前言

深度學習是一項強大的新技術，我們認為，每一門學科都應該利用它。由於各種領域的專家比較可能發現它的新用法，所以我們希望有更多來自各種背景的人參與其中，開始使用深度學習。

這就是 Jeremy 創辦 fast.ai，透過免費的線上課程和軟體來讓深度學習更容易使用的原因。Sylvain 是 Hugging Face 的研究工程師，他之前是 fast.ai 的研究科學家，也曾經擔任數學和計算機科學教師，為即將進入法國精英大學的學生做好準備。我們合寫這本書是為了讓盡量多的人掌握深度學習。

本書對象

如果你完全沒有接觸過深度學習與機器學習，我們最歡迎這種人了，我們只期望你知道怎麼寫程式，最好是使用 Python。

沒有經驗？沒問題！

沒有寫過任何程式也沒關係！我們會在前三章讓主管、產品經理等人清楚地了解他們必須知道且最重要的事項。當你在書中看到程式時，可以先試著閱讀它們，用直覺來了解它們在做什麼，我們也會逐行解釋它們。與其了解語法的細節，不如對於事情的來龍去脈有高層次的認識。

有自信的深度學習實踐者也可以在這裡學到很多東西，本書將告訴你如何做出世界級的成果，包括來自最新研究的技術。你將會看到，你不需要完成高階的數學訓練或經年累月的學習，只需要具備一些常識和毅力就夠了。

你需要知道的知識

如上所述，閱讀本書的先決條件只有知道如何寫程式（只要一年的經驗就夠了），最好使用 Python，而且至少讀過高中數學。就算你已經忘了大部分的東西也沒關係，我們會視情況稍微複習它們。Khan Academy 也有很棒的免費線上課程可以提供協助（*https://www.khanacademy.org*）。

我們的意思不是深度學習不會用到高中以上的數學，而是我們會在你需要特定的基礎知識時教導它們（或是告訴你可以學習的資源）。

本書會從大局開始討論，逐漸深入表層之下的區域，所以有時你可能要先將它擱著，學習其他的主題（程式的寫法或一些數學）。這種做法沒有任何問題，我們也希望你用這種方式來閱讀這本書。你可以先瀏覽它，並且只在需要時，尋找其他的資源。

請注意，如果你使用 Kindle 或其他的電子書閱讀器，可能要按兩下圖像才能看到完整尺寸的版本。

線上資源

本書的所有範例程式都以 Jupyter notebook 的形式放在網路上（別擔心，第 1 章會告訴你什麼是 Jupyter notebook），它是本書的互動式版本，你可以在上面實際執行程式碼，並且試驗它。詳情見本書的網站（*https://book.fast.ai*），網站裡面也有各種工具的最新設定資訊，以及額外的紅利章節。

你將學到什麼

看完這本書後，你會知道：

- 如何訓練模型，在以下的領域取得頂尖的結果
  - 電腦視覺，包括圖像分類（例如按品種來對寵物照片進行分類）和圖像定位及偵測（例如找出圖像中的動物）
  - 自然語言處理（NLP），包括文件分類（例如影評情緒分析）和語言建模
  - 內含分類資料、連續資料與混合資料的表格式資料（例如銷售預測），包括時間序列
  - 協同過濾（例如電影推薦）

- 如何將模型轉換成 web 應用程式
- 深度學習模型如何運作、為何那樣運作，以及如何使用那些知識來改善模型的準確度、速度與可靠度
- 在實務上非常重要且最新的深度學習技術
- 如何閱讀深度學習研究論文
- 如何從零開始實作深度學習演算法
- 如何思考工作的道德含義，以確保你可以讓世界更美好，而且工作成果不會被濫用，傷害他人

雖然你可以在目錄看到完整的清單，但是為了稍微滿足你的好奇心，以下是我們將討論的一些技術（如果你完全不知道其中的一些技術，不用擔心，你很快就會學到它們了）：

- 仿射函數與非線性
- 參數與觸發輸出
- 隨機初始化與遷移學習
- SGD、動力、Adam 和其他優化法
- 摺積
- 批次標準化
- dropout
- 資料擴增
- 權重衰減
- ResNet 與 DenseNet 架構
- 圖像分類與回歸
- embedding
- 遞迴神經網路（RNN）
- 分割
- U-Net
- 還有更多內容！

各章問題

每一章的結尾都有一些問題，它是讓你複習你在每一章學到的東西的好地方，因為（我們希望！）在每一章結束時，你將能夠回答裡面的所有問題。其實，有一位校閱者（謝謝您，Fred！）說，他喜歡先把問題看一遍，再閱讀那一章，這樣他就知道該注意哪些地方了。

※ 本書採單色印刷，彩色圖片可至本書的網站查看（*https://book.fast.ai*）。

操作步驟：

1.於網頁左側欄選擇「Notebook Servers」下方的「Colab」。

2.於右側的「opening a chapter of the book」欄位下方將顯示各章內容的連結。

序

深度學習已經在短時間內變成一種用途廣泛的技術了，它可以解決電腦視覺、機器人、醫療保健、物理、生物學等領域的問題，並且自動解決問題。深度學習有一項深得人心的特性在於它相對簡單，現在已經有人做出強大的深度學習軟體，讓一般人都可以快速且輕鬆地上手了，你可以在幾週之內了解並熟悉這些技術的基本知識。

這個特性開啟了一個充滿創造力的世界，你可以用它來處理已取得資料的問題，然後看著電腦奇妙地為你解決它。但是，你也會發現自己離一個巨大的障礙越來越近，雖然你已經做出一個深度學習模型了，但是它的效果不如你的預期，此時就是進入下一個階段，尋找並閱讀最先進的深度學習研究的時候了。

