Chapter 04

卷積神經網路

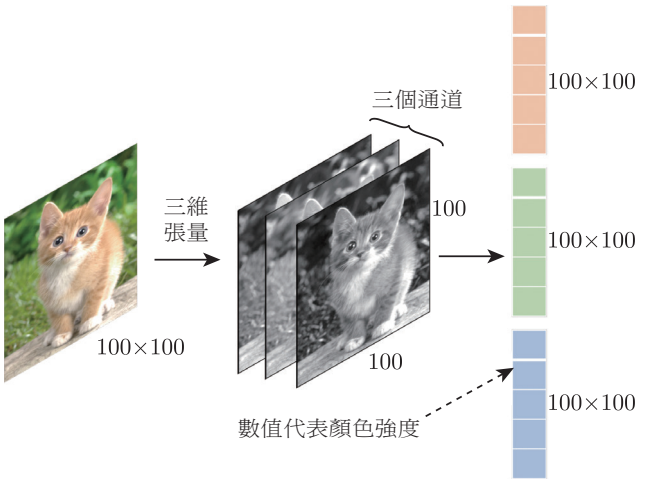
本章從卷積神經網路開始，探討神經網路的架構設計。卷積神經網路是一種非常典型的網路架構，常用於圖片分類等任務。透過卷積神經網路，我們可以知道網路架構如何設計，以及為什麼合理的網路架構可以最佳化神經網路的表現。

所謂圖片分類，就是給機器一幅圖片，由機器去判斷這幅圖片裡面有什麼樣的東西 —— 是貓還是狗、是飛機還是汽車。怎麼把圖片當作模型的輸入呢？對於機器，圖片可以描述為三維張量（張量可以想像成維度大於 2 的矩陣）。一幅圖片就是一個三維的張量，其中一維的大小是圖片的寬，另一維的大小是圖片的高，還有一維的大小是圖片的**通道**（**channel**）數目。

Q 什麼是通道？

A 彩色圖片的每個像素都可以描述為紅色（red）、綠色（green）、藍色（blue）的組合，這 3 種顏色就稱為圖片的 3 個色彩通道。這種顏色描述方式稱為 RGB 色彩模型，常用於在螢幕上顯示顏色。

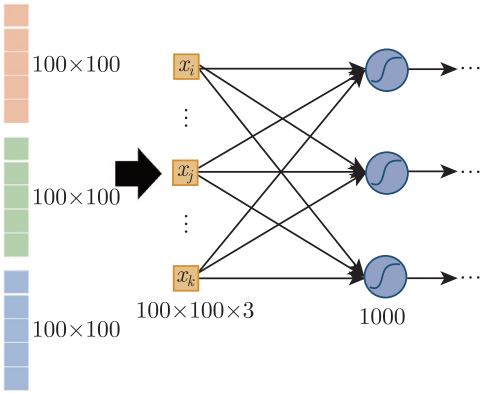
神經網路的輸入往往是向量，因此，在將代表圖片的三維張量輸入神經網路之前，需要先將它「拉直」，如圖 4.1 所示。在這個例子中，張量有 $100\times100\times3$ 個數字，所以一幅圖片由 $100\times100\times3$ 個數字組成，把這些數字排成一排，就是一個巨大的向量。這個向量可以作為神經網路的輸入，而這個向量的每一維裡面的數值則是某個像素在某一通道下的顏色強度。



▲ 圖 4.1 把圖片作為輸入

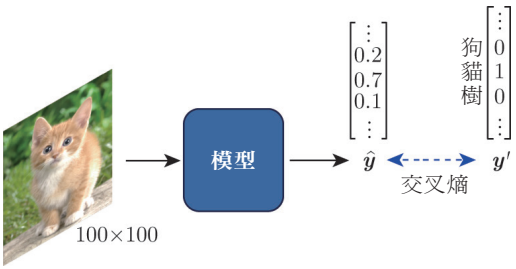
圖片有大有小，而且不是所有圖片的尺寸都是一樣的。常見的處理方式是把所有圖片先調整成相同的尺寸，再輸入圖片的辨識系統。在下面的討論中，預設輸入的圖片尺寸已固定為 100 像素 ×100 像素。

