnostic meta-learning methods (MAML) on low-resource dependency parsing. Instead of optimizing parsing accuracies of training languages, MAML optimizes for parsing accuracies on each language after fine-tuning, which effectively reduces the mismatch of training- and testing-time objectives. We first apply MAML on delexicalized dependency parsing to analyze the performance of different variants of MAML-based methods (MAML, Reptile, FOMAML), and the impact of various hyperparameter settings on parsing accuracies. We find that Reptile is suitable for both zero-shot transfer and low-resource fine-tuning, while MAML and FOMAML can quickly adapt to target languages. Then we extend MAML-based methods to a real-world scenario – lexicalized dependency parsing and find that in most cases, conventional multilingual training works well enough, leaving some room for improvement in MAML-based methods. We also perform an analysis of the ability of different methods to adapt to target languages' characteristics, providing useful observation for improving MAML-based methods.

# Contents

# 圖目錄

# 表目錄

# 第一章　導論

## 1.1　研究動機

為何自然語言處理系統能從少量資料中習得未見過的語言語法是一重要的問題？

首先，從達成人工智慧的角度觀之，杭氏（Noam Chomsky）的刺激貧乏（the poverty of the stimulus）[2] 指出嬰兒能夠在只看過有限語料的情況下習得任何母語的語法，且許多相異的語法都可以解釋這些他們所接收到的語料，然而嬰兒仍然成功習得該語言的語法並成功產生語句與他人溝通。古氏（J.H. Greenberg）[3] 發現存在某些文法結構比其他可能的文法結構更為常見，這說明人類學習語言可能不只仰賴該語言的資料，也仰賴他們對語言的歸納偏置（linguistic inductive bias）的了解，即對語言可能具備之性質的假設，使得他們能夠在吸收有限的語料後，從所有可能解釋語料的語法中選出較有可能符合歸納偏置的語法。語言的歸納偏置可能來自人類擁有相同的腦部結構、或由於語言做為溝通工具所具備的特性 [4]。因此從達成人工智慧的目標觀之，既然嬰孩有辦法在接觸少量的語料後成功習得語法，那麼自然語言處理系統也應該具備這樣的能力。

2016 年由尼氏（Joakim Nivre）等人提出的 Universal Dependencies 句法樹庫 [5]（後稱 UD 句法樹庫）截至 UD2.5 版，已有 90 種語言、累計超過兩千萬詞被收錄 [6]，其中不乏許多資料不足語言之依存句法樹庫，為資料不足語言的系統提供了一個絕佳的測試場域。

從實務上處理資料不足語言的角度觀之，由於擁有大量語料的語言佔世界上所有的語言比例非常少；其餘的語言大多不是只存在少量語料，就是根本沒有語料可供學習。以 UD 句法樹庫為例子，2020 年 5 月出版的 UD 2.6 版句法樹庫共

收錄約 90 種語言，相較於世界上現存的語言數目 7117 種[1]僅佔約 1%，可見如果欲處理地球上所有的語言，只有少量甚或沒有訓練資料的語言比例遠大於有堪用數量訓練資料的語言。也因此若自然語言處理系統有辦法在給予寥寥數句語料後就習得該語言的語法，將有助於降低開發資料不足語言的自然語言處理系統的門檻。

近年來，巨量資料與硬體資源的進步大幅推動了以深層類神經網路（Deep Neural Networks）為基礎的自然語言處理系統的發展。深層類神經網路在今日的自然語言處理系統已是不可或缺的一項元件。然而深層類神經網路有一大缺點：並不適合拿來處理少量資料的學習。深層類神經網路在近年來取得的巨大成功，可歸因於其強大的表現力，特徵工程（feature engineering）的捨棄、對資料假設的減少，使得它比以往的模型更能夠掌握自然語言內部複雜的結構。而上述深層類神經網路的優點是在有巨量資料的前提下才得以成立，若只有少量資料，則對資料假設的缺乏，過強的表現力，特徵工程的不足，反而會導致模型直接過擬合（overfitting）在少量資料上。為了克服深層類神經網路不適合處理少量資料的問題，許多研究者紛紛投入以類神經網路進行少量樣本學習的研究，希望可以打造出具有優異的準確率，又不會過擬合在少量資料的深層類神經網路模型。

機器學習領域在探討模型對資料的假設時，經常使用歸納偏置（inductive bias）一詞來描述這些假設的集合 [8]。由於深度類神經網路對資料的歸納偏置相較其他模型較弱，當處理少量資料時，流行的作法是先使用同領域與其相似的資料對網路進行預訓練或協同訓練，使模型從這些領域內相似的資料中習得該領域資料的歸納偏置，並以參數的方式儲存。依此類推，在自然語言處理領域上處理資料不足語言的任務時，常見的作法有使用相似語系或擁有相似性質的語言之相同任務的資料（如使用同為日耳曼語系的英文依存文法資料幫助德文依存句法分析），或

[1]取自 Ethnologue 第二十三版 [7]。

使用該語言相關任務的資料進行訓練（如以詞性標注任務幫助依存句法分析）。

在依存句法分析方面，過去已經有不少文獻探討如何藉助資料充足語言的標註資料改進資料不足語言的依存句法剖析 [9, 10, 11]；此類研究旨在提高跨語言句法知識轉移的功效，茲列舉如下：根據語言類型學（linguistic typology）的知識 [12] 選擇性地共享源語言（source language）與目標語言（target language）的模型參數 [13]，或將其當做模型輸入，幫助模型利用其對不同語言統一的語法規則描述，進而得以泛化到更多未見過的目標語言 [14, 9, 15, 16]。衡量轉移學習用於不同模型（轉換器（transformer）或遞歸式類神經網路（RNN））及演算法（如圖式（graph-based）或轉換式（transition-based）剖析器（parser））的難易度 [17]；如何挑選用於訓練的源語言以提高目標語言表現 [18]。

雖然已經有許多文獻試圖改進資料不足語言的依存句法分析，大多數的方法仍有一些可以改進的地方：許多模型的訓練目標仍侷限於優化源語言的句法分析準確率，與最終欲優化的目標語言的句法分析準確率不同，頂多在模型選擇時（model selection）以目標語言上的驗證準確率（validation accuracy）作為選擇基準，仍造成訓練與測試目標的不匹配。再者，模型選擇時需要先得知目標語言，不同的目標語言所選擇的模型不同，無法只憑單一模型就在各種語言上進行精細校正（fine-tuning）。

本研究將模型無關元學習方法應用於依存句法分析，在資料充足語言上進行預訓練，以提高精細校正在目標語言後的句法分析準確率為目標對精細校正的初始參數進行優化，將精細校正的過程整合進模型優化的演算法中，解決資料不足句法分析訓練與測試目標不匹配的問題。

## 1.2 研究方向

本研究改進多語言句法分析預訓練幫助資料不足語言分析的演算法，分析不同預訓練方法以單一模型精細校正在多種資料不足語言上的優劣，詳細貢獻條列如下：

- 首先為了去除語言中與句法無關的性質對結果的影響，使用詞性標記做為特徵，在去詞化（delexicalized）依存句法分析的任務上，分析模型無關元學習方法及其變形在多種不同實驗設置下（步數、資料量、模型大小）精細校正在多種資料不足語言的表現及其優劣。
- 接著使用大型多語語言模型編碼器產生特徵，在詞化（lexicalized）依存句法分析任務上，分析模型無關元學習方法及其變形在多種不同實驗設置下（步數、資料量）精細校正在多種資料不足語言的表現及其優劣。

## 1.3 章節安排

本論文之章節安排如下：

- 第二章：介紹本論文相關背景知識。
- 第三章：模型無關元學習方法及其變形用於去詞化依存句法分析。
- 第四章：模型無關元學習方法及其變形用於詞化依存句法分析。
- 第五章：本論文之結論與未來研究方向。

# 第二章　背景知識

## 2.1　機器學習（Machine Learning）

機器學習為實現人工智慧的一種途徑，利用過往資料歸納出解決問題的方法。例子包括：機器學習根據有無人為標註資料與否，可分爲監督式學習及非監督式學習兩類；以下主要介紹監督式學習。

### 2.1.1　機器學習問題架構

**符號定義**

我們首先介紹監督式學習所用到的符號：

- 輸入（input）：$x \in \mathcal{X}$；$\mathcal{X}$ 代表輸入空間。
- 輸出（output）：$y :\in \mathcal{Y}$；$\mathcal{Y}$ 代表輸出空間。
- 函數空間 (function space)：$\mathcal{F} = \{\ f \mid f : \mathcal{X} \to \mathcal{Y}\}$
- 資料空間：$\mathcal{D} = \mathcal{X} \times \mathcal{Y}$
- 資料集：$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\} \overset{i.i.d.}{\sim} \mathcal{D}$
- 假說集合（hypothesis set）：$\mathcal{H}, \mathcal{H} \subset \mathcal{F}$
- 損失函數（loss function）：$\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$
- 理想函數（oracle function）：$f^* : \mathcal{X} \to \mathcal{Y} \quad s.t. \quad \mathbb{E}_{x,y\sim\mathcal{D}}\left[\mathbb{1}_{f^*(x)\neq y}\right] = 0$
- 學習演算法（learning algorithm）：$a : (\mathcal{H}, D) \mapsto h, \quad h \in \mathcal{H}$

**輸入與輸出**

輸入與輸出的例子包括：

- 銀行根據過往核發信用卡的申請人的背景及是否批准核發的資料。$x$：申請人背景；$y$：核發與否。
- 新聞分類。$x$：新聞內容；$y$：新聞類別，如政治、體育。
- 影像分類。$x$：影像；$y$：影像類別，如貓、狗。

**損失函數**

定義好輸入與輸出之後，我們需要給定衡量模型輸出與正確答案的偏差的函數，稱為損失函數 $\ell(\cdot, \cdot)$；常見的任務及其損失函數羅列如下：

- 二元分類 (binary classification)：$\mathcal{Y} = \{0, 1\}$
  - 0-1 損失函數（0-1 loss function）：
    $$\ell(\hat{y}, y) = \begin{cases} 1 & \text{if } \hat{y} \neq y \\ 0 & \text{if } \hat{y} = y \end{cases}$$
  - 交叉熵函數（cross-entropy function）：
    $$\ell(\hat{y}, y) = y \cdot \log p(\hat{y}) + (1 - y) \cdot \log(1 - p(\hat{y})),$$
    $$\hat{y} = \begin{cases} 1 & \text{if } p(\hat{y}) > 0.5, \\ 0 & \text{otherwise} \end{cases}$$
- 多元分類（multiclass classification）：$\mathcal{Y} = \{0, 1, \ldots, N\}$
  - 交叉熵函數：
    $$\ell(\hat{y}, y) = \sum_{i=1}^{N} y_i \log(\hat{y}_i),$$
    $$y_i = \begin{cases} 1 & \text{if } y = i, \\ 0 & \text{if } y \neq i \end{cases}$$

**經驗風險最小化（Empirical Risk Minimization）**

給定輸入資料與相對應的答案，經驗風險最小化 $a$ 的目標是在假說集合 $\mathcal{H}$ 內尋找損失函數在資料分佈 $\mathcal{D}$ 上的期望值最小：

$$h^* = \arg\min_{h \in \mathcal{H}} \mathbb{E}_{x,y \sim \mathcal{D}} \left[ \ell(h(x), y) \right] \tag{2.1}$$

但資料的變化千千萬萬，我們得不到世界上所有的資料，因此我們並不知道真實的資料分佈 $\mathcal{D}$，只能用蒐集到的資料集 $D = \{(x_i, y_i)\}_{i=1}^N$ 去近似它：

$$h^* = \arg\min_{h \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} \ell(h(x_i), y_i) \tag{2.2}$$

### 2.1.2 機器學習模型

機器學習中所謂的模型通常包含了假說集合、學習演算法、與其配合的損失函數。以下列舉常見的模型及其組成：

- 支撐向量機：
  - 假說集合：徑向基函數（radial basis function）、多項式（polynomial）等
  - 學習演算法：二次規劃（quadratic programming）
  - 損失函數：鉸接損失函數（hinge loss）
- 類神經網路：
  - 假說集合：類神經網路
  - 學習演算法：梯度下降（gradient descent）

– 損失函數：交叉熵函數（cross-entropy loss）用於多元分類、均方差函數（MSE loss）用於迴歸（regression）等

## 2.2 深度類神經網路（Deep Neural Networks）

深度類神經網路為一種仿造生物神經網路的計算模型，具有強大的建模能力，及可以近似任意連續函數的理論保證 [19]。過去數十年由於電腦計算能力的不足造成研究進展緩慢；不過自 2000 年代中期以後，隨著電腦算力逐漸增強，逐漸在圖像、語音領域取得優異的成績。其架構仿造動物神經元的設計，每個神經元會接收來自其他數個神經元 $x_1, x_2, \ldots, x_n$ 的訊號，將這些訊號依自身的權重 $w_1, w_2, \ldots, w_n$ 加權後加上偏差 $b$ ，最後通過激活函數 $\sigma : \mathbb{R} \to \mathbb{R}$ 決定該神經元的輸出 $y = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right)$ ，傳遞給下個神經元。

### 2.2.1 前饋式類神經網路 (FeedForward Neural Network)

前饋式類神經網路 (FeedForward Neural Network) 為早發明、也是最簡單的一種類神經網路，可分爲單層與多層前饋式類神經網路。

單層的類神經網路由權重（weight）$\mathbf{W}$、偏差（bias）$\mathbf{b}$ 與激活函數 (activation function) $f$ 所組成。其運作方式接受輸入神經元組（向量）$\mathbf{x}_{in} \in \mathbb{R}^n$ ，將其乘以權重 $\mathbf{W} \in \mathbb{R}^{m \times n}$ ，加上偏差 $\mathbf{b} \in \mathbb{R}^m$ ，最後通過激活函數 $f$ 得到輸出神經元組（向量）$\mathbf{x}_{out} \in \mathbb{R}^m$ ：

$$\mathbf{x}_{out} = f(\mathbf{W}\mathbf{x}_{in} + \mathbf{b}) \tag{2.3}$$

多層類神經網路則將多個單層類神經網路連接起來，將一單層類神經網路的

輸出神經元組作為下層類神經網路的輸入神經元組：

$$\mathbf{x}_\ell = f(\mathbf{W}\mathbf{x}_{\ell-1} + \mathbf{b}) \tag{2.4}$$

因此給定一 L 層類神經網路、輸入神經元組 $\mathbf{x}_0$、及輸出神經元組 $\mathbf{y}$，該類神經網路計算方式如下：

$$\begin{aligned}
\mathbf{x}_1 &= f_1(\mathbf{W}_1\mathbf{x}_0 + \mathbf{b}_1) \\
\mathbf{x}_2 &= f_2(\mathbf{W}_2\mathbf{x}_1 + \mathbf{b}_2) \\
\mathbf{x}_3 &= f_3(\mathbf{W}_3\mathbf{x}_2 + \mathbf{b}_3) \\
&\vdots \\
\mathbf{x}_L &= f_L(\mathbf{W}_L\mathbf{x}_{L-1} + \mathbf{b}_L)
\end{aligned} \tag{2.5}$$

在所有的類神經網路裡，激活函數必須是非線性的函數，否則根據線性函數的性質，線性激活函數所組成的多層類神經網路只等價於一簡單的仿射變換。

### 2.2.2 類神經網路訓練（Deep Neural Network Training）

訓練類神經網路通常使用梯度下降法（gradient descent）來進行模型的優化（optimization）。給定模型參數 $\theta$、模型函數 $h_\theta$、損失函數 $\ell$、輸入 $\mathbf{x}$、正確輸出 $\mathbf{y}$、更新過後的參數 $\theta'$，梯度下降法更新參數的公式如下：

$$\theta' = \theta - \alpha\frac{\partial \ell(h_\theta(\mathbf{x}), \mathbf{y})}{\partial \theta} \tag{2.6}$$

其中 $\alpha$ 為梯度下降的步數大小，稱為學習率（Learning rate），太大則可能錯過局部最佳點，太慢則缺乏效率，需要細心調整。

如果進行梯度下降法時使用整個資料集的資料，則稱為批次梯度下降（batch gradient descent）：

$$\theta' = \theta - \alpha \frac{\partial \sum_{i=1}^{N} \ell\left(h_\theta\left(\mathbf{x}_i\right), \mathbf{y}_i\right)}{\partial \theta} \tag{2.7}$$

當損失平面（loss surface）接近 convex、或有一全域最優點的時候，批次梯度下降可以迅速找到該點，獲得好的結果；但當損失平面高度非 convex、並有很多局部最優點的時候，梯度下降法（gradient descent）缺乏逃出不夠好的局部最優點的隨機性，因而需要隨機梯度下降法（stochastic gradient descent）來提供隨機性：

$$\theta' = \theta - \alpha \frac{\partial \ell\left(h_\theta\left(\mathbf{x}_i\right), \mathbf{y}_i\right)}{\partial \theta} \tag{2.8}$$

隨機梯度下降法每次更新只用一筆資料，計算出的梯度較雜亂（noisy），優點是使參數有機會逃出局部最優點，缺點則是梯度太過雜亂，且一次只使用一筆資料，無法利用圖形處理器（Graphics Processing Unit, GPU）平行處理所帶來的速度優勢縮短訓練時間；小批次隨機梯度下降法（mini-batch stochastic gradient descent）則將一次更新所使用的資料數設於上述兩者之間：

$$\theta' = \theta - \alpha \frac{\partial \sum_{i=1}^{N'} \ell\left(h_\theta\left(\mathbf{x}_i\right), \mathbf{y}_i\right)}{\partial \theta} \tag{2.9}$$

其中 $1 \ll N' \ll N$。小批次隨機梯度下降法不若批次梯度下降法缺乏隨機性，也沒有隨機梯度下降法太隨機且過慢的缺點，爲現行主流機器學習界採用的方法。

### 2.2.3 遞歸式類神經網路（Recurrent Neural Network）

遞歸式類神經網路為一種接受序列為輸入的類神經網路，特別適合處理自然語言、語音、音樂等序列資料。其特色在於資料並不是一次全部輸入網路，而是依照序列自身的排序一一輸入網路；而網路除了接受當前時間點的輸入 $\mathbf{x}_t$ 之外，還另外接受前一個時間點的隱含狀態 $\mathbf{h}_{t-1}$，處理後得到這個時間點的隱含狀態 $\mathbf{h}_t$ 與模型輸出 $\mathbf{y}_t$ 。

**艾氏遞歸式類神經網路（Elman RNN）**

首先我們介紹最基本的由艾氏（Jeffrey L. Elman）於 1990 年提出的艾氏遞歸式類神經網路 [20]：給定在 t 時間點的隱含狀態 $\mathbf{h}_t \in \mathbb{R}^m$、序列資料 $\{\mathbf{x}_i\}_{i=1}^T$, $\mathbf{x}_t \in \mathbb{R}^n$，輸出向量 $y_t \in \mathbb{R}^o$，艾氏遞歸式類神經網路的運作方式如下：

$$\begin{aligned} \mathbf{h}_t &= \sigma_h\left(\mathbf{W}_h\mathbf{x}_t + \mathbf{U}_h\mathbf{h}_{t-1} + \mathbf{b}_h\right) \\ \mathbf{y}_t &= \sigma_h\left(\mathbf{W}_y\mathbf{h}_{t-1} + \mathbf{b}_y\right) \end{aligned} \tag{2.10}$$

其中 $\mathbf{W}_h \in \mathbb{R}^{m\times n}$、$\mathbf{U}_h \in \mathbb{R}^{m\times m}$、$\mathbf{W}_h \in \mathbb{R}^{m\times n}$、$\mathbf{W}_y \in \mathbb{R}^{o\times n}$。

**長短期記憶遞歸式類神經網路（LSTM RNN）**

長短期記憶遞歸式類神經網路（Long-Short Term Memory RNN, LSTM RNN）由施氏（Jürgen Schmidhuber）[21] 於 1997 年提出，將閘門（gate）的概念引入遞歸式類神經網路來控制資訊的流動，取得巨大的成功。總共有三個閘門用來控制 LSTM 中資訊的流動：輸入閘（input gate）、輸出閘（output gate）、遺忘閘 (forget

gate)。

$$\mathbf{i}_t = \sigma_g\left(\mathbf{W}_i\mathbf{x}_t + \mathbf{U}_i\mathbf{h}_{t-1} + \mathbf{b}_i\right) \tag{2.11a}$$

$$\mathbf{o}_t = \sigma_g\left(\mathbf{W}_o\mathbf{x}_t + \mathbf{U}_o\mathbf{h}_{t-1} + \mathbf{b}_o\right) \tag{2.11b}$$

$$\mathbf{f}_t = \sigma_g\left(\mathbf{W}_f\mathbf{x}_t + \mathbf{U}_f\mathbf{h}_{t-1} + \mathbf{b}_f\right) \tag{2.11c}$$

$$\tilde{\mathbf{c}}_t = \sigma_h\left(\mathbf{W}_c\mathbf{x}_t + \mathbf{U}_c\mathbf{h}_{t-1} + \mathbf{b}_c\right) \tag{2.11d}$$

$$\mathbf{c}_t = \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tilde{\mathbf{c}}_t \tag{2.11e}$$

$$\mathbf{h}_t = \mathbf{o}_t \circ \sigma_h(\mathbf{c}_t) \tag{2.11f}$$

其中 $\mathbf{x}_t$ 為輸入向量、$\mathbf{f}_t$ 為遺忘閘向量、$\mathbf{i}_t$ 為輸入閘向量、$\mathbf{o}_t$ 為輸出閘向量、$\mathbf{h}_t$ 為隱含狀態向量、$\mathbf{c}_t$ 為單元狀態（cell state）向量、$\sigma_g$ 為 sigmoid 函數、$\sigma_h$ 為 tanh 函數。LSTM 用輸入閘來控制要讓多少輸入向量的資訊流入單元狀態向量；遺忘閘向量用來決定要留住/遺忘多少上個時間點的單元狀態向量；輸出閘向量則負責控制單元狀態的資訊有多少流入隱含狀態向量。

### 2.2.4 轉換器類神經網路（Transformer Neural Network）

轉換器類神經網路為梵氏（Ashish Waswani）等人 [22] 於 2017 年提出，捨棄了常被詬病時間點之間無法平行處理而顯得過慢的遞歸式結構，轉而使用可以平行計算的自專注機制（self-attention mechanism）作爲模型的主架構，在 2017 年之後蔚爲流行，頗有取代遞歸式類神經網路之勢。

#### 自專注機制（Self-Attention Mechanism）

我們首先介紹自專注機制：以輸入層（input layer）為例，嵌入層（embedding layer）將 $w_i$ 編碼成詢向量（query vector）$q_i \in \mathbb{R}^{d_k}$、鑰向量 $k_i \in \mathbb{R}^{d_k}$ 與值向量

$v_i \in \mathbb{R}^{d_v}$；則自專注模組會輸出該字 $w_i$ 對句子中其他字的專注權重。定義 $\alpha_{ij}$ 為 $w_i$ 對 $w_j$ 的專注權重，則

$$\alpha_{ij} = \text{softmax}\left(\frac{q_i k_j^\top}{\sqrt{d_k}}\right) = \frac{\exp(\frac{q_i k_j^\top}{\sqrt{d_k}})}{\sum_{j'=1}^{N} \exp(\frac{q_i k_j'^\top}{\sqrt{d_k}})} \tag{2.12}$$

特別的是，詢向量與鑰向量之間的內積 $q_i k_j^\top$ 會先除以 $\sqrt{d_k}$，再送入軟性最大化層（softmax layer）；這是為了確保當詢與鑰向量的維度 $d_k$ 變大，內積的大小不會跟著變大，使得通過軟性最大化層後產生梯度消失（gradient vanishing）的現象而妨害訓練；瓦氏將此專注機制稱為縮放式點積專注法（Scaled Dot-Product Attention）。自專注對 $w_i$ 的輸出 $y$ 則為此權重對所有字的值向量 $v$ 的加權之和：

$$y = \sum_{j=1}^{N} \alpha_{ij} v_j \tag{2.13}$$

將句子中所有字的詢向量、鑰向量與值向量組合成矩陣(如 $V = \begin{bmatrix} v_1 & v_2 & \dots & v_N \end{bmatrix}$)，我們可以寫出自專注機制的矩陣形式：

$$\text{SelfAtt}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V \tag{2.14}$$

事實上，自專注機制是一種圖神經網路的特例：圖神經網路的每個節點透過對與其有連結的其他節點進行互動來更新自己的表示（representation）；自專注機制則可以視為一個以字為節點的全連接圖（fully-connected graph）的圖神經網路。

**多頭自專注（Multi-Head Self Attention）**

梵氏在同篇論文中發現使用多組不同的專注模組，對效能有正面的提升；因此他提出多頭自專注，將維度為 $d_{\text{model}}$ 的詢向量、鑰向量與值向量線性投影到多

個維度為 $d_k$、$d_k$、$d_v$ 的子空間（subspace），在子空間中分別進行自專注機制運算後，再將他們重組起來並投影回維度為 $d_{\text{model}}$ 的空間：

$$\begin{aligned} \text{MultiHead}(Q,K,V) &= \text{Concat}(\text{head}_1,\ldots,\text{head}_h)W^O \\ \text{其中head}_i &= \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{2.15}$$

投影矩陣的維度分別為 $W_i^Q \in \mathbb{R}^{d_{\text{model}}\times d_k}$，$W_i^K \in \mathbb{R}^{d_{\text{model}}\times d_k}$，$W_i^V \in \mathbb{R}^{d_{\text{model}}\times d_v}$，$W^O \in \mathbb{R}^{hd_v\times d_{\text{model}}}$。之後的研究發現多頭自專注有利於模型對不同的語言現象，如詞的位置、不同語法功能、少見詞等分別進行專注機制運算 [23]。

**位置編碼（Positional Encoding）**

由於自專注機制不若遞歸式類神經網路，透過輸入網路的先後順序暗中給了網路字與字的相對位置資訊，自專注機制若沒有特別的模組為每個字的位置建模，有序的字組成的句子與無序的詞袋 (bag-of-words) 並無二致。因此梵氏引入位置編碼，直接將位置資訊以向量的形式表示：

$$\begin{aligned} PE_{\text{pos,2i}} &= \sin(\text{pos}/10000^{2i/d_{\text{model}}}) \\ PE_{\text{pos,2i+1}} &= \cos(\text{pos}/10000^{2i/d_{\text{model}}}) \end{aligned} \tag{2.16}$$

其中 pos 代表位置，而 $i$ 代表維度。使用正弦與餘弦函數的優點，梵氏的解釋是模型可以透過這些函數的週期性學習字與字的相對位置。

## 2.3 分佈式表示（distributed representation）

如何將語義以電腦可理解的方式表示，可以說是自然語言處理的核心問題。以下介紹最爲流行的一派方法，也就是以分佈式語義學（distributional semantics）為基礎的分佈式表示。

### 2.3.1 詞向量（Word Vectors）

最簡單的方法為尋找一個足夠有代表性的語料庫，經過分詞後，統計詞與詞之間共同出現的次數，構建出詞之間的共現矩陣。底下用只有四句話的語料展示如何構建共現矩陣（co-occurrence matrix）：

- 李宏毅幾班？
- 李宏毅五年二十班。
- 李仲翊幾班？
- 李仲翊三年五班。

以上四句話的共現矩陣為：

| | 李宏毅 | 李仲翊 | 幾 | 班 | 三 | 五 | 年 | 二十 |
|---|---|---|---|---|---|---|---|---|
| 李宏毅 | 0 | 0 | 1 | 2 | 0 | 1 | 1 | 1 |
| 李仲翊 | 0 | 0 | 1 | 2 | 1 | 1 | 1 | 0 |
| 幾 | 1 | 1 | 0 | 2 | 0 | 0 | 0 | 0 |
| 班 | 2 | 2 | 2 | 0 | 1 | 2 | 2 | 1 |
| 三 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |
| 五 | 1 | 1 | 0 | 2 | 1 | 0 | 2 | 1 |
| 年 | 1 | 1 | 0 | 2 | 1 | 2 | 0 | 1 |
| 二十 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |

表 2.1: 共現矩陣的示例。

表2.1中，每個字的分佈表示即爲該欄/列向量（在對稱的共現矩陣中欄向量與列向量相同）。但此共現矩陣的大小為語料庫中詞種數的平方：若詞種數為 $|V|$，則共現矩陣大小為 $|V|^2$，這樣的矩陣過於龐大（矩陣內每格均爲整數，若一整數有 4 bytes，詞種數為 100,000，則此矩陣共有 $4 \times 100,000 \times 100,000 = 4$ GB），其隱含的資訊量應能用更少的維度表示；因此後人提出許多降維（Dimension Reduction）配合矩陣平滑化（Smoothing）（或可達成類似目的）的方法得出低維度的詞向量，以下舉最成功的文字向量（Word2Vec）為例：

**Word2Vec**

文字向量（word2vec）為米氏（mikolov）於 2013 年提出及開發，以批次訓練分解共現矩陣 $M$（co-occurrence matrix）為詞矩陣 $W$ 及語境（context）矩陣 $C$，其優點是不需要事先統計並儲存龐大的共現矩陣（若有十萬詞，則 矩陣大小 $= 100,000^2$），每次分批讀入小部分語料後更新 $W$ 及 $C$ 即可。損失函數方面，捨棄需要更新所有詞參數的軟性最大化（softmax），而使用噪聲對比估計（Noise Contrastive Estimation）損失函數，有效降低計算負擔。其運作方式如下：

給定一詞 $w$ 及其語境 $c$，文字向量的目標旨在最大化該詞的詞向量 $\vec{w}$ 及該語境的語境向量 $\vec{c}$ 的內積；具體的機率模型則使用 S 函數（sigmoid function）來表示詞-語境對 $(w, c)$（word-context pair）出現在語料庫 $D$ 的機率：

$$P(D = 1|w, c) = \sigma(\vec{w} \cdot \vec{c}) = \frac{1}{1 + e^{-\vec{w} \cdot \vec{c}}} \tag{2.17}$$

若只有最大化出現在語料庫中詞-語境對 $(w, c)$，到最後所有 $\vec{w}$ 與 $\vec{c}$ 都會朝向同個方向，這不是我們所希望得到的詞向量，因此還需要做負取樣（negative sampling），也就是隨機取樣不會出現在語料庫裡的負樣本 $(\vec{w}, \vec{c_N})$ 並降低其出現的

機率（實際操作上並不會檢查隨機從詞彙裡取樣語境 $\vec{c_N}$ 得到的詞-語境對 $(w, \vec{c_N})$ 是否出現在語料庫 $D$，而直接假定出現的機率不大）。因此詞向量的損失函數如下：

$$\ell = -\sum_{w \in V_W} \sum_{c \in V_C} \#(w,c) \left( \log \sigma(\vec{w}, \vec{c}) + k \cdot \mathbb{E}_{c_N \sim P_D} [\log \sigma \left( -\vec{w}, \vec{c_N} \right)] \right) \tag{2.18}$$

其中 $k$ 為每個正樣本搭配的負樣本的個數，$\#(w,c)$ 為詞-語境對 $(w,c)$ 出現在語料庫 $D$ 中的次數，$\vec{c_N}$ 為取樣自一元分佈（unigram distributation）$P_D(c) = \frac{\#(c)}{|D|}$ 的語境詞。

文字向量取得了巨大的成功，從 2014 年到 2018 年都是各種自然語言處理任務的標準配備，直到上下文化表示的出現（見章節2.3.2）。

### 2.3.2 語境化表示（Contextualized Representations）

詞向量在各種自然語言處理任務中雖然取得了巨大的成功，其一詞一向量的本質並不適合處理多義詞、代名詞指涉、或甚至同一詞其語義在不同語境下的微妙差異。為了解決這樣的問題，研究者開始思考一詞多個向量的可能性，如詞義向量（word sense embeddings），給予不同詞義不同的向量（如 bank 有河岸或銀行兩種意思）[24, 25]。而將一詞多向量的想法推行到極致，就是語境化表示的想法：對同一詞，只要出現在不同語境，就給予不同的向量。

**ELMo**

馬氏 [26] 於 2018 年提出的 ELMo (Embeddings from Language Models) 為第一個使用語境化表示得到巨大成功的模型，展示了普通的兩層雙向 LSTM（詞嵌入

使用字符 CNN）用語言模型的目標函數訓練在大量語料上，其隱藏層即蘊含了豐富的語境化表示，使得 ELMo 在六種自然語言處理任務相對於基準模型有 6% 到 20% 的進步率。