然而，深度學習蘊含大量的知識，在它背後隱藏著多達三十年的理論、技術和工具，在閱讀這些研究時，你會發現作者用非常複雜的方式來解釋簡單的事情，科學家們在這些論文裡面使用異國文字和數學符號，你也無法找到教科書或部落格文章以易懂的方法介紹必要的背景知識，工程師和程式員都假設你已經知道 GPU 如何工作，並且了解一些鮮為人知的工具了。

此時，你希望有良師益友可以請益，他曾經經歷你的處境，了解工具和數學，可以教導你最棒的研究、最先進的技術、高階的工程方法，並且以有趣的方式把它變得簡單。我在十年前剛進入機器學習領域時也經歷過你的情況，多年來，我費盡心思地了解包含一些數學知識的論文，雖然我有很多優秀的導師給我很大的幫助，但是我仍然花了好幾年才適應機器學習和深度學習，這促使我和其他創作者一起設計出 PyTorch 這個讓深度學習更平易近人的軟體框架。

Jeremy Howard 和 Sylvain Gugger 也經歷過你的情況，他們也想要學習和使用 ML，儘管他們從未接受過任何正式的 ML 科學或工程培訓。Jeremy 與 Sylvain 和我一樣經歷多年的緩慢學習才成為專家和領導者，但是他們和我不同的是，他們無私地投入大量的精力來確保別人不會重蹈他們的覆轍。他們打造了一個很棒的課程，稱為 fast.ai，讓只具備基本程式知識的人就可以輕鬆地使用先進的深度學習技術，這個課程已經讓成千上萬位熱切的畢業生成為卓越的實踐者了。

Jeremy 和 Sylvain 孜孜不倦地完成這本書，在書中，他們創造了一段深度學習的神奇旅程，用簡單的文字來介紹每一個概念，為你帶來頂尖的深度學習技術和最先進的研究，並且讓它們易於了解。

在這本 500 多頁的有趣旅程中，你將了解電腦視覺的最新進展，深入研究自然語言處理，並學習一些基本的數學知識。這個過程不僅充滿樂趣，也可以協助你將想法付諸實踐。你可以將 fast.ai 社群視為一個大家庭，裡面有成千上萬位從業者，每一位和你一樣的人都可以討論和構思大大小小的解決方案，無論問題是什麼。

很高興你發現這本書，希望它能夠鼓勵你好好利用深度學習，不管問題的本質是什麼。

— *Soumith Chintala*
PyTorch 的共同創辦人

第一章

你的深度學習旅程

哈囉！感謝你允許我們加入你的深度學習旅程，無論你已經走了多遠！在這一章，我們將告訴你這本書的內容，介紹深度學習背後的重要概念，並且在不同的任務中，訓練我們的第一個模型。就算你沒有技術或數學背景也沒關係（有這些背景也一樣！），我們寫這本書就是為了讓盡可能多的人了解深度學習。

深度學習是讓大家使用的

很多人認為若要透過深度學習來獲得很棒的結果，就要使用各種難以取得的東西，但是正如你將在這本書中看到的，那些人錯了。表 1-1 是使用世界級的深度學習時**完全不需要**的東西。

表 1-1　使用深度學習時不需要的東西

迷思（不需要）	真相
許多數學	高中數學就夠了。
許多資料	我們看過有人用不到 50 筆資料項目做出破紀錄的成果。
許多昂貴的電腦	你可以免費使用最先進的工作所需的資源。

深度學習是一項使用多層的神經網路來提取和轉換資料的電腦技術，其用例包含人類語音辨識和動物照片分類。神經網路的每一層都會從之前的神經層接收輸入，並且逐步改進它們。這些神經層都是用演算法來訓練的，演算法可將它們的誤差最小化，及改善它們的準確度，讓網路學會如何執行特定的任務，下一節會詳細介紹訓練演算法。

深度學習具備強大的能力、靈活性和簡單性，這就是我們認為各種學科都要使用它的原因，包括社會科學、物理科學、藝術、醫學、金融、科學研究等。以一個人為例，儘管 Jeremy 沒有醫學背景，但他也創辦了 Enlitic，一家使用深度學習演算法來診斷疾病的公司，這家公司在成立幾個月之後，就宣布它的演算法辨識惡性腫瘤的準確度比放射科醫生還要高（*https://oreil.ly/aTwdE*）。

以下是活用深度學習的領域的上千項任務之中，獲得最佳成果的幾項任務：

自然語言處理（*NLP*）

回答問題、語音辨識、產生文件摘要、分類文件、在文件中找出名稱、日期等，以及搜尋談到某個概念的文章

電腦視覺

衛星和無人機照片判讀（例如用於災害復原）、人臉辨識、產生圖像的標題、閱讀交通號誌、協助自駕車定位行人和車輛

醫學

在放射照片中發現異常，包括 CT、MRI 與 X 光照、計算病理幻燈片裡面的特徵數量、以超音波測量特徵、診斷糖尿病性視網膜病變

生物學

蛋白質摺疊、蛋白質分類、各種基因學任務，例如腫瘤正常排序，和分類臨床可行基因突變、細胞分類、分析蛋白質 / 蛋白質互動

圖像生成

將圖像改成彩色、提升圖像解析度、移除圖像雜訊、將圖像轉換成著名藝術家的美術風格

推薦系統

web 搜尋、產品推薦、首頁布局

玩遊戲

西洋棋、圍棋、大部分的 Atari 遊戲，和許多即時策略遊戲

機器人

處理難以定位（例如透明的、反光的、缺乏紋理的）或難以發現的物體

其他的應用

金融與物流預測、將文字轉成語音，及其他…

值得注意的是，雖然深度學習有這麼多的應用，但幾乎所有的深度學習都是用一種創新的模型來建構的：神經網路。

但事實上，神經網路不是全新的東西，為了在這個領域中具備更廣闊的視野，我們必須先了解一些歷史。

神經網路：簡史

在 1943 年，神經生理學家 Warren McCulloch 與邏輯學家 Walter Pitts 合作開發一個人造神經元的數學模型，他們在論文「A Logical Calculus of the Ideas Immanent in Nervous Activity」裡面說：