如圖 4.2 所示，如果把向量當作全連接網路的輸入，則輸入的特徵向量（feature vector）的長度就是 $100\times100\times3$。這是一個非常長的向量。由於每個神經元與輸入向量中的每個數值間都需要一個權重，因此當輸入的向量長度是 $100\times100\times3$ 且第 1 層有 1000 個神經元時，第 1 層就需要 $1000\times100\times100\times3 = 3\times10^7$ 個權重，數量巨大。更多的參數為模型帶來了更大的彈性和更強的能力，但也提高了過擬合的風險。模型的彈性越大，就越容易過擬合。為了避免過擬合，在做圖片辨識的時候，考慮到圖片本身的特性，並不一定需要全連接，即不需要每個神經元與輸入向量中的每個數值間都有一個權重。接下來我們針對圖片辨識任務，對圖片本身的特性進行一些觀察。



▲ 圖 4.2 全連接網路

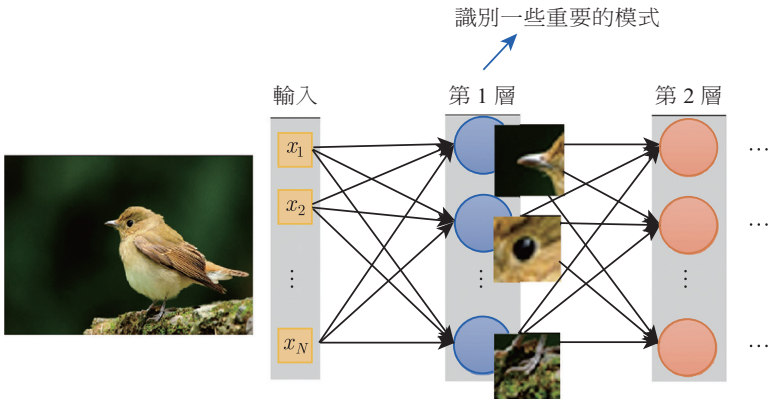
模型的輸出應該是什麼呢？模型的目標是分類，因此可將不同的分類結果表示成不同的獨熱向量 $\boldsymbol{y}'$。在這個獨熱向量裡面，對應類別的值為 1，其餘值為 0。例如，我們規定向量中的某些維度代表狗、貓、樹等分類結果，若分類結果為貓，則貓所對應的維度的數值就是 1，其他東西所對應的維度的數值就是 0，如圖 4.3 所示。獨熱向量 $\boldsymbol{y}'$ 的長度決定了模型可以辨識出多少不同種類的東西。如果獨熱向量 $\boldsymbol{y}'$ 的長度是 5，則代表模型可以辨識出 5 種不同的東西。現在比較強的圖片辨識系統往往可以辨識出 1000 種以上，甚至上萬種不同的東西。如果希望圖片辨識系統可以辨識上萬種東西，則標籤就是維度上萬的獨熱向量。模型的輸出透過 softmax 函式以後，得到 $\hat{\boldsymbol{y}}$。我們希望 $\boldsymbol{y}'$ 和 $\hat{\boldsymbol{y}}$ 的交叉熵越小越好。



▲ 圖 4.3 圖片分類

4.1 觀察 1：檢測模式不需要整幅圖片

假設我們的任務是讓神經網路辨識出圖片中的動物。對於一個圖片辨識的類神經網路裡面的神經元而言，它要做的就是檢測圖片裡面有沒有出現一些特別重要的模式（pattern），這些模式分別代表不同的物體。比如有三個神經元分別看到鳥嘴、眼睛、鳥爪 3 個模式，這代表類神經網路看到了一隻鳥，如圖 4.4 所示。



▲ 圖 4.4 使用神經網路來檢測模式

人們在判斷一種物體的時候，往往也是抓最重要的特徵。看到這些特徵以後，從直覺上就會自認為看到了某種物體。對於機器，也許這是一種有效的判斷圖片中是何物體的方法。但假設用神經元來判斷某種模式是否出現，也許並不需要每個神經元都去看一幅完整的圖片。因為不需要看整幅完整的圖片就能判斷重要的模式（比如鳥嘴、眼睛、鳥爪）是否出現，如圖 4.5 所示，要想知道圖片中有沒有一個鳥嘴，只需要看一個非常小的範圍。這些神經元不需要把整幅圖片當作輸入，而只需要把圖片的一小部分當作輸入，就足以檢測某些特別關鍵的模式是否出現，這是第 1 個觀察。

Chapter 08

生成模型

越來越多的生成式軟體在極大程度上改變了我們的生活。例如，我們可以透過一張照片，讓軟體自動產生一段音樂，或是讓軟體自動產生一段影片。本章具體介紹它們背後的基礎模型 —— **生成模型**（**generative model**）。