**BERT**

在這之後許多試圖改進 ELMo 的語境化表示模型如雨後春筍般相繼出現，如 GPT[27]、ULMFit[28] 等，其中最成功且流行的當屬 2019 年戴氏 [29] 基於轉換器的架構設計的轉換器模型的雙向編碼器表示（Birectional Encoder Representations from Transformers, 下稱 BERT）。過去的語境化表示如 ELMo、GPT[27] 等語境化表示通常為單向（unidirectional）語言模型，而傳統的雙向 LSTM 語言模型也只是左到右與右到左語言模型的淺層級聯（concatenation），其本質仍是兩個獨立訓練的單向語言模型。BERT 改進了傳統的雙向 LSTM 語言模型，利用轉換器模型的架構優勢設計了遮蔽式語言模型（masked language model），相當於給模型進行克漏字測驗的訓練，或也可以看成去噪自編碼器（denoising autoencoder）的一個例子，使模型可以同時利用雙邊的語境進行預測，使其隱藏層蘊含的訊息更為豐富，而這是兩個各只利用單邊語境的傳統雙向語言模型不能做到的。

## 2.4 依存句法剖析（Dependency Parsing）

### 2.4.1 句法簡介

句法（syntax）可定義為支配句子結構，決定詞、子句如何組成其上級結構的一系列規則。根據梅氏（Igor Mel’čuk）的文字-語意理論（Meaning-text theory，見圖2.1），一段語句的生成可以被描述為從語意表徵（semantic representation，SemR）到語音表徵（phonetic representation，PhonR）之間一連串的轉換（見圖

n）：語意表徵先經由一普遍適用的語法規則產生其深層語法表徵（deep syntactic representation，DSyntR），爾後經由各語言的語法規則（syntactic rules）轉換為該語言特有的表層語法表徵（surface syntactic representation，SSyntR）；接著再透過各語言的線性化（linearization）規則轉換為構詞表徵（morphological representation，MorphR）；最後轉化為語音表徵，成為一般人所聽見的語句。從語意到語音的映射為一多對多的函數：同樣的語意可以用不同詞彙及語法結構表示（一對多），即改述（paraphrasing）；不同語意表徵的語句亦有可能經轉換後恰巧有相同的語音表徵（多對一），如圖2.2，英文語句 “I saw a man with a telescope.”，兩種不同語意表徵的語句（「我看見一個帶望遠鏡的男人」與「我用望遠鏡看見一個男人」）分別產生不同的語法表徵，最後因英文的線性化規則恰巧產生相同的語音表徵。此示例說明了句法剖析對於語意理解的重要性：同樣的一句話可能根據不同的句法結構導出不一樣的語意。

依存句法（dependency grammar）則為句法理論的一支，將句法結構視為詞與詞之間的相依關係，以有向鏈結（directed links）將詞關聯在一起；相較於成分句法（constituency grammar）只能表示連續結構（continuous constituents），當遇到不連續結構時只能以最相近的連續結構近似，依存句法則無此限制，因此特別適合分析語序相對自由的語言。

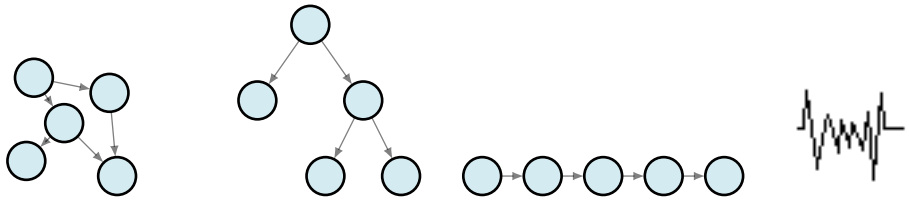
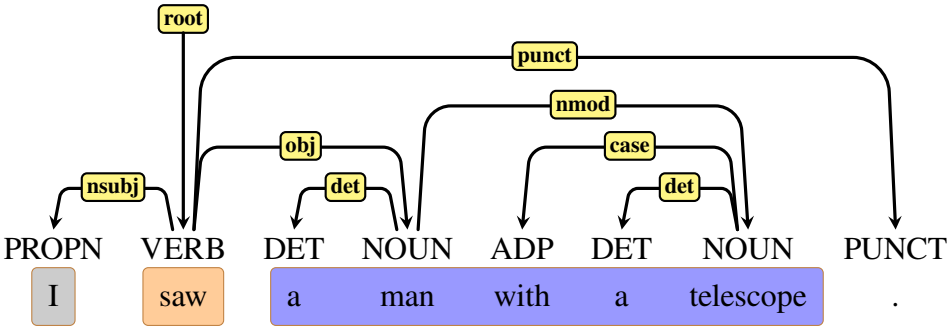
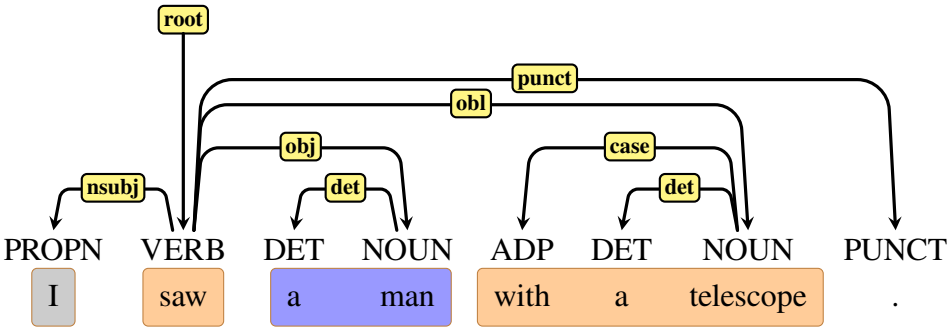



圖 2.1: 語義-文字理論。

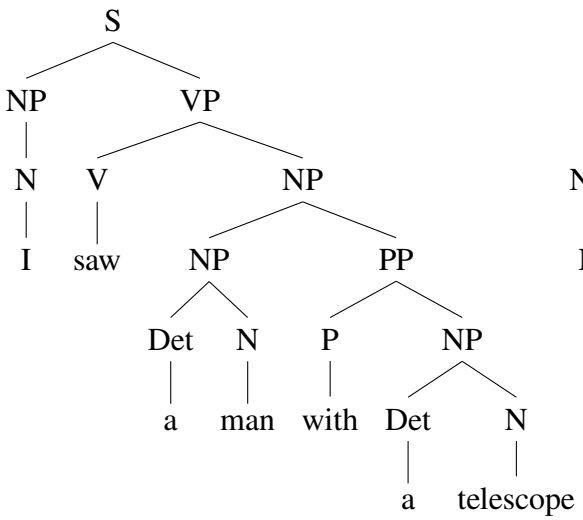



(a) 介系詞片語 “with a telescope” 依附於名詞 “a man” 上；
翻譯：我看見一個帶望遠鏡的男人。

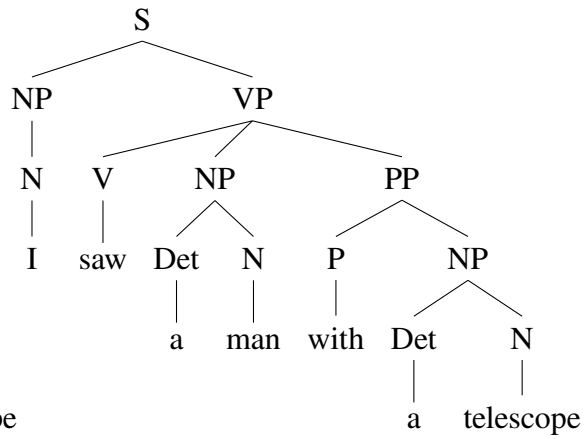



(b) 介系詞片語 “with a telescope” 依附於動詞 “saw” 上。
翻譯：我用望遠鏡看見一個男人。

圖 2.2: 介系詞片語依附（Prepositional phrase attachment, PP attachment）造成的結構歧義性（Structural ambiguity）。




(a) 圖2.2a的成分句法剖析




(b) 圖2.2b的成分句法剖析

### 2.4.2 定義及問題描述

給定一句話 $\mathbf{x} = (w_1, w_2, \ldots, w_N)$，$w_n$ 代表一個詞（word），定義該句裡所有詞的集合為 $W = \{w_i\}|_{i=1}^{N}$。以詞 $w \in W$ 為節點，，及詞之間所有可能的邊 $\mathcal{R} = \{(w_i, w_j) \mid w_i, w_j \in W, w_i \neq w_j\}$：依存句法樹為一單一根有向樹（single-rooted arborescence）$t = (W, R),\ R \subset \mathcal{R}$，亦即其為一滿足以下限制的有向圖：

- 整個有向圖只有一個根節點（root，無入邊的節點）。
- 除了根節點以外，每個節點皆有剛好一個父節點。
- 每個節點都存在剛好一條路徑從根節點到該節點。

在句法學中，每個詞（節點）的父節點稱作該詞的中心詞（head）；而詞擁有的子節點則稱作該詞的依附詞（dependent）。因此給定句子 $\mathbf{x}$，其所有合法單一根有向樹集合 $T(\mathbf{x})$，一剖析器 $\mathbf{x} \mapsto t,\ t \in T(\mathbf{x})$，其目標便是從所有可能的樹集 $T(\mathbf{x})$ 中找出一顆樹 $t'$，使得其與正確的樹 $t^*$ 之差異愈小愈好。

#### 依存關係之標籤

在依存句法樹裡，每個依存關係 $r$ 都會有其標籤 $l$ 描述該依存關係的性質，而剖析器給定句子 $\mathbf{x}$ 預測完依存關係 $R$ 後也需要正確預測每個依存關係 $r \in R$ 的標籤 $l$。不同語言的句法有許多相同與不同之處，為了求同存異，UD 句法樹庫在句法關係的分類上採取兩階層的架構，在第一個階層定義了 37 種所有語言共享的普適句法關係（universal syntactic relations），每個語言原本定義的各種語言專屬的句法關係都會被歸類到第一層的普適句法關係中的某一種關係，而語言內的句法關係若需要更細緻的分類，則可以放進第二層的語言專屬關係（language specific relations），標籤的形式為 普適句法關係 : 語言專屬關係。

**評估指標（evaluation metric）**

現行評斷依存句法剖析器輸出好壞的數值為標籤不計依附分數（Unlabeled Attachment Score, UAS）與標籤依附分數（Labeled Attachment Score, LAS）。由於句法剖析樹每個詞都有其中心詞（除了根節點以外）的特性，依附分數利用此性質，以詞為單位，計算剖析器成功預測每個詞的中心詞的比例：

$$\text{標籤不計依附分數} = \frac{\#\text{中心詞與正確答案相同的詞}}{\#\text{詞}}。 \tag{2.19}$$

而沒有中心詞的根節點詞，剖析器則要正確判斷其沒有中心詞。

標籤依附分數則計算中心詞與通往中心詞的邊其依存關係標籤兩者均預測正確的比例：

$$\text{標籤依附分數} = \frac{\#\text{中心詞、依存關係標籤與正確答案相同的詞}}{\#\text{詞}}。 \tag{2.20}$$

### 2.4.3 圖類剖析器（Graph-based Parser）

圖類剖析器的運作方式簡介如下：圖類剖析器為所有合法的單一根有向樹進行評分，其方式為將每顆樹 $t = (W, R)$ 的分數拆分成其所有組成邊的分數之加總：

$$\text{Score}(t) = \exp\left(\sum_{r \in R} \text{s}(r)\right) \tag{2.21}$$

其中 $\text{Score}(t)$ 為樹 $t$ 的評分函數，$\text{s}(r)$ 為邊 $r$ 的評分函數。

| 句法關係 | 英文描述 | 中文描述 |
|---|---|---|
| acl | clausal modifier of noun (adjectival clause) | 形容詞子句 |
| advcl | adverbial clause modifier | 副詞子句 |
| advmod | adverbial modifier | 副詞 |
| amod | adjectival modifier | 形容詞 |
| appos | appositional modifier | 同位語 |
| aux | auxiliary | 助動詞 |
| case | case marking | 格位 |
| cc | coordinating conjunction | 並列連詞 |
| ccomp | clausal complement | 子句補語 |
| clf | classifier | 分類詞 |
| compound | compound | 複合（名詞、動詞等） |
| conj | conjunct | 並列 |
| cop | copula | 系詞 |
| csubj | clausal subject | 子句主詞 |
| dep | unspecified dependency | 未定義依存關係 |
| determiner | determiner | 限定詞 |
| discourse | discourse element | 話語元素 |
| dislocated | dislocated elements | 錯位 |
| expl | expletive | 虛主詞 |
| fixed | fixed multiword expression | 固定多詞表達 |
| flat | flat multiword expression | 扁平多詞表達 |
| goeswith | goes with | 同詞 |
| iobj | indirect object | 間接受詞 |
| list | list | 列舉 |
| marker | marker | 標記 |
| nmod | nominal modifier | 名詞修飾語 |
| nsubj | nominal subject | 名詞主語 |
| nummod | numeric modifier | 數值修飾語 |
| obj | object | 受詞 |
| obl | oblique nominal | 間接格名詞 |
| orphan | orphan | 孤懸 |
| parataxis | parataxis | (句子) 並列 |
| punct | punctuation | 標點 |
| reparandum | overridden disfluency | 修護語 |
| root | root | 根 |
| vocative | vocative | 呼格 |
| xcomp | open clausal complement | 開放子句補語 |

表 2.2: UD 句法樹庫裡的普適句法關係。

而最佳的生成有向樹 $t^*$ 就是分數最高的生成有向樹：

$$t^* = \arg\max_{t} \ \mathrm{Score}(t) = \arg\max_{t=(W,R)\in T(\mathbf{x})} \ \exp\left(\sum_{r\in R} \mathrm{s}(r)\right) \tag{2.22}$$

上述最佳生成有向樹 $t^*$ 可由執行最大生成有向樹演算法（maximum-spanning aborescence algorithm，見演算法1）得出。因此圖類剖析器旨在學習一個評分函數 $\mathrm{s}((w_i, w_j))$ ，使得給定訓練資料裡的樹 $G = (V, E)$，使得正確句法樹中的邊 $(w_i, w_j) \in E$ 其分數 $\mathrm{s}((w_i, w_j))$ 被拉高，其餘沒有出現在句法樹中的邊 $(w_i, w_{j'}) \notin E$ 其分數 $\mathrm{s}((w_i, w_{j'}))$ 被拉低。

**全域似然性（global likelihood）**

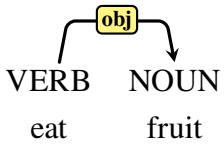
為了拉高正確句法樹的分數與拉低錯誤句法樹的分數，給定參數 $\theta$ 的剖析器機率函數 $p_\theta(t|\mathbf{x}) : T(\mathbf{x}) \to [0, 1]$，我們可以定義句法樹的模型機率為該句法樹的分數除以所有可能句法樹集合 $T$ 的分數總和：

$$p_\theta(t) = \frac{\mathrm{Score}(t)}{\sum\limits_{t' \in T} \mathrm{Score}(t')} \circ \tag{2.23}$$

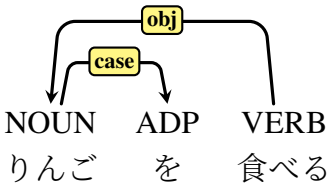
其中分母又稱為句法樹的配分函數（partition function）。分母的配分函數看似難以計算，幸虧所有可能句法樹的分數總和可以透過克氏矩陣-樹定理（Kirchhoff’s Matrix-Tree Theorem）快速計算出來 [30]。因此我們只要優化正確句法樹的機率 $p_\theta(t)$，正確句法樹的分數與錯誤句法樹的分數自然會分別被拉高與拉低。2007 年的古氏（Terry Koo）[31] 與 2017 年的馬氏（Xuezhe Ma）[32] 均採用該定理有效率地計算配分函數。

**中心語選擇交叉熵（head-word selection cross-entropy）**

相較於上節使用矩陣-樹定理得出精確的句法樹機率，多氏 [33] 在該篇論文中採用較為簡單的演算法近似句法樹機率；如前述，由於句子中的每個詞都剛好需要從其他詞中選擇一詞作為其中心詞（或者指定該詞為「根」），令 $h(w)$ 為詞 $w$ 的中心




(a) 英文有中心詞前置的傾向。




(b) 日文有中心詞後置的傾向。

圖 2.4: 不同語言有不同的中心詞方向性參數。

詞，$W^+ = W + \{\texttt{root}\}$（$(\texttt{root}, w)$ 代表 $w$ 為根節點），他將句法樹的機率用每個詞選擇其中心詞的機率相乘來代表：

$$p_\theta(t) = \prod_{w \in W} \frac{e^{\mathrm{s}((h(w),w))}}{\sum_{w' \in W^+, w' \neq w} e^{\mathrm{s}((w',w))}} \circ \tag{2.24}$$

### 2.4.4 中心詞方向性（head-directionality）

不同語言的語法其中一個關鍵的差異為中心詞的方向性。若一種語言的依存關係其中心詞在依附詞之前的頻率較高，則我們稱此語言為中心詞前置（head-initial）的語言；反之，若一種語言的依存關係其中心詞在依附詞之後的頻率較高，則我們稱此語言為中心詞後置（head-final）的語言。圖2.4以英文及日文為例子，說明中心詞方向性在依存句法樹上的實例。圖2.5則展示了 UD 句法樹庫收錄所有含有訓練集的語言其中心詞方向性，可以看到不同語言的方向性有著不小的差異，從中心詞前置傾向明顯的阿拉伯語（ar）到嚴格遵守中心詞後置規則的日語（ja），展現了人類語言在方向性上的多樣性。

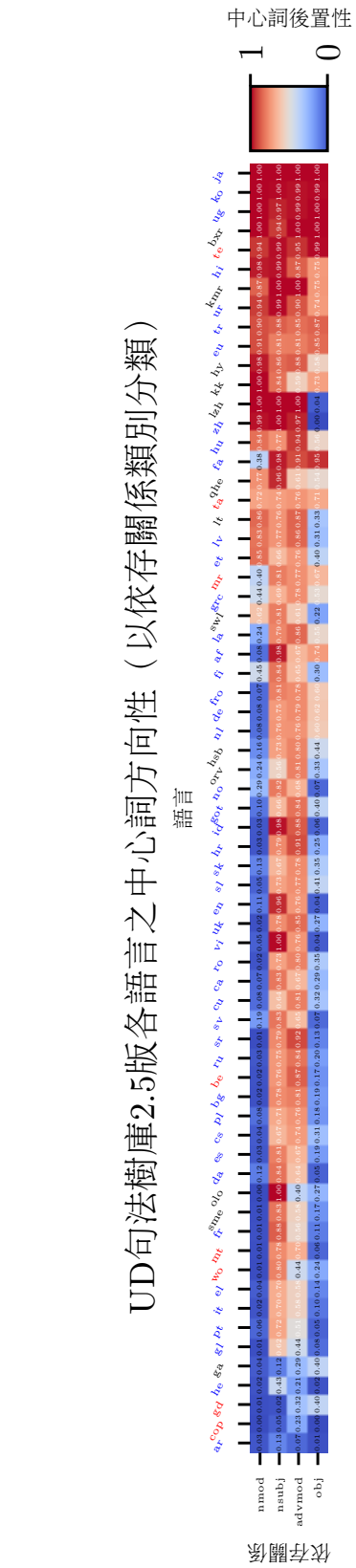



圖 2.5: UD 句法樹庫 2.5 版各語言之中心詞方向性。圖中數字為每種關係中心詞後置佔該關係所有出現次數的比例。藍色與紅色的語言分別為訓練與未見過語言。

**procedure 最大生成有向樹** $(G, s)$

$G = (V,\ E)$

邊評分函數 $s : E \to \mathbb{R}$

$E' = \{(w_i, w_j) \,|w_j \in V, w_i = \arg\max_{w_i} s\,(w_i, w_j)\}$

$G' = (V, E')$

**if** $G'$ 無環 **then**

回傳 $G'$

**else**

尋找一邊集合 $E_C$ 使得其為 $G'$ 中的環

$G_C =$ **收束** $(G', E_C, s)$

$\mathbf{y} =$ **最大生成有向樹** $(G_C, s)$

尋找一節點 $v \in C$ 使得 $(v', v) \in \mathbf{y}, (v'', v) \in C$

回傳 $\mathbf{y} \cup C - \{(v'', v)\}$

**procedure 收束** $(G, E_C, s)$

令 $G_C$ 為 $G$ 除去 $C$ 中的節點後的子圖

將節點 $c$ 加入 $G_C$ 中，代表原本的環 $C$

**for** $v \in V - C : \exists_{v' \in C}\ (v', v) \in E$ **do**

將邊 $(c, v)$ 加入 $G_C$, 其分數 $s(c, v) = \max_{v' \in C} s(v', v)$

**for** $v \in V - C : \exists_{v' \in C}\ (v, v') \in E$ **do**

將邊 $(v, c)$ 加入 $G_C$,

$s(v, c) = \max_{v' \in C} [s(v, v') - s(a(v'), v') + s(C)]$,

$a(v)$ 為 $v$ 之父節點 $C$,

$s(C) = \sum_{v \in C} s(a(v), v)$

回傳 $G_C$

Algorithm 1: 最大有向樹演算法。

## 2.5 基於優化的元學習（Optimization-based Meta Learning）

### 2.5.1 模型無關元學習（Model-agnostic Meta Learning，MAML）

模型無關元學習於 2017 年由芬氏 (Chelsea Finn) 提出，為基於優化的元學習方法的一種，將元學習的宗旨「學習如何學習」（learning to learn）理解為對某些任務學習一個好的初始模型參數，使得模型在碰到新任務時能夠學得更快更好。

現在我們將上面的敘述改寫為數學語言：給定一任務分佈 $\mathcal{T}$、從任務分佈中採樣出的任務 $\tau \sim \mathcal{T}$、每個任務對應的訓練損失函數 $\mathcal{L}_{\tau,A}$ 與測試損失函數 $\mathcal{L}_{\tau,B}$ 、模型無關元學習目標是找尋一參數 $\phi$，使得參數在任務分佈下採樣出的每個任務，用訓練資料分別進行 $k$ 次更新後的參數（此步驟稱為內循環，inner-loop），在各自任務上的測試損失函數最小：

$$\phi^* = \arg\min_{\phi} \mathbb{E}_{\tau\sim\mathcal{T}} \left[\mathcal{L}_{\tau,B}\left(U^k_{\tau,A}(\phi)\right)\right] \tag{2.25}$$

$$U^k_{\tau}\left(\phi\right) = \underbrace{U_{\tau}(...U_{\tau}(U_{\tau}}_{\texttt{k times}}(\phi))...) \tag{2.26}$$

其中更新函數 $U_{\tau}(\phi)$ 可以用任何優化器實現，比如陽春 SGD（Vanilla SGD）：

$$U_{\tau,\text{SGD}}\left(\phi\right) = \phi - \alpha\nabla_{\phi}\mathcal{L}_{\tau}\left(\phi\right) \tag{2.27}$$

或者 Adam 優化器 $U_{\tau,\text{ADAM}}(\phi)$[34]。模型無關元學習針對式2.25的解法是再做一次

梯度下降法（此步驟稱為外循環，outer-loop）：

$$g_{\text{MAML}} = \frac{\partial}{\partial \phi} \mathcal{L}_{\tau,B} \left( U_{\tau,A}(\phi) \right) \tag{2.28}$$

$$= U'_{\tau,A}(\phi) \mathcal{L}'_{\tau,B}(\widetilde{\phi}), \quad \text{其中} \quad \widetilde{\phi} = U_{\tau,A}(\phi) \tag{2.29}$$

式2.29的 $U'_{\tau,A}(\phi)$ 為更新函數 $U_{\tau,A}$ 的雅可比矩陣（Jacobian Matrix），含有參數的二階導數。下以陽春 SGD 為例，推導為何上式有二階導數：

令初始參數為 $\phi_0$，更新 $k$ 次後的參數為 $\phi_k$，則更新 $k$ 次的函數的導數（以下省略 $A$）$\left(U_\tau^k\right)'(\phi_0)$ 可以寫成：

$$\left(U_\tau^k\right)'(\phi_0) = \frac{\partial \phi_k}{\partial \phi_0} \tag{2.30}$$

$$= \prod_{i=0}^{k-1} \frac{\partial \phi_{i+1}}{\partial \phi_i} \tag{2.31}$$

我們將單步更新的導數寫出來：

$$\frac{\partial \phi_{i+1}}{\partial \phi_i} = \frac{\partial}{\partial \phi_k} \left( \phi_k - \alpha \nabla_{\phi_k} \mathcal{L}_\tau \left( \phi_k \right) \right) \tag{2.32}$$

$$= I - \alpha \nabla^2_{\phi_k} \mathcal{L}_\tau \left( \phi_k \right) \tag{2.33}$$

因此 $k$ 步更新的導數可以寫成單步更新參數之黑塞矩陣 (Hessian) 的連乘：

$$\left(U_\tau^k\right)'(\phi_0) = \prod_{i=0}^{k-1} \left( I - \alpha \nabla^2_{\phi_i} \mathcal{L}_\tau \left( \phi_i \right) \right) \tag{2.34}$$

代入式2.25後，我們得到

$$g^k_{\text{MAML}} = \left(U^k_{\tau,A}\right)'(\phi)\mathcal{L}'_{\tau,B}(\widetilde{\phi}) \tag{2.35}$$

$$= \mathcal{L}'_{\tau,B}(\phi_k)\prod_{i=0}^{k-1}\left(I - \alpha\nabla^2_{\phi_i}\mathcal{L}_{\tau,A}\left(\phi_i\right)\right) \tag{2.36}$$

$$= \nabla_{\phi_k}\mathcal{L}_{\tau,B}(\phi_k)\prod_{i=0}^{k-1}\left(I - \alpha\nabla^2_{\phi_i}\mathcal{L}_{\tau,A}\left(\phi_i\right)\right) \tag{2.37}$$

由於模型的參數維度非常大，計算其黑塞矩陣需要耗費大量計算資源，因此之後的研究者紛紛對模型無關元學習提出不需要計算黑塞矩陣的一階近似元學習方法。以下介紹一階模型無關元學習（First-order MAML）與爬蟲類元學習（Reptile）[35] 兩種變形。

### 2.5.2 一階模型無關元學習（First-order MAML）

一階模型無關元學習直接將原本模型無關元學習之更新函數的導數 $\left(U^k_{\tau,A}\right)'(\phi)$ 設為單位矩陣：

$$g^k_{\text{FOMAML}} = g^k_{\text{MAML}}\Big|_{\left(U^k_{\tau,A}\right)'(\phi)=I} \tag{2.38}$$

$$= \mathcal{L}'_{\tau,B}(\phi_k) \tag{2.39}$$

### 2.5.3 爬蟲類元學習（Reptile）

爬蟲類元學習直接將其梯度設爲各個任務參數更新前後差值 $\phi - U^k_{\tau_A}(\phi)$ 的平均：

$$g^k_{\text{REP}} = \mathbb{E}_\tau\left[\left(\phi - U^k_{\tau_A}(\phi)\right)\right] \tag{2.40}$$

當 $k = 1$ 時，此演算法與普通的多工訓練（multi-task training）並無二致。但當 $k > 1$ 時，此更新含有 $L_{\tau_A}$ 的高次導數，使得其行為與多工訓練多所不同。

# 第 三 章　使用元學習在資料不足的去詞化依存句法剖析

## 3.1　簡介

本章介紹使用模型無關元學習進行多語言預訓練，以幫助資料不足語言的依存句法剖析的系列實驗。

模型無關元學習本來是為了處理圖片分類領域少量樣本學習的問題，不久後也為資料不足（但資料量大於少量樣本學習）的學習任務所用，如古氏（Jiatao Gu）將元學習方法引入資料不足的機器翻譯，並勝過使用普通多工學習的基準模型 [36]。芬氏 (Chelsea Finn) 在 2018 年提出的模型無關元學習 (model-agnostic meta-learning) [37] 為所有使用梯度下降法 (gradient descent) 進行最佳化的模型提供了一項簡潔且有效的方法處理資料不足任務。在語言轉移學習的框架下，其目標是替未見過的語言 (unseen languages) 尋找一合適參數初始值，使得少量步數梯度更新後，參數在該語言的測試集上表現最佳，其稱此在少量步數更新就能大幅增進爲見過語言表現的能力為「快速適應」（fast adaptation）[37]。

在元學習出現以前，若希望利用相似語言中所蘊含的資訊給予模型語言普遍具備的歸納偏置（下稱普適語言偏置，universal linguistic biases），多工學習（multi-task learning）為一主要的方法 [38]。藉由共享特徵抽取網路並用以同時訓練多種相關語言，相關語言的資訊得以透過反向傳播（backpropagation）注入類神經網路中，使模型相較於解釋單一語言，更加偏好能夠解釋所有相關語言的假說，有效幫助模型達成泛化（generalization）。

然而多工訓練的目標，是提高訓練語言 (training languages) 在其測試集 (testing

set) 上的準確率，而提高訓練語言的準確率，未必就代表在資料不足語言上的準確率也會隨之提高；有可能出現訓練語言與資料不足語言差異過大，而導致多工訓練模型無法幫助資料不足語言的任務表現。

不若單純的多工訓練，模型無關元學習於訓練階段的目標並非提高在訓練語言上的表現，而是直接最佳化模型在未見過語言上，訓練與測試環境沒有不匹配之處，希望減輕模型只在訓練語言的測試集上有好表現，而無法推廣到資料不足語言上的問題。

本章為了了解模型無關元學習及其各種一階近似的變形（見第2.5節）在多語言依存句法剖析上的行為，去除詞的影響，只使用詞性標記作為特徵進行依存句法剖析。另外，為了更深入了解不同模型組態（model configurations）對於模型無關元學習系列方法的影響，以模型大小為操作變因，試圖了解小模型與大模型是否會改變不同方法對依存句法剖析的表現。其次，調整不同模型無關元學習內循環步數，並觀察其與依存句法剖析的準確率之間的關係。最後，藉由觀察不同方法經過不同精細校正階段後在目標語言方向性分佈的改變軌跡，試圖分析模型無關元學習是否有達到其宣稱的快速適應的能力。

## 3.2 多語言去詞化依存句法剖析（multilingual delexicalized dependency parsing）

由於詞化依存句法剖析有太多變因，包括使用的預訓練模型，其對不同語言的偏置等等，都會影響句法剖析模型於目標語言上的行為；且詞化依存句法剖析參數量較多，使用二次微分的計算量與佔用空間均較大，訓練耗時，有時甚至會發生用來進行平行矩陣運算的圖形處理器記憶體不足的問題。因此為了排除語言本身

句法以外性質對句法剖析的影響及計算資源的考量，本節先進行去詞化依存句法剖析的實驗，也就是只使用句法樹庫提供的詞性標記做為詞的表徵進行句法剖析，詞化依存句法剖析留待第4章再行探討。

### 3.2.1 詞性標記（POS tags）

UD 句法樹庫中大部分的語言均提供兩種詞性標記：專為該語言設計的詞性標記（XPOS），通常為該句法樹庫尚未整合進 UD 時原本的詞性標記，與各語言統一的普適詞性標記 [39]（Universal POS tags, UPOS），其捨棄各語言細緻的詞性分別，整合語言間相似性質的詞性，以達到所有語言共享同一組詞性集合的目標。如前置介系詞（prepositions）與後置介系詞（postpositions）在 UPOS 的框架下就被整合成介系詞（adpositions）而不做前後置之分。由於 UPOS 有更好的跨語言通用性，本研究去詞化依存句法剖析的詞性標記輸入均使用 UPOS。

### 3.2.2 圖類剖析器 – 深層雙仿射層注意力網路（Graph-based Parser – Deep Biaffine Attention）

2017 年由多氏 [33] 提出的深層雙仿射層注意力網路（下稱**雙仿射**），憑藉其簡單的模型架構及強大的實務表現，成爲近年來最常被採用的句法剖析模型。與其他圖類剖析器一樣，**雙仿射**的目標為學習邊評分分數：$s(w_i, w_j)$，使得正確句法樹出現的可能性提高。

給定編碼器函數 $\mathbf{r}(w) \in \mathbb{R}^n$、雙線性矩陣 $\mathbf{U}^{(1)} \in \mathbb{R}^{n \times n}$、線性矩陣 $U^{(2)}$, $U^{(3)} \in \mathbb{R}^n$ 與偏差 $\mathbf{b}$，**雙仿射**的邊評分函數為：

$$s(w_i, w_j) = \mathbf{r}(w_i)^\top \mathbf{U}^{(1)} \mathbf{r}(w_j) + \mathbf{r}(w_i)^\top U^{(2)} + \mathbf{r}(w_j)^\top U^{(3)} + \mathbf{b} \,. \tag{3.1}$$

