推薦序

Foreword

無庸置疑，科技領域對我們日常生活的影響可說是與日俱增。這些變化之快，如同它們從來沒變過一樣，並且早已充斥在我們的身邊 —— 在智慧型手機中、車子、智慧音箱與各種我們用來提升效率、福祉與連結性的事物之中。機器學習是現今時代中最具劃時代意義的科技之一。產業、學界與工程社群不斷地去理解、促進並探索這項驚人科技的能耐，並對於各個產業開啟了更多潛在可能性與全新應用。

我是 Arm 公司的機器學習產品經理。在這個角色上，我身處於機器學習（Machine Learning, ML）革命的核心，而這正發生在智慧型手機、汽車產業、遊戲、AR/VR 與其他各種裝置上。我已預見在不久的將來，所有的電子裝置都會具備某種 ML 功能 —— 從世界最大最快的超級電腦，一路到最小的低功耗微控制器。身處 ML 領域讓我得以認識科技界最聰明也最耀眼的人們 —— 他們勇於挑戰傳統產業存在已久的“正統”、提出各種艱澀問題，並藉由運用 ML 技術來開啟嶄新的價值。

我第一次碰到 Gian Marco 時，我連“ML”都還不太知道怎麼說，但當時他早已是業界老兵了。我對於他專業知識的深度與廣度感到驚訝不已，當然還有他在處理各種困難問題的驚人能耐。與 Arm 團隊共事，他讓 **Arm Compute Library（ACL）**成為了 Arm 平台上功能最強大的機器學習函式庫。ACL 的全面成功可說是打遍天下無敵手，且已被部署在全球數以百萬計的裝置上了 —— 包含伺服器、旗艦級智慧型手機與智慧烤箱等等。

當 Gian 告訴我，他正在寫一本關於 ML 的書的時候，我馬上聯想到的是「哪個部分？」ML 生態系實在是太多元了，需要考量許多截然不同的技術、平台與框架。同時，由於他對於 ML 各方面的淵博學識，我知道他

就是這項任務的絕佳人選 . 再者，Gian Marco 在敘事風格上明確且邏輯清晰，有其個人的獨特魅力。

Gian Marco 所寫的這本書用一系列的實務案例，帶領讀者揭開了 TinyML 世界的神秘面紗。每個範例都是以清晰且風格一貫的專案來呈現，並提供了簡易的逐步教學。從最根本的原理開始，作者逐一解釋了每個專案會用到的電子電路與軟體技術的重要基礎。本書接著談到了會用到的平台與技術，然後才介紹 ML —— 我們運用這項技術來開發、訓練神經網路，並部署在目標裝置上。這本書真的是名副其實的「一應俱全」。每個專案都比前一個更具挑戰性，並巧妙搭配了現有與發展中的技術。你所學到的不只是「如何做」，還會理解「為什麼要這麼做」。論到各種邊緣裝置，本書確實對於 ML 這項技術提供了極其宏觀全面的視野。

ML 不斷地讓科技在各方面上都陷入了「混亂」，並已成為軟體開發者的必備技能。本書透過了現成可用的平價技術來幫助你快速上手。不論你是 ML 新手或已具備了一定的經驗值，本書中的各個專案提供了知識面的堅固基石，也同時也為日後的自我發展與各項實驗保留了足夠的空間。不論是把本書視為課本或是參考工具書，你都能為日後開發各種 ML 應用打好紮實的基礎。本書能讓你的團隊眼界更寬廣、讓產品效率更好、效能更高，甚至還能催生出全新的功能。

—Ronan Naughton

ARM 機器學習資深產品經理

CHAPTER 1

TinyML 入門

終於，我們即將邁入**微型機器學習（TinyML）**的世界。

我們會先介紹這個新興領域，並介紹應用**機器學習（ML）**在低功率**微控制器**上的發展機會與挑戰。本章主要將介紹 TinyML 的基本要素、電力消耗與微控制器，這些是讓 TinyML 有別於應用在雲端、桌上型電腦和智慧手機上這類傳統 ML 的關鍵。尤其是微控制器程式設計部分，對於沒有太多關於嵌入式程式設計經驗的讀者來說相當重要。