> 由於神經活動具備「全有或全無」的特性，神經事件以及它們之間的關係可以用命題（propositional）邏輯來處理。我們發現，每一個網路的行為都可以用這些項（term）來處理。

McCulloch 與 Pitts 發現，我們可以用簡單的加法與閾值來代表真實的神經元，做成簡化的模型，如圖 1-1 所示。Pitts 自學成才，在 12 歲時就收到劍橋大學的錄取通知書，他原本可以和偉大的 Bertrand Russell 一起學習，但是他拒絕這個邀請，事實上，他一生都沒有接受任何高級學位或權威職位的邀請，他的著作大都是在無家可歸的時刻完成的。儘管他沒有得到官方認可的職位，而且與世界越來越隔絕，但他與 McCulloch 的合作影響深遠，後來，名為 Frank Rosenblatt 的心理學家承接了他的工作。

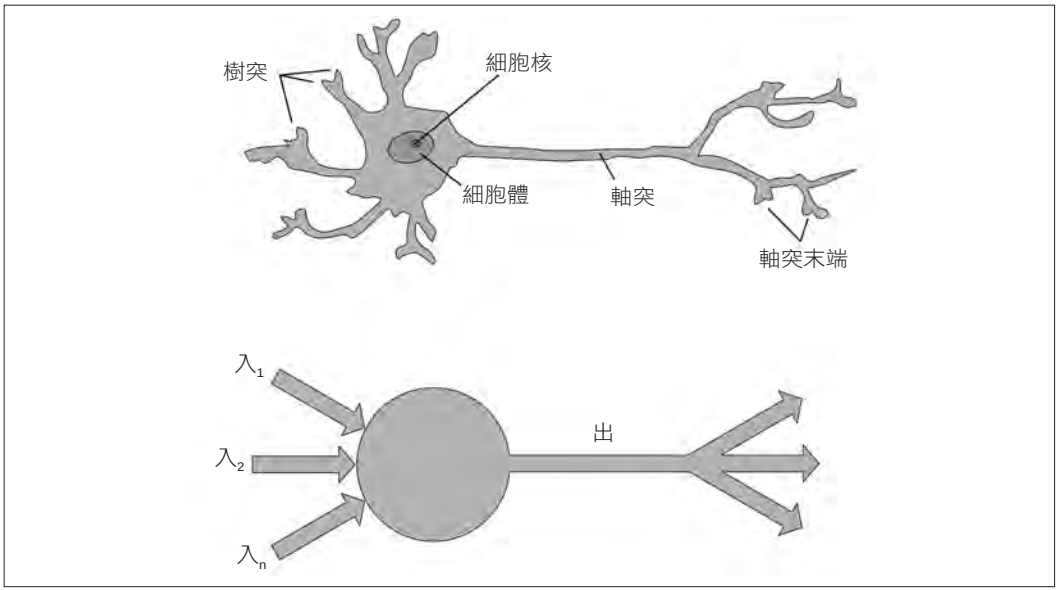

圖 1-1　自然的與人工的神經元

Rosenblatt 進一步開發人造神經元，賦予它學習的能力，更重要的是，他致力於製造第一個使用這些原理的設備，Mark I Perceptron。Rosenblatt 在「The Design of an Intelligent Automaton」裡面如此描述這項作品：「我們正在見證這種機器的誕生，它是完全不需要依靠人類的訓練或控制，就能感知、識別與認識周圍環境的機器。」他做出來的感知器能夠成功地認出簡單的形狀。

麻省理工學院的 Marvin Minsky 教授（他與 Rosenblatt 讀同一所高中，且分數比他少一分！）與 Seymour Papert 合著了一本關於 Rosenblatt 的發明的書，稱為 *Perceptrons*（MIT Press），他們指出，這些設備的單一神經層無法學習一些簡單但很重要的數學函數（例如 XOR），在同一本書裡，他們也展示了這種限制可以用多層的設備來解決。不幸的是，在這些見解中，只有第一個得到廣泛的認同。於是，在接下來的 20 年裡，全世界的學術界幾乎完全放棄神經網路。

或許 David Rumelhart、James McClelland 與 PDP Research Group 在 1986 年透過 MIT Press 出版的 *Parallel Distributed Processing*（PDP）是過去 50 年來，最關鍵的神經網路研究成果。他們在第 1 章提出與 Rosenblatt 相似的希望：

> 人類之所以比今日的電腦更聰明，是因為人腦採用一種基本的計算架構，這種架構更適合處理人類擅長的「自然資訊處理任務」的核心層面…我們將介紹一種模擬這種認知過程的計算框架，它看起來比其他框架更接近人腦的計算風格。

PDP 的假設是：傳統的電腦程式的工作方法與人腦非常不同，這應該是電腦程式很不擅長處理（在當時）人腦可以輕鬆勝任的工作（例如認出照片裡面的物體）的原因。作者聲稱，PDP 的方法「比其他的框架更接近」人腦的工作方式，因此它處理這種任務的效果更好。

事實上，PDP 提出的方法很像今日的神經網路所使用的方法。該書定義平行分散式處理需要以下條件：

- 一組**處理單元**
- 一個**觸發狀態**
- 各個單元都有一個**輸出函數**
- 單元之間的**連接模式**
- 透過網路的連結來傳遞活動模式的**傳播規則**
- 將傳入一個單元的輸入與該單元當時的狀態結合，來讓該單元產生一個輸出的**觸發規則**
- 根據經驗修改連接模式的**學習規則**
- 運作系統的**環境**