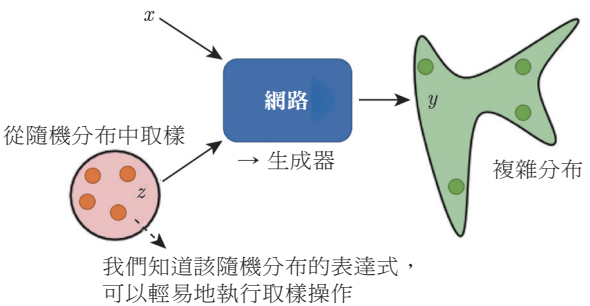
8.1 生成對抗網路

到目前為止，我們學習到的網路在本質上都是一個函式，即提供一個輸入，網路就可以輸出一個結果。此外，前幾章已經介紹了各式各樣的網路，它們可以應對不同類型的輸入和輸出。例如，當輸入是一張圖片時，可以使用卷積神經網路等模型進行處理；當輸入是序列資料時，可以使用以遞迴神經網路架構為基礎的模型進行處理，其中輸出既可以是數值、類別，也可以是一個序列。這些網路已經可以解決我們日常會遇到的大多數問題。

8.1.1 生成器

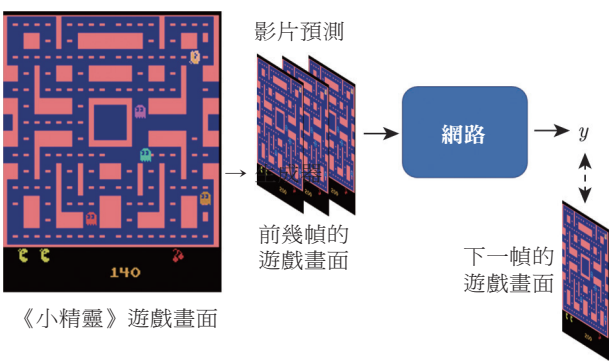
與先前介紹的模型所不同的是，生成模型中的網路會被當作**生成器**（**generator**）使用。具體來說，在輸入時會將一個隨機變數 z 與原始輸入 x 一併輸入模型，這個變數是從隨機分布中取樣得到的。輸入時可以採用向量拼接的方式將 x 和 z 一併輸入，或在 x 和 z 長度一樣時，將它們的和作為輸入。變數 z 的特別之處在於其非固定性，即每一次我們使用網路時，

都會從一個隨機分布中取樣得到一個新的 z。通常，我們對該隨機分布的要求是，它必須足夠簡單，可以較為容易地進行取樣，或者可以直接寫出該隨機分布的函式，如**高斯分布**（**Gaussian distribution**）、**均勻分布**（**uniform distribution**）等。所以每次在輸入 x 的同時，我們都從隨機分布中取樣得到 z，得到最終的輸出 y。隨著取樣得到的 z 的不同，我們得到的輸出 y 也會不一樣。同理，對於網路來說，其輸出也不再固定，而變成一個複雜的分布。我們將這種可以輸出一個複雜分布的網路稱為生成器，如圖 8.1 所示。



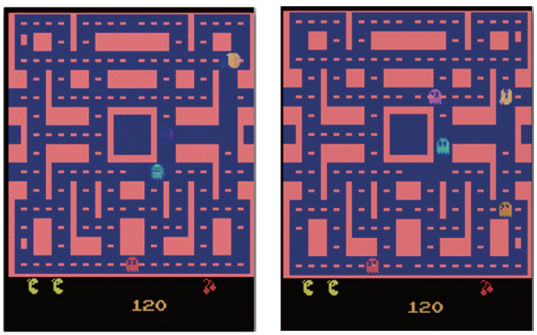
▲ 圖 8.1 生成器示意圖

如何訓練這個生成器呢？思考一下，我們為什麼訓練生成器，又為什麼需要輸出一個分布呢？下面介紹一個影片預測的例子，給模型一個影片短片，讓它預測接下來發生的事情。影片環境是《小精靈》遊戲，預測下一幀的遊戲畫面，如圖 8.2 所示。