介紹完 TinyML 的基本架構後，我們將建立一個簡易 LED 應用程式的開發環境，同時也代表著 TinyML 實作之旅正式展開。

相較於之後的章節，本章關於理論架構的討論會比較多，可以幫助你熟悉這項快速發展的技術之概念與常用術語。

本章主題如下：

- TinyML 簡介
- 深度學習概要
- 認識功率與能量的差異

- 微控制器的程式設計
- 認識 Arduino Nano 33 BLE Sense 與 Raspberry Pi Pico
- 設定 Arduino Web Editor、TensorFlow 和 Edge Impulse
- 於 Arduino Nano 和 Raspberry Pi Pico 上執行草稿碼

技術需求

本章所有實作範例所需項目如下：

- Arduino Nano 33 BLE Sense 開發板，1 片
- Raspberry Pi Pico 開發板，1 片
- Micro-USB 傳輸線，1 條
- 安裝 Ubuntu 18.04+ 或 Windows 10 x86-x64 的筆記型或 PC

TinyML 簡介

本書所有的專案都將針對**微型機器學習**，也就是 **TinyML**，提出實際的解決方案。本節將介紹什麼是 TinyML，以及它所帶來的無窮潛力。

◉ 什麼是 TinyML ？

TinyML 是機器學習與嵌入式系統中的一種技術，讓智慧型應用程式可以運作於極低功率的裝置上。這種裝置的記憶體和運算能力通常都十分有限，但可以透過感測器感知周圍的實際環境並根據 ML 演算法做出的決定採取行動。

在 TinyML 中，ML 和部署平台不只是兩個獨立的實體，還需要充分了解彼此。事實上，如果沒有將目標裝置的特性納入考慮就直接設計 ML 架構的話，要部署出有效且正常運作的 TinyML 應用程式便不是那麼容易。

另一方面，如果不了解所涉及的軟體演算法，也想要設計出高能源效率的處理器來擴充這些裝置的機器學習功能也是不可能的。

本書的 TinyML 目標裝置為各種微控制器，接下來將說明為何做此選擇。

◉ 為什麼要在微控制器上使用 ML ？

選擇微控制器的首要原因是它們被廣泛地應用在各種領域裡，像是汽車、消費性電子產品、廚房家電、醫療保健和電信等。如今微控制器早已無所不在，隱身於日常生活中的各種電子設備裡。

隨著**物聯網（IoT）**的興起，微控制器的市場也跟著蓬勃發展。市場研究公司 IDC[1] 於 2018 年發表的一份報告指出，全球微控制器的銷售量達 281 億個，預計將在 2023 年成長到 382 億個 [2]。同一年智慧型手機和個人電腦的銷售量分別是 15 億支和 6720 萬台，可見這個數字是多麼驚人。因此，TinyML 象徵著 IoT 裝置向前邁出了重要的一步，大幅增加了能夠在區域網路執行 ML 作業之微型聯網物件的數量。