你將會在本書看到，現代的神經網路滿足以上的每一個需求。

在 1980 年代，大部分的模型都是用雙層的神經元做成的，它可以避免 Minsky 與 Papert 提出來的問題（這是他們為了使用上述的框架而採取的「單元之間的連接模式」）。事實上，在 80 年代與 90 年代，神經網路被廣泛地用在實際專案中。然而，對於理論議題的誤解再次阻礙了這個領域的發展，雖然在理論上，只要加入一層額外的神經元就可以讓神經網路近似任何一種數學函數了，但是在實務上，這種網路通常過於龐大且過於緩慢，無法實際使用。

儘管研究人員早在 30 年前就指出使用更多層神經元才能實際獲得良好的表現，但是直到最近十年，這個原則才被廣泛認可及應用。由於多層結構的採用，也因為電腦硬體的改進、有更多資料可用，還有經過調整的演算法讓我們可以更快速且更輕鬆地訓練神經網路，現代的神經網路終於展現它們的潛力了。我們終於擁有 Rosenblatt 所承諾的：「完全不需要依靠人類的訓練或控制，就能感知、識別與認識周圍環境的機器。」

這就是你會在本書學習建構的東西，但是在那之前，因為我們要花很多時間相處，所以讓我們先認識一下彼此…

我們是誰？

我們是 Sylvain 與 Jeremy，這趟旅程的導遊，但願你會發現我們很適合這個職位。

Jeremy 已經使用及教導機器學習將近 30 年了，他在 25 年前開始使用神經網路，在這段時間裡，他帶領很多以機器學習為核心的公司和專案，包括創辦首家專門開發深度學習與醫學的公司 Enlitic，以及在全球最大的機器學習社群 Kaggle 擔任總裁和首席科學家，他與 Rachel Thomas 博士一起創辦 fast.ai，本書的基礎就是這個機構設計的課程。

有時我們會用專欄直接和你對話，例如這段 Jeremy 所說的：

Jeremy 說

大家好，我是 Jeremy ！你應該會覺得很有趣的是，我沒有受過任何正規科技教育。我有哲學學士學位，但成績不太好，我比較喜歡進行實際專案而不是理論研究，所以我在大學期間進入一家名為 McKinsey & Company 的管理顧問公司全職工作。如果你寧願親自動手做事，也不想花好幾年的時間學習抽象的概念，你就可以了解我的想法！沒有太多數學或正式科技背景的人（也就是跟我一樣的人）可以從我的專欄看到適合他們的資訊。

另一方面，Sylvain 非常了解正規的技術教育。他已經寫了 10 本數學教科書，涵蓋整個高級法國數學課程！

Sylvain 說

與 Jeremy 不同的是，我沒有經年累月的程式和機器學習演算法的撰寫經驗，而是透過觀看 Jeremy 的 fast.ai 課程影片進入機器學習的世界。因此，如果你還沒有打開過終端機，並且在命令列上寫過命令，你就知道我的出身！有數學或正式技術背景，但缺乏實際程式設計經驗的人（也就是跟我一樣的人）可以在我的專欄裡面看到適合他們的資訊。

目前已經有成千上萬位來自世界各地、各行各業的學生上過 fast.ai 課程了。Sylvain 是 Jeremy 在課程裡看過的學生中，令他印象最深刻的一位，這導致 Sylvain 加入 fast.ai，與 Jeremy 一起編寫 fastai 軟體程式庫。

以上總總意味著你可以從兩個不同的世界得到最大的好處：有人比任何人都知道這套軟體，因為這套軟體是他們寫出來的，其中有一位數學專家，以及一位程式設計與機器學習專家；以及既能理解身為數學局外人的感受，又能理解程式設計和機器學習邊緣人的感受的人。

有在看運動比賽的人都知道，如果評論員有兩位，你也需要加入第三位「特別評論員」。我們的特別評論員是 Alexis Gallagher。Alexis 有多彩多姿的背景，他做過數理生物學研究員、編劇、即興藝人、McKinsey 顧問（跟 Jeremy 一樣！）、Swift 程式員與 CTO。

Alexis 說

我想，是時候學習這門 AI 課程了！畢竟，我嘗試過許多其他東西…但是我其實沒有製作機器學習模型的背景。而且…這會不會很難？我會和你一樣用這本書來學習。我的專欄有我在這個旅程中認為有幫助的學習小撇步，希望它們也可以幫助你。

如何學習深度學習？

曾經撰寫 *Making Learning Whole*（Jossey-Bass）的哈佛大學教授 David Perkins 對教學有很多看法，他的基本理念是教導全局，也就是說，如果你要教棒球，你就要先帶學生去看棒球賽，或是先讓他們下場打球，而不是教他們如何從頭開始纏線做出棒球，不是教他們拋物線的物理原理，也不是教他們當球被球棒擊中時的摩擦係數。

哥倫比亞大學數學博士、前布朗大學教授、K–12 數學教師 Paul Lockhart 在一篇有影響力的文章「A Mathematician's Lament」（*https://oreil.ly/yNimZ*）裡面創造一個惡夢般的虛構世界，在裡面，教音樂和藝術的方法與教數學一樣，小孩必須先花十幾年的時間來掌握樂譜和理論，並且上好幾堂課，學會將樂譜轉換成不同的音調，才可以開始聽音樂和演奏音樂。在藝術課裡面，學生要學習顏色和塗色器，但是在大學之前不准實際畫畫。聽起來很荒謬？這就是教數學的方法——我們要求學生花好幾年的時間死記硬背，學習枯燥、與現實脫節的**基本知識**，我們聲稱這種做法可以帶來回報，最終卻讓大多數人放棄這門學科。