上式可分為三部分解釋：$\mathbf{r}(w_i)^\top U^{(2)}$ 代表 $w_i$ 接受任何依附詞的可能性；$\mathbf{r}(w_j)^\top U^{(3)}$ 代表 $w_j$ 接受任何中心詞的可能性；$\mathbf{r}(w_i)^\top \mathbf{U}^{(1)}\mathbf{r}(w_j)$ 則代表 $w_i$ 與 $w_j$ 之間存在連結（邊）的可能性。

### 3.2.3 多工學習基準模型（multi-task baseline）

多工學習要求單一參數同時在各個語言上取得好表現，因此其演算法的參數更新梯度為當前參數在所有語言的損失之期望值對該參數的微分：

$$g_{\text{multi}} = \nabla_\phi \left( \mathbb{E}_\tau \left[ \mathcal{L}_\tau \left( \phi \right) \right] \right) \tag{3.2}$$

### 3.2.4 修訂版爬蟲類元學習

原始版本的爬蟲類元學習 [35] 無論是在內循環或外循環的開頭都不會重新啟動內循環的優化器，當使用有動能（momentum）的優化器如 Adam 時，當下的梯度更新會受之前用其他語言計算而得的梯度影響，造成語言間不必要的干擾。為了避免此現象，原作者將一階動能項 $\beta_1$ 設為 0。然而初始實驗發現 $\beta_1 = 0$ 的 Adam 在精細校正時時會對準確率造成負面影響。再者，爬蟲類元學習將外循環原始梯度（raw gradient）直接設為各語言內循環（亦即精細校正）前後參數的差異的平均，此數值大小很大程度上取決於內循環優化器的學習率，恐與一般模型精細校正時接收到的梯度分佈差異過大。

為處理此問題，本研究稍稍修改了爬蟲類元學習的演算法，希望在不改變內循環優化器設置的前提下保留爬蟲類元學習加總內循環所有梯度的優點：與其將內循環梯度設為前後參數的差異的平均 $\phi - U^k_{\text{ADAM}}(\phi)$，內循環原始梯度經過內循環優化器處理過後的產物，本研究將內循環梯度直接設為內循環**原始**梯度的平均

（此處優化器以 Adam 為例）：

$$g_{\text{REP}} = \mathbb{E}_{\tau}\left[\frac{1}{k}\sum_{i=1}^{k}\nabla_{\phi_i}\mathcal{L}_{\tau}\left(\phi_i^{\texttt{adam}}\right)\right] \tag{3.3}$$

其中

$$\phi_i^{\texttt{adam}} = U_{\text{ADAM}}\left(\phi_{i-1}^{\texttt{adam}}\right). \tag{3.4}$$

此數值雖然是由內循環優化器計算出來，但此處取其原始梯度，受內循環學習率影響較小。此修訂版爬蟲類元學習與普通的多工學習的差異比起原始的爬蟲類元學習要來的更小：外循環的梯度為內循環原始梯度的平均，與多工學習類似；但原始梯度仍是由內循環更新過的參數計算出來的（除了內循環的第一步），更接近原本的爬蟲類元學習。初始實驗發現此修訂版爬蟲類元學習表現不俗，且相較原始爬蟲類元學習的表現來的更加穩定。本研究稍稍濫用爬蟲類元學習的名稱，往後皆以爬蟲類元學習指稱本節所提出的修訂版爬蟲類元學習。

## 3.3 實驗設置

我們從 53 種訓練語言（73 個訓練句法樹庫）中選取有官方驗證集（development set）的 46 種訓練語言（66 個訓練句法樹庫）作爲訓練語言（見表3.2）。預訓練完成後，我們分別對該模型進行。我們挑選 UD 2.5 版中 8 種不在訓練語言中的語言做為測試用的未見過語言（unseen languages，見表3.3）。精細校正時，為排除語料多寡對各語言表現的影響，各語言從中取樣 96 句句子做為精細校正用語料。由於批次大小為 16（句），因此精細校正 1 回合相當於梯度更新 6 步。為觀察各方法精細校正不同步數後的進步軌跡，每次實驗均呈現各方法精細校正前、精細校正 1 步（1/6 回合）、1 回合與 80 回合的 UAS/LAS。實驗在訓練與測試時時均使用正確

| 超參數 | 值 |
|---|---|
| 詞性嵌入維度 | 100 |
| 詞性嵌入維度（小模型） | 10 |
| 編碼器 | 雙向 LSTM |
| 編碼器層數 | 3 |
| 編碼器隱維度 | 100 |
| 編碼器隱維度（小模型） | 10 |
| 依存邊維度 | 200 |
| 依存邊維度（小模型） | 20 |
| 依存標籤維度 | 200 |
| 依存標籤維度（小模型） | 20 |
| 詞性丟棄機率 | 0.33 |
| 批次大小 $b$ | 16 |
| 語言數 $l$ | 10 |
| 訓練樣本數/回合 | 64000 |
| 訓練回合數 | 10 |
| 優化器 | Adam |
| $\beta_1, \beta_2$ | 0.9, 0.9 |
| 權重衰減參數 | 0.01 |
| 基礎學習率 | $3e^{-4}$ |
| 最大梯度範數 | 5.0 |
| 內循環步數 | 2 |

(a) 預訓練超參數。

| 超參數 | 值 |
|---|---|
| 優化器 | Adam |
| 基礎學習率 | $3e^{-4}$ |
| $\beta_1, \beta_2$ | 0.9, 0.9 |
| 批次大小 | 16 |
| 訓練回合數 | 80 |
| 最大梯度範數 | 5.0 |

(b) 精細校正超參數。

表 3.1: 去詞化依存句法剖析模型超參數一覽。

| 語言 | 句法樹庫編碼 |
|---|---|
| Afrikaans | af_afribooms |
| Ancient Greek | grc_proiel |
| Ancient Greek | grc_perseus |
| Arabic | ar_padt |
| Basque | eu_bdt |
| Bulgarian | bg_btb |
| Catalan | ca_ancora |
| Chinese | zh_gsd |
| Croatian | hr_set |
| Czech | cs_cac |
| Czech | cs_fictree |
| Czech | cs_pdt |
| Danish | da_ddt |
| Dutch | nl_alpino |
| Dutch | nl_lassysmall |
| English | en_ewt |
| English | en_gum |
| English | en_lines |
| Estonian | et_edt |
| Finnish | fi_ftb |
| Finnish | fi_tdt |
| French | fr_gsd |
| French | fr_sequoia |
| French | fr_spoken |
| Galician | gl_ctg |
| Galician | gl_treegal |
| German | de_gsd |
| Gothic | got_proiel |
| Greek | el_gdt |
| Hebrew | he_htb |
| Hindi | hi_hdtb |
| Hungarian | hu_szeged |
| Indonesian | id_gsd |

| 語言 | 句法樹庫編碼 |
|---|---|
| Italian | it_isdt |
| Italian | it_postwita |
| Japanese | ja_gsd |
| Korean | ko_gsd |
| Korean | ko_kaist |
| Latin | la_ittb |
| Latin | la_proiel |
| Latin | la_perseus |
| Latvian | lv_lvtb |
| Norwegian | no_bokmaal |
| Norwegian | no_nynorsk |
| Norwegian | no_nynorsklia |
| Old Church Slavonic | cu_proiel |
| Old French | fro_srcmf |
| Persian | fa_seraji |
| Polish | pl_lfg |
| Polish | pl_sz |
| Portuguese | pt_bosque |
| Romanian | ro_rrt |
| Russian | ru_syntagrus |
| Russian | ru_taiga |
| Serbian | sr_set |
| Slovak | sk_snk |
| Slovenian | sl_ssj |
| Slovenian | sl_sst |
| Spanish | es_ancora |
| Swedish | sv_lines |
| Swedish | sv_talbanken |
| Turkish | tr_imst |
| Ukrainian | uk_iu |
| Urdu | ur_udtb |
| Uyghur | ug_udt |
| Vietnamese | vi_vtb |

表 3.2: 預訓練所使用的訓練句法樹庫/語言。

的斷句、斷詞，並使用句法樹庫提供的正確詞性做為詞的表徵。

至於多語言訓練的部分，孔氏 [40] 與烏氏 [41] 進行多語言訓練的方法，是將全部語言的句法樹庫接在一起、在一個小批次 (batch) 中混合多個句法樹庫訓練。這樣的做法可能會導致資料量大的語言取樣頻率過高；我們的方法則是每次更新從全部語言裡取樣 $l$ 種語言，每種語言取樣 $b$ 個句子，一個批次總共有 $b \times l$ 個句

| 語言 | 句法樹庫編碼 |
|---|---|
| Wolof | wo_wtb |
| Scottish Gaelic | gd_arcosg |
| Coptic | cop_scriptorium |
| Telugu | te_mtg |
| Belarusian | be_hse |
| Marathi | mr_ufal |
| Maltese | mt_mudt |
| Tamil | ta_ttb |

表 3.3: 測試用未見過的句法樹庫/語言。

子。不同於孔氏與烏氏，這樣的方法防止模型過度對資料充足語言的特性建模，但也可能使得資料不足語言的句子被過度取樣而產生過擬合的現象。

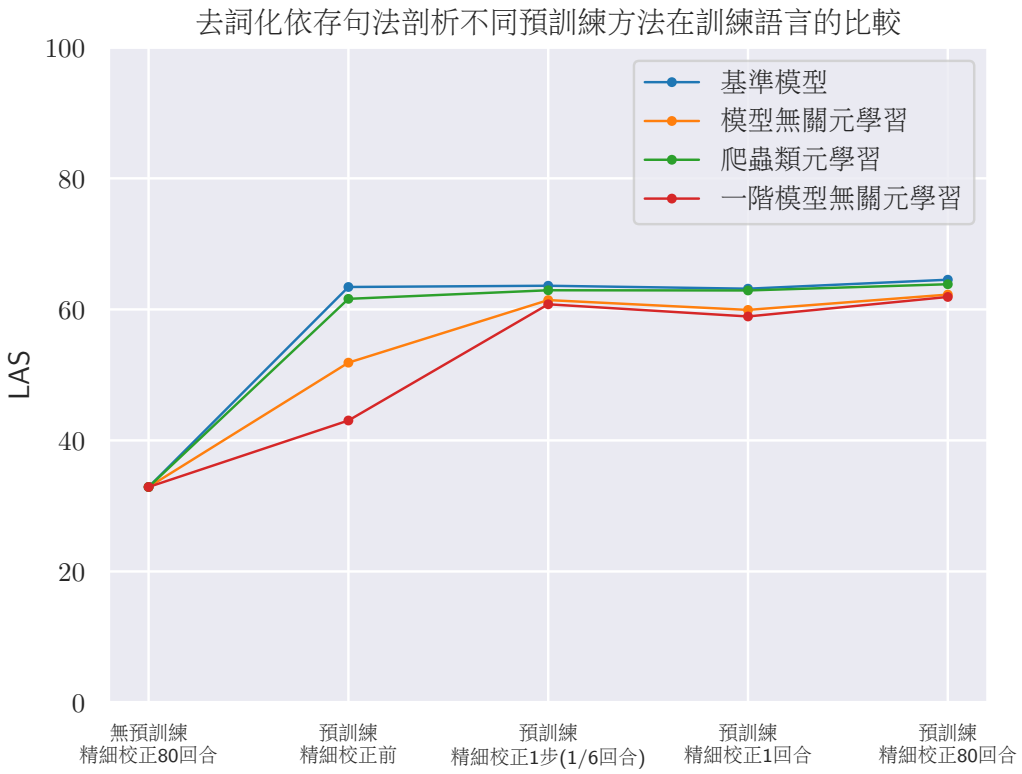
超參數的設置見表3.1。此處本章以模型大小做為操縱變因，探討其對不同預訓練方法的影響。其中一般模型的詞性標記嵌入維度、編碼器的輸入維度與編碼器的隱維度均設為 100；經過雙向 LSTM 之後輸出維度為 200 的向量，因此將其後的依存邊與依存標籤維度設為 200。而小模型詞性標記嵌入維度、編碼器的輸入維度與編碼器的隱維度則設為 10，經過雙向 LSTM 之後輸出維度為 20 的向量，其後的依存邊與依存標籤維度設為 20，均為一般模型的十分之一。

## 3.4 實驗結果

以下比較去詞化依存句法剖析各方法表現、分析不同內循環步數對各種模型無關元學習方法的影響、並觀察小模型在上述設置下與普通模型有何不同之處。

### 3.4.1 去詞化依存句法剖析不同方法比較

圖3.1為去詞化依存句法剖析不同預訓練方法產生的模型在目標語言上經過不同步數的精細校正後的測試集 LAS 數值。由圖中可以觀察到：



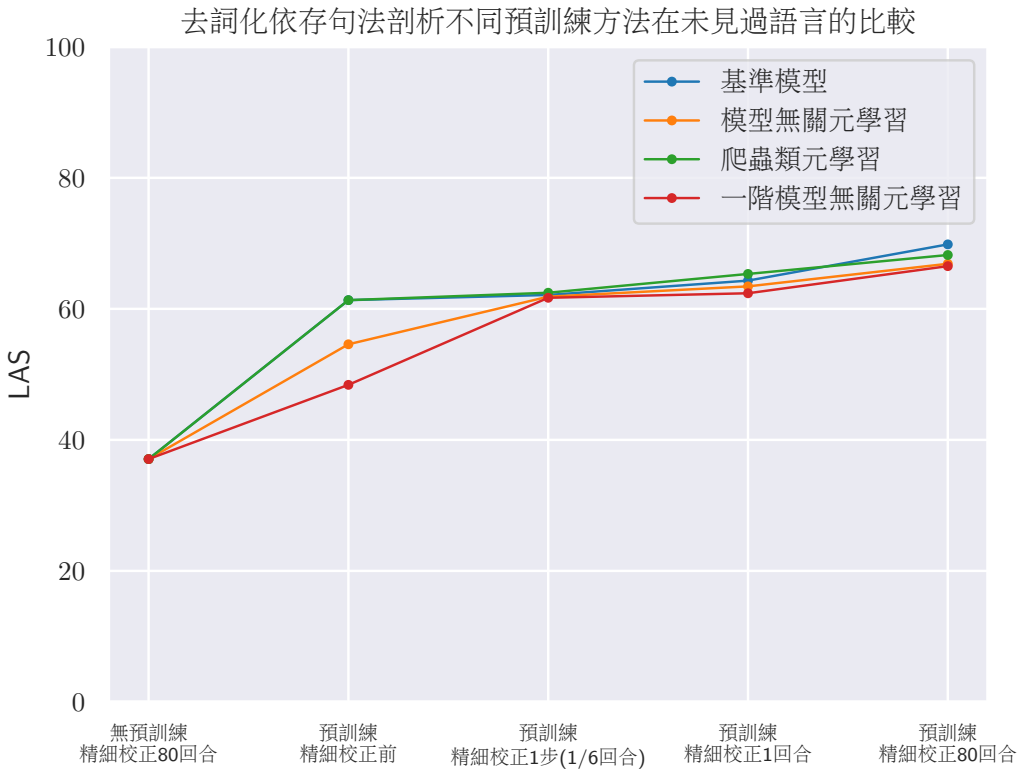




圖 3.1: 去詞化依存句法剖析不同預訓練方法精細校正後的平均 LAS 折線圖。

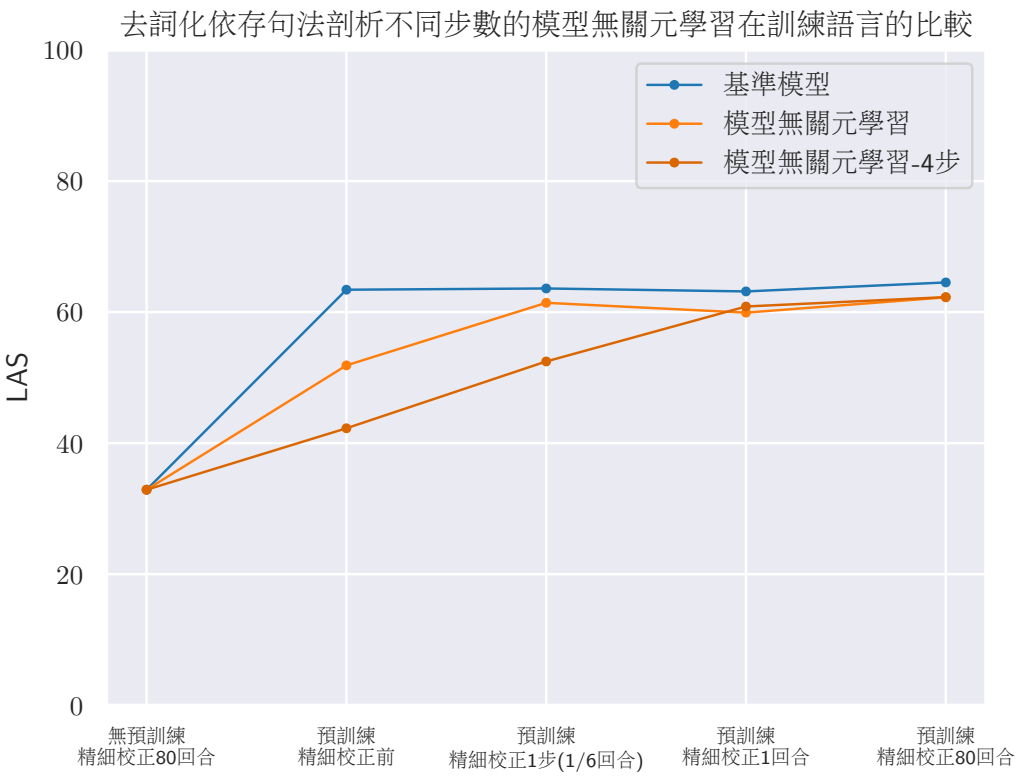
- 在訓練語言上，基準模型無論接觸多少目標語言的資料（0 回合、1 步、1 回合、80 回合），其表現均大幅贏過模型無關元學習模型，這說明基準模型的確達到其訓練目標-同時剖析所有的訓練語言。

- 在未見過語言上，模型無關元學習模型精細校正 1 步的 LAS 相對於精細校正前 LAS 的進步量贏過基準模型與爬蟲類元學習模型，顯示模型無關元學習方法的確有快速適應的能力，但其代價是在精細校正前的 LAS 大幅落後另外兩者；且精細校正 1 步、1 回合與 80 回合的表現均略遜基準模型與爬蟲類元學習模型，說明快速適應後的模型無關元學習無法利用更多的資料與更多的精細校正步數持續精進。

- 在未見過語言上，爬蟲類元學習模型精細校正前、精細校正 1 步與 1 回合的表現均可與基準模型匹敵或略勝一籌，只有在精細校正 80 回合的準確率稍遜基準模型，顯示其做為基準模型與模型無關元學習混和的方法，既取基準模型同時剖析所有語言的能力，又能如模型無關元學習模型在接觸少量未見過語言句法資料下有所進步，為模型無關元學習方法中最佳。

- 在所有語言上，一階模型無關元學習模型精細校正前後的分數雖然落後其他所有方法，但在精細校正一步後展現出所有模型最大的進步量（相對於精細校正前）；顯示其預訓練後的參數中仍蘊含快速適應所需的資訊，惟精細校正前大幅落後的數值說明其做為同時剖析所有語言的模型可能較不適合。

每個語言各自詳細的 UAS/LAS 數值可見圖5.1、5.2、5.3與5.4。各方法以語言為單位統計顯著勝過其他方法的次數可見表3.4。



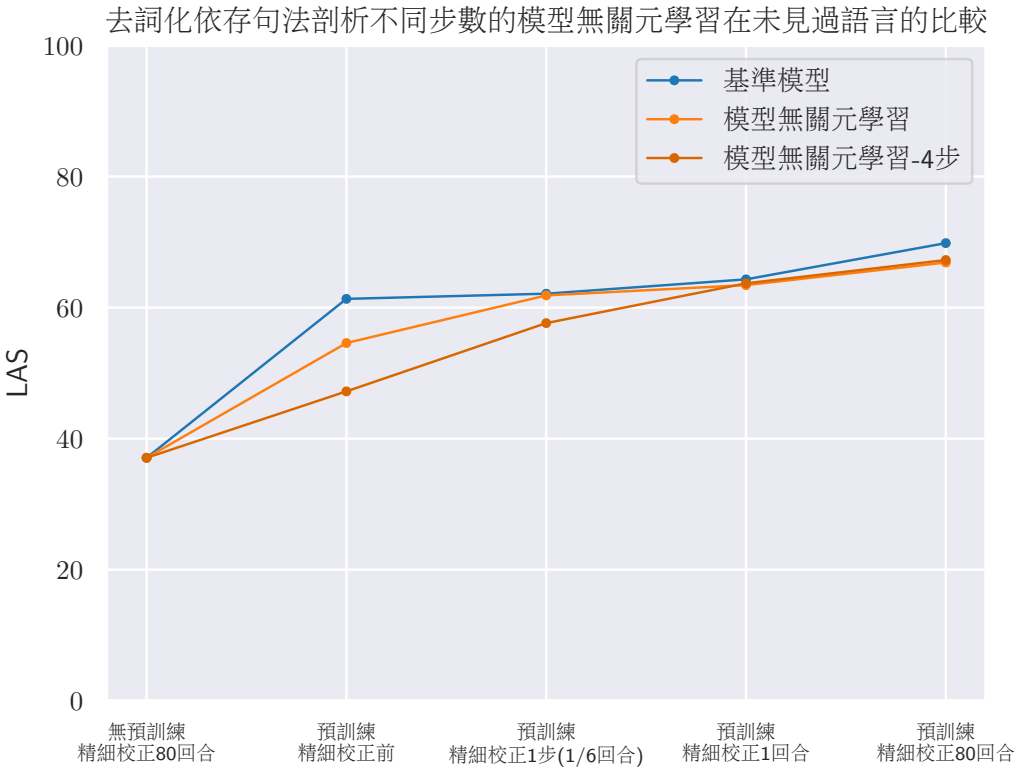
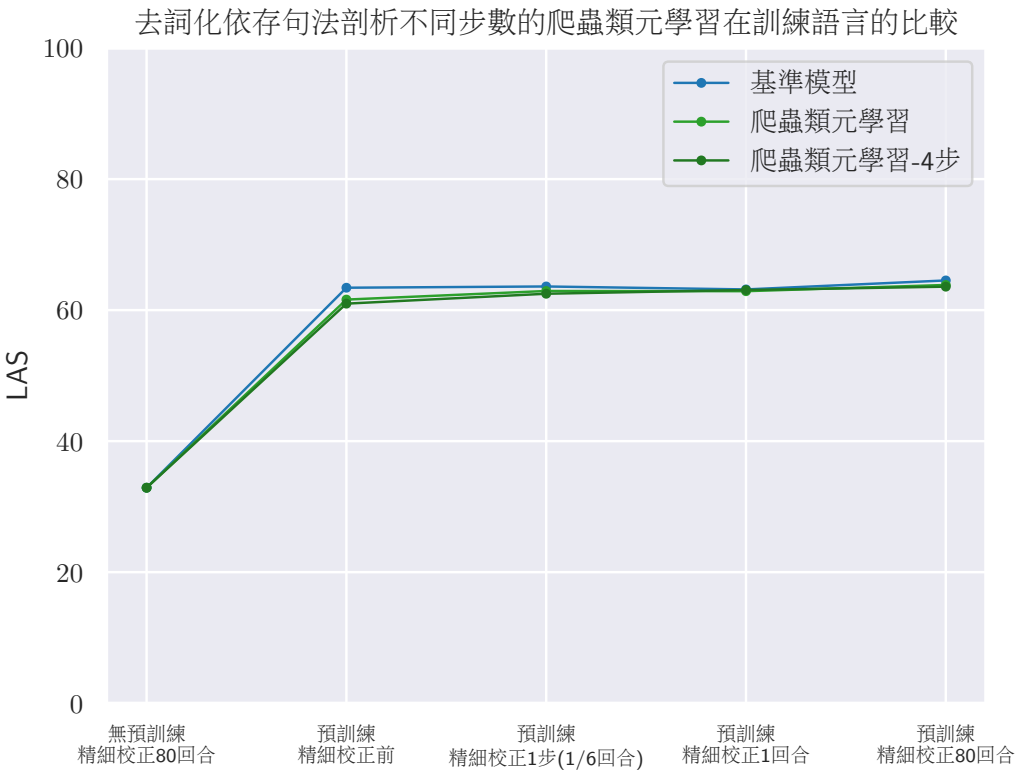




圖 3.2: 去詞化依存句法剖析不同步數的模型無關元學習精細校正後的平均 LAS 折線圖。



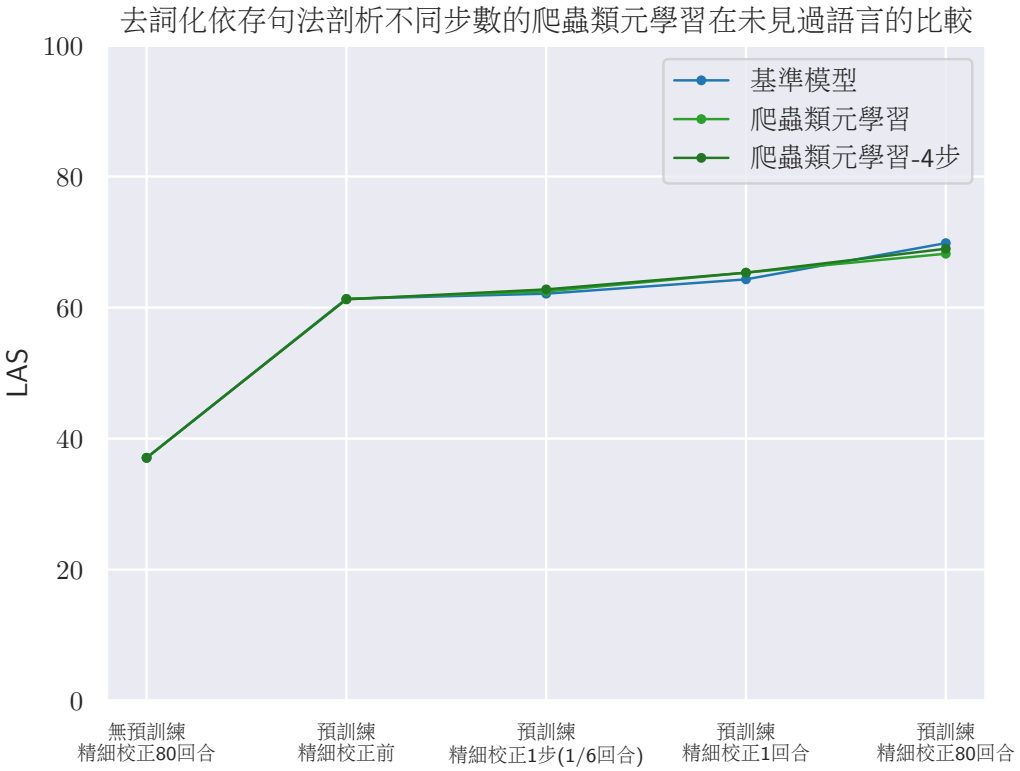
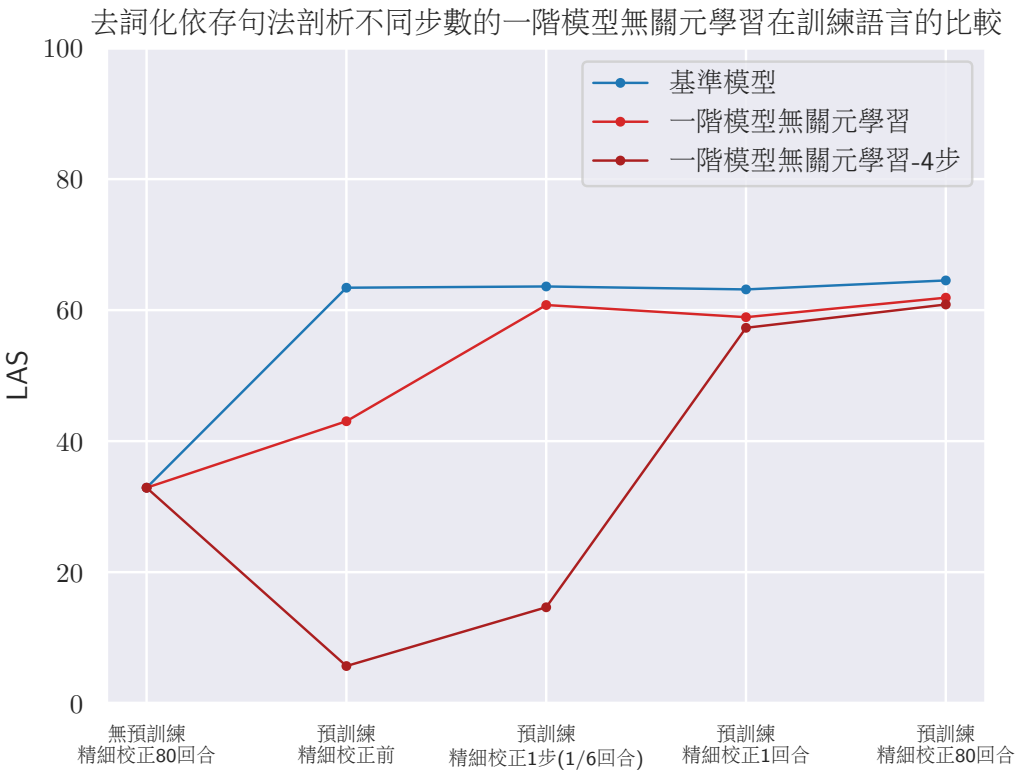




圖 3.3: 去詞化依存句法剖析不同步數的爬蟲類元學習精細校正後的平均 LAS 折線圖。



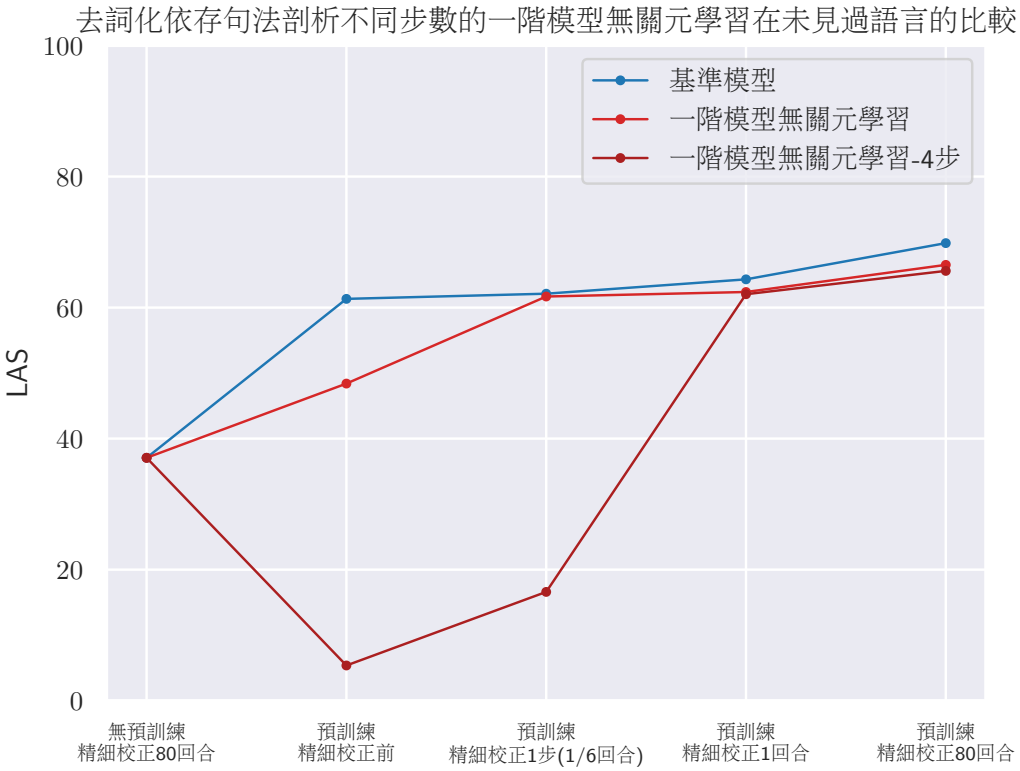




圖 3.4: 去詞化依存句法剖析不同步數的一階模型無關元學習精細校正後的平均 LAS 折線圖。

| 精細校正回合 | 基準模型 | 模型無關元學習 | 爬蟲類元學習 | 一階模型無關元學習 | 無 |
|---|---|---|---|---|---|
| 精細校正前 | **11** | 0 | 0 | 0 | 2 |
| 精細校正 1 步 | **5** | 1 | 0 | 0 | 7 |
| 精細校正 1 回合 | **4** | 0 | 1 | 0 | 8 |
| 精細校正 80 回合 | **7** | 0 | 2 | 0 | 4 |