選擇微控制器的第二個理由是因為價格便宜，開發簡易且功能足以執行複雜的**深度學習（DL）**演算法。

但話說回來，為什麼不能將計算放在性能更好的雲端上呢？也就是說，為什麼需要在區域網路執行 ML ？

◉ 為什麼要在區域網路執行 ML ？

主要原因有三個－等待時間、耗電量和隱私性：

- **縮短延遲時間**：從雲端收發資料並非那麼即時，可能會影響到一些需要在特定時間內完成反應的應用程式。

1 https://www.idc.com

2 http://www.arm.com/blogs/blueprint/tinyML

- **降低耗電量**：即使是藍牙等低功率的通訊協定，從雲端接發資料也較為耗電。

  下圖為 Arduino Nano 33 BLE Sense 開發板板載元件的功耗分析，也是本書將使用的兩個微控制器開發板之一：

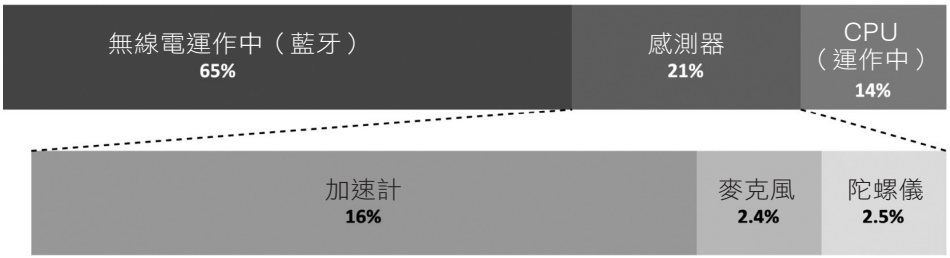

圖 1-1 Arduino Nano 33 BLE Sense 開發板功耗分析

  從以上功耗分析可以看出，CPU 運算比藍牙通訊來得省電（14% 比 65%），因此最好是多依賴計算並盡量減少傳輸以降低電力快速耗盡的風險。一般來說，嵌入式裝置中最耗電的元件就是無線電了。

- **隱私性**：本地端 ML 代表用戶隱私可以受到保護，避免敏感訊息外洩。

現在我們已經知道在微型裝置上使用 ML 的好處了，那麼，將 ML 帶到邊緣裝置上的機會與挑戰是什麼呢？

◉ TinyML 的發展機會與挑戰

TinyML 非常適合應用在任何無法輕易使用正規電源，而應用程式又必須盡可能地透過電池長時間運作的裝置上。

仔細想想其實不難發現，日常生活中已經充斥著應用了 ML 的電池供電裝置。例如，智慧手錶和運動手環等穿戴裝置，可以辨別人類活動以追蹤使用者的健康目標或偵測出跌倒等緊急情況。

這些日常用品為適用於所有用途的 TinyML 應用程式，因為它們可由電池供電，且需要藉由裝置中的 ML 來為感測器所獲取的資料賦予更多意義。

然而，電池供電的解決方案可不僅止於穿戴裝置。一些情況下我們可能會需要透過裝置來監控環境，例如，在森林中部署應用 ML 的電池供電裝置來偵測火災，並防止火勢的蔓延。

TinyML 有著無限可能，剛才簡單介紹的幾個例子只是一小部分。

然而，發展機會多也意味著需要面對一些嚴峻的挑戰。挑戰來自於運算能力，因為裝置多半受限於記憶體容量和處理速度。我們必須在容量僅有幾 KB 的系統上運作，而且有時候處理器還沒有浮點運算加速器。

另一方面，部署環境也可能不太友善。像是灰塵、極端氣候條件等環境因素都可能妨礙並影響應用程式的運作。

接下來的章節將介紹一些 TinyML 常見的部署環境。

◉ TinyML 的部署環境

TinyML 應用程式可以在**集中式系統**或**分散式系統**上運作。

集中式系統的應用程式不一定需要和其他裝置溝通。

關鍵字檢測是一個常見的例子。如今我們時常會透過語音與智慧型手機、相機、無人機和廚房家電溝通互動。用來喚醒智慧管家的神奇指令如 *OK Google*、*Alexa* 等便是 ML 模組在區域網路中持續運作的最佳範例。應用程式需要在不向雲端發送資料的情況下於低功率系統上運作才能維持有效性與即時性，並盡可能地不消耗電力。