不幸的是，許多深度學習的教學資源在課程的一開始也這樣做，他們要求學生遵循 Hessian 的定義，以及損失函數的 Taylor 近似理論，卻不提供實際運作的範例程式。我們不是在抨擊微積分，我們熱愛微積分，Sylvain 甚至在大學教過它，但我們認為微積分不適合當成學習深度學習的起點！

在深度學習裡，如果你想要修正模型來讓它有更好的效果，此時就是學習相關理論的時機，但是在這之前，你必須先有一個模型。我們會用實際的例子來教導幾乎所有東西，我們將在建構這些範例的同時，介紹越來越深的知識，並且告訴你如何讓專案越來越好。也就是說，你將會一步一步地學習所需的理論基礎，藉著這種方式，你將明白它為什麼重要，以及它如何運作。

所以，我們承諾，在這本書，我們將遵守這些原則：

教導全局

我們會先告訴你如何使用一個完整的、可動作的、可用的、先進的深度學習網路，運用簡單的、表達力強大的工具來解決真正的問題。接下來，我們會逐漸深入介紹這些工具是如何製造的，以及製造這些工具的工具是如何製造的，以此類推…

一定會用範例來教導

我們會提供可以讓你直覺理解的背景和目的，而不是先秀出代數符號公式。

盡量簡化

我們已經花了好幾年的時間建構各種工具與教學方法來簡化上述的複雜主題了。

排除障礙

到目前為止，深度學習還是一場排他性的遊戲，我們想要拆除圍牆，讓所有人都可以加入。

深度學習最困難的部分是手工製作的工作（artisanal）：你怎麼知道資料夠不夠多？它的格式是否正確？有沒有正確地訓練模型？如果沒有，該怎麼處理？這就是我們認為應該「做中學」的原因。與基本的資料科學技術一樣，你只能藉著累積實際經驗來提升深度學習技術，在理論上花太多時間會適得其反，學習的關鍵是直接開始寫程式，並試著解決問題，理論可以等你有了背景和動機時再來學習。

在旅途中，有時你會遇到困難，覺得被卡住了，此時不要放棄！回到尚未卡住的地方，從那裡開始仔細閱讀，找到開始不明白的地方，寫程式做實驗，並且用 Google 搜尋教學來處理你遇到的問題——通常你會找到從不同角度出發，並且很有幫助的資源。此外，在第一次閱讀時有一些不了解的地方（尤其是程式碼）是可以預期的，也是很正常的事情。有時依序了解內容並逐步前進並不容易，或許你可以從旅途的其他地方知道更多背景，掌握大局之後，困難即可迎刃而解。所以如果你被卡在某一節，可以先試著不理會它，繼續看下去，並且記下它，之後再回來研究。

切記，在深度學習中取得成功不需要任何學術背景。在研究界和業界中，許多重要的突破性成果都是沒有博士學位的人做出來的，例如，這篇被引用 5,000 次以上，在過去十年來最具影響力的論文「Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks」（*https://oreil.ly/JV6rL*）是 Alec Radford 在念大學時寫出來的。即使是努力克服艱鉅挑戰打造自駕車的特斯拉 CEO Elon Musk 也說過（*https://oreil.ly/nQCmO*）：

> 博士學位絕對不是必須的，最重要的是深刻理解 AI，以及做出真正有用的 NN（後者真的很不容易），就算你只有高中學歷也不用在意。

但是，若要成功，你一定在個人專案裡運用本書的知識，而且一定要堅持不懈。

你的專案與心態

無論你想要根據植物葉子的照片來鑑定植物是否患病、自動產生針織圖案、用 X 光診斷結核病，還是確定浣熊什麼時候會使用你的貓門，我們都會盡快讓你先用深度學習來處理你的問題（使用為了解決其他的問題而訓練好的模型），再逐漸探討更多細節。在下一章，你會在 30 分鐘之內，學會如何使用深度學習，以頂尖的準確度解決你自己的問題！（如果你渴望馬上開始寫程式，那就直接跳到那裡吧！）坊間有人散播邪惡的訊息，說你需要 Google 規模的計算資源和資料量才可以進行深度學習，假的！

那麼，哪一些任務是優秀的測試範例？你可以訓練模型來分辨畢卡索和莫內的畫作，或分辨你的女兒的照片，而不是兒子的。把重心放在你感興趣和熱愛的事物上很有幫助，在一開始，先為自己設定四五個小專案，而不是努力解決一個大問題的效果比較好，因為我們很容易陷入困境，太早有太大的野心往往適得其反。當你掌握基本技術之後，你就可以開始把目標放在真正會讓你感到驕傲的事情上了！

Jeremy 說

幾乎所有問題都可以用深度學習來解決。舉例來說，我創辦的第一家公司叫做 FastMail，它在 1999 年成立時，提供增強型 email 服務（現在仍然如此）。在 2002 年，我用原始形式的深度學習（單層神經網路）來協助分類 email，以及防止顧客收到垃圾郵件。

擅長使用深度學習的人都有愛玩與好奇的性格。已故的物理學家理察·費曼是我們認為應該很擅長深度學習的人物之一：他之所以能夠理解次原子粒子運動，是因為他對板子在空中旋轉時如何擺動很有興趣。

接著，我們來看看你將學到什麼，首先是軟體。

軟體：PyTorch、fastai 與 Jupyter（以及為何它不重要）

我們曾經使用幾十種程式包和許多程式語言來完成上百個機器學習專案，在 fast.ai 裡，我們用當今常見的主流深度學習和機器學習程式包來撰寫課程。當 PyTorch 在 2017 年問世之後，我們花了超過 1,000 個小時來測試它，並且決定在未來的課程、軟體開發和研究中使用它，從那時起，PyTorch 變成世界上發展速度最快的深度學習程式庫，已經被頂尖會議的多數研究論文採用，這通常意味著它會被業界採用，因為那些論文最終會被用在商業產品和服務上。我們發現 PyTorch 是最靈活且最富表達力的深度學習程式庫，而且它沒有為了提升簡單性而犧牲速度，而是兩者兼備。