(a) 訓練語言部份。

| 精細校正回合 | 基準模型 | 模型無關元學習 | 爬蟲類元學習 | 一階模型無關元學習 | 無 |
|---|---|---|---|---|---|
| 精細校正前 | **1** | 0 | **1** | 0 | 6 |
| 精細校正 1 步 | 0 | **1** | **1** | 0 | 6 |
| 精細校正 1 回合 | 1 | 0 | **2** | 0 | 5 |
| 精細校正 80 回合 | **4** | 0 | 0 | 0 | 4 |

(b) 未見過語言部份。

表 3.4: 去詞化依存句法分析各預訓練方法在各精細校正階段 LAS 統計顯著勝過所有其他方法次數。

### 3.4.2 去詞化依存句法剖析各方法不同內循環步數比較

圖3.2、3.3與3.4 為去詞化依存句法剖析各個預訓練方法不同內循環步數在目標語言上經過不同步數的精細校正後的測試集 LAS 數值。未註明內循環步數之方法其步數皆為 2 步。我們有幾項發現：

- 圖3.2中，模型無關元學習-2 步在精細校正前與精細校正 1 步均贏過模型無關元學習-4 步，而模型無關元學習-4 步則在精細校正 1 回合與 80 回合略勝一籌，說明內循環步數愈長、達到好表現所需要的精細校正步數愈長；短內循環步數更快適應未見過語言，但長內循環步數經過較多的精細校正步數後效果愈佳，均與其各自優化的目標相符。

- 圖3.3中，爬蟲類元學習-2 步在訓練語言上稍較爬蟲類元學習-4 步為佳，但在未見過語言上爬蟲類元學習-4 步卻略為反超爬蟲類元學習-2 步，可看出爬蟲類元學習愈長的內循環步數可能愈不容易過擬合在訓練語言上，而能夠推廣至未見過的語言上。

- 圖3.4中，一階模型無關元學習-4 步在精細校正前表現大幅落後一階模型無關元學習-4 步，甚至無預訓練的單語言精細校正 80 回合都贏過它，但卻在接觸目標語言 1 回合（6 步）後急起直追追上一階模型無關元學習-2 步，快速適應的能力是所有預訓練方法中最好，顯示其預訓練後（精細校正前）的參數雖然表現差勁，其參數中可能其實蘊含快速適應所需的資訊。雖然所有精細校正階段的數值仍皆略遜基準模型，其快速適應的能力著實值得更多實驗探究背後成因。

### 3.4.3 去詞化依存句法剖析小結

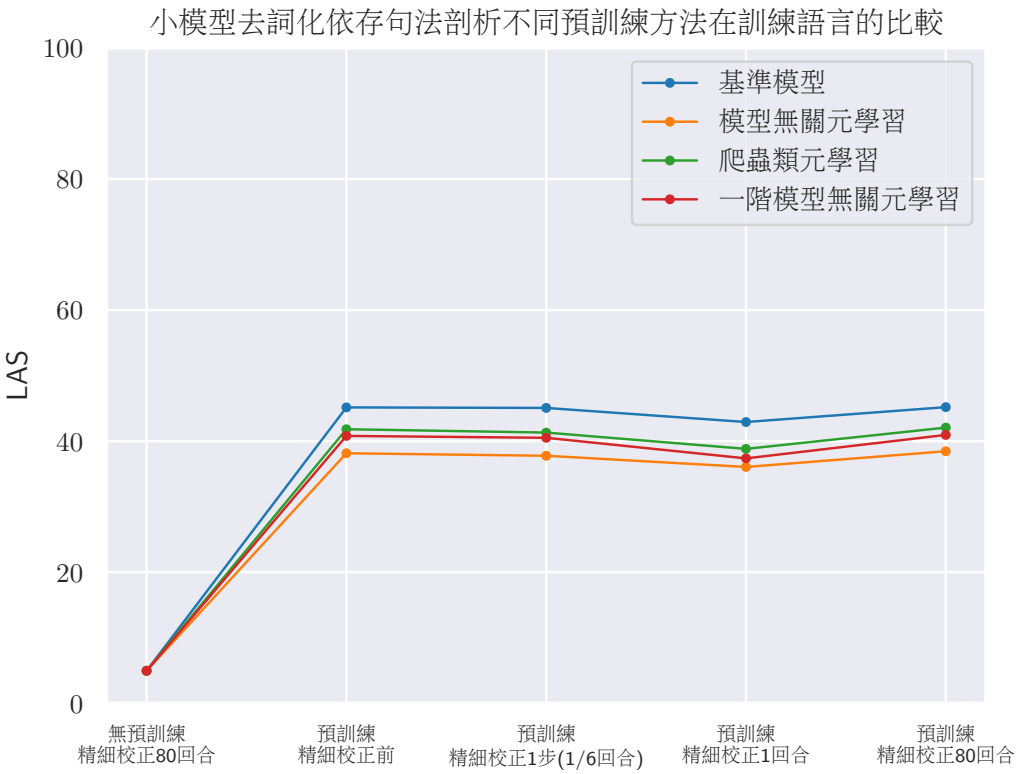
- 我們發現爬蟲類元學習在訓練語言與未見過語言上，無論是精細校正前或經過不同步數的精細校正，其表現均穩定的與基準模型匹敵或稍稍勝過，顯示其無論做為同時剖析所有語言的單一模型，或者作為資料不足語言剖析器的初始參數於其上進行精細校正都相當合適。

- 模型無關元學習在未見過語言上進行小步數的精細校正較其他方法略有優勢，說明其較基準方法更適合做為資料不足語言剖析器的初始參數，在目標語言語料極少或只容許少量梯度更新的情境下使用可以收到不錯的效果。

- 一階模型無關元學習雖然在數值上劣於其他預訓練方法，但其精細校正後與未接觸目標語言前相比的進步量為所有方法之冠，似乎較模型無關元學習更有快速適應的能力。

### 3.4.4 小模型去詞化依存句法剖析不同方法比較

圖3.1為小模型的去詞化依存句法剖析不同預訓練方法產生的模型在目標語言上經過不同步數的精細校正後的測試集 LAS 數值。我們可以發現幾點與普通大小模型



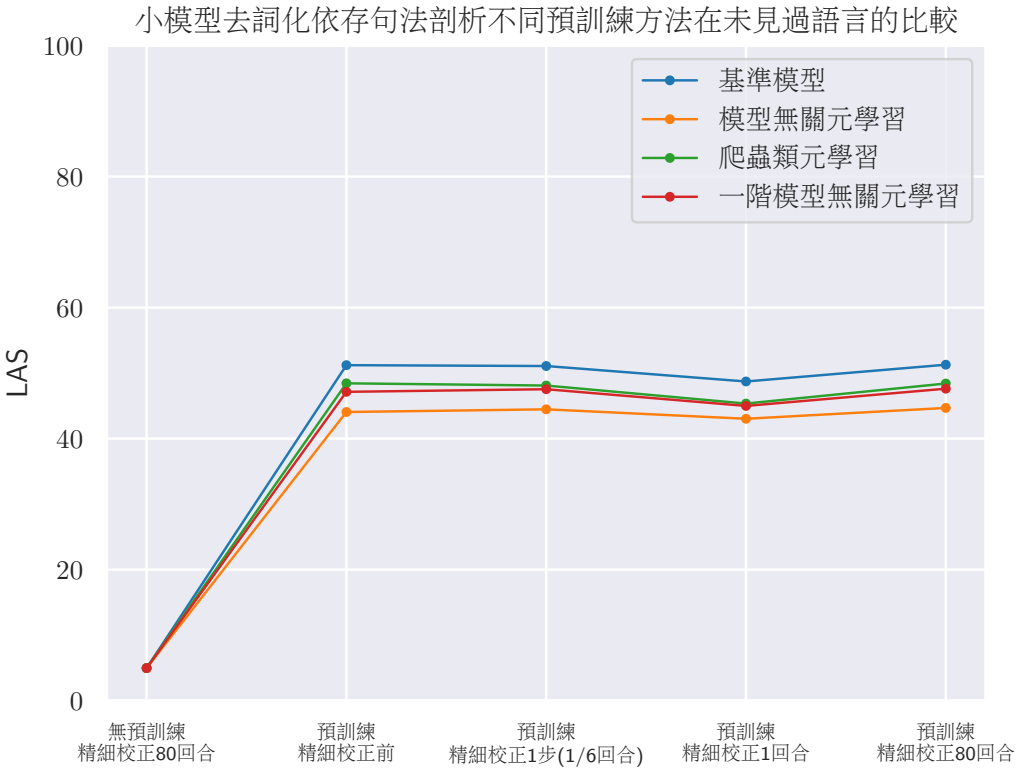




圖 3.5: 小模型去詞化依存句法剖析不同預訓練方法精細校正後的平均 LAS 折線圖。

不同的現象：

- 預訓練過的模型其 LAS 數值從 60 左右降到 35-45 的區間，有約 15-25 的降幅。在同樣梯度步數更新的情況下，模型變小導致表現變差是正常的現象。
- 方法的表現依照與基準模型演算法相似程度排序，愈類似基準模型的方法表現愈好：與基準方法最像的爬蟲類元學習居次；同為一階但只取內循環最後一步梯度的一階模型無關元學習在後；而使用二階微分的模型無關元學習則殿底。且方法之間的表現排序在所有語言與各種精細校正步數下均一致，不似普通大小模型各有千秋。

| 精細校正回合 | 基準模型 | 模型無關元學習 | 爬蟲類元學習 | 一階模型無關元學習 | 無 |
|---|---|---|---|---|---|
| 精細校正前 | **12** | 0 | 0 | 0 | 1 |
| 精細校正 1 步 | **12** | 0 | 0 | 0 | 1 |
| 精細校正 1 回合 | **11** | 0 | 1 | 0 | 1 |
| 精細校正 80 回合 | **11** | 0 | 0 | 0 | 2 |

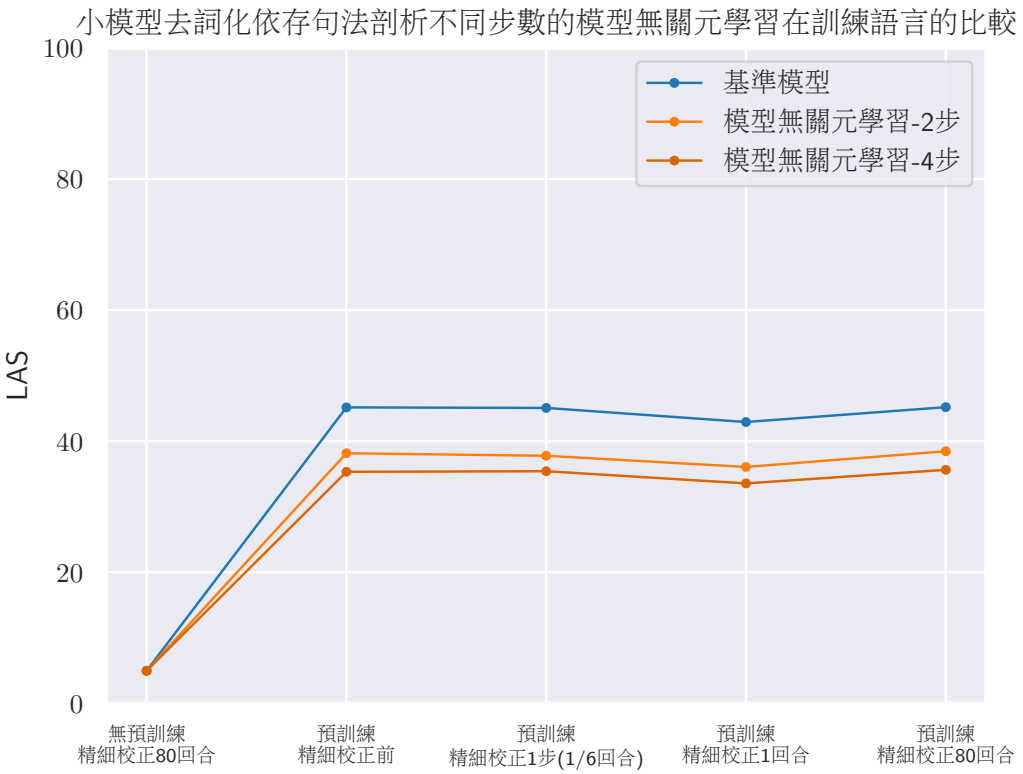
(a) 訓練語言部份。

| 精細校正回合 | 基準模型 | 模型無關元學習 | 爬蟲類元學習 | 一階模型無關元學習 | 無 |
|---|---|---|---|---|---|
| 精細校正前 | **5** | 0 | 0 | 0 | 3 |
| 精細校正 1 步 | **5** | 0 | 0 | 1 | 2 |
| 精細校正 1 回合 | **5** | 0 | 0 | 1 | 2 |
| 精細校正 80 回合 | **5** | 0 | 0 | 1 | 2 |

(b) 未見過語言部份。

表 3.5: 小模型去詞化依存句法分析各預訓練方法在各精細校正階段 LAS 統計顯著勝過所有其他方法次數。

每個語言各自詳細的 UAS/LAS 數值可見圖5.5、5.6、5.7與5.8。各方法以語言為單位統計顯著勝過其他方法的次數可見表3.5。



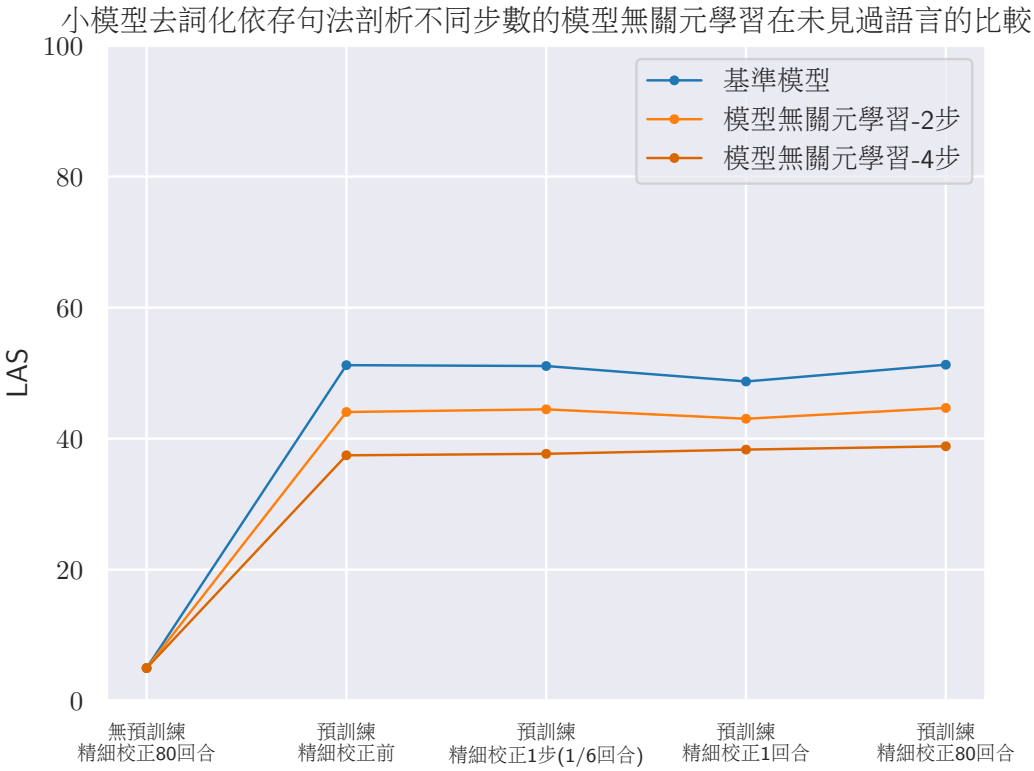
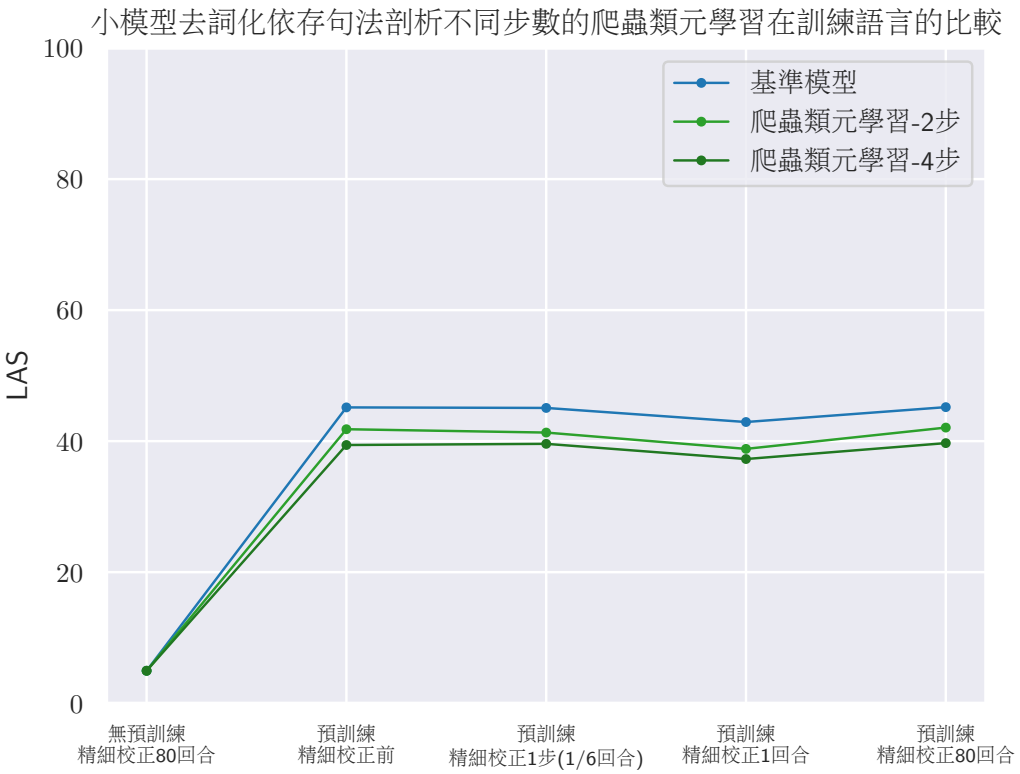




圖 3.6: 小模型去詞化依存句法剖析不同步數的模型無關元學習精細校正後的平均 LAS 折線圖。



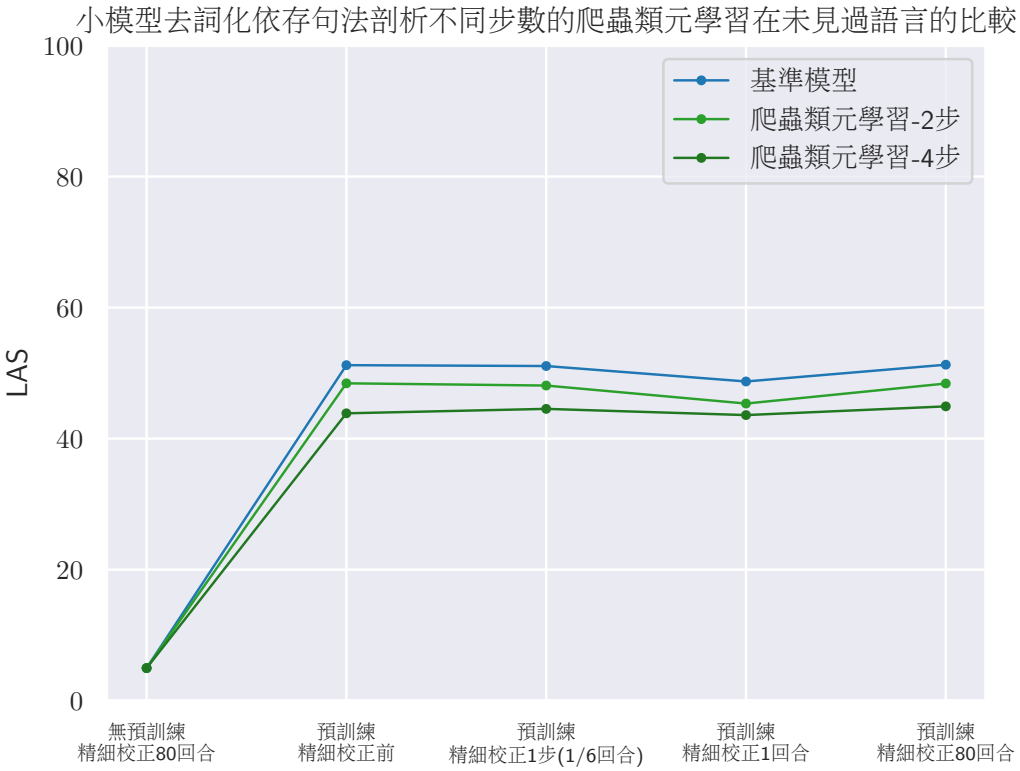
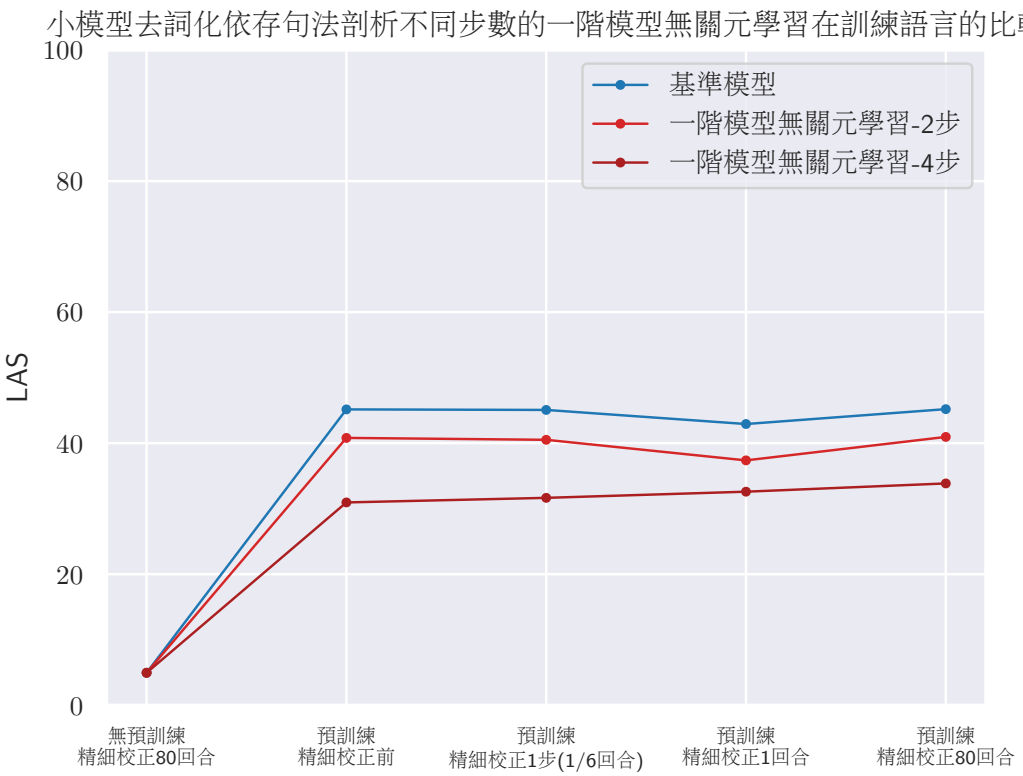




圖 3.7: 小模型去詞化依存句法剖析不同步數的爬蟲類元學習精細校正後的平均 LAS 折線圖。



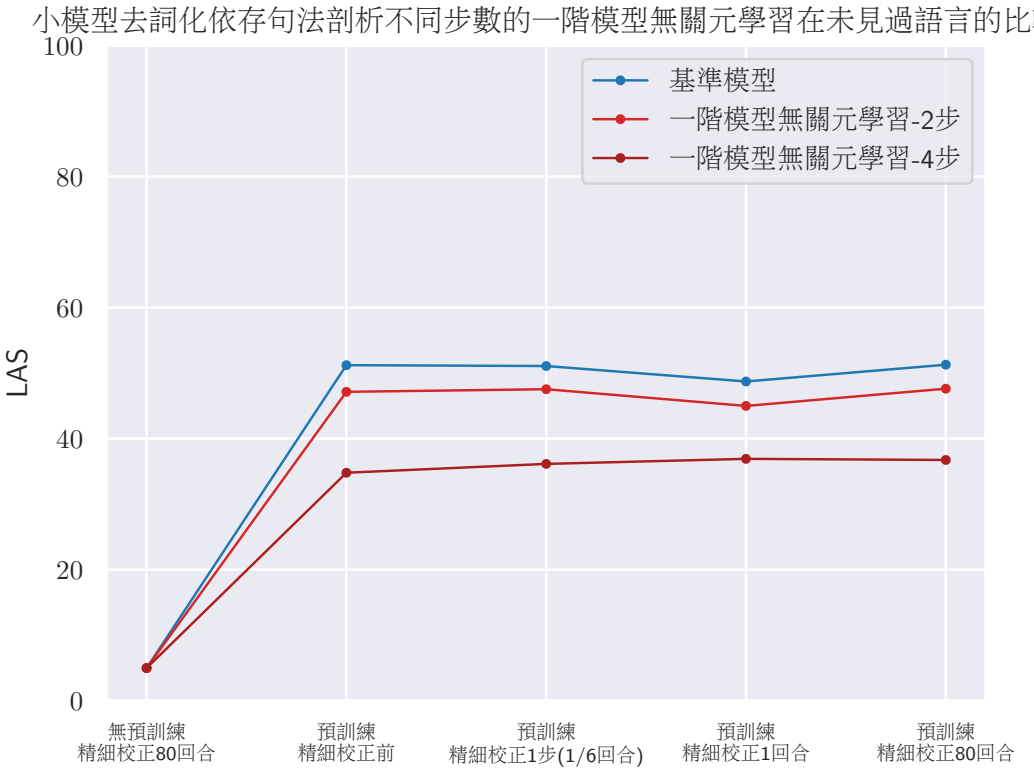




圖 3.8: 小模型去詞化依存句法剖析不同步數的一階模型無關元學習精細校正後在測試集上的平均表現。

### 3.4.5 小模型去詞化依存句法剖析各方法不同內循環步數比較

圖3.6、3.7與3.8 為去詞化依存句法剖析各個預訓練方法不同內循環步數在目標語言上經過不同步數的精細校正後的測試集 LAS 數值。未註明內循環步數之方法其步數皆為 2 步。

從這些圖中，我們發現愈多內循環步數的模型其表現愈差。若將愈多內循環步數詮釋為與基準方法愈不像，那麼我們其實得到與第3.4.4節類似的觀察：在小模型的去詞化依存句法剖析中，與基準方法愈類似的方法，其表現愈好。

### 3.4.6 小模型去詞化依存句法剖析小結

與普通模型不同，在小模型去詞化依存句法剖析中我們發現簡單的基準模型的表現勝過其他所有模型無關元學習的系列方法；且與基準模型演算法愈為相似的方法表現愈好。我們需要更多的實驗與分析來幫助解釋這樣的現象。

這裡我們提出一個可能的原因：或許大模型的參數夠多，使得為所有語言共同建模的參數比例變少，為各別語言建模的參數比例變多，造成其預訓練出來的模型不易推廣到未見過的語言。此時模型無關元學習可以促使大模型提高為所有語言共同建模的參數比例，使其更偏好能夠解釋所有語言的假說，而非只是個別語言句法剖析模型的聯集而已。然而在小模型的情境下，基準模型沒有足夠的參數為個別語言建模，又需要利用有限的參數在訓練語言上取得好表現，因此偏好選擇能夠解釋更多語言的假說，以提高參數利用的效率。因此小模型的基準模型比較沒有死記硬背各語言的問題，在這種情況下模型無關元學習相較於基準模型就沒有特別的優勢了。

## 3.5 分析與討論

在此章節中，我們想要探討各種預訓練方法在精細校正的過程對目標語言特性的適應方式有何不同：是從接觸目標語言前就充分掌握目標語言特性，抑或是在接觸的過程中快速適應？

我們挑選語言形態學中常被提到的中心語方向性參數（head-directionality parameter）作為觀察的對象，在第2.4.4節時介紹過，該特性為各語言句法主要的差異之一，因此各方法產生的句法樹之中心語方向性分佈可作為其對該語言語法的掌握程度重要的指標。

### 3.5.1 計數模型

有些中心詞方向性特別強烈的語言，我們並不清楚其詞性標記序列分佈是否有可能使模型容易產生中心詞方向性強烈的句法樹。如日語是非常遵守中心詞後置原則的語言，經常作為中心詞（如動詞）的詞性可能本來就會置於句子後半，我們並不清楚這樣的現象是否會導致中心詞後置的句子出現的比例特別高。為了排除語言（及其句法樹庫）的詞性標記序列本身可能造成之對方向性的偏置，我們需要製作一個不仰賴上下文、只與中心詞及依附詞有關，且只含有適用於所有語言的中心詞-依附詞機率，而不偏重任一單一語言的剖析器模型來剖析各語言句法樹庫的詞性標記序列。這樣的模型可以做為判斷預訓練方法有沒有習得目標語言中心語方向性的虛無假設（null hypothesis）。倘若預訓練方法產生的句法樹其中心語方向性與正確句法樹中心語方向性的差異沒有低於上述模型與正確句法樹的差異，則我們不能宣稱該預訓練方法學到了目標語言的中心語方向性。

因此，我們訓練了一個使用所有訓練語言語料統計出的計數模型，簡介如下：給定中心詞詞性 $p_{head}$、依附詞詞性 $p_{dep}$，統計所有訓練語言句法樹庫的中心詞選

擇條件機率 $p(\mathrm{p}_{\mathrm{head}}|\mathrm{p}_{\mathrm{dep}})$，其中條件為依附詞的詞性。為了平衡各語言句法樹庫大小不一的情形，上述機率會先在各自語言內部進行正規化，之後再以語言為單位進行正規化，以達到類似第3.3節中平衡各語言句法樹庫取樣頻率的效果。

### 3.5.2 去詞化依存句法剖析各方法產生句法樹之方向性分析

圖3.9、3.10、3.11與3.12 呈現了去詞化依存句法剖析各預訓練方法經過不同步數的精細校正後產生的句法樹其方向性的變化及與正確句法樹方向性的差異。

由圖3.9可以看出在精細校正前，爬蟲類元學習與基準模型產生的句法樹其中心詞方向性就與正確答案相差不遠；之後不管精細校正 1 步、1 回合、80 回合（圖3.10、3.11、3.12），其分佈與正確答案的相近程度依然保持精細校正前的水準，並有些許進步，顯示這兩種方法無論有沒有接觸過目標語言，知不知道目標語言的任何資訊，透過詞性標記序列就可以充分掌握目標語言之中心詞方向性。而模型無關元學習與一階模型無關元學習產生的句法樹其中心詞方向性與正確答案則不若前述兩者相近，與圖3.1中的 LAS 得到的發現類似，顯示這兩個方法可能並未對目標語言的中心詞方向性有太多的假設；但接觸目標語言並進行梯度更新以後（圖3.10），它們快速掌握了目標語言的中心詞方向性，顯示這兩個方法的確有快速適應的能力。計數模型產生之句法樹與正確答案中心詞方向性雖有相關性，但之間的差異較預訓練方法大，顯示不同中心詞方向性的語言其詞性標記序列分佈對任意方法產生的句法樹中心詞方向性影響有限。

