

深度學習的未來 - 程式合成 (program synthesis)

短短幾年的時間內，深度學習已取得了巨大的突破，特別是在機器感知 (perception) 領域：只要輸入足夠的訓練資料 (影像、影片與聲音)，便可以從中萃取出幾乎是人類所能萃取的任何資訊。深度學習雖然早已被過度炒作，不過在充分發展其潛能下，仍會有十多年的興盛光景。

文／旗標科技

我們在機器學習領域所期待的必要轉型發展，就是要脫離那些純態樣識別 (pattern recognition)、也就是只適用特定領域的模型，並朝向擁有抽象和推理能力、可用於解決大部分問題的極度普適模型邁進。目前能夠進行基本推理形式的 AI 程式都是由人類工程師手動撰寫程式碼而成，例如那些依賴於搜索演算法、圖形處理和形式邏輯的 AI 系統。

甚麼是程式合成？

改變現有這種開發方式的關鍵，可能會是程式合成 (program synthesis)。雖然目前還是很小眾的領域，但在未來幾十年內，它可能會有大的躍進。程式合

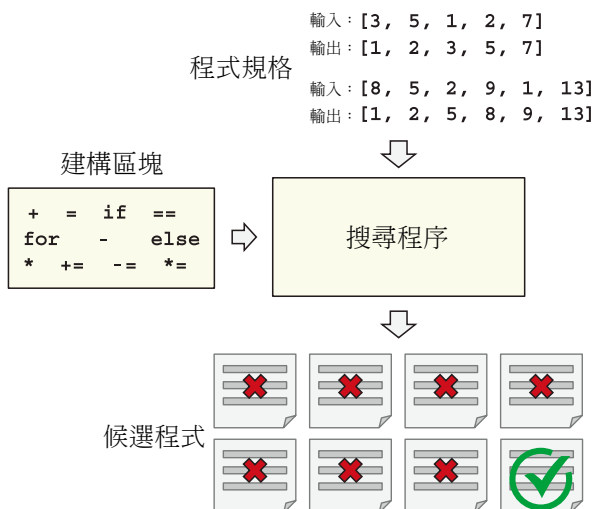


圖 1 程式合成圖解：給定程式規格和一組建構區塊，搜尋程序就會將建構區塊整合為候選程式，並測試是否符合規格。在找出有效的程式後，搜尋過程才會停止

成會使用搜尋演算法（可能是基因程式設計中的基因搜尋法）來探索大量可能的程式，並自動生成簡單的程式（見圖 1）。當它找到一個符合需求規格的程式時，搜尋就會停止。這些規格的形式通常是一組輸入-輸出組合。這很容易令人聯想到機器學習：給定以輸入-輸出形式呈現的訓練資料後，我們要找到能將輸入對應到輸出，並能普適到新輸入上的程式。不同之處在於，我們不是在寫死的程式（神經網路）中學習參數值，而是透過離散搜尋來生成原始程式碼（見表 1）。

▼ 表 1 機器學習與程式合成的比較

機器學習	程式合成
模型：可微分的參數式函數	模型：程式語言算符的運算圖
引擎：梯度下降	引擎：離散搜尋(如：基因搜尋法)
需要大量資料才能產生可信的結果	不需大量資料，只需幾個訓練範例就能運作

程式合成，讓我們能在 AI 系統中添加程式導向的抽象能力，也就是拼圖中缺少的那一塊。

深度學習及程式合成的融合

當然，深度學習不會消失，程式合成並不是它的替代品，而是它的補充品。程式合成是目前為止我們的人工大腦中缺少的那一半。我們要結合兩者並加以利用，主要的方法如下：

- 開發出能同時整合深度學習模組和離散演算法模組的系統。
- 使用深度學習，讓程式搜尋過程更有效率。

現在，最強大的 AI 系統是混合式 (hybrid) 的系統。這種系統能同時利用深度學習模型和人工寫成的符號操縱程式。以 DeepMind 的 AlphaGo 為例，它展現出來的智慧大部分都是由人類程式設計師所設計、硬編碼出來的（如蒙地卡羅樹搜尋法），而「從資料學習」只發生在特定的子模組（價值網路和策略網路）中。再來想想看自動駕駛的例子：自駕車之所以能處理各式各樣的情況，是因為它一直維持著周遭事物的模型（也就是一個 3D 模型），其中充滿了人類工程師寫死的假設。這個模型會透過深度學習感知模組不斷地更新，進而了解汽車的周圍環境。

對 AlphaGo 和自駕車而言，「人類創造的離散程式」和「學習而來的連續模型」之結合，是讓表現更上一層樓的原因。如果少了兩者中的任何一方，都是不可能達成如此表現的。目前為止，這種混合系統的離散演算法元素，都是由人類工程師寫死的。但在未來，這樣的系統或許不需要人類的參與，可以完全自行學習。

現在，想像一個神經網路以程式原語 (programming primitive) 進行擴增，但它不是只有寫死的 for 迴圈，而是包含了大量程式設計原語，讓模型可以自由操作、擴展處理功能，例如：if 條件式、while 敘述、變數的創建、長期記憶的磁碟儲存空間、算符排序、高階資料結構（串列、圖形和雜湊表）等等。這種模型能表示的程式空間，會比目前深度學習模型所能表示的廣泛許多，且其中一些程式還能達到更高的普適性。更重要的是，雖然有些特定模組還是可微分的，但程序本身不會是可微分的端到端程式，因此必須透過離散程式搜尋和梯度下降的組合來生成。

我們將擁有一個由「提供推理和抽象能力的正規演算法模組」，以及「提供非正規直覺和模式識別能力的幾何模組」組合而成的混合體。整套系統能在

只需極少，或甚至無人參與的情況下學習而成。這將大大擴展能透過機器學習解決的問題範圍，也就是在給定適當的訓練資料下，能夠自動生成的程式空間。

使用深度學習引導程式搜尋

目前程式合成面臨一個大難題：它的效率極低。程式合成的運作方式，是在搜尋空間中嘗試每一個可能的程式，直到找出符合規範的程式為止。隨著需求規格的複雜度逐漸增加，或用來寫程式的原語詞彙表擴增，程式搜尋過程就會遇到所謂的組合爆炸 (combinatorial explosion)，也就是它要考慮的程式組合增長極快（事實上，比單純的指數增長還要快）。因此，程式合成目前只能用來生成非常短的程式，短期內是無法為電腦生成一個新的作業系統的。

深度學習能幫程式合成做到相同的事：儘管我們想生成的每個具體程式本質上都可能是離散的物件，會執行無法被內插的資料操作，但目前為止的證據表明：包含了全部有用程式的空間，看起來可能很像一個連續的流形。這意味著，一個經過數百萬次「成功的程式生成週期」的深度學習模型，可能會開始對「通過程式空間的路徑」，也就是搜尋過程應採取的、「從規格通往相應程式」的那條路徑產生很強的直覺。這就像軟體工程師可能對自己要寫的程式架構，以及通往目標所需的那些中間函式和類別，會產生的直覺一樣。

鴻海研究院執行長 李維斌曾說：「掌握本質，了解其所能與不能，是面對技術快速迭代、世局難以預測的鑰匙。」AI 三個月是一個世代，超過一年就是史前了。**Keras 大神歸位：深度學習全面進化！用 Python 實作 CNN、RNN、GRU、LSTM、GAN、VAE、Transformer** 一書，帶您認清深度學習的真實面貌，在快速發展的 AI 領域中，隨時保持戰鬥狀態。



Learning by doing
從做中學
系列

跟著
Keras 大神
的腳步——
升級你的 **深度學習** 技能！

PLATO
旗標科技