PyTorch 最適合當成低階的基礎程式庫，來提供高階功能的基本操作。fastai 程式庫就是一種在 PyTorch 之上加入高階功能的流行程式庫。它也特別適合這本書，因為它獨家提供深層的軟體架構（這個分層式 API 甚至有一篇討論它的學術論文（*https://oreil.ly/Uo3GR*））。本書會在深入探討深度學習的基本知識時，進入越來越深的 fastai 層。這本書使用第 2 版的 fastai 程式庫，它是從零開始改寫的，提供許多獨特的功能。

然而，要學習哪一種軟體其實不重要，因為從一個程式庫換成另一個只需要花費幾天的學習時間。真正重要的是正確地學習深度學習的基礎和技術，我們的程式會盡量清楚地傳達你需要知道的概念。我們會在教導高階的概念時使用高階的 fastai 程式，在教導低階的概念時使用低階的 PyTorch，甚至使用純 Python 程式。

雖然最近新的深度學習程式庫正如雨後春筍般出現，但你必須為未來幾個月和幾年的快速變化做好準備。隨著更多人進入這個領域，他們也會帶來更多技能和想法，並嘗試更多東西。你應該假設，你今天學到的任何一種程式庫和軟體在一到兩年之內就會落伍。你只要看一下 web 程式領域的程式庫和技術堆疊變化的頻率就可以知道這一點，何況 web 領域比深度學習成熟許多，而且成長速度慢得多。我們堅信，學習的重點在於了解底層的技術、如何應用它們，還有在新工具和技術出現時，如何快速上手。

在本書的最後，你將了解 fastai 的幾乎所有內部程式（還有大部分的 PyTorch），因為在每一章裡，我們都會往下深掘一層，來告訴你在建構與訓練模型時發生什麼事。這意味著你會學到現代深度學習所使用的、最重要的最佳實踐法，不僅知道如何使用它們，也知道它們如何實際運作與實作。如果你想要在別的框架裡面實施這些做法，你將擁有必要的知識。

對學習深度學習而言，最重要的事情是寫程式與做實驗，所以你必須有一個可以寫程式來做實驗的好平台。Jupyter（*https://jupyter.org*）是最流行的程式實驗平台，本書將使用它。我們將告訴你如何使用 Jupyter 來訓練模型和做實驗，以及檢查每一個資料預先處理和模型開發流程。Jupyter 之所以成為 Python 最流行的資料科學工具是有原因的，它有強大的功能、靈活，而且容易使用，相信你也會喜歡它！

我們來看一下如何實際使用它，並且訓練我們的第一個模型吧！

你的第一個模型

如前所述，我們會先教你怎麼做事，再解釋為什麼要做那些事情。我們將按照這種由上而下的做法，先訓練一個照片分類模型，以 100% 的準確度辨識狗與貓。為了訓練這個模型與進行實驗，你必須先做一些設定，別擔心，它沒有看起來那麼難。

Sylvain 說

雖然設定的部分看起來有點令人卻步，絕不要跳過它，尤其是當你沒有或很少終端機或命令列等工具的使用經驗時。它們大部分都不是必要的，而且你將發現，你只要用常用的網頁瀏覽器就可以設定最簡單的伺服器了。務必跟著本書一起做實驗來學習。

取得 GPU 深度學習伺服器

為了做這本書談到的幾乎所有事情，你的電腦必須具備 NVIDIA GPU（遺憾的是，其他品牌的 GPU 並未完整支援主要的深度學習程式庫），但是我們不建議你買一台這種電腦，事實上，即使你有這種電腦，我們也不建議你現在就使用它！設置電腦很花時間與精力，現在你應該把所有精力放在深度學習上，因此，我們建議你租用已裝好必要設備的電腦，它們的租金在使用的情況下最低每小時只需要 $0.25，有些選項甚至是免費的。

術語：圖形處理單元（*GPU*）

也稱為顯示卡，它是在電腦內部的特殊處理器，可同時處理成千上萬個任務，專門為了在電腦上顯示遊戲的 3D 環境而設計。這些基本任務很像神經網路所處理的任務，因此 GPU 運行神經網路比一般的 CPU 快好幾百倍。所有現代電腦都有 GPU，但搭載能夠處理深度學習的正確 GPU 的電腦很少。

因為服務供應商會來來去去，與這本書搭配的最佳 GPU 伺服器會隨著時間而改變。本書的網站（*https://book.fast.ai*）有一個推薦清單，現在就去按照說明，連接一個 GPU 深度學習伺服器。別擔心，大部分的平台都只需要 2 分鐘的設定時間，而且許多平台都不需要付費即可使用，甚至不需要綁定信用卡。

Alexis 說

我的拙見是，接受這個建議！如果你喜歡電腦，或許你會試著設定自己的電腦，小心！雖然你可以這樣做，但是它出乎意外地複雜、難懂而且令人分心。這就是為什麼本書的書名不是 *Ubuntu 系統管理*、*NVIDIA 驅動程式安裝*、*apt-get*、*conda*、*pip* 及 *Jupyter Notebook 設置雜談*，它們應該各自用一本書來說明。我曾經在工作時設計與部署機器學習生產環境基礎設施，這的確很有成就感，但與建立模型無關，就像維修飛機跟駕駛飛機是兩回事一樣。