▲ 圖 8.2 影片預測 —— 以《小精靈》遊戲為例

為了預測下一幀的遊戲畫面，我們只需要給網路輸入前幾幀的遊戲畫面。而要得到這樣的訓練資料，只需要在玩《小精靈》遊戲的同時進行錄製。訓練網路，讓網路的輸出 y 與真實圖片越接近越好。當然在實作中，為了保證訓練高效率，我們會將每一幀的遊戲畫面分割為很多塊作為輸入，同時分別進行預測。接下來為了簡化，假設網路是一次輸入整個遊戲畫面的。如果使用前幾章介紹的基於監督式學習的訓練方法，則得到的結果可能會十分模糊，遊戲中的角色甚至可能消失或出現殘影，如圖 8.3 所示。



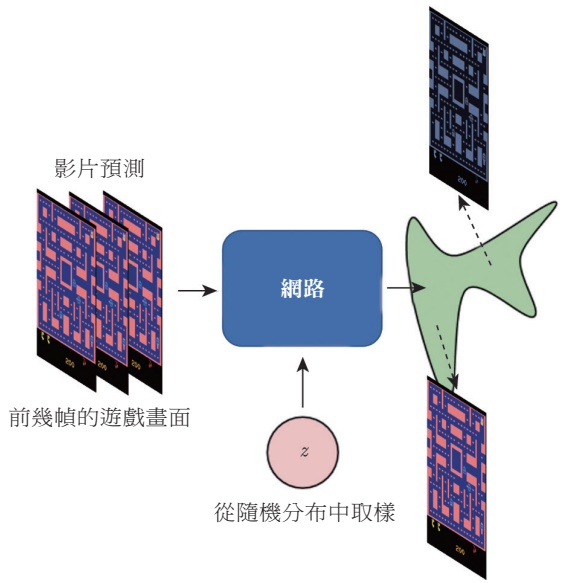
預測值 1　　預測值 2

▲ 圖 8.3 基於監督式學習的《小精靈》遊戲的預測值

造成該問題的原因是，監督式學習中的訓練資料對於同樣的轉角同時儲存了角色向左轉和角色向右轉兩種輸出。在訓練的時候，對於一條向左轉的訓練資料，網路得到的指示就是要學會代表遊戲角色向左轉的輸出。同理，對於一條向右轉的訓練資料，網路得到的指示就是學會代表遊戲角色向右轉的輸出。但實際上這兩種資料可能會被同時訓練，網路「兩面討好」，就會得到一個錯誤的結果 —— 向左轉是對的，向右轉也是對的。

應該如何解決這個問題呢？答案是讓網路有機率地輸出一切可能的結果，或者說輸出一個機率分布，而不是進行原來的單一輸出，如圖 8.4 所示。當給網路一個隨機分布時，網路的輸入會加上一個 z，這時輸出就變成了一個非固定的分布，其中包含了向左轉和向右轉的可能。舉例來說，假設選擇的 z 服從一個二項分布，即只能取 0 或 1 並且兩種可能各占 50%。網路就可以在 z 取樣到 1 的時候就向左轉，取樣到 0 的時候就向右轉，這樣問題就解決了。

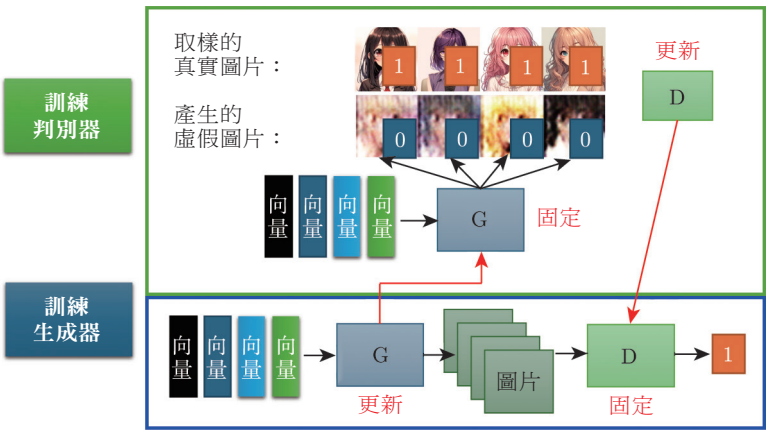
回到生成器的討論中，我們什麼時候需要這類生成模型呢？答案是當我們的任務需要「創造性」的輸出，或者我們想知道一個可以輸出多種可能性的模型，且這些輸出都是對的模型的時候。這可以類比為讓很多人一起處理一個開放式的問題，或是腦力激盪，大家的回答五花八門，但都是正確的。所以也可以理解為生成模型讓模型有了創造力。再舉兩個更具體的例子。對於畫圖，假設畫一個紅眼睛的角色，每個人可能畫出來的或者心中想的都不一樣。對於聊天機器人，它也需要有創造力。比如我們問聊天機器人：「你知道有哪些童話故事嗎？」，聊天機器人會回答《安徒生童話》、《格林童話》等，沒有標準的答案。所以對於生成模型來說，它需要能夠輸出一個分布，或者說多個答案。在生成模型中，非常知名的就是**生成對抗網路**（**Generative Adversarial Network，GAN**）。