### 3.5.3 小模型去詞化依存句法剖析各方法產生句法樹之方向性分析

圖3.13、3.14、3.15與3.16 呈現了小模型去詞化依存句法剖析各預訓練方法經過不同步數的精細校正後產生的句法樹其方向性的變化及與正確句法樹方向性的差

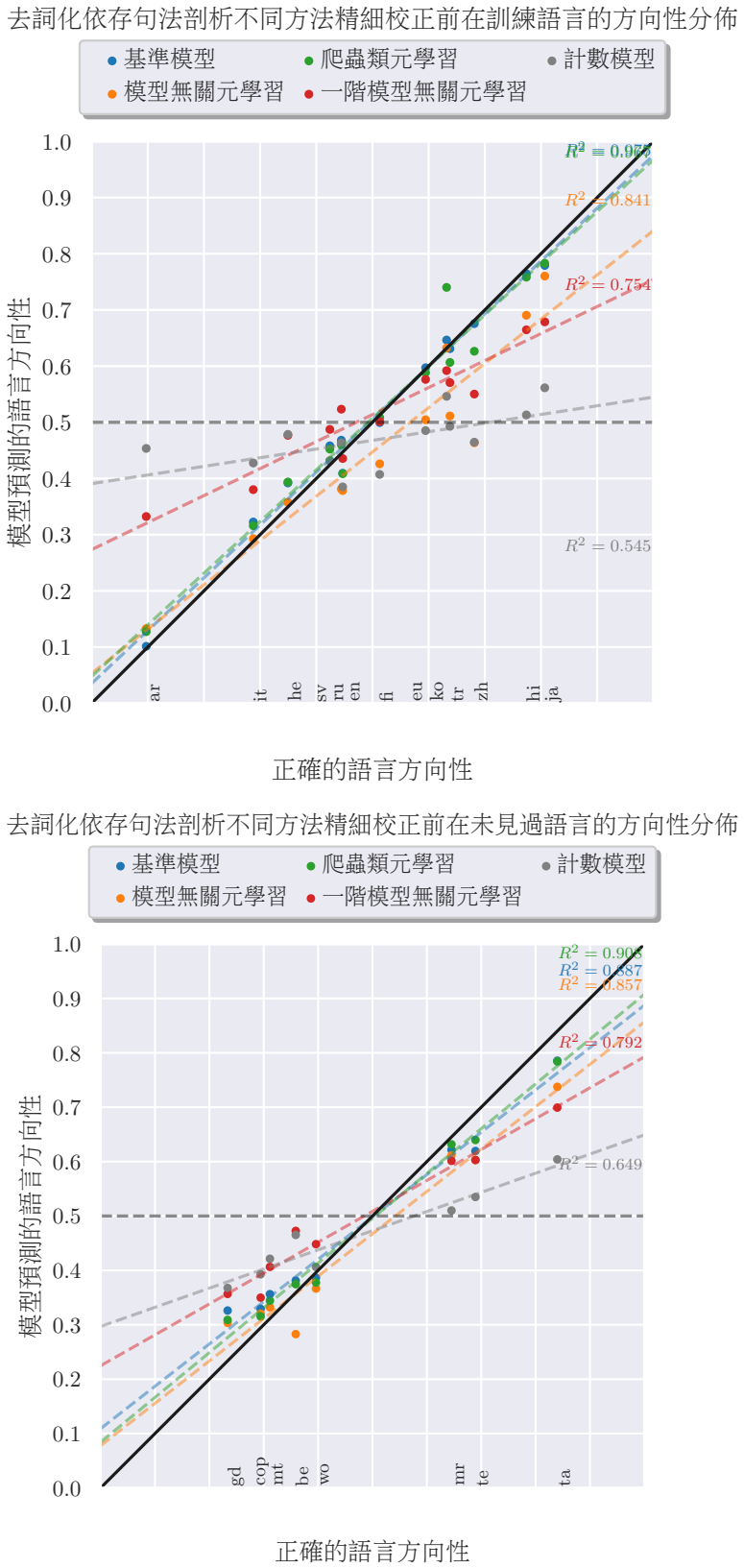
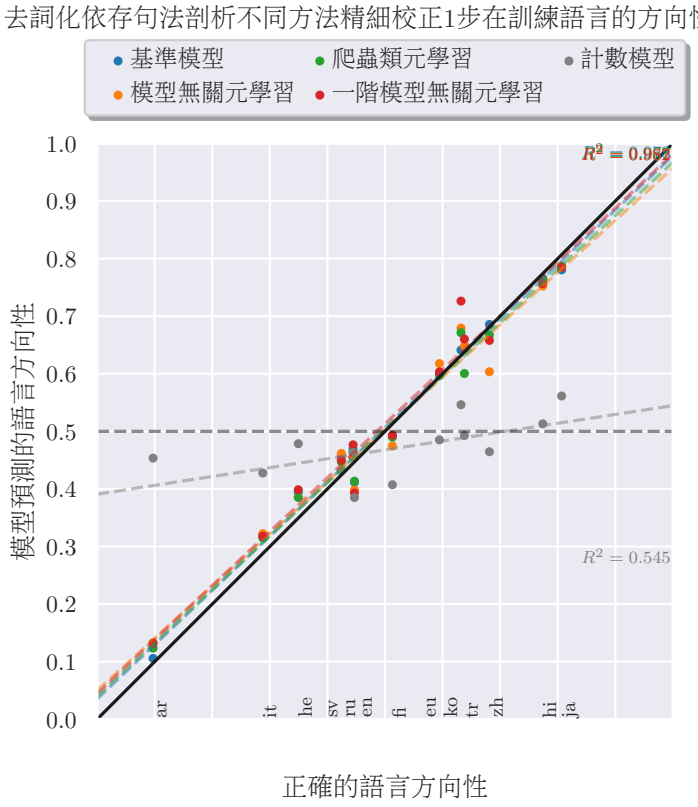



圖 3.9: 去詞化依存句法剖析不同方法在各語言精細校正前的方向性分佈。



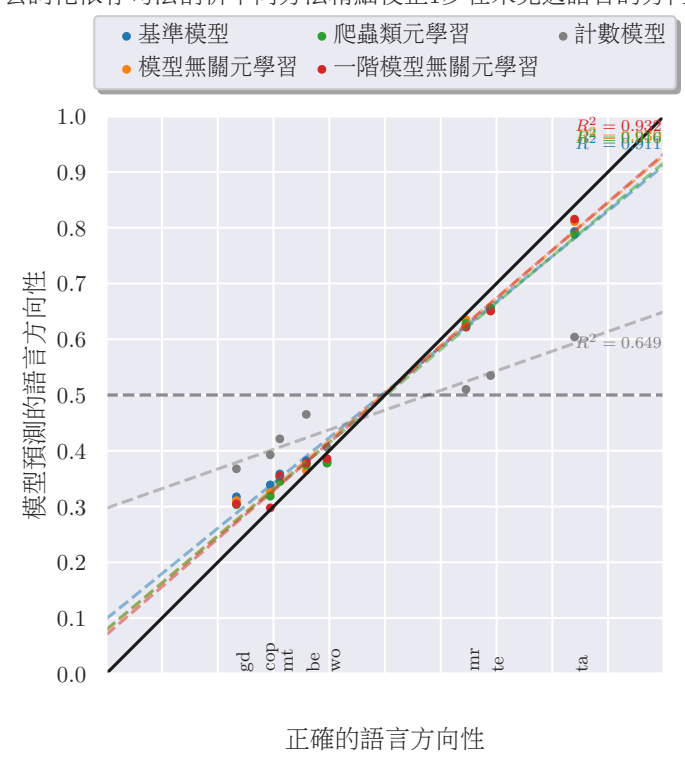




圖 3.10: 去詞化依存句法剖析不同方法在各語言精細校正 1 步（$\frac{1}{6}$ 回合）後的方向性分佈。

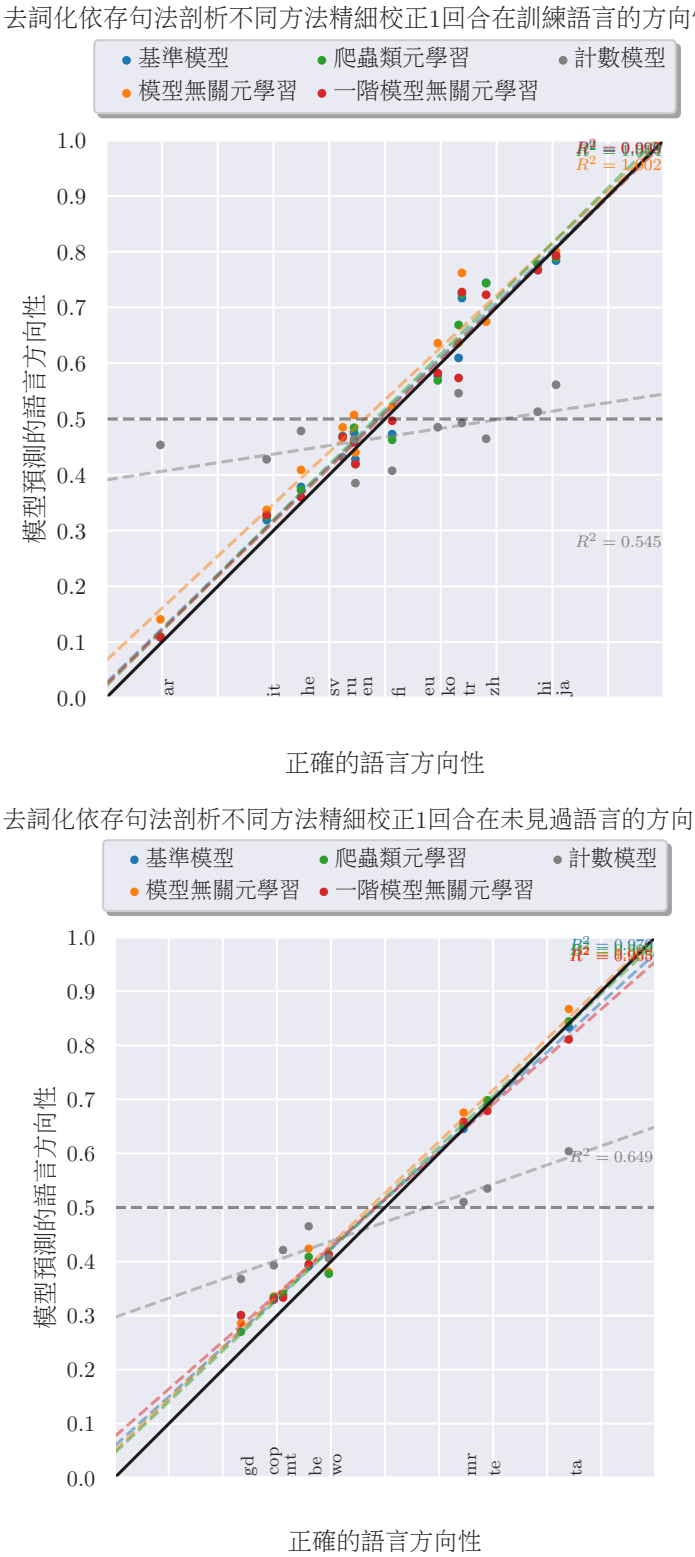
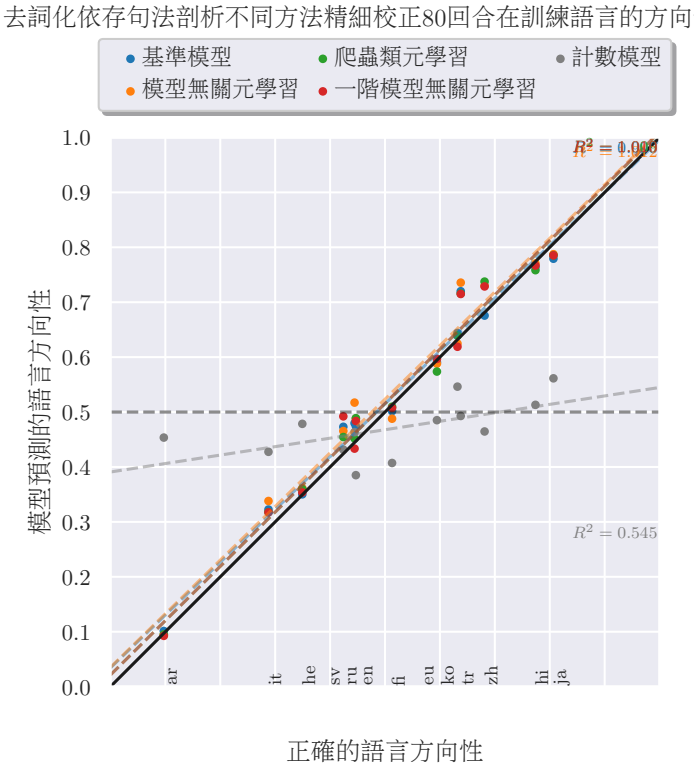



圖 3.11: 去詞化依存句法剖析不同方法在各語言精細校正 1 回合後的方向性分佈。



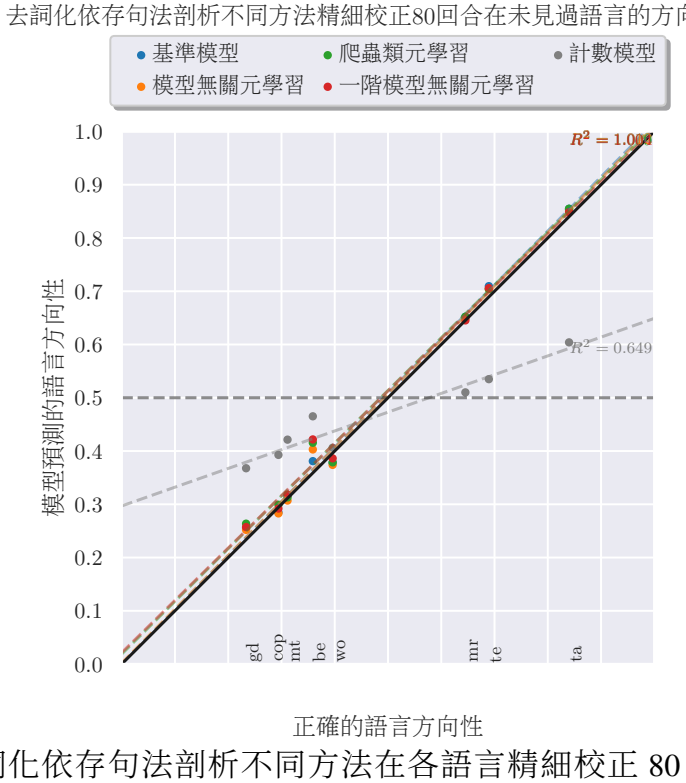




圖 3.12: 去詞化依存句法剖析不同方法在各語言精細校正 80 回合後的方向性分佈。

異。從圖中可以發現使用小模型的預訓練方法其中心詞方向性與正確答案之間的差異都較普通模型要來的更大，與圖3.5中的 LAS 得到的發現類似。值得注意的是，精細校正少量步數時（圖3.14中的 1 步與圖3.15中的 1 回合）所有方法中心詞方向性的與正確答案的差異反而稍較精細校正前微微上升，即使是專門為精細校正的過程優化的模型無關元學習方法也有類似的現象；這部份需要更多的分析實驗來釐清其中原因。

## 3.6 小結

此章將模型無關元學習系列方法應用在特徵較為單純的去詞化依存句法剖析任務上，比較了基準模型、模型無關元學習、一階模型無關元學習與爬蟲類元學習四種方法及其不同內循環步數的變形在訓練語言與未見過語言上精細校正 1 步、1 回合與 80 回合的表現，發現爬蟲類元學習在未見過語言精細校正少量步數的情況下有略勝基準模型的表現，而模型無關元學習與其一階變形則透過其精細校正 1 步前後的大幅進步展示了其快速適應的能力。在不同內循環步數的實驗方面，也發現愈多內循環步數的爬蟲類元學習與模型無關元學習在更長的精細校正步數有較好的準確率，與其演算法設計的初衷高度符合。另外，也試圖改變模型參數多寡，發現在小模型的情境下簡單的基準模型就可以勝過模型無關元學習系列方法，說明模型無關元學習做為一正則化的方法，可能更適合在有較大模型表現力的情境下使用。最後，藉由中心詞方向性這個不同語言間句法的一大差異，觀察不同預訓練方法在精細校正的過程中對語言句法特性掌握程度的變化軌跡，發現基準模型與爬蟲類元學習在未接觸目標語言之前就已經大致掌握該語言的句法特性，而模型無關元學習與其一階變形則在精細校正前後快速習得目標語言的句法特性，提供了除了 UAS/LAS 準確率以外另一方面的觀察。

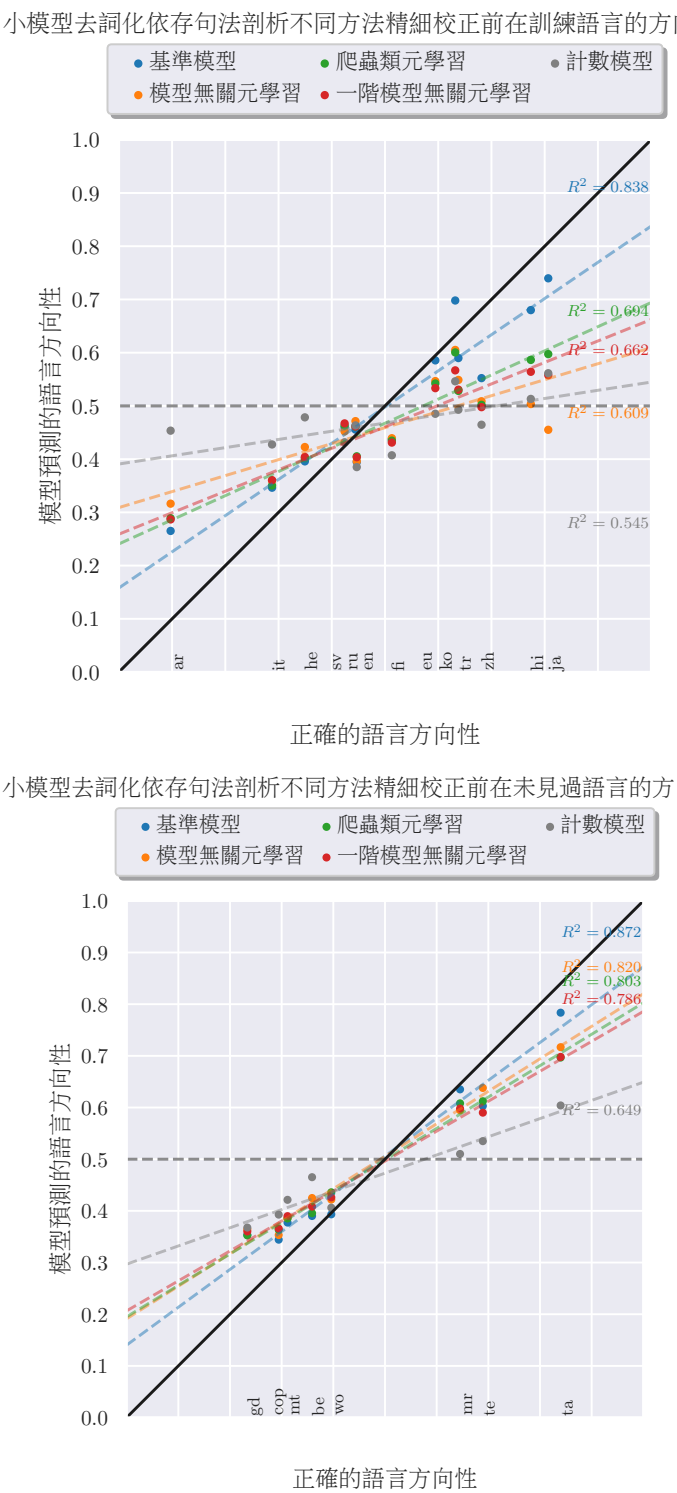
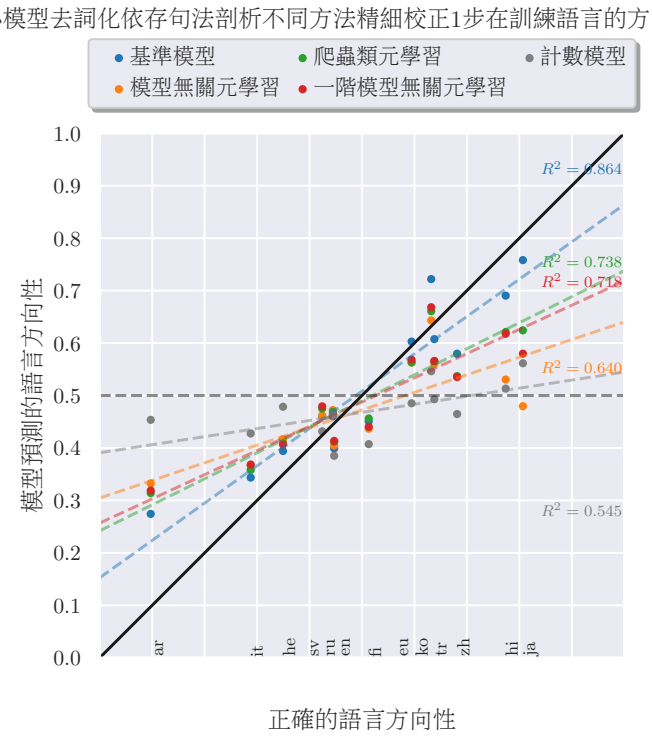



圖 3.13: 小模型去詞化依存句法剖析不同方法在各語言精細校正前的方向性分佈。



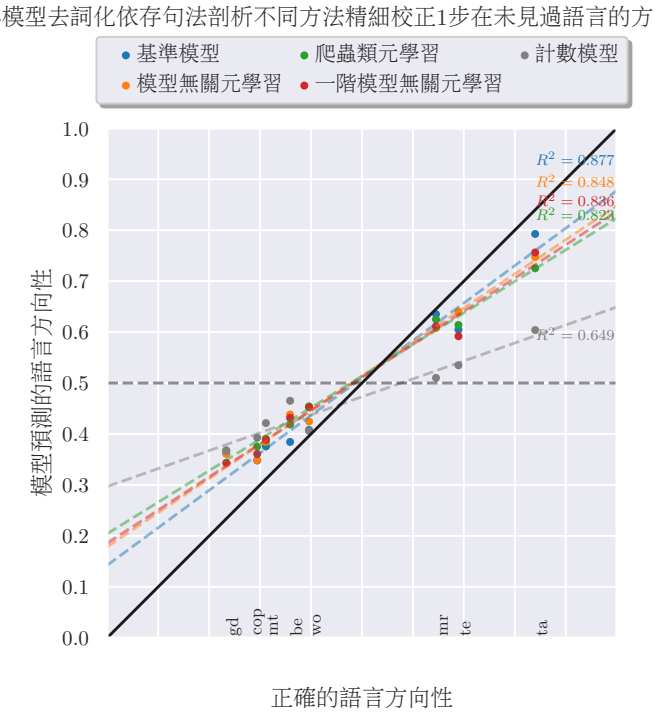
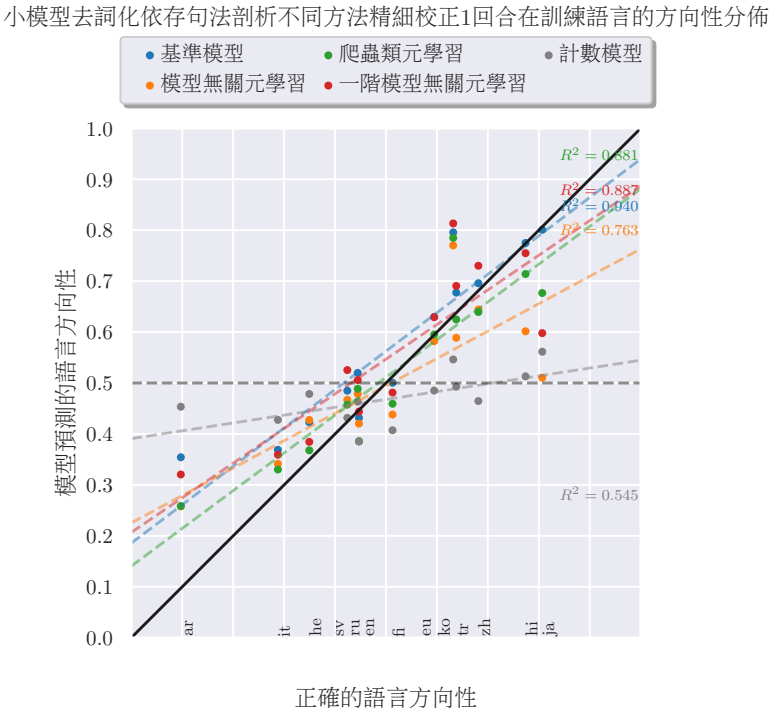




圖 3.14: 小模型去詞化依存句法剖析不同方法在各語言精細校正 1 步（$\frac{1}{6}$ 回合）後的方向性分佈。



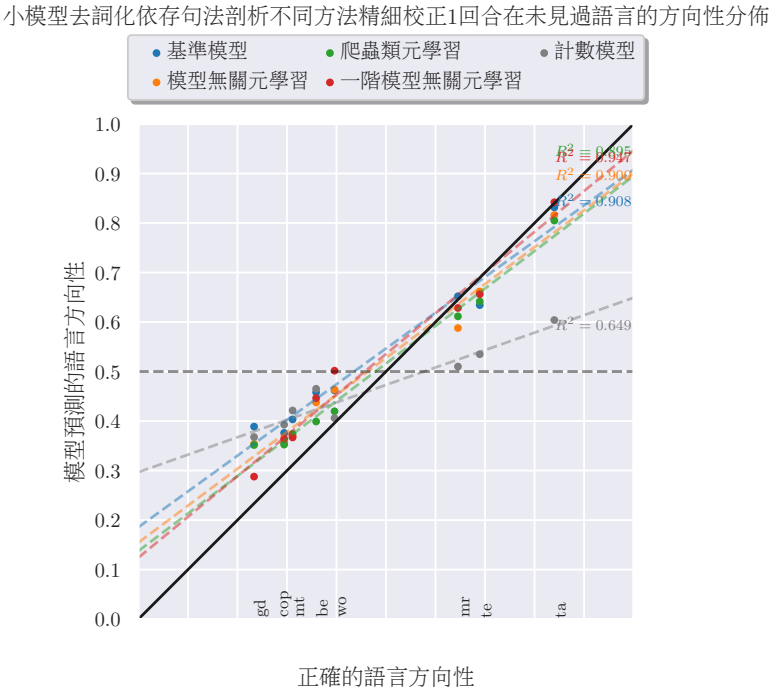




圖 3.15: 小模型去詞化依存句法剖析不同方法在各語言精細校正 1 回合後的方向性分佈。

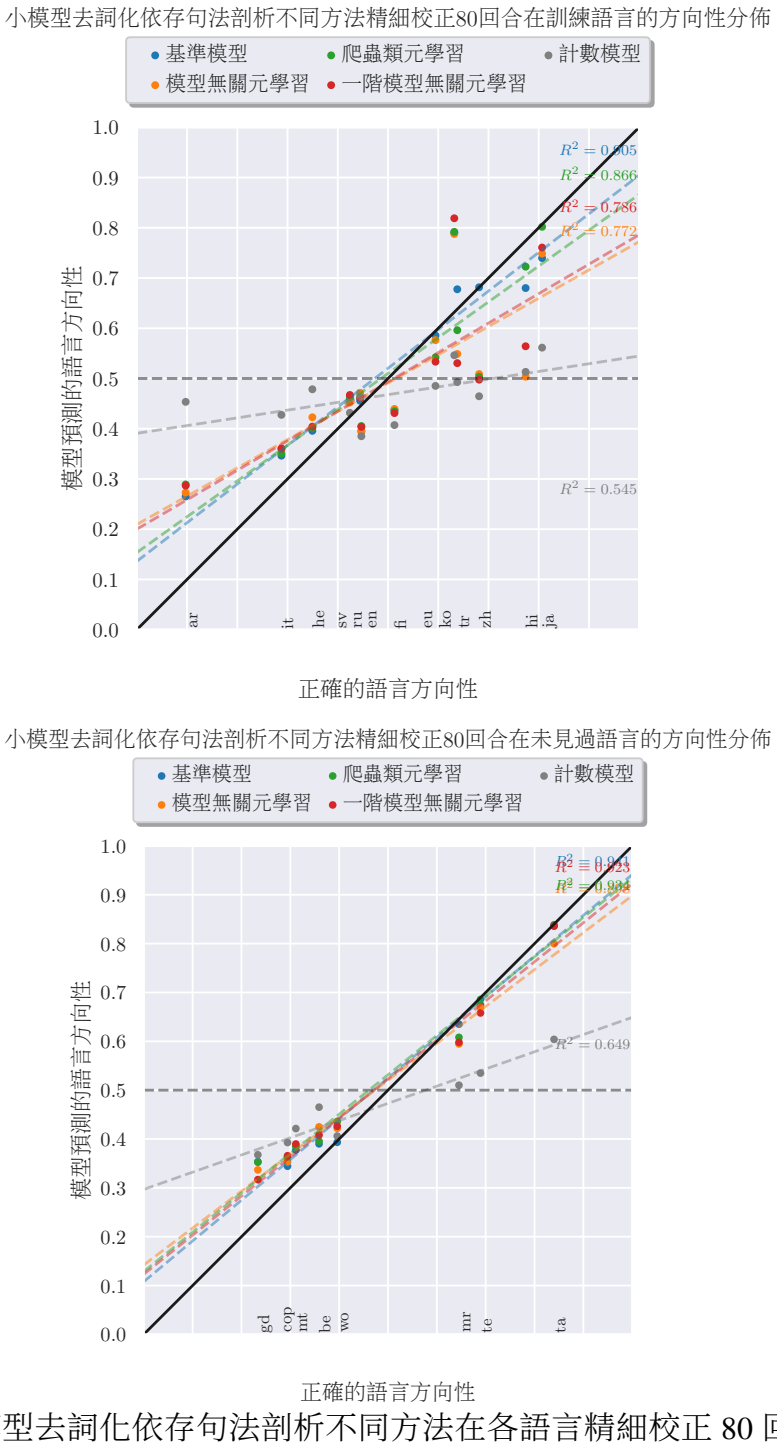



圖 3.16: 小模型去詞化依存句法剖析不同方法在各語言精細校正 80 回合後的方向性分佈。

# 第 四 章　使用元學習在資料不足的詞化依存句法剖析

## 4.1　簡介

接續上一章的去詞化依存句法分析，此章我們將模型無關元學習系列方法推展到真實的應用場景。在真實情境下我們沒有語言的詞性標記，只有其純文字，在每種語言書寫系統均大不相同的情況下，用單一編碼器將他們各自編碼成向量表示，並輸入分類器中，輸出依存句法樹，是為多語言詞化依存句法分析（multilingual lexicalized dependency parsing）。

此章架構簡介如下：第4.2節首先介紹多語言詞化依存句法剖析模型的架構，包括使用的多語言編碼器、預訓練與精細校正時分別採用的模型組態等；第4.3節介紹詞化依存句法剖析的實驗設置，說明為了將依存句法剖析應用到實際場景，將第3.3節所介紹的去詞化依存句法剖析其實驗設置做了哪些微調改變；第4.4節呈現實驗結果，觀察模型無關元學習系列方法應用在詞化依存句法剖析的表現與非詞化版本有何不同；第4.5節的分析與討論與第3.5節相同，觀察詞化句法剖析不同預訓練方法經過精細校正後目標語言方向性分佈的改變軌跡，並與非詞化版本比較；最後在4.6節簡單總結此章的發現。

## 4.2 多語言詞化依存句法剖析模型架構

### 4.2.1 多語言基於轉換器模型的雙向編碼器表示（multilingual BERT）

句法剖析的架構中，在大型預訓練語言模型出現以前，編碼器函數 $\mathbf{r}(w)$ 常見的選擇為數層隨機初始化的 LSTM 或轉換器；大型預訓練語言模型出現後，以其為編碼器函數的初始訓練參數進行精細校正（fine-tuning）[1]所訓練出的剖析器紛紛取得更好的成績。多語言的句法剖析在大型預訓練語言模型出現之前較少文獻直接讓各語言共同分享編碼器函數，真正共享參數的也多為接受詞性標記而非文字的去詞化依存句法剖析（delexicalized dependency parsing），其原因主要可歸結為多語言模型需要設計統一的記符集（token set）來表示每個語言各異的書寫系統（writing system）產生的文字。在單語言時可直接用該語言經斷詞後所統計出的常見詞作為記符（token）；單語言的常見詞數目通常設定在 10000-40000 詞即非常堪用，剩下的低頻詞並不影響模型表現太多；但多語言模型若不減少任一語言之記符數而直接結合各語言的詞彙做為記符集，此記符集將變得太大，且相近語言無法透過相似的構詞共享參數，如西班牙文的「學生」一詞 “estudiante” 與其英文的對應 “student” 有共同的詞子字串 “stud”，上述直接結合的方法便無法讓模型學習到這些共通性。文獻上已經提出許多解決辦法，以下列舉三項：

- 使用語素分割器（morpheme segmenter）：利用專家知識構造出基於規則或統計的語素分割器分割語素，並以語素為記符，分割結果符合人類知識，使相似詞可以正確的共用語素記符向量的參數爲其優點，惟某些語言可能不存在