通常集中式 TinyML 應用程式的目的在於觸發更多更耗電的功能，而本身不需要將任何資料送進雲端的特性也使程式可以保有隱私。

分散式系統中的裝置（即**節點**或**感測器節點**）會在區域網路中執行 ML，但也會與附近的裝置或主機溝通以實現共同的目標，如下圖：

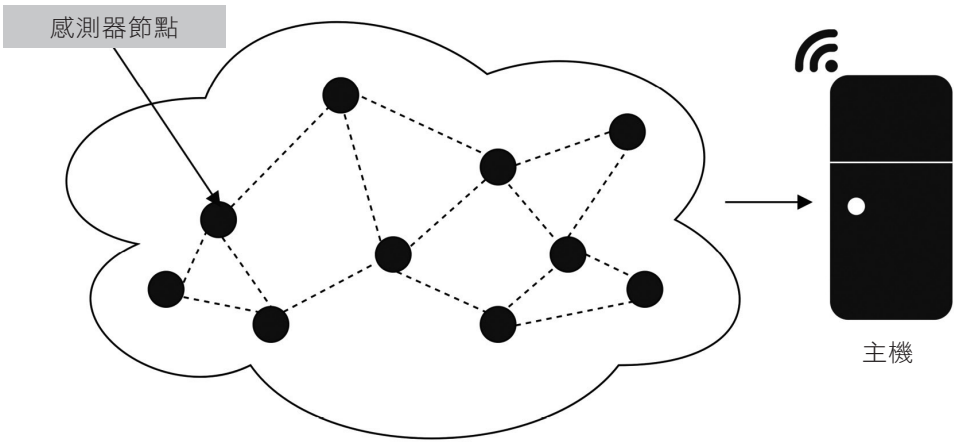

圖 1-2 無線感測網路

Note

因為節點是網路的一部分，且通常會經由無線技術通訊，因此也被稱為**無線感測網路（WSN）**。

儘管此情境會與傳輸資料的功耗影響形成對比，裝置會需要互相合作以建立關於作業環境具有意義且精確的認知。對一些需要全面性地理解量值的擴散狀況的應用程式來說，知道特定節點的溫度、濕度、土壤含水量以及其他物理量可能沒有太大的意義。

假設有一個可以提高農耕效率的應用程式。WSN 有助於決定農田的哪些區域需要更多或更少水，進而讓灌溉更有效率與自動化。當然啦，高效率的通訊協議對網路的壽命來說很關鍵，而 TinyML 在實現此一目標上發揮了作用。由於發送原始資料太過耗電，ML 可以先執行部分運算以減少需要傳輸的資料量和頻率。

TinyML 提供了無限可能，而 **tinyML 基金會**是找出這個快速發展的 ML 領域搭配嵌入式系統所蘊藏的無窮潛力的絕佳去處。

◉ tinyML 基金會

tinyML 基金會 [3] 為支持並連繫 TinyML 世界的非營利專業組織。

在 Arm、Edge Impulse、Google 和 Qualcomm 等多家公司的支持下，tinyML 基金會在全球各地發展出一個多元豐富的社群（如美國、英國、德國、義大利、奈及利亞、印度、日本、澳洲、智利和新加坡），涵蓋了硬體、軟體、系統工程師、科學家、設計師、產品經理與商業人士等各方好手。

基金會持續在網路與實體活動中推廣不同的免費計畫，以吸引各領域的專家和新手參與以鼓勵知識共享與交流，並透過 TinyML 建立一個更健康永續的世界。

Tips

歡迎在不同國家 / 地區的 Meetup 小組清單 [4] 免費加入離你最近的小組 [5]，並隨時了解最新的 TinyML 技術與即將舉辦的活動。

介紹完 TinyML 後，要來深入探索它的組成要素了。接下來將分析讓裝置能夠做出智慧決策的關鍵：深度學習 (DL)。

深度學習概要

ML 是微型裝置能夠做出智慧決策的關鍵。這些軟體演算法高度依賴正確的資料以學習以經驗為基礎的模式與行動。俗話說，資料就是 ML 的一切，因為它決定了應用程式的成敗。

3 www.tinyml.org

4 https://www.meetup.com

5 https://www.meetup.com/en-AU/pro/TinyML/

本書把深度學習視為機器學習的一個特殊的分支，可針對原始影像、文字或聲音等不同類型資料進行複雜的分類任務。這些演算法具有最先進的準確率，並且在一些分類問題上的表現也比人類優秀。這項技術讓聲控虛擬助理、臉部辨識系統和自動駕駛等許許多多的概念得以實現。