▲ 圖 8.4 基於生成模型的《小精靈》遊戲的預測結果

下面我們透過讓機器產生動畫人物的臉部來具體地介紹 GAN。**無限制生成**（**unconditional generation**）不需要原始輸入 x，與無限制生成相對的就是需要原始輸入 x 的**條件型生成**（**conditional generation**）。如圖 8.5 所示，對於無限制的 GAN，它的唯一輸入就是 z，這裡假設 z 為取樣自常態分布的向量。它通常是一個低維的向量，例如 50 維或 100 維。

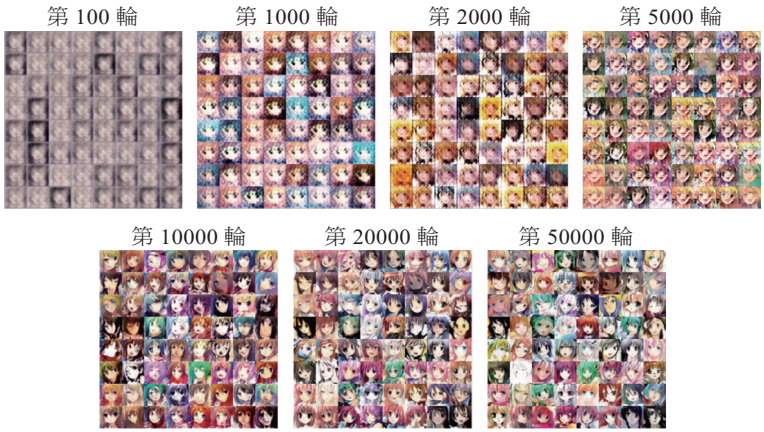
我們總結一下 GAN 演算法的兩個關鍵步驟：固定生成器，訓練判別器；固定判別器，訓練生成器。接下來就是重複以上的訓練，訓練完判別器，就固定判別器，訓練生成器；訓練完生成器，就用生成器產生更多的新圖片，給判別器做訓練用；訓練完判別器，再訓練生成器，就這樣重複下去。當其中一個進行訓練的時候，另一個就固定住，期待它們都可以在自己的目標上達到最佳效果，如圖 8.10 所示。



▲ 圖 8.10 GAN 的完整訓練過程

8.3 GAN 的應用案例

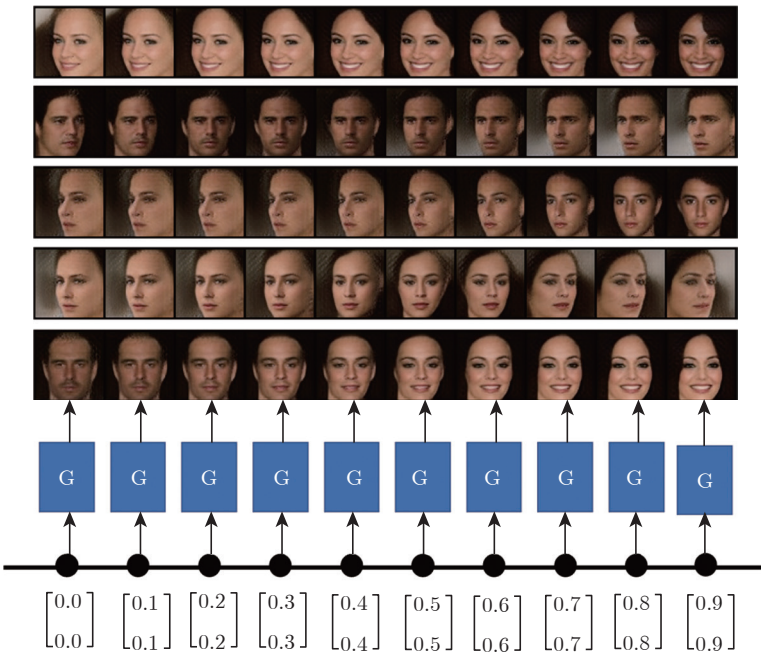
下面介紹 GAN 的一些應用案例。首先介紹 GAN 產生動漫人物人臉的例子，如圖 8.11 所示，這些分別是訓練 100 輪、1000 輪、2000 輪、5000 輪、10000 輪、20000 輪和 50000 輪的結果。可以看到，訓練到 100 輪時，產生的圖片還比較模糊；訓練到 1000 輪時，出現了眼睛；訓練到 2000 輪時，嘴巴產生出來了；訓練到 5000 輪時，已經開始有一點人臉的輪廓了，並且機器學到了動漫人物大眼睛的特徵；訓練到 10000 輪時，外部輪廓已經可以明顯感覺到了，只是還有些模糊；訓練 20000 輪後產生的圖片完全可以以假亂真，訓練 50000 輪後產生的圖片已經十分逼真。