[1]注意到此處的「精細校正」是指對大型預訓練語言模型的參數進行微調，而貫串全論文的模型無關元學習的精細校正是指對模型無關元學習預訓練完成的模型參數進行微調，兩者微調的對象有些微差別。本章內除了本節的「精細校正」是指前者之外，他處均指後者。而此章在大型預訓練語言模型上所做的「多語言依存句法剖析預訓練」，是對模型無關元學習而言的術語，若照大型預訓練語言模型相關文獻的說法，應稱為「多語言依存句法剖析精細校正」，在此特別釐清。

準確率高的語素分割器，通用性不足。

- 使用字符（character）做為記符：優點為不需要某些語言可能沒有的語素分割器，但字符顆粒度過小，單句話的記符數變多，會增加模型處理時間。
- 使用次級詞分割（subword tokenization）演算法：次級詞是比詞小但比字符大的記符，由演算法統計出語言中的詞彙較常獨立出現的子字串（substring）做為新的記符，將詞取代為多個子字串的結合，也可看做是非監督式演算法計算出的語素。常見的演算法包括字節對編碼（Byte-pair encoding） [42]、WordPiece [43] 等。

其中次級詞雖然分割品質受演算法及訓練語料大小影響良窳不一，但其兼備語素分割器共用子字串與字符不需要人類知識的優點，因此現行單語言與多語言的大型預訓練語言模型均採用次級詞做為記符來取代原本以詞或字符為單位的表示法。

本研究與目前孔氏提出的多語言句法剖析的最佳單一模型 Udify[40] 一樣採用**多語言基於轉換器模型的雙向編碼器表示**（下稱 mBERT）作為編碼器函數來編碼語料中的衆多語言。

令 $\mathbf{r}(w)$ 為編碼器函數，$\text{mBERT}(w)_i$ 為記符 $w$ 通過 mBERT 第 $i$ 層的輸出，由於許多文獻 [26, 29] 均指出與其讓下游任務只接受最後一層的輸出，讓模型在精細校正（fine-tuning）時自由混合幫助較大的輸出層更有助於模型表現，而特氏也發現 [44] 若交由每個任務自由混合預訓練模型不同層數的輸出，不同任務所給予的層權重分佈大不相同，其中與句法相關的任務（如詞性標註、句法標註）傾向給予接近輸入的層較大的權重，而與語意相關的任務則給予接近輸出的層較大的權重，顯示原本只取最後一層的方法恐非最佳策略；因此這裏採用彼氏（Matthew Peters）提出的層專注機制（layer attention），給予每一層輸出專注權重，

讓模型決定哪一層的輸出對句法剖析較有幫助：

$$\mathbf{r}\left(w\right) = \alpha \sum_{i=1}^{L} \text{mBERT}\left(w\right)_i \cdot \text{softmax}(\mathbf{c})_i \tag{4.1}$$

其中 $L$ 為 mBERT 的層數（本研究使用 `bert-base-multilingual-cased` 版本，$L = 12$），$\alpha$ 為可調整的純量，$\mathbf{c} \in \mathbb{R}^L$ 為層專注權重。

為了防止模型過於仰賴特定層的資訊而造成過擬合，這裏採用孔氏提出的 [40] 的層丟棄（layer dropout），在訓練時每個層專注權重 $c_i$ 有 $p = 0.1$ 的機率被設為 $-\infty$，使權重重新分配到其他的層上，迫使模型整合 mBERT 全部層輸出的資訊，而非偏重特定某幾層。

### 4.2.2 適應器 (adapter)

適應器為雷氏（Sylvestre-Alvise Rebuffi）[1] 所提出在影像領域的轉移學習方法，後由何氏（Neil Houlsby）引進自然語言處理常用的轉換器模型 [45]，其指出當時自然語言處理的轉移學習方法多半使用大型預訓練轉換器模型進行全模型精細校正在目標任務上，但何氏認為全模型精細校正需要調整模型中所有的參數，每種任務都會產生一個全新的模型，太耗費儲存空間與計算資源，且大型預訓練轉換器模型已經含有大量句法語意等任務所需資訊，應不需要變動參數過多；因此他提出固定原本的大型預訓練轉換器模型參數，但在被固定的模型層中加入具殘差網路性質的適應器，模型只需為每個任務調整適應器少量的參數，任務間還是共享原本的大型預訓練轉換器模型參數，而實驗數據也顯示，加入適應器的轉換器模型可以在所需調整的參數量遠低於全模型精細校正下，在目標任務中達成與其相似的表現。其架構為一前饋層組成的兩層瓶頸網路（一層投射到較小維度，一層投射回原本維度），再加上殘差連結（residual），置於轉換器中前饋層後、層正

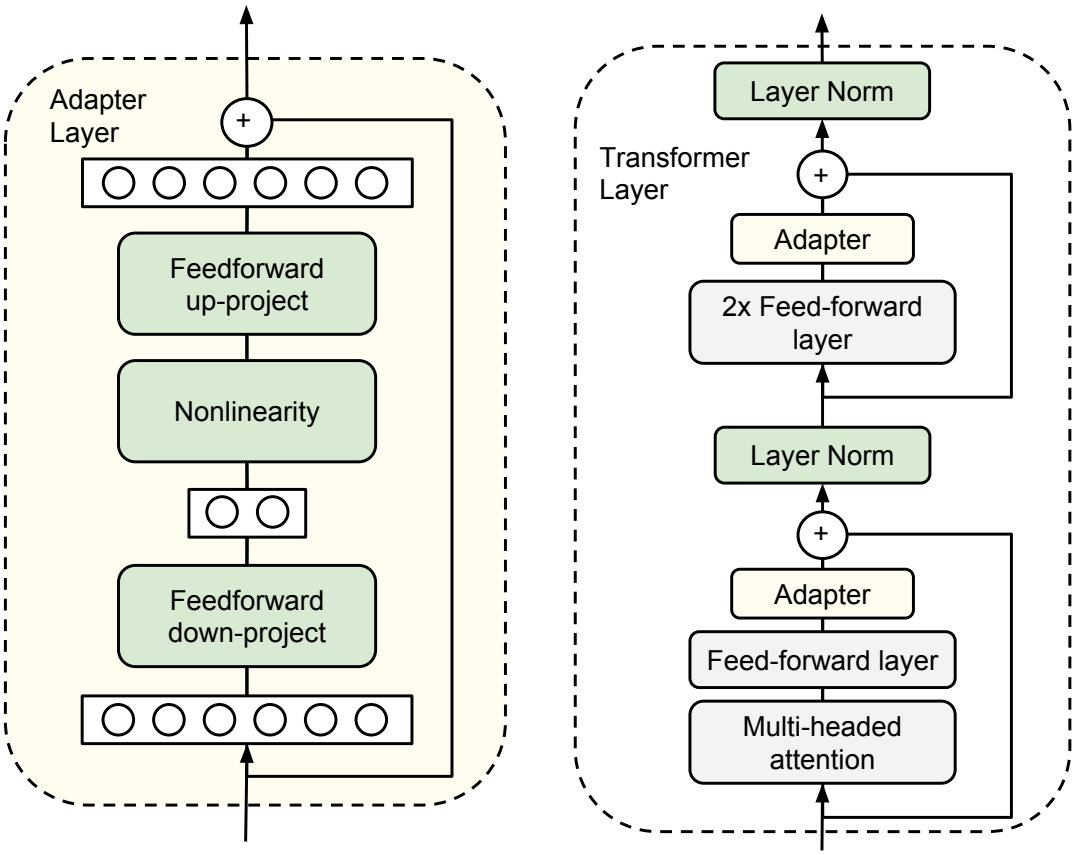



圖 4.1: **左側**：適應器架構；**右側**：加入適應器後的轉換器架構（圖取自 [1]）。

規化之前的位置，細節可見圖4.1。

## 4.3 實驗設置

| 超參數 | 值 |
|---|---|
| 依存標籤維度 | 256 |
| 依存邊維度 | 768 |
| 丟棄機率 | 0.5 |
| BERT 丟棄機率 | 0.2 |
| BERT 遮蔽機率 | 0.2 |
| 層丟棄機率 | 0.1 |
| 批次大小 $b$ | 16 |
| 語言數 $l$ | 10 |
| 訓練樣本數/回合 | 64000 |
| 訓練回合數 | 10 |
| 優化器 | Adam |
| $\beta_1, \beta_2$ | 0.9, 0.9 |
| 權重衰減參數 | 0.01 |
| 基礎學習率 | $3e^{-4}$ |
| 最大梯度 | 5.0 |

(a) 預訓練超參數。

| 超參數 | 值 |
|---|---|
| 丟棄機率 | 0.5 |
| BERT 丟棄機率 | 0.2 |
| BERT 遮蔽機率 | 0.2 |
| 層丟棄機率 | 0.1 |
| 批次大小 | 16 |
| 訓練回合數 | 80 |
| 優化器 | Adam |
| $\beta_1, \beta_2$ | 0.9, 0.99 |
| 權重衰減參數 | 0.01 |
| 基礎學習率 | $3e^{-4}$ |
| 最大梯度範數 | 5.0 |

(b) 精細校正超參數。

表 4.1: 詞化依存句法剖析模型超參數一覽。

本節實驗設置與第3.3節去詞化的依存句法剖析設置大致相同，不再贅述。超參數部份，適應器模型除了優化器的學習率調整方法不同以外，大部分的超參數都與烏氏 [41] 的設置一樣；見表4.1a。由於模型無關元學習需要的計算量龐大，詞化依存句法分析的參數量又比去磁化依存句法分析的參數量多非常多，本研究可取得的運算資源無法在合理的時間將該模型訓練完成，因此詞化依存句法分析的實驗均不使用模型無關元學習；又由於一階模型無關元學習在去詞化依存句法分析表現在四種方法中居末，因而決定不將其應用在詞化的依存句法分析任務上；因此以下詞化依存句法分析的實驗皆進行在下列三種預訓練方法：基準模型、爬蟲類元學習-2 步、爬蟲類元學習-4 步。

精細校正的超參數請見表4.1b。值得注意的是，未見過語言中的 8 種語言有 4 種並未出現在 mBERT 的預訓練語言裡，分別為 Wolof（wo）、Scottish Gaelic（gd）、Coptic（cop）與 Maltese（mt）[2]。沒有經過 mBERT 預訓練過的語言，又未在句法樹預訓練的語言中出現，模型無關元學習系列方法在這些語言上的表現也是值得關注的一點。

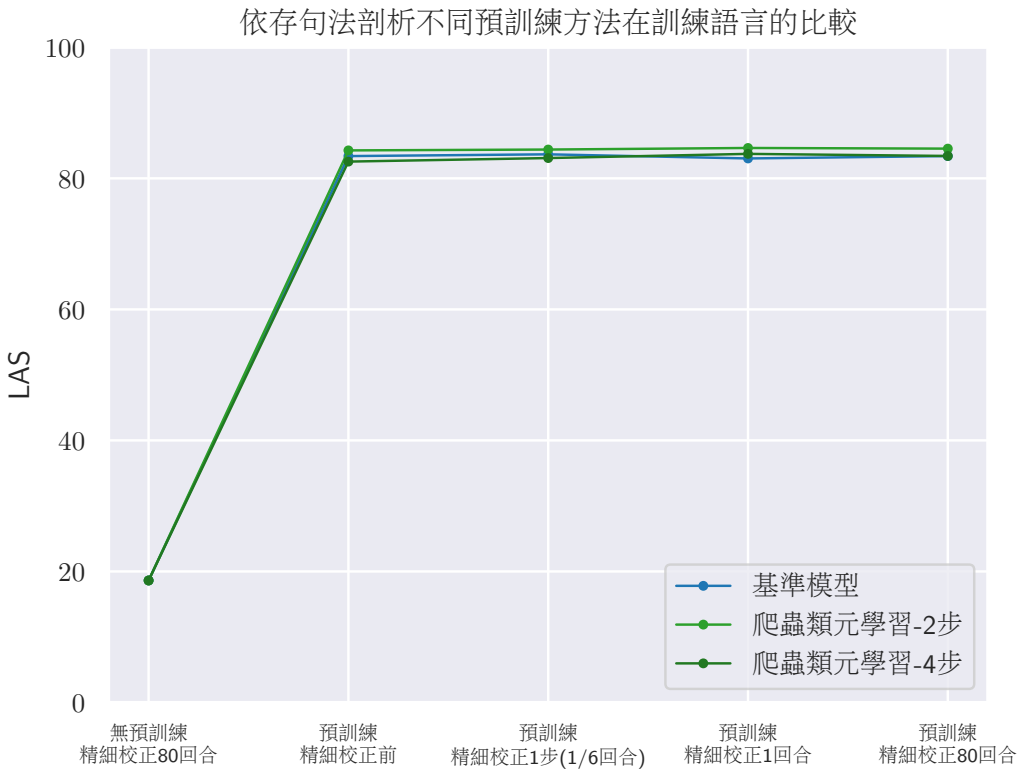
## 4.4 實驗結果

圖4.2為詞化依存句法剖析不同預訓練方法產生的模型在目標語言上經過不同步數的精細校正後的測試集 LAS 數值。每個語言各自詳細的 UAS/LAS 數值可見圖5.9、5.10、5.11與5.12。

我們似乎發現與圖3.1中去詞化依存句法剖析頗有差異的現象：爬蟲類元學習兩種方法（2 步、4 步）在訓練語言與基準模型表現相差不多，甚至稍稍贏過基準模型；然而在未見過的語言上，爬蟲類元學習在各個精細校正階段明顯落後基準模型，且有段不小的差距。關於訓練語言上的表現，我們首先檢視圖5.9中不同預訓練方法在各語言精細校正前的詳細表現，我們會觀察到基準模型雖然在平均表現上以微小差距落後，但它在超過半數的訓練語言上都贏過爬蟲類元學習（在 ar、zh、en、he、ko、sv、tr 等語言上統計顯著），而爬蟲類元學習只有在芬蘭語（Finnish, 語碼 fi）上統計顯著贏過基準模型，但在芬蘭語上基準模型表現的非常差，大幅拉低平均表現，因此產生圖3.1中爬蟲類元學習小勝基準模型的錯覺。

因此我們認為在詞化依存句法剖析上，光看全部語言的平均無法如實反映不同方法的表現，而應該看不同方法在各語言統計顯著贏過其他方法的次數。從表4.2可看出除了在訓練語言上精細校正 1 回合以外，基準模型無論是在訓練語言

[2]mBERT 選取維基百科中文章數前 100 名的語言作為訓練語料。100 種語言的清單可見：https://github.com/google-research/bert/blob/master/multilingual.md



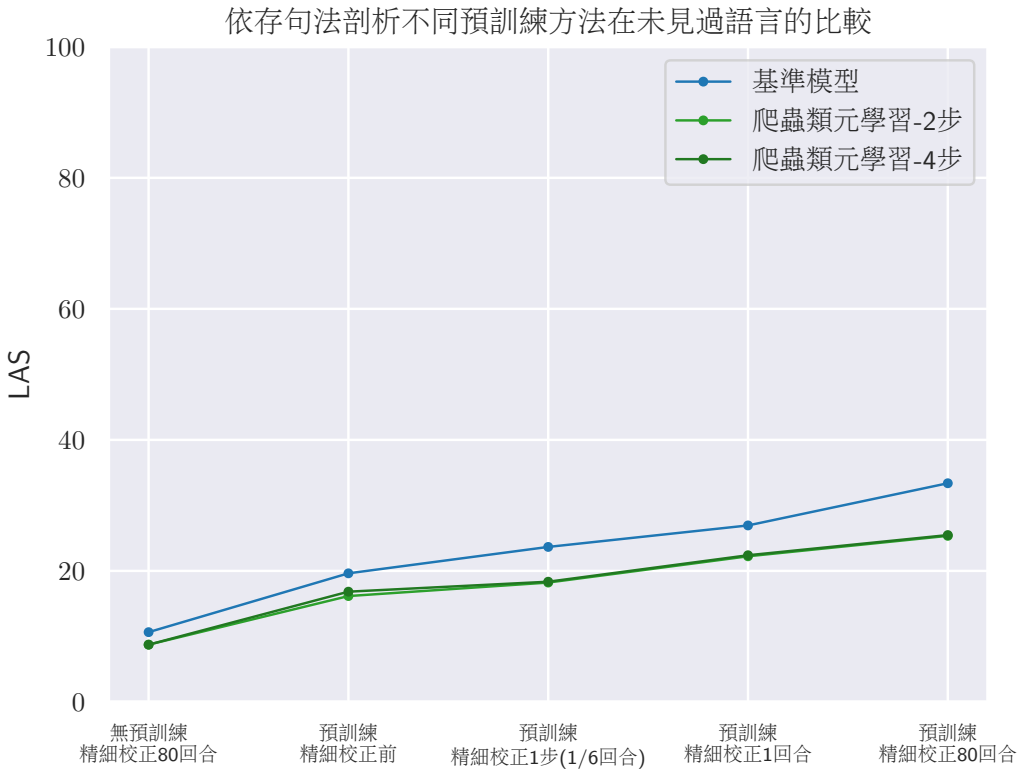




圖 4.2: 依存句法剖析不同預訓練方法精細校正後的平均 LAS 折線圖。

| 精細校正回合 | 基準 | 爬蟲類元學習-2 步 | 爬蟲類元學習-4 步 | 無 |
|---|---|---|---|---|
| 精細校正前 | **6** | 0 | 1 | 6 |
| 精細校正 1 步 | **6** | 3 | 0 | 4 |
| 精細校正 1 回合 | 2 | **5** | 0 | 6 |
| 精細校正 80 回合 | **6** | 3 | 0 | 4 |

(a) 訓練語言部份。

| 精細校正回合 | 基準 | 爬蟲類元學習-2 步 | 爬蟲類元學習-4 步 | 無 |
|---|---|---|---|---|
| 精細校正前 | **7** | 1 | 0 | 0 |
| 精細校正 1 步 | **7** | 1 | 0 | 0 |
| 精細校正 1 回合 | **7** | 1 | 0 | 0 |
| 精細校正 80 回合 | **7** | 1 | 0 | 0 |

(b) 未見過語言部份。

表 4.2: 詞化依存句法分析各預訓練方法在各精細校正階段 LAS 統計顯著勝過所有其他方法次數。

抑或是未見過語言都大幅勝過爬蟲類元學習-2 步及爬蟲類元學習-4 步，顯示簡單的多語言協同訓練對使用 mBERT 做為模型初始參數的多語言剖析器來說已經十分有效，爬蟲類元學習在詞化依存句法剖析的情境下並沒有太多優勢。

而爬蟲類元學習唯一穩定勝過基準模型的語言 Coptic（cop），檢查 mBERT 的詞彙後發現 Coptic 所使用的字母為該語言專屬，並沒有在 mBERT 的詞彙中，所以 Coptic 語言對 mBERT 來說全部都是詞彙以外的詞（OOV word, out-of-vocabulary word）；因此在可以說是完全訓練時分佈以外（out-of-distribution）的情境下，爬蟲類元學習才有辦法勝過基準模型。

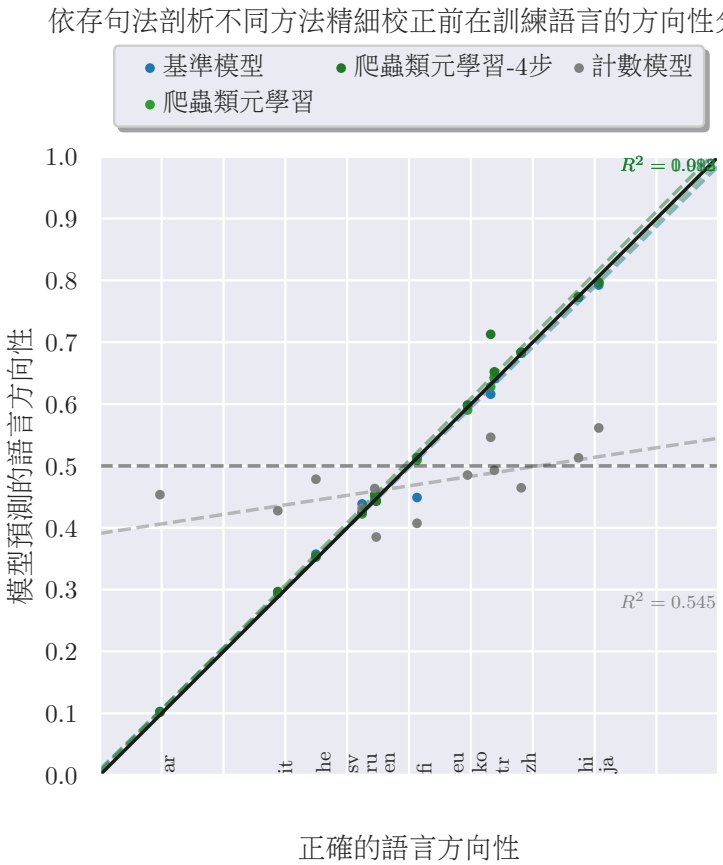
## 4.5 分析與討論

此節中我們試圖觀察與第3.5節一樣的現象，也就是不同方法在精細校正的過程中對中心詞方向性的掌握程度變化。從圖4.3、4.4、4.5、4.6 可以看出對未見過語言的中心詞方向性掌握度較去詞化依存句法剖析時為低，到最後精細校正 80 回合時仍有許多語言（如 mr）未收斂到正確的中心詞方向性，展現了模型同時要學習該

語言詞彙的句法功能與該語言句法方向性的困難度；其中爬蟲類元學習-2 步與 4 步產生的句法樹其中心詞方向性與正確答案的差異又比基準模型要來得大一些，與柱狀圖 UAS 觀察到的現象類似。

## 4.6 小結

我們將模型無關元學習從理想的去詞化依存句法剖析推廣到真實場景的詞化依存句法分析，使用大型多語言預訓練語言模型 mBERT 做為依存句法剖析的編碼器，將各種不同語言的句子映射到同一個向量空間，進行多語言句法剖析預訓練，並精細校正到未見過的語言上。我們發現實驗採用的模型無關元學習之變形-爬蟲類元學習，並無法與基準模型匹敵，mBERT 搭配普通的多語言協同訓練的基準模型就已經表現得足夠好，最後在方向性分析上觀察到三種預訓練方法都無法充分的掌握目標語言的中心詞方向性，說明了用訓練語言的純文字（而非單純的詞性標記）訓練出來的依存句法剖析器轉移到目標語言的困難程度。



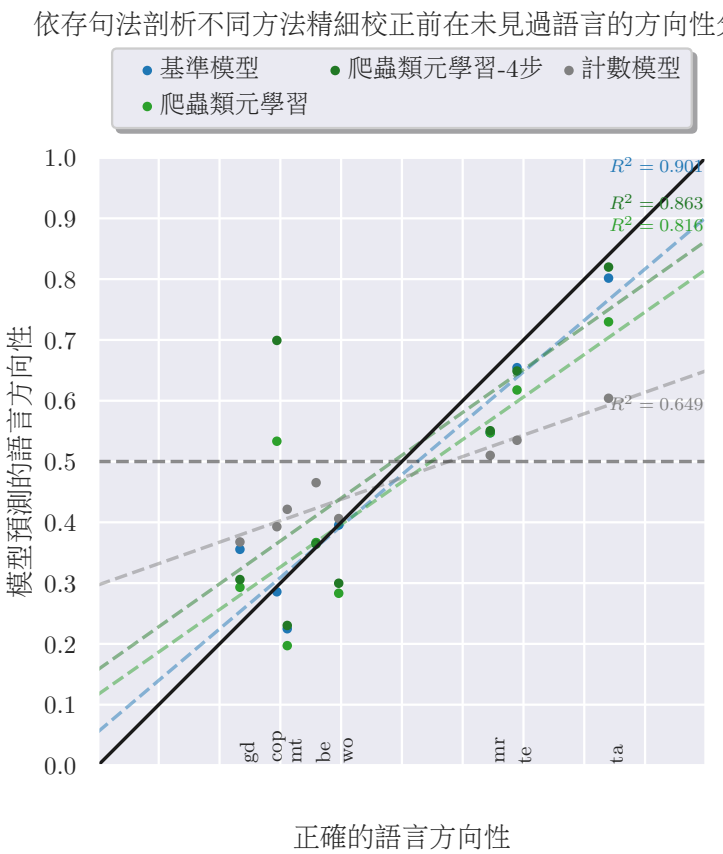
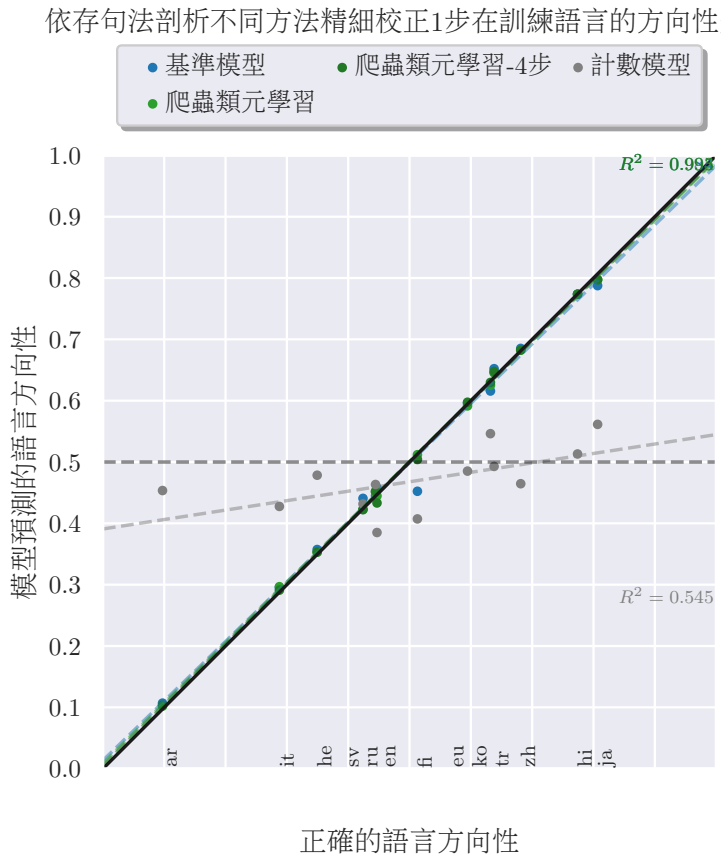




圖 4.3: 依存句法剖析不同方法在各語言精細校正前的方向性分佈。



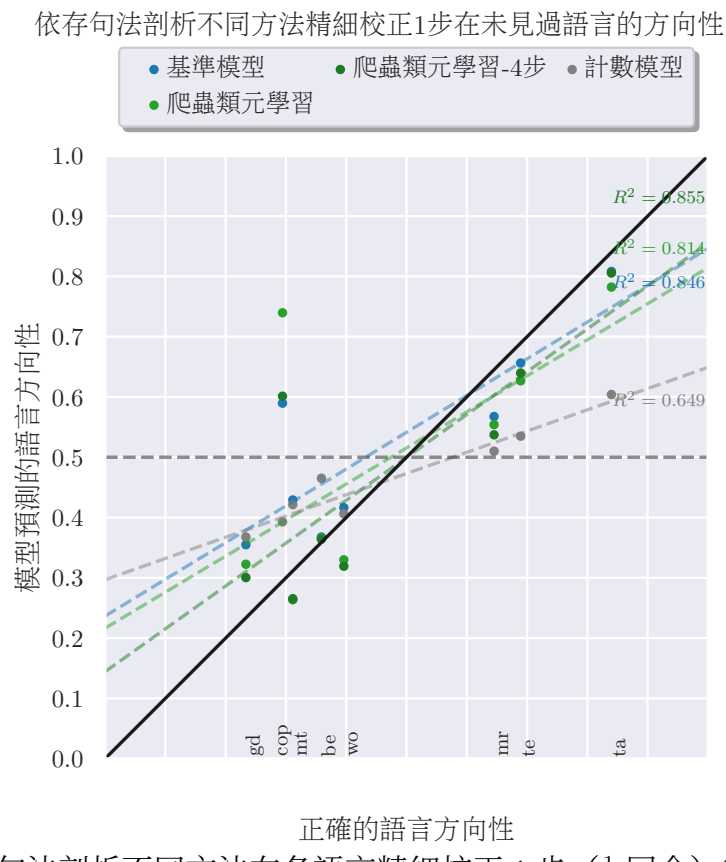




圖 4.4: 依存句法剖析不同方法在各語言精細校正 1 步（$\frac{1}{6}$ 回合）後的方向性分佈。

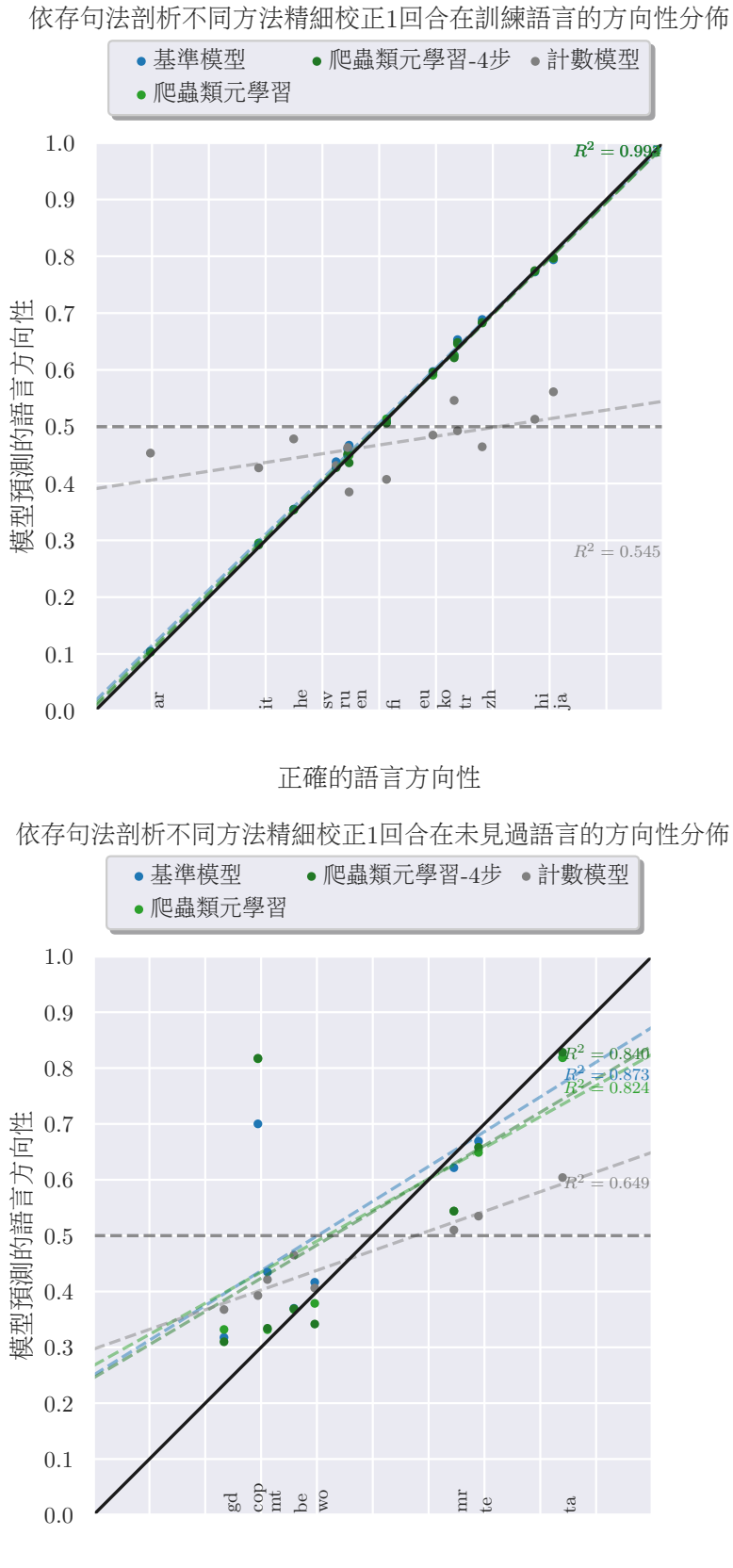



圖 4.5: 依存句法剖析不同方法在各語言精細校正 1 回合後的方向性分佈。

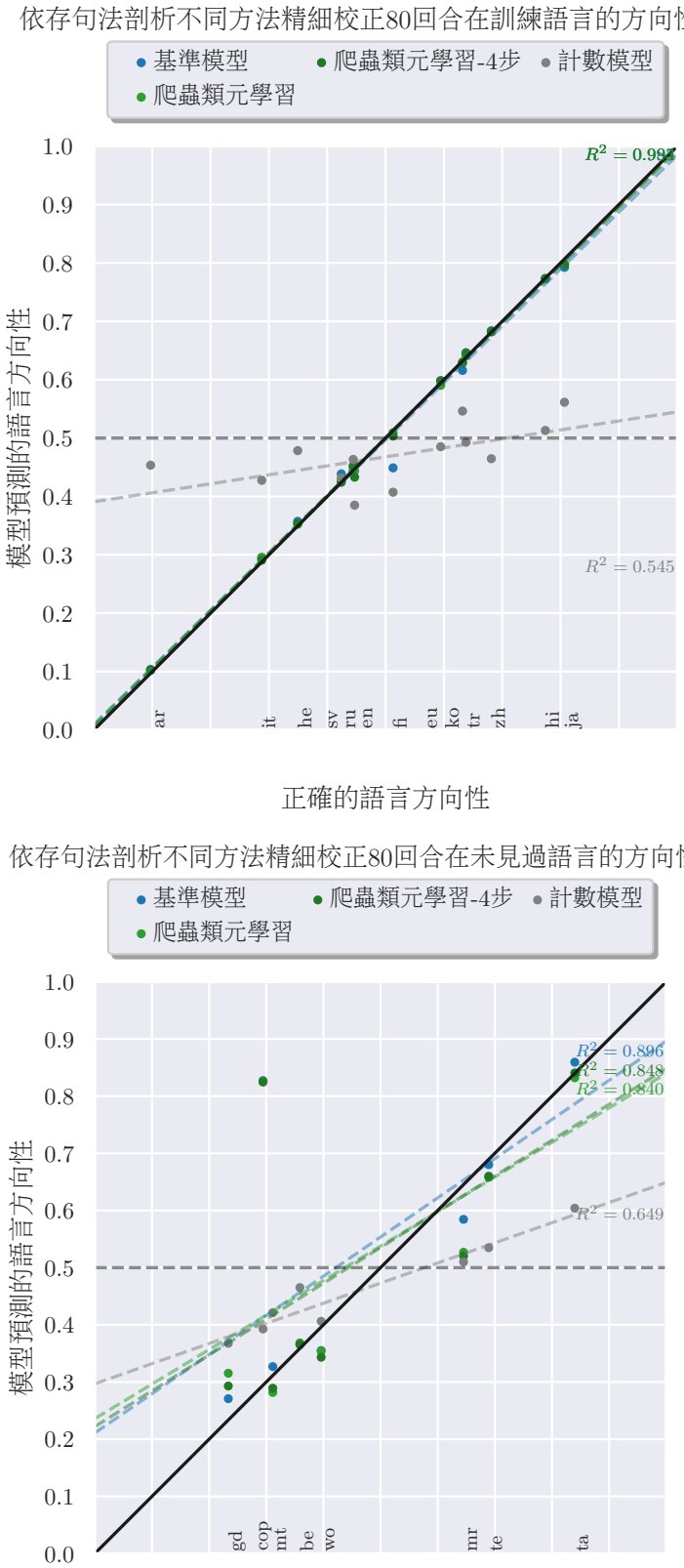



圖 4.6: 依存句法剖析不同方法在各語言精細校正 80 回合後的方向性分佈。

# 第 五 章　結論與展望

## 5.1　研究貢獻與討論

本研究提出使用模型無關元學習預訓練於資料充足語言，並用以改善資料不足語言的依存句法剖析。第一章介紹資料不足語言依存句法剖析的問題，並說明其在人工智慧理論與應用上的重要性。第二章介紹機器學習、依存句法剖析與模型無關元學習的背景知識。為將語言自身無關句法之特性影響依存句法剖析表現的變因排除，第三章先使用普適詞性標記做為表徵進行依存句法剖析，並發現爬蟲類元學習在精細校正前與精細校正少量步數均有與基準模型匹敵或略佳的表現；而模型無關元學習與其一階近似雖然在精細校正前表現較差，但可以在寥寥數步內就將表現提升到與基準模型差不多的準確率，顯示其擁有快速適應的能力。第四章將模型無關元學習推廣至只使用純文字的詞化依存句法分析，發現爬蟲類元學習相較於基準模型並無特別優勢；最後執行了預訓練方法在精細校正過程中學習目標語言中心詞方向性之進程的分析，為日後改進模型無關元學習應用於依存句法剖析提供了有用的參考資料。