在網站上介紹的每一種選項都有教學，在完成教學之後，你會看到圖 1-2 的畫面。

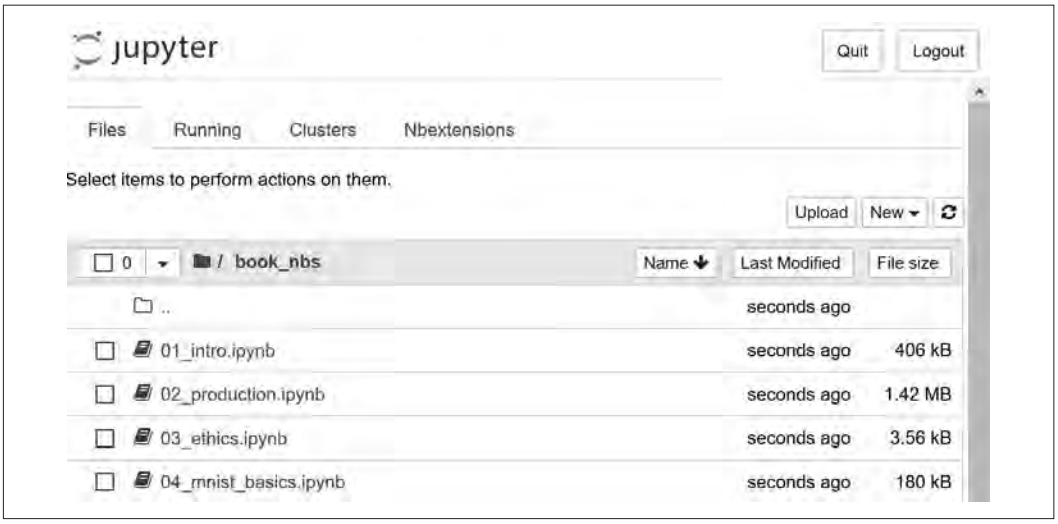

圖 1-2　Jupyter Notebook 的初始畫面

現在你可以執行第一個 Jupyter notebook 了！

術語：*Jupyter Notebook*

這套軟體可讓你將格式化的文字、程式碼、圖像、影片等全部放入一個互動式文件中。由於 Jupyter 在學術界與業界受到廣泛的使用，而且已經造成廣大的影響，所以獲得軟體界最高榮譽，ACM Software System Award。Jupyter notebook 是資料科學家開發深度學習模型和與之互動時最常使用的軟體。

執行你的第一個 notebook

notebook 是按照本書的介紹順序，按章編號的。因此，你看到的第一個 notebook 就是現在要使用的 notebook，你將使用這個 notebook 來訓練一個可以辨識狗與貓照片的模型。為此，你需要下載一個包含狗與貓照片的資料組，並且使用它來**訓練模型**。

※ 本書採單色印刷，彩色圖片可至本書的網站查看（*https://book.fast.ai*）。

資料組只是一堆資料——它可能是圖像、email、金融指標、聲音，或任何其他東西。坊間有許多免費的資料組很適合用來訓練模型，其中許多資料組是學術界為了協助進行研究而創造的，也有很多資料組是為了舉辦競賽而製作的（坊間有一些讓資料科學家比較誰做出來的模型最準確的競賽！），有些是其他程序（例如財務歸檔）的副產品。

full 與 *clean notebook*

網站上面有兩個資料夾，裡面有不同的 notebook 版本。*full* 資料夾裡面有用來製作你正在看的這本書的 notebook，包含所有的文字與輸出。*clean* 版本有一些標題與程式 cell（儲存格），但沒有任何輸出與文章。建議你在看完每一節之後，闔上書本，跑一下 clean notebook，看看能不能在執行每一個 cell 之前知道它會顯示什麼結果，並且試著回想一下那段程式想要展示什麼。

直接點選 notebook 即可將它打開，它長得像圖 1-3（注意，在不同的平台上可能有一些細節不同，你可以忽略這些差異）。

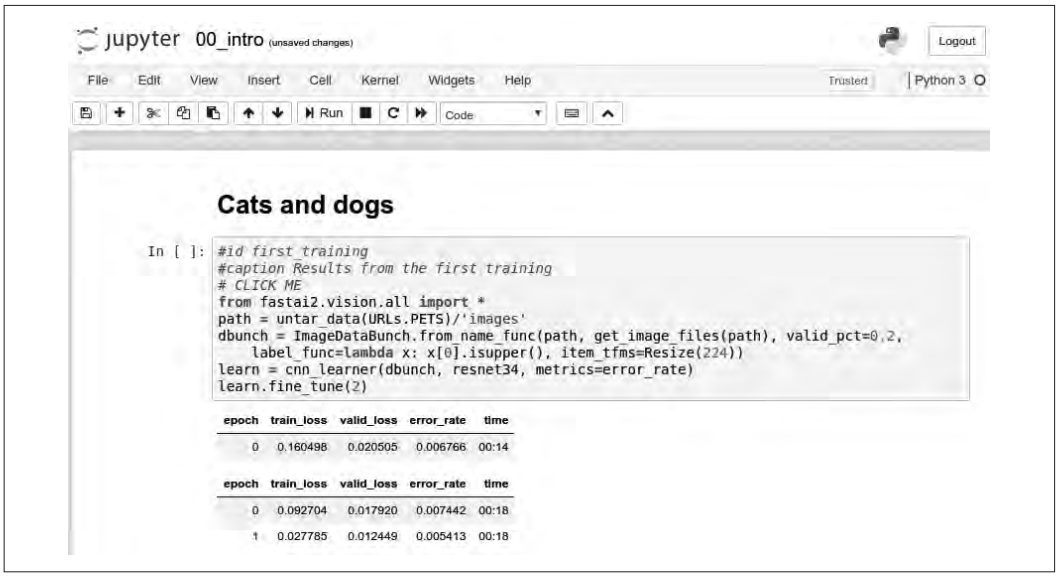

圖 1-3　Jupyter notebook

notebook 裡面有 *cell*，cell 有兩種：

- 容納格式化的文字、圖像等的 cell。它們使用 *Markdown* 格式，你很快就會學到這種格式。
- 容納可執行的程式碼的 cell，它的輸出就在它的下面（可能是一般的文字、表格、圖像、動畫、聲音，甚至是互動式 app）。