▲ 圖 8.11 GAN 產生動漫人物人臉的視覺化效果

除了產生動漫人物的人臉以外，當然也可以產生真實的人臉，如圖 8.12 所示。產生高解析人臉的技術叫作**漸進式 GAN**（**progressive GAN**），圖 8.12 是由機器產生的人臉。我們可以使用 GAN 產生我們從來沒有看過的人臉，如圖 8.13 所示。舉例來說，先前我們介紹的 GAN 中的生成器，就是輸入一個向量，輸出一張圖片。此外，我們還可以對輸入的向量做內差，在輸出部分，我們會看到兩張圖片之間連續的變化。比如輸入一個向量，透過 GAN 產生一個表情看起來非常嚴肅的男人；同時輸入另一個向量，透過 GAN 產生一個微笑著的女人。輸入介於這兩個向量之間的數值向量，我們就可以看到這個男人逐漸笑了起來。再比如，輸入一個向量，產生一個往左看的人；同時輸入另一個向量，產生一個往右看的人，在這兩個向量之間做內差，機器並不會傻傻地將兩張圖片重疊在一起，而是產生一張正面的人臉。神奇的是，我們在訓練的時候其實並沒有輸入正面的人臉，但機器可以自己學到，只要對這兩張圖片做內差，就可以得到一張正面的人臉。

▲ 圖 8.12 漸進式 GAN 產生人臉的效果



▲ 圖 8.13 GAN 產生連續人臉的過程（圖中的 G 代表生成器）

不過，如果我們不加約束，GAN 就會產生一些很奇怪的圖片。比如，使用 BigGAN 演算法 [1] 會產生一個左右不對稱的玻璃杯子，甚至產生一個網球狗，如圖 8.14 所示。

▲ 圖 8.14 GAN 產生不符合常理的視覺化例子

19.3 ChatGPT 背後的關鍵技術 —— 預訓練

在澄清了一些對 ChatGPT 的常見誤解以後，接下來介紹 ChatGPT 是怎麼被訓練出來的。ChatGPT 背後的關鍵技術就是預訓練。預訓練其實有各式各樣的稱呼，有時候又叫自監督學習，預訓練得到的模型則叫基石模型。關於 ChatGPT 這個名字的由來，名字中的 Chat 代表聊天，G 指生成，P 指預訓練，T 指 Transformer。

我們先來看看一般的機器學習是什麼樣子。想像我們要訓練一個翻譯系統，要把英文翻譯成中文，如果要找一個函式，它可以把英文翻譯成中文，一般的機器學習方法是這樣的：首先收集大量成對的中英文對照例句，告訴機器如果輸入「I eat an apple」，輸出就應該是 「我吃蘋果」；如果輸入「You eat an orange」，輸出就應該是「你吃柳丁」。要讓機器學會將英文翻譯為中文，首先得有人去收集大量的中英文成對的例句。這種需要成對的東西來學習的技術，叫作監督式學習。有了這些成對的資料以後，機器就會自動找出一個函式，這個函式包含了一些翻譯的規則，比如機器知道輸入是「I」時輸出就是「我」，輸入是「You」時輸出就是「你」。接下來我們給機器一個句子，期待機器可以得到正確的翻譯結果。