## 5.2　未來展望

### 5.2.1　訓練語言的選擇對不同預訓練方法的影響

本研究預訓練所使用的訓練語言在所有實驗中是固定不變的。然而訓練語言對預訓練句法剖析模型來說是人類語言所有可能句法空間的取樣樣本，當面對取樣語言數不足或性質不平衡的取樣語言時，不同模型預訓練演算法不同之處將更容易

被突顯出來。因而探討語言句法空間的取樣偏差（sampling bias）對預訓練模型的影響，也是一項值得深入研究的問題。

### 5.2.2 不同句法樹機率定義對不同預訓練方法的影響

我們在2.4.3提到過兩種定義句法樹機率的方法，並只取較簡單的中心詞選擇交叉熵法對句法樹機率進行近似。然其該方法定義的機率終究只是一種近似，對句法樹來說並非合法的機率分佈（所有句法樹的機率總和可能不等於 1）。相較於中心詞選擇交叉熵法，全域似然性法不在中心詞選擇上做正規化，而是對所有句法樹的分數加總計算出其配分函數，這樣的正規化對句法樹來說方為合法的機率分佈。雖然全域似然性法在單語言句法剖析時勝過中心詞選擇交叉熵法（見 [32] 表 3），我們並不清楚在多語言句法剖析裡，模型必須同時對句法樹機率分佈迥異的多種語言建模的情況下，全域似然性法相較於中心詞選擇交叉熵法是否仍有優勢，因此我們認為這是未來實驗上值得探討的一個方向。

### 5.2.3 不同依存句法剖析演算法對不同預訓練方法的影響

本研究的依存句法剖析主要侷限在圖類剖析器的實驗上，但依存句法剖析尚有轉移類依存句法剖析器（transition-based dependency parser），將依存句法剖析視為以一個個對樹進行修改的動作將句法樹組裝起來的過程，有一個緩衝區（buffer）用來貯存尚未處理的詞，一個堆疊區用來表示當下樹的狀態，以及事先定義好的數個動作（action）將緩衝區的詞一個個接上堆疊區的句法樹，直到完成為止。

阿氏（Wasi Ahmad）曾經在 [17] 中比較過轉移類方法與圖類方法在跨語言轉移學習的優劣，他發現除了在相近語言（如英文與荷蘭文）間轉移類方法較圖類方法為佳之外，圖類方法整體來說有較好的跨語言轉移能力，且在不相干的語言

間尤甚。轉移類方法中的不同語言的動作序列（action sequence）差異過大可能是其轉移不易的原因。然而 [17] 只探討從單一語言（英文）的句法剖析模型出發轉移到其他語言的情況，主要探討不同單語言句法剖析模型轉移的難易度。在多語言預訓練的框架下則可探討元學習相較於多語言學習是否可以拉近轉移類方法與圖類方法在跨語言轉移能力的差距。

### 5.2.4 不同編碼器對不同預訓練方法的影響

阿氏在 [17] 中也比較了轉換器（Transformer）與遞歸類神經網路（RNN）作為依存句法剖析的編碼器在跨語言轉移學習上的表現差異。他發現轉換器相較於遞歸類神經網路更容易進行跨語言轉移學習。本研究於去詞化依存句法剖析使用遞歸類神經網路，於詞化依存句法剖析則使用轉換器組成的 mBERT，並沒有在控制其他變因的情況下探討過兩種不同架構的編碼器是否會對元學習與多語言學習在零樣本與精細校正的行為有系統性的影響，也是一項值得以實驗釐清的問題。

# 參 考 文 獻


[1] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Efficient parametrization of multi-domain deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8119–8127.

[2] N. Chomsky, "Rules and representations," *Behavioral and brain sciences*, vol. 3, no. 1, 1980.

[3] J. H. Greenberg, "Universals of language." 1963.

[4] G. K. Zipf, "Human behavior and the principle of least effort," 1949.

[5] J. Nivre, M.-C. de Marneffe, F. Ginter, Y. Goldberg, J. Hajic, C. D. Manning, R. T. McDonald, S. Petrov, S. Pyysalo, N. Silveira, R. Tsarfaty, and D. Zeman, "Universal dependencies v1: A multilingual treebank collection," in *LREC*, 2016.

[6] J. Nivre, M.-C. de Marneffe, F. Ginter, J. Hajivc, C. D. Manning, S. Pyysalo, S. Schuster, F. Tyers, and D. Zeman, "Universal dependencies v2: An evergrowing multilingual treebank collection," in *LREC*, 2020.

[7] D. M. Eberhard, G. F. Simons, and C. D. Fennig, *Ethnologue, languages of the world*, 23rd ed. SIL International, 2020.

[8] T. M. Mitchell, *The need for biases in learning generalizations*. Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ., 1980.

[9] Y. Zhang and R. Barzilay, "Hierarchical low-rank tensors for multilingual transfer parsing," in *Proceedings of the 2015 Conference on Empirical Methods in*

*Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 1857–1867. [Online]. Available: https://www.aclweb.org/anthology/D15-1213

[10] Ž. Agić, A. Johannsen, B. Plank, H. Martínez Alonso, N. Schluter, and A. Søgaard, "Multilingual projection for parsing truly low-resource languages," *Transactions of the Association for Computational Linguistics*, vol. 4, pp. 301–312, 2016. [Online]. Available: https://www.aclweb.org/anthology/Q16-1022

[11] M. S. Rasooli and M. Collins, "Cross-lingual syntactic transfer with limited resources," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 279–293, 2017. [Online]. Available: https://www.aclweb.org/anthology/Q17-1020

[12] M. S. Dryer and M. Haspelmath, Eds., *WALS Online*. Leipzig: Max Planck Institute for Evolutionary Anthropology, 2013. [Online]. Available: https://wals.info/

[13] T. Naseem, R. Barzilay, and A. Globerson, "Selective sharing for multilingual dependency parsing," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Jeju Island, Korea: Association for Computational Linguistics, Jul. 2012, pp. 629–637. [Online]. Available: https://www.aclweb.org/anthology/P12-1066

[14] O. Täckström, R. McDonald, and J. Nivre, "Target language adaptation of discriminative transfer parsers," in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Atlanta, Georgia: Association for Computational

Linguistics, Jun. 2013, pp. 1061–1071. [Online]. Available: https://www.aclweb.org/anthology/N13-1126

[15] L. Aufrant, G. Wisniewski, and F. Yvon, "Zero-resource dependency parsing: Boosting delexicalized cross-lingual transfer with linguistic knowledge," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, Dec. 2016, pp. 119–130. [Online]. Available: https://www.aclweb.org/anthology/C16-1012

[16] P. Littell, D. R. Mortensen, K. Lin, K. Kairis, C. Turner, and L. Levin, "URIEL and lang2vec: Representing languages as typological, geographical, and phylogenetic vectors," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 8–14. [Online]. Available: https://www.aclweb.org/anthology/E17-2002

[17] W. Ahmad, Z. Zhang, X. Ma, E. Hovy, K.-W. Chang, and N. Peng, "On difficulties of cross-lingual transfer with order differences: A case study on dependency parsing," 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)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 2440–2452. [Online]. Available: https://www.aclweb.org/anthology/N19-1253

[18] Y.-H. Lin, C.-Y. Chen, J. Lee, Z. Li, Y. Zhang, M. Xia, S. Rijhwani, J. He,

Z. Zhang, X. Ma, A. Anastasopoulos, P. Littell, and G. Neubig, "Choosing transfer languages for cross-lingual learning," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 3125–3135. [Online]. Available: https://www.aclweb.org/anthology/P19-1301

[19] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural networks*, vol. 4, no. 2, pp. 251–257, 1991.

[20] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008. [Online]. Available: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[23] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, "Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 5797–5808. [Online]. Available: https://www.aclweb.org/anthology/P19-1580

[24] J. Reisinger and R. J. Mooney, "Multi-prototype vector-space models of word meaning," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Los Angeles, California: Association for Computational Linguistics, Jun. 2010, pp. 109–117. [Online]. Available: https://www.aclweb.org/anthology/N10-1013

[25] A. Neelakantan, J. Shankar, A. Passos, and A. McCallum, "Efficient non-parametric estimation of multiple embeddings per word in vector space," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1059–1069. [Online]. Available: https://www.aclweb.org/anthology/D14-1113

[26] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 2227–2237. [Online]. Available: https://www.aclweb.org/anthology/N18-1202

[27] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[28] J. Howard and S. Ruder, "Universal language model fine-tuning for text classification," *arXiv preprint arXiv:1801.06146*, 2018.

[29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," 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)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://www.aclweb.org/anthology/N19-1423

[30] W. T. Tutte, *Graph Theory*. Addison-Wesley Menlo Park, 1984, vol. 11.

[31] T. Koo, A. Globerson, X. Carreras, and M. Collins, "Structured prediction models via the matrix-tree theorem," in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, Czech Republic: Association for Computational Linguistics, Jun. 2007, pp. 141–150. [Online]. Available: https://www.aclweb.org/anthology/D07-1015

[32] X. Ma and E. Hovy, "Neural probabilistic model for non-projective MST parsing," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 59–69. [Online]. Available: https://www.aclweb.org/anthology/I17-1007

[33] T. Dozat and C. D. Manning, "Deep biaffine attention for neural dependency parsing," *ArXiv*, vol. abs/1611.01734, 2017.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[35] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[36] J. Gu, Y. Wang, Y. Chen, V. O. K. Li, and K. Cho, "Meta-learning for low-resource neural machine translation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 3622–3631. [Online]. Available: https://www.aclweb.org/anthology/D18-1398

[37] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *ICML*, 2017.

[38] R. Caruana, "Multitask learning," *Machine learning*, vol. 28, no. 1, pp. 41–75, 1997.

[39] S. Petrov, D. Das, and R. McDonald, "A universal part-of-speech tagset," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey: European Language Resources Association (ELRA), May 2012, pp. 2089–2096. [Online]. Available: http://www.lrec-conf.org/proceedings/lrec2012/pdf/274_Paper.pdf

[40] D. Kondratyuk and M. Straka, "75 languages, 1 model: Parsing universal dependencies universally," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 2779–2795. [Online]. Available: https://www.aclweb.org/anthology/D19-1279

[41] A. Üstün, A. Bisazza, G. Bouma, and G. van Noord, "Udapter: Language adaptation for truly universal dependency parsing," *arXiv preprint arXiv:2004.14327*, 2020.

[42] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1715–1725. [Online]. Available: https://www.aclweb.org/anthology/P16-1162

[43] M. Schuster and K. Nakajima, "Japanese and korean voice search," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 5149–5152.

[44] I. Tenney, D. Das, and E. Pavlick, "BERT rediscovers the classical NLP pipeline," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 4593–4601. [Online]. Available: https://www.aclweb.org/anthology/P19-1452

[45] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," *arXiv preprint arXiv:1902.00751*, 2019.

# 附　　錄

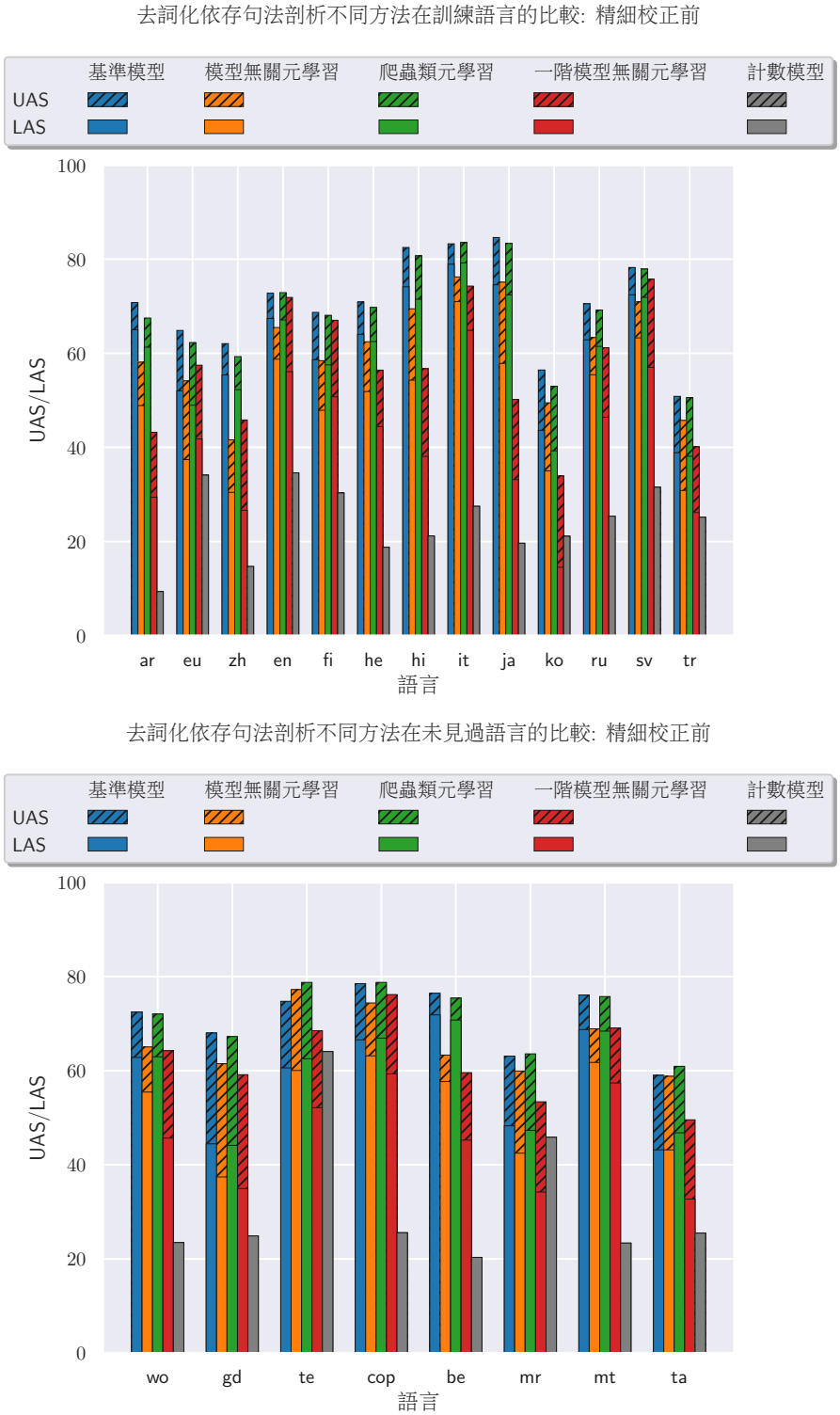



圖 5.1: 去詞化依存句法剖析不同方法在各語言精細校正前的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。

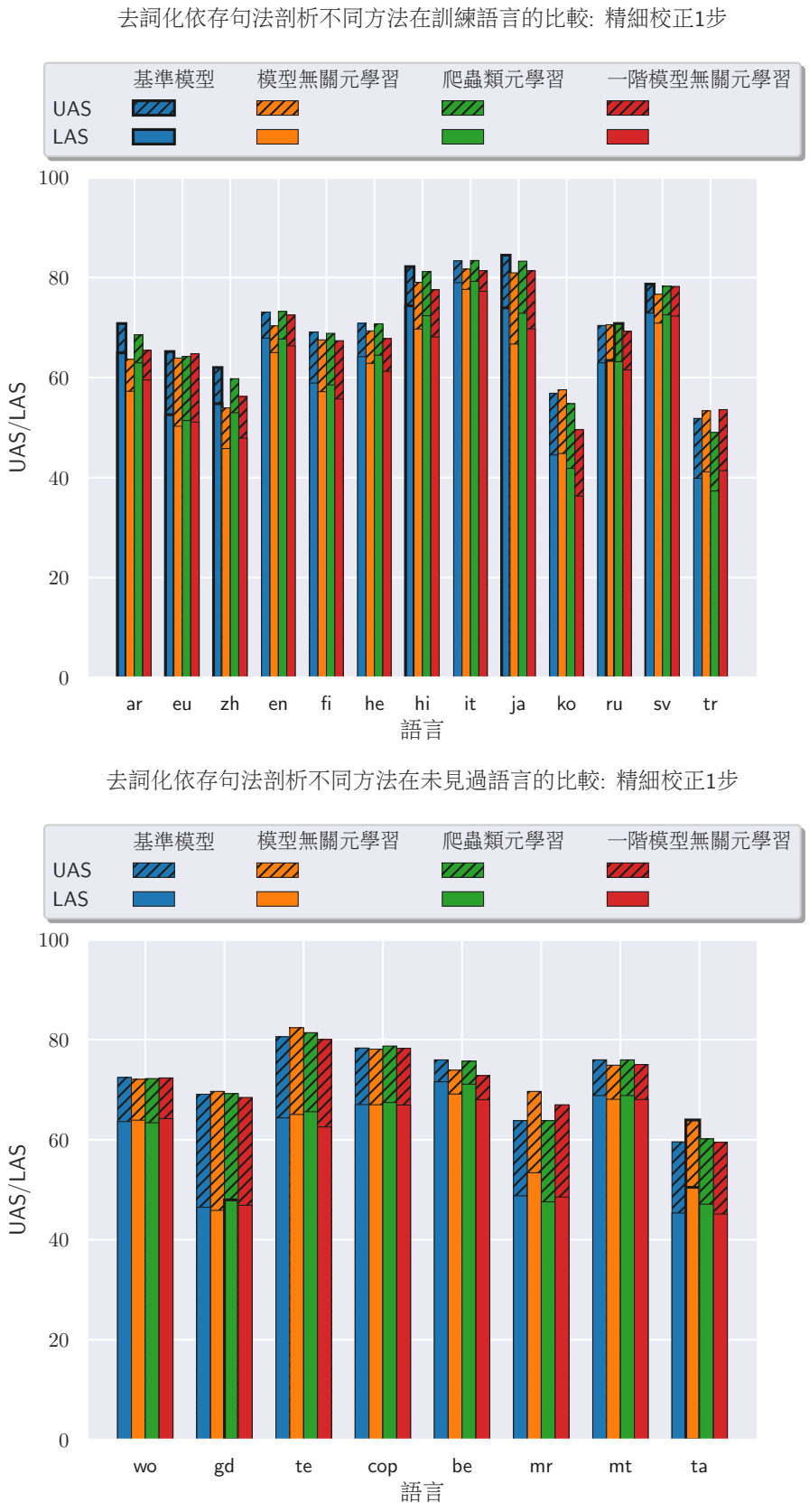



圖 5.2: 去詞化依存句法剖析不同方法在各語言精細校正 1 步（$\frac{1}{6}$ 回合）後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。

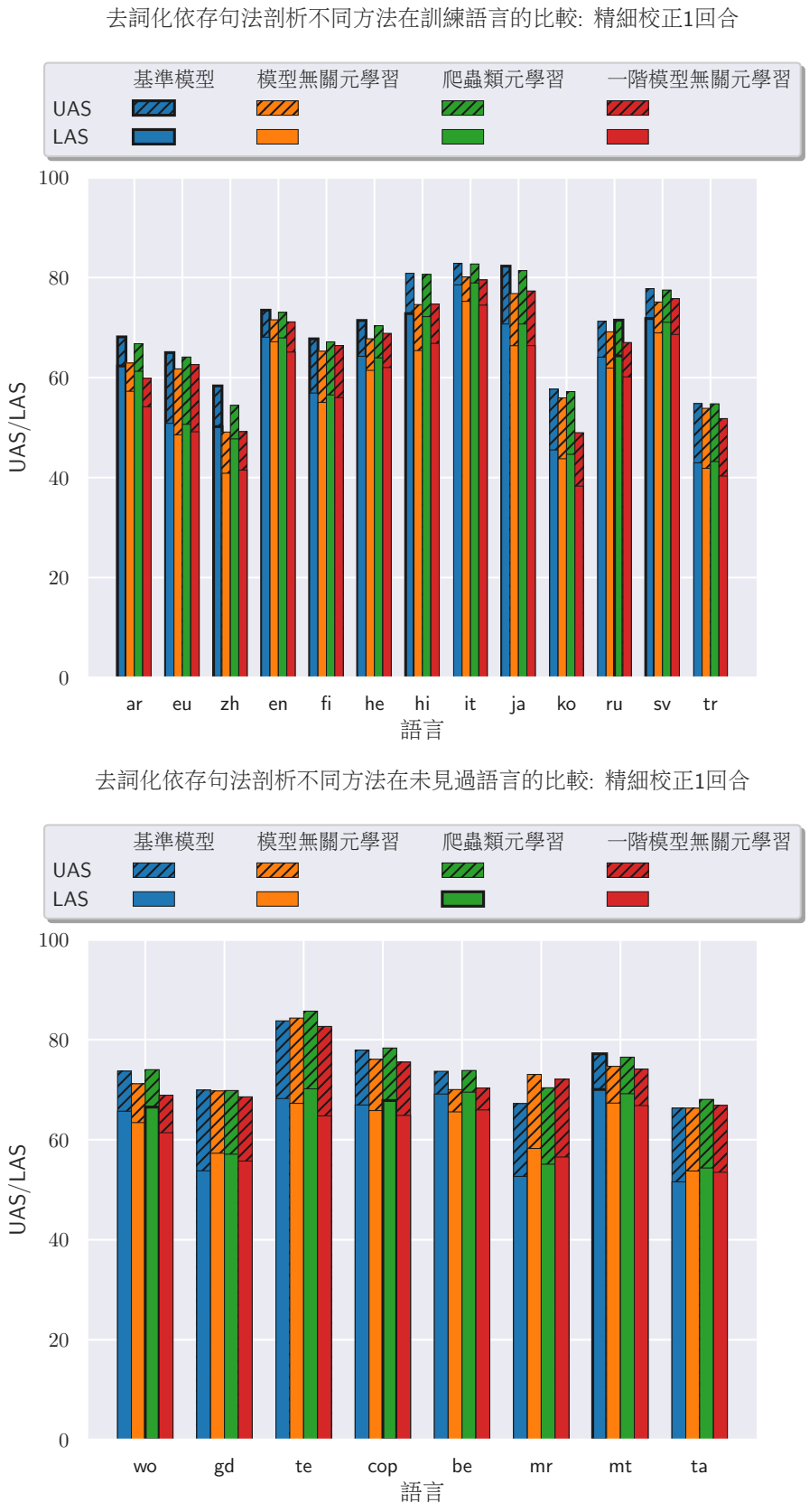



圖 5.3: 去詞化依存句法剖析不同方法在各語言精細校正 1 回合後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。

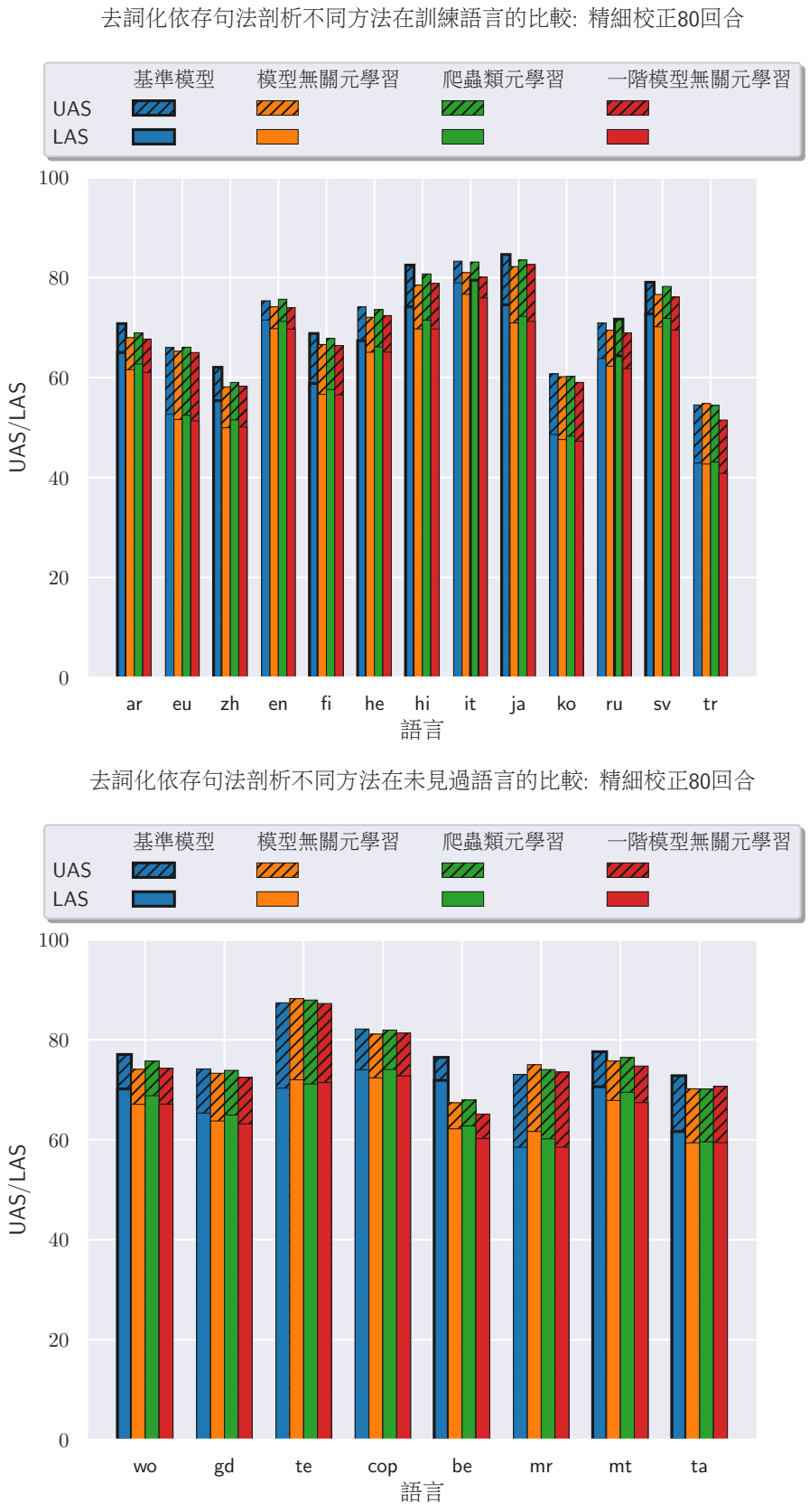



圖 5.4: 去詞化依存句法剖析不同方法在各語言精細校正 80 回合後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。

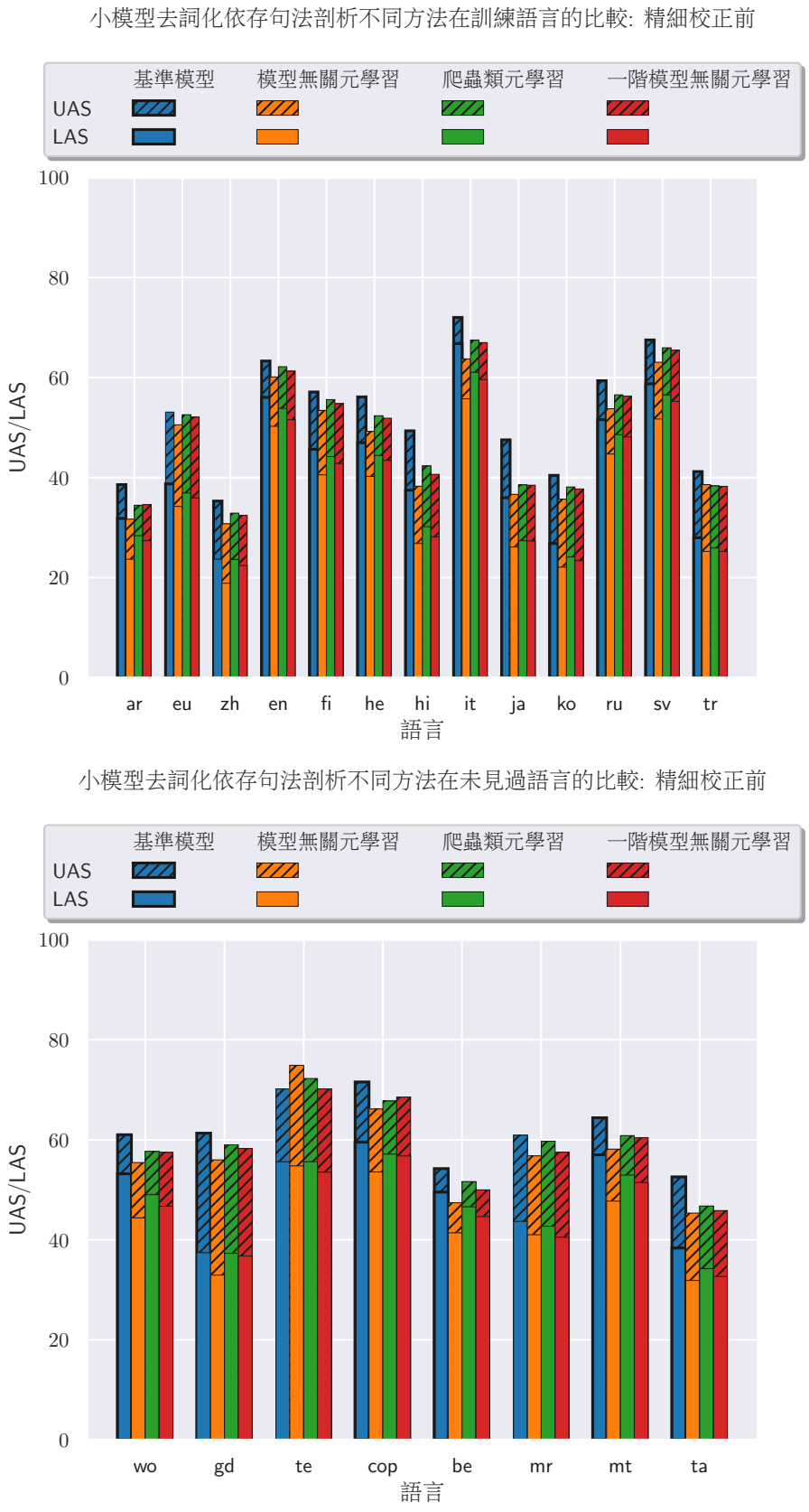



圖 5.5: 小模型去詞化依存句法剖析不同方法在各語言精細校正前的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。