Jupyter notebook 可能處於兩種模式之一：編輯模式或命令模式。在編輯模式中，你可以像平常一樣，用鍵盤在 cell 裡面輸入字母。但是在命令模式中，你不會看到任何閃爍的游標，而且每一個鍵盤的按鍵都有特殊的功能。

在繼續看下去之前，按下鍵盤的 Escape 鍵來切換至命令模式（如果你已經在命令模式了，這不會怎樣，因此儘管按下它）。你可以按下 H 來顯示所有功能，按下 Escape 來移除這個協助畫面。注意，與大多數的程式不同的是，在命令模式下，你不需要按下 Control、Alt 之類的按鍵來輸入命令，只要按下所需的字母按鍵即可。

你可以按下 C 來複製一個 cell（你必須先選擇那個 cell，讓它的周圍有個框線，如果它還沒有被選取，按下它一次），然後按下 V 來貼上它的副本。

按下開頭為「# CLICK ME」的 cell 來選擇它。該行的第一個字元代表接下來的東西是 Python 的注釋，所以當 cell 執行時，它會被忽略。或許你不相信，該 cell 的其餘部分是一個完整的系統，可以建立和訓練一個辨識貓與狗的先進模型。我們來訓練它！你只要按下鍵盤的 Shift-Enter，或按下工具列的 Play 按鈕就可以訓練它了。以下這些事情會在你等待的幾分鐘之內發生：

1. 從 fast.ai 資料組收集區，將一個稱為 Oxford-IIIT Pet Dataset（*https://oreil.ly/c_4Bv*）的資料組下載到你使用的 GPU 伺服器上，該資料組裡面有 7,349 張 37 個品種的貓與狗照片。
2. 從網際網路下載一個已經用 130 萬張照片**預先訓練的獲獎模型**。
3. 用最新的遷移學習技術來**微調**這個預先訓練的模型（pretrained model，以下簡稱預訓模型），建立一個為了辨識狗與貓而特別製作的模型。

前兩個步驟只需要在你的 GPU 伺服器上執行一次，當你再次執行這個 cell 時，它會使用已經下載的資料組與模型，不會再次下載它們。我們來看一下這個 cell 的內容與結果（表 1-2）：

```
# CLICK ME
from fastai.vision.all import *
path = untar_data(URLs.PETS)/'images'
```

```
def is_cat(x): return x[0].isupper()
dls = ImageDataLoaders.from_name_func(
    path, get_image_files(path), valid_pct=0.2, seed=42,
    label_func=is_cat, item_tfms=Resize(224))

learn = cnn_learner(dls, resnet34, metrics=error_rate)
learn.fine_tune(1)
```

表 1-2　第一次訓練的結果

epoch	train_loss	valid_loss	error_rate	time
0	0.169390	0.021388	0.005413	00:14

epoch	train_loss	valid_loss	error_rate	time
0	0.058748	0.009240	0.002706	00:19

你得到的結果應該不會與這裡的完全相同。訓練模型的過程有很多小的隨機變化來源。然而，這個例子所顯示的錯誤率通常小於 0.02。

訓練期

下載預訓模型與資料組可能要花好幾分鐘，取決於你的網路速度。執行 `fine_tune` 可能會花 1 分鐘左右。本書的模型通常要用幾分鐘來訓練，你的模型也是如此，所以最好看看你能不能活用這段時間，例如，在訓練模型時，繼續閱讀下一節，或打開另一個 notebook，用它來進行一些程式實驗。

本書是用 Jupyter notebook 寫成的

我們用 Jupyter notebook 來寫這本書，所以書中的每一張圖表、表格與計算幾乎都有程式碼，可讓你自行重現它們。這就是為什麼本書的一些程式碼後面幾乎都立刻有一個表格、一張圖片或一些文字。你可以在本書的網站（*https://book.fast.ai*）找到所有程式碼，並且試著自行執行與修改每一個範例。

你剛才已經看到本書的 cell 如何輸出一個表格了。以下是輸出文字的 cell：

```
1+1
```

```
2
```

Jupyter 一定會印出或顯示最後一行的結果（如果有的話），例如，這個 cell 會輸出一張圖像：

```
img = PILImage.create('images/chapter1_cat_example.jpg')
img.to_thumb(192)
```

那麼，我們怎麼知道這個模型好不好？你可以在表格的最後一欄看到 *error rate*（**錯誤率**），它是照片被錯誤辨識的比例。我們將錯誤率當成 metric（這是衡量模型品質的數據，選擇的標準是直覺而且容易理解）。如你所見，這個模型近乎完美，儘管訓練時間只有幾秒鐘（不含一次性的下載資料組與預訓模型的時間）。事實上，你剛才實現的準確度已經比 10 年前任何人做出來的結果好超級多了！

最後，我們來確認這個模型確實可以工作。找一張狗或貓的照片，如果你手上沒有，只要搜尋 Google Images 並且從那裡下載一張照片即可，接下來執行定義 `uploader` 的 cell，它會輸出一個按鈕，你可以按下它並選擇想要分類的照片：

```
uploader = widgets.FileUpload()
uploader
```

現在你可以將已上傳的檔案傳給模型。務必使用只有一隻狗或一隻貓的清楚照片，不要上傳線圖、卡通或類似的圖片。這個 notebook 將告訴你它認為那張圖是狗還是貓，以及它有多大的信心。希望你的模型有很棒的表現：

```
img = PILImage.create(uploader.data[0])
is_cat,_,probs = learn.predict(img)
print(f"Is this a cat?: {is_cat}.")
print(f"Probability it's a cat: {probs[1].item():.6f}")
```

```
Is this a cat?: True.
Probability it's a cat: 0.999986
```