如果把監督式學習的概念套用到 ChatGPT 上，結果應該是這樣的：首先要找很多的人類老師，讓他們設定好 ChatGPT 的輸入和輸出的關係，比如告訴 ChatGPT，當有人問世界第一高峰是哪個時，就回答珠穆朗瑪峰。總之，要找大量的人給 ChatGPT 正確的輸入和輸出，有了這些正確的輸入和輸出以後，就可以讓機器自動地學習一個函式，完成如下功能：當輸入「世界第一高峰是哪個」的時候，輸出「珠」的機率最大，接下來輸出「穆」的機率也比較大，以此類推。有了這些訓練資料以後，機器就可以找到一個函式，這個函式能夠滿足我們的要求 —— 給定一個輸入的時候，它的輸出和人類老師給的輸出十分接近。但顯然僅僅這樣做是不夠的，因為如果機器只根據人類老師的教導找出函式，它的能力將是非常有限的，因為人類老師可以提供的成對資料是非常有限的。舉例來說，假設資料裡沒有任何一句話提到青海湖，當有人問機器中國第

一大湖是哪個的時候，它不可能回答青海湖，因為它很難憑空產生這個專有名詞。人類老師可以提供給機器的資訊是很少的，所以機器如果只靠人類老師提供的資料來訓練，它的知識就會非常少，很多問題也就沒有辦法回答。

ChatGPT 的成功其實依賴於另外一個技術，這個技術可以製造出大量的資料。網路上的每一段文字都可以拿來教機器做文字接龍，比如從網路上隨便爬取到一個句子 —— 世界第一高峰是珠穆朗瑪峰，我們把前半段當作機器的輸入，後半段不管是不是正確答案，都告訴機器後半段就是正確答案。接下來就讓機器去學習一個函式，這個函式應該做到當輸入「世界第一高峰是」的時候，輸出「珠」的機率越大越好。如此一來，網路上的每一個句子都可以拿來教機器做文字接龍。事實上，ChatGPT 的前身 GPT 所做的事情就是單純地從網路上的大量資料中學習做文字接龍。

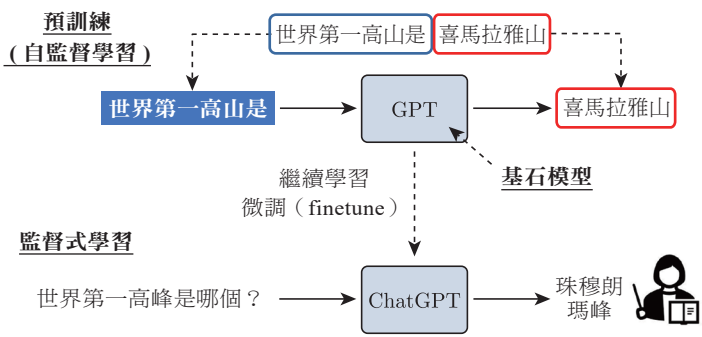
早在 ChatGPT 之前就有一系列的 GPT 模型。GPT-1 在 2018 年的時候就已經出現了，只是那個時候沒有受到很大的注意。GPT-1 其實是一個很小的模型，只有 1 億 1700 萬個參數。它的訓練資料集也不大，只有 1 GB 的訓練資料。但是一年之後，OpenAI 公開了 GPT-2，GPT-2 的參數量是 GPT-1 的近 10 倍，訓練資料則是前者的約 40 倍。有了這麼大的模型，還有了這麼多的資料去訓練機器，根據網路上的資料做文字接龍，會有什麼樣的效果呢？當年最讓大家津津樂道的一個結果是，你可以和 GPT-2 說一段話，接下來它就開始胡說。舉個例子，和 GPT-2 說有一群科學家發現了獨角獸，接下來 GPT-2 就開始亂編這些獨角獸的資訊。這個能力在今天看來沒什麼，AI 就應該做到這樣的事情，但在 2019 年，學術界對此非常震驚，大家覺得它的回答有模有樣。事實上，GPT-2 也是可以回答問題，甚至可以輸出文章的摘要。所以其實早在 GPT-2 的時候，機器就已經有回答問題的能力了。

在今天，包含 15 億個參數的模型在大家眼中可能不是一個特別大的模型，但是在 2019 年，人們十分震驚於世界上居然有如此巨大的模型。從結果來看，就算只是從大量的網路資料中學習做文字接龍，GPT-2 也已經有了回答問題的能力。2020 年發行的 GPT-3 是 GPT-2 的約 100 倍大，訓練資料約 570 GB。文字量差不多是《哈利‧波特》全集的 30 萬倍。