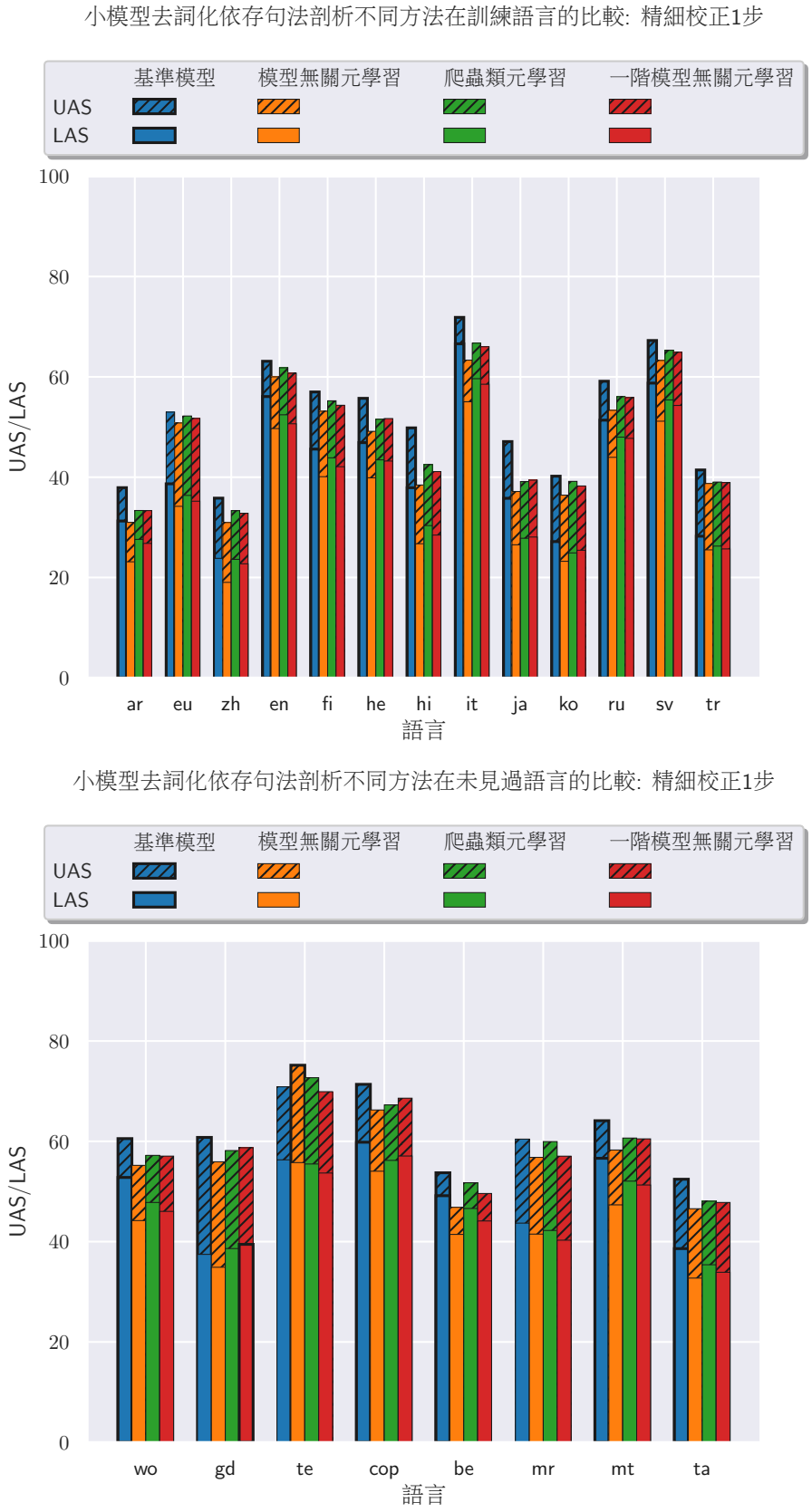



圖 5.6: 小模型去詞化依存句法剖析不同方法在各語言精細校正 1 步（$\frac{1}{6}$ 回合）後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。

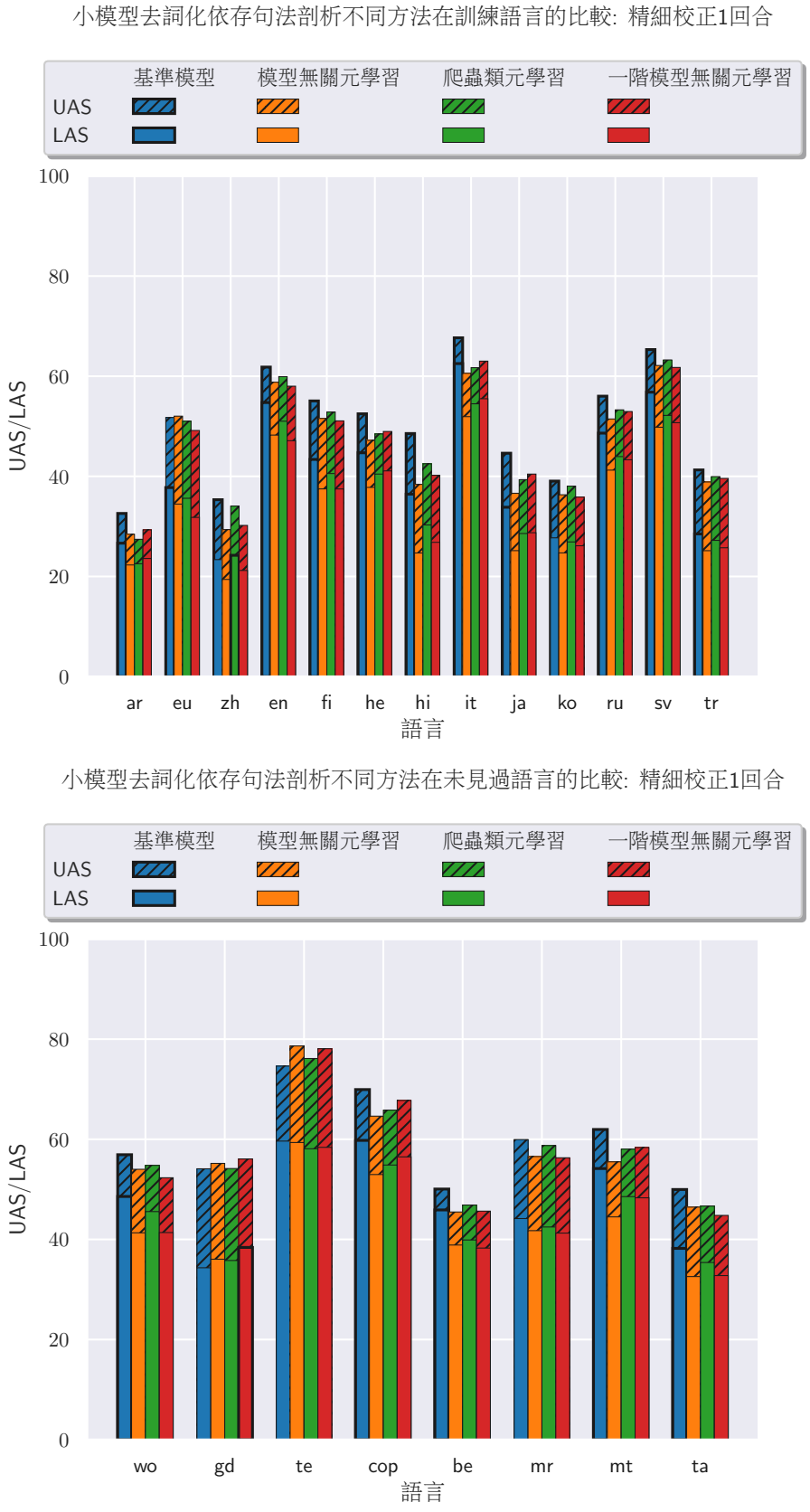



圖 5.7: 小模型去詞化依存句法剖析不同方法在各語言精細校正 1 回合後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。

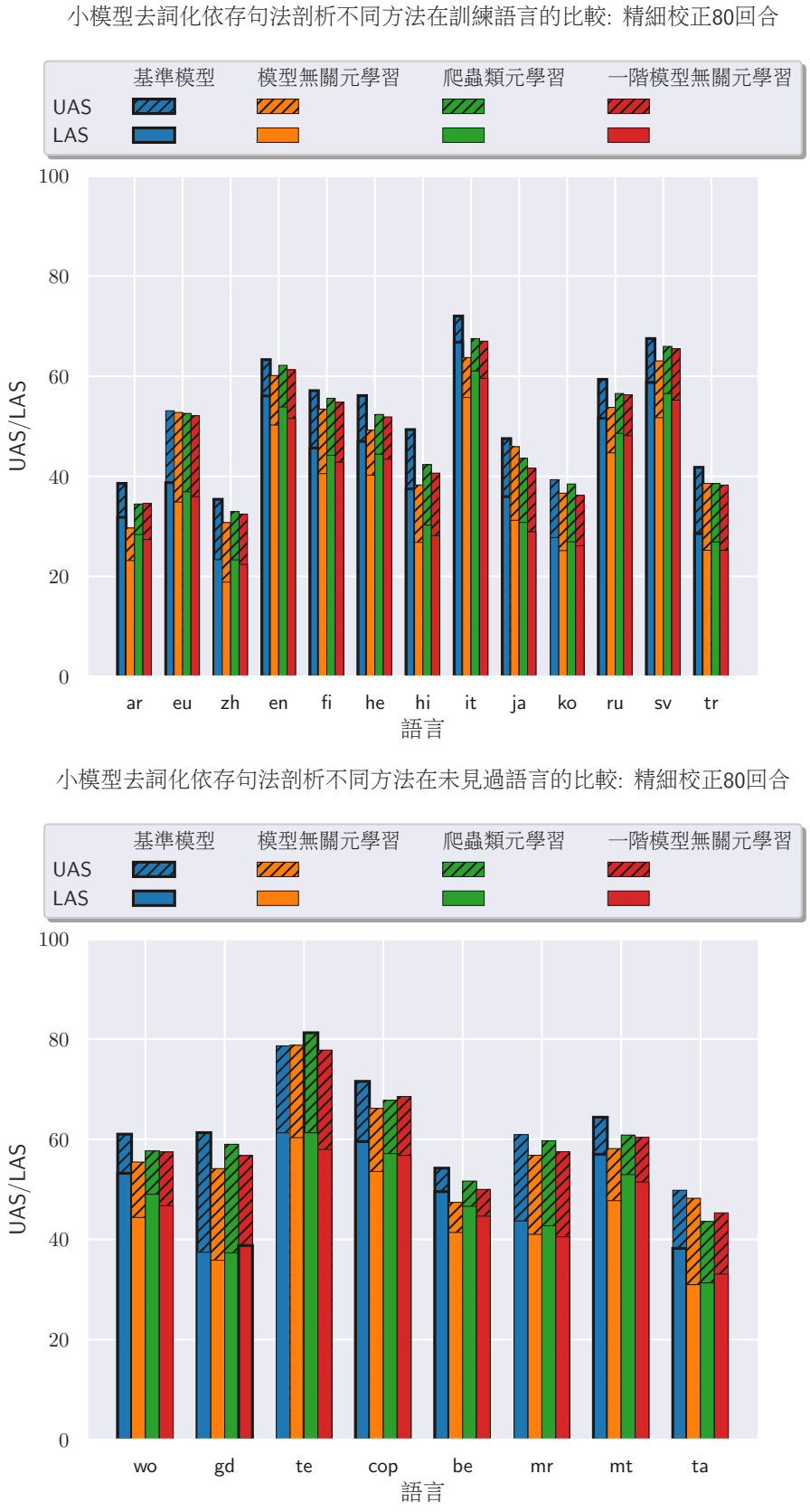
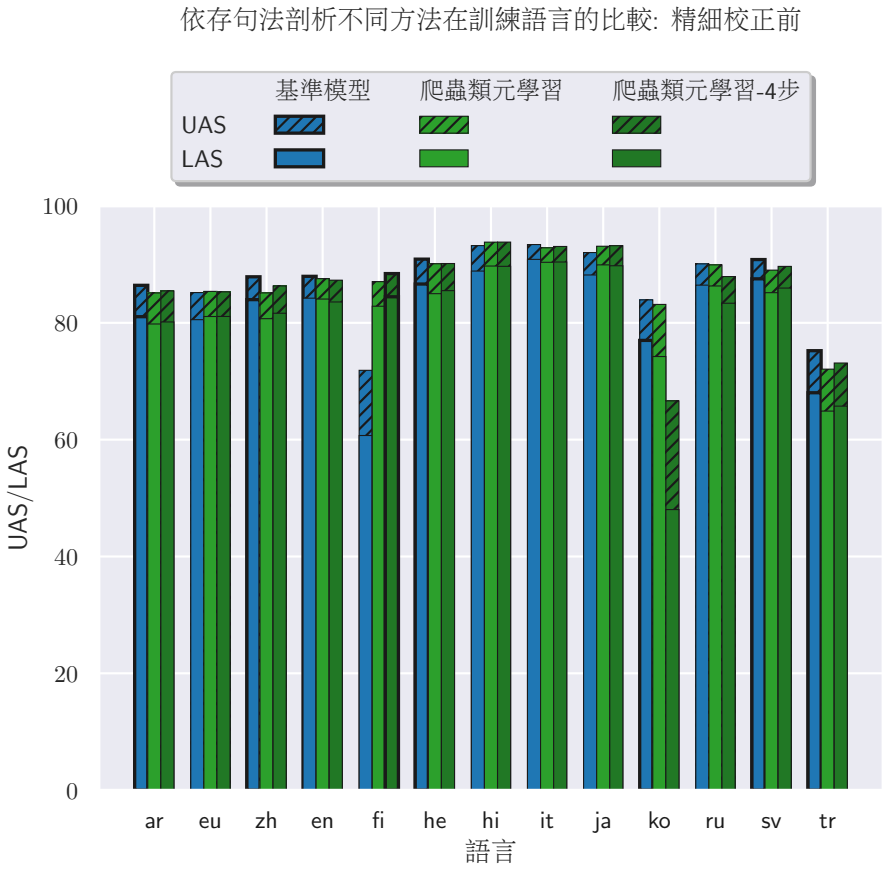



圖 5.8: 小模型去詞化依存句法剖析不同方法在各語言精細校正 80 回合後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。



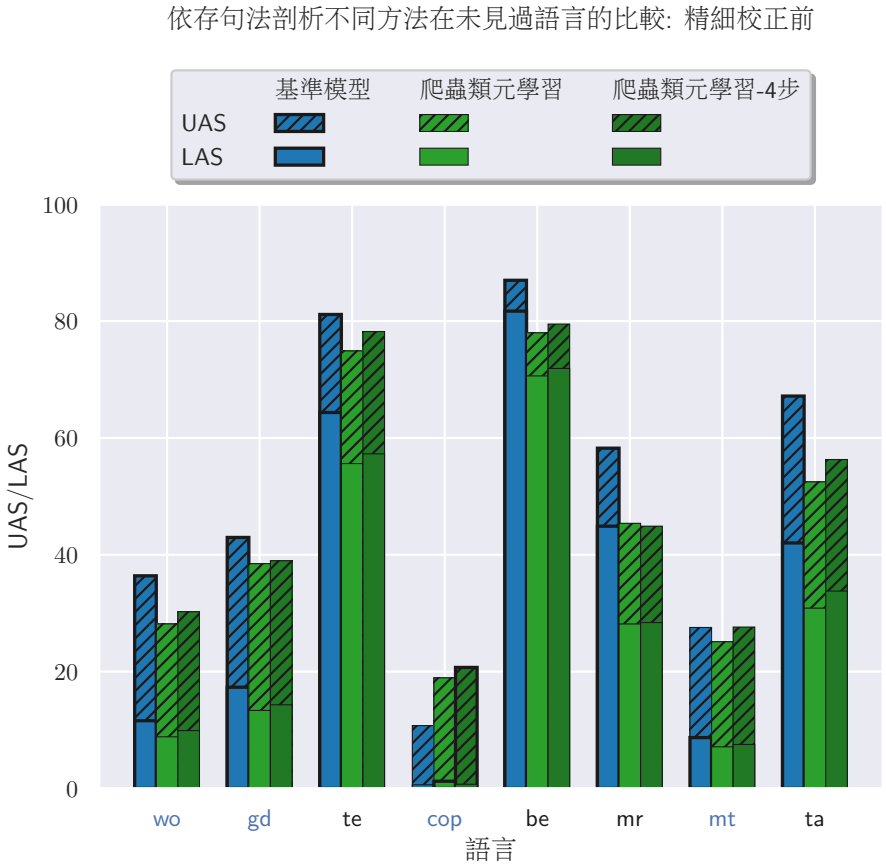
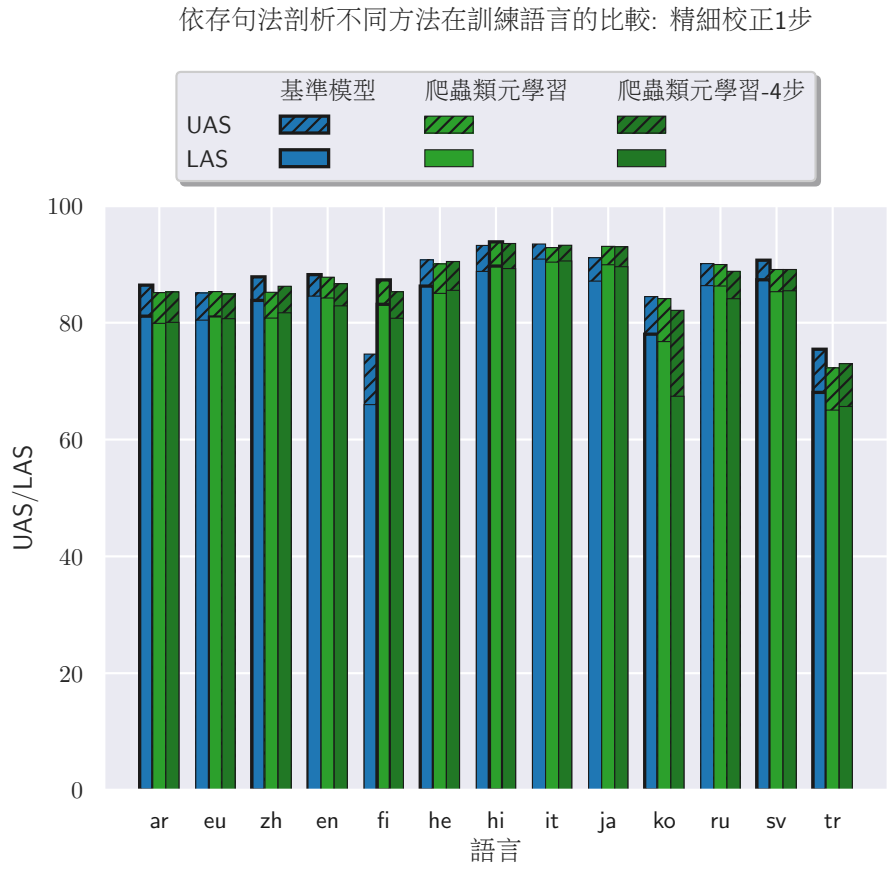




圖 5.9: 依存句法剖析不同方法在各語言精細校正前的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。上色之語言為不在 mBERT 的預訓練語言者。



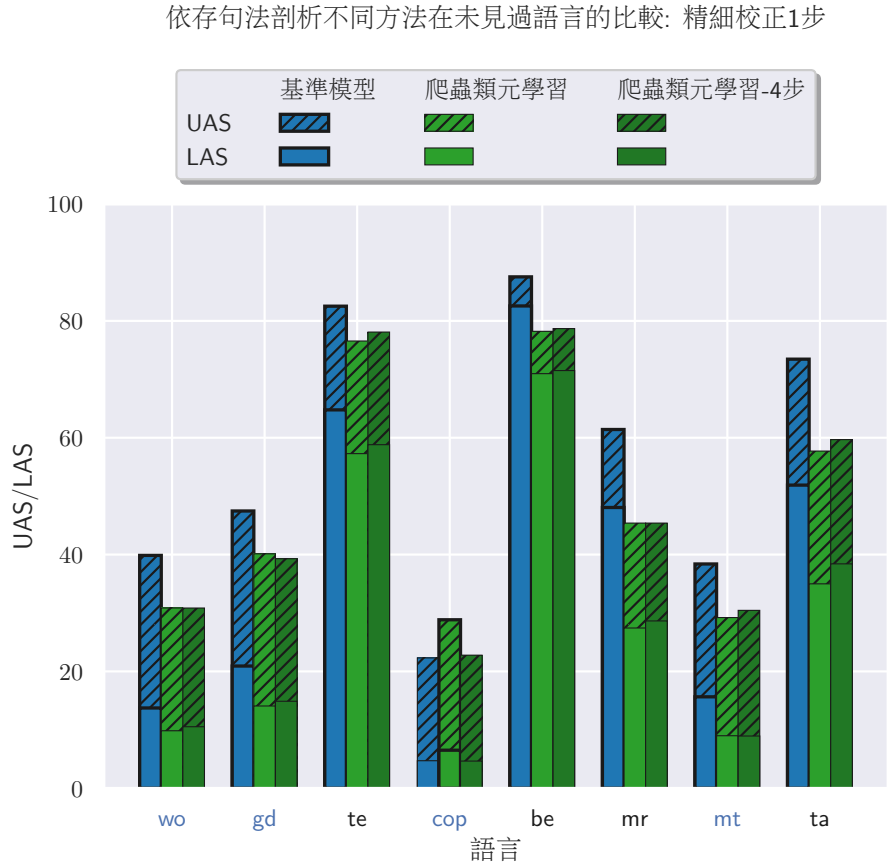
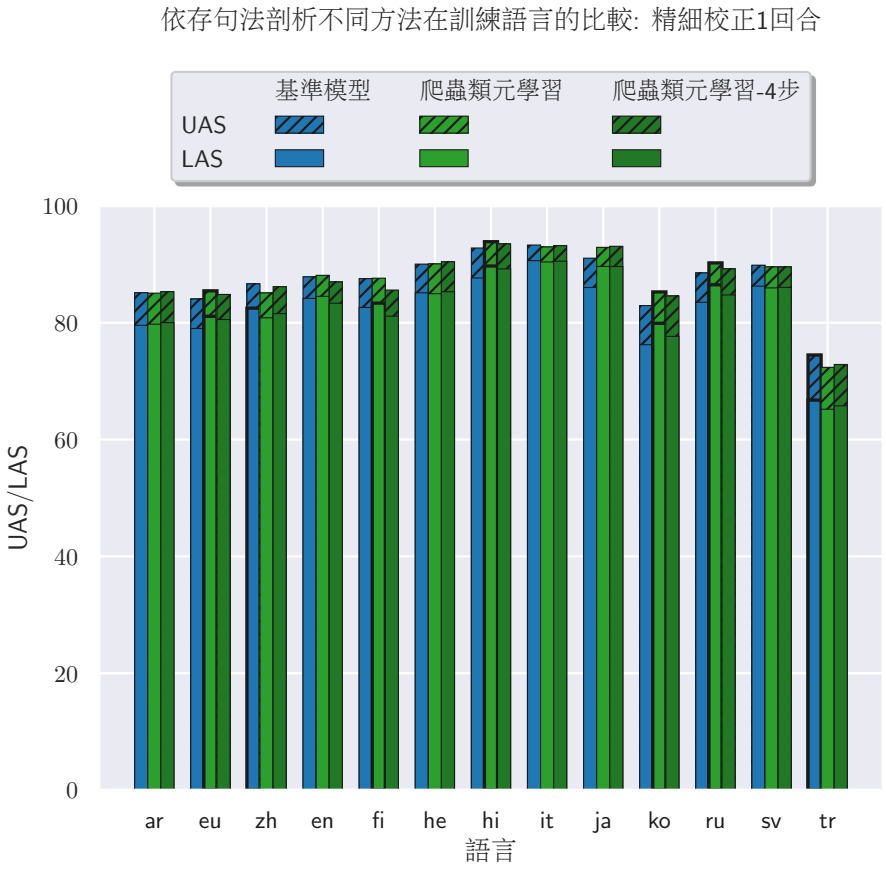




圖 5.10: 依存句法剖析不同方法在各語言精細校正 1 步（$\frac{1}{6}$ 回合）後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。上色之語言為不在 mBERT 的預訓練語言者。



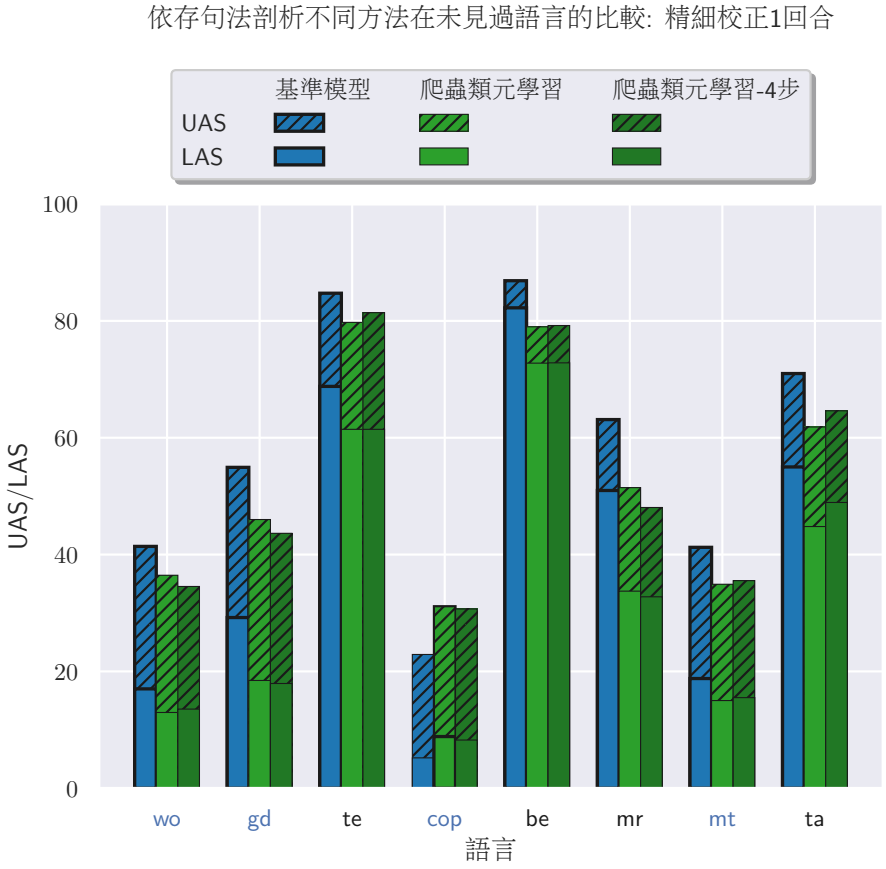




圖 5.11: 依存句法剖析不同方法在各語言精細校正 1 回合後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。上色之語言為不在 mBERT 的預訓練語言者。

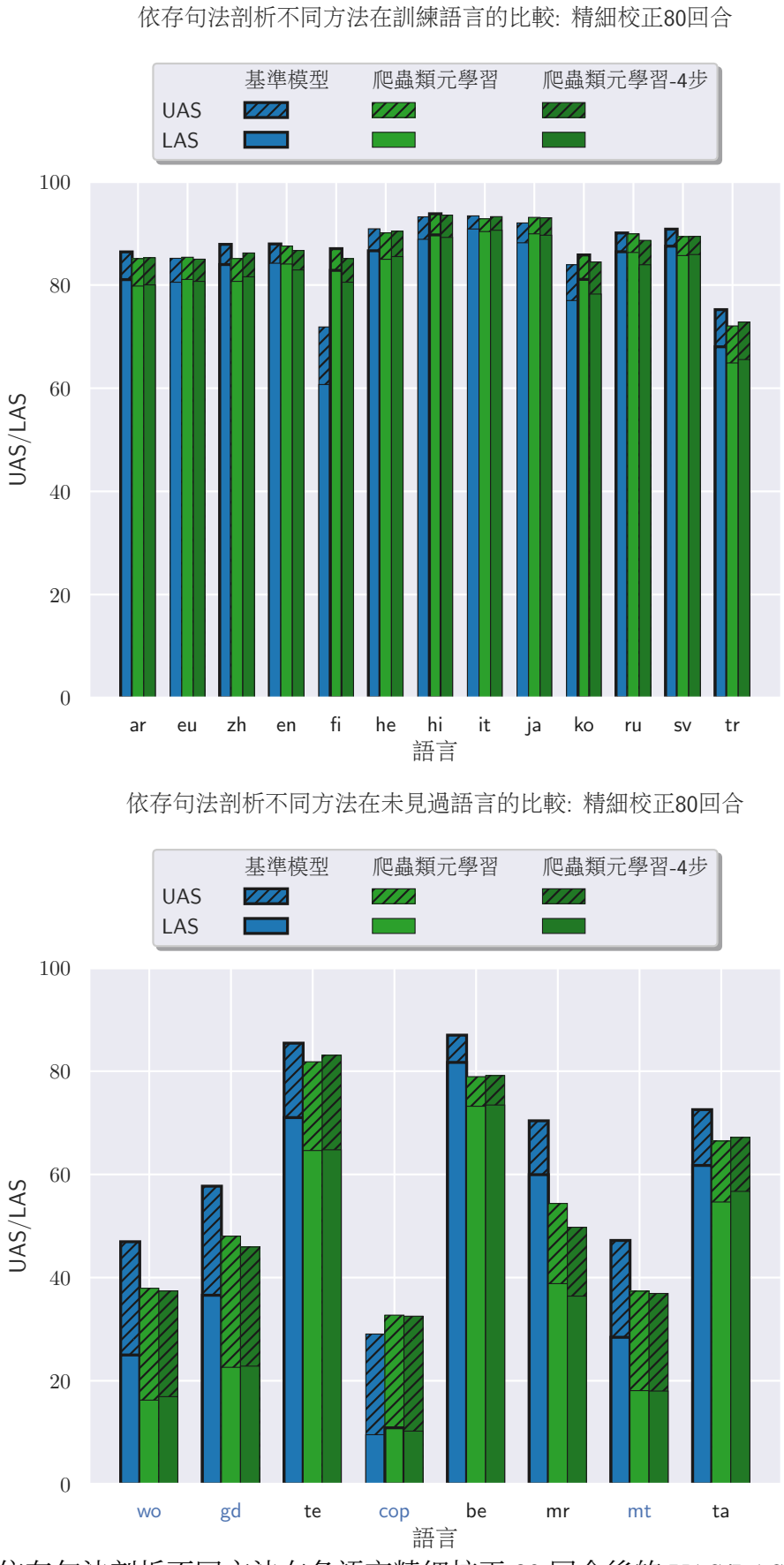



圖 5.12: 依存句法剖析不同方法在各語言精細校正 80 回合後的 UAS/LAS 長條圖。邊框較粗者為統計顯著勝過其他方法之意。上色之語言為不在 mBERT 的預訓練語言者。