Q GPT-3.5 是什麼呢？

A 事實上，沒有任何一篇文章明確地告訴我們 GPT-3.5 指的是哪一個模型，目前 OpenAI 官方的說法是，只要是在 GPT-3 的基礎上做一些微調得到的模型都叫 GPT-3.5，這個名稱並不特指某個模型。

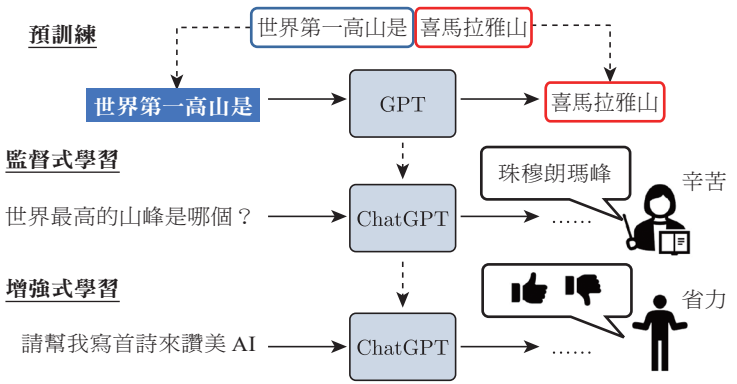
GPT-3 可以做什麼樣的事情呢？GPT-3 剛公開時引起了非常大的轟動，那時候的人們認為 GPT-3 實在太大了，它甚至會寫程式碼。它為什麼可以做到這件事情呢？因為 GitHub 上有很多程式碼，裡面還有程式碼註解，所以 GPT 在做文字接龍的時候，看到這些程式碼註解，也就能夠將程式碼產生出來。所以 GPT-3 可以寫程式碼，好像也不是特別讓人震驚的事情。不過，GPT-3 看起來是有非常大的能力上限的，雖然能力很強，但它給的答案不一定是我們想要的。所以怎麼再強化 GPT-3 的能力呢？下一步就需要人工介入了，到 GPT-3 為止，訓練是不需要人監督的，但是從 GPT-3 到 ChatGPT，就需要人類老師的介入了，所以 ChatGPT 其實是監督式學習以後的結果。在進行監督式學習之前，透過大量網路資料學習的這個過程，就稱為預訓練，如圖 19.4 所示。這個繼續學習的過程則叫微調，預訓練有時候又叫自監督學習。機器在學習時需要成對的資料，但這些成對的資料不是人類老師提供的，而是用一些方法產生的，這就叫自監督學習。



▲ 圖 19.4 ChatGPT 中的預訓練

預訓練對 ChatGPT 的效能又有多大的幫助呢？ChatGPT 是多語言互動的，不管用中文、英文還是日文問它，它都會給出答案。很多人可能覺得它背後有一個比較好的翻譯引擎，因為 OpenAI 並沒有針對 GPT 的多語言能力進行公開說明，所以我們也不能排除它使用翻譯引擎的可能性。但我們猜測它應該不需要用到翻譯引擎，因為很有可能只要教 ChatGPT 幾種語言的問答，它就可以自動學會其他語言的問答。舉一個多語言模型 Multi-BERT 的例子，這是在 ChatGPT 出現之前非常熱門的一個自監督學習的語言模型，它在 104 種語言上做過預訓練。Multi-BERT 有一個神奇的技能，假設讓它進行閱讀能力測驗。我們只用英文微調，接下來用中文進行測驗，很神奇的是，Multi-BERT 可以回答中文的問題，而且裡面沒有包含翻譯等流程。

另外，我們知道 ChatGPT 中不僅使用了監督式學習，還使用了增強式學習。ChatGPT 使用的是增強式學習中的 PPO 演算法。如圖 19.5 所示，在增強式學習中，不是直接給機器答案，而是告訴機器現有的答案好還是不好。增強式學習的好處是，相較於監督式學習（監督式學習中的人類老師是比較辛苦的），人類老師可以偷懶，只需指導大的方向。此外，增強式學習更適合用在人類自己都不知道答案的時候。舉例來說，如果要寫首詩來讚美 AI，其實很多人當場是寫不出來的，但機器可以寫出來。對人類來說，判斷機器寫的這首詩是不是一首好詩則簡單許多。



▲ 圖 19.5 ChatGPT 中的增強式學習