深度學習架構和演算法的完整討論超出了本書的範疇，但本節將彙整一些可以幫助你理解本書其他內容的一些要點。

◉ 深度神經網路

深度神經網路是由幾個學習模式的堆疊層所組成。

每一層都包含了數個神經元，即受到人腦啟發的**人工類神經網路（ANNs）**的基本計算元素。

神經元透過線性變換產生一筆輸出，即加權後的輸入總和與稱為**偏誤（bias）**的定值之相加結果，如下圖：

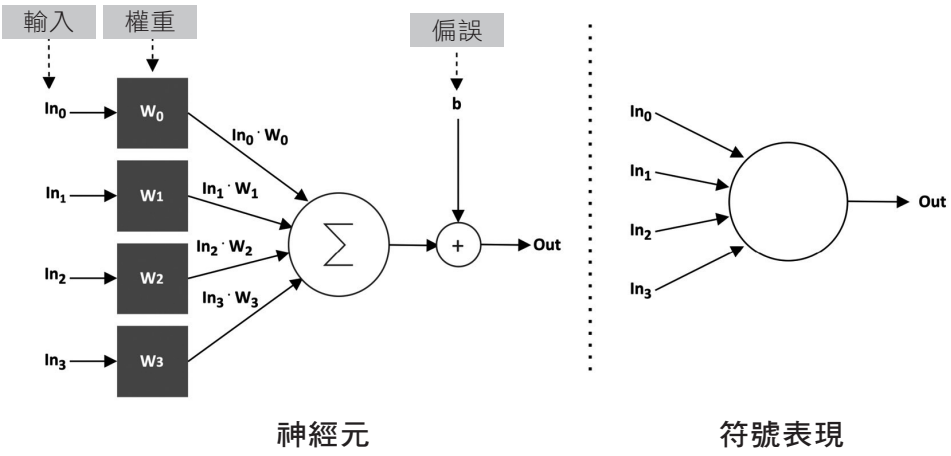

圖 1-3 – 神經元表現

加權後總和的係數就稱為**權重（weight）**。

在迭代訓練過程後的權重和偏誤可以幫助神經元學習複雜的模式。

然而，神經元只能透過線性變換解決簡單的線性問題。因此，被稱為**觸發**的非線性函數通常會搭配神經元的輸出來幫助網路學習複雜的模式。觸發為施加於神經元輸出上的非線性函數：

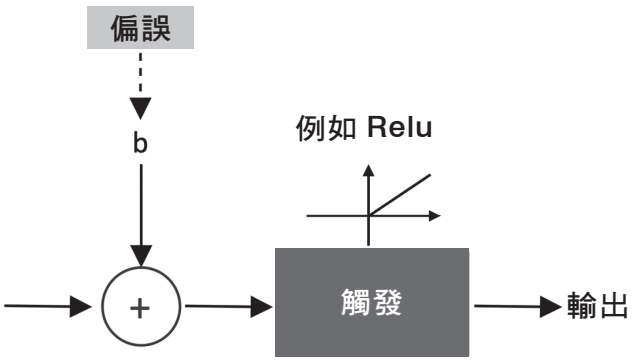

圖 1-4 觸發函式

整流線性單位函式（ReLU）為一種被廣泛採用的觸發函式，如以下程式碼所示：

```
float relu(float input) {
  return max(input, 0);
}
```

其運算的單純性讓它比其他需要更多運算資源的非線性函數，例如雙曲正切或 S 型函數等來得更受歡迎。

接下來將說明神經元是如何連接來解決複雜的視覺辨識作業。

◉ 卷積神經網路

卷積類神經網路（CNN）為專門用於視覺辨識作業的深度神經網路。

我們可把 CNN 視為由多個密集層（也就是**全連接層**）所組成的傳統**全連接神經網路**的進階版。

正如下圖，全連接網路的重要特徵之一便是每一個神經元都會與上一層的所有輸出神經元連接：

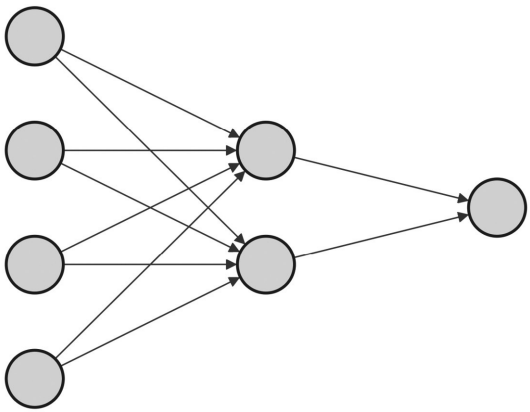

圖 1-5 全連接神經網路

不幸的是，這個方法不適合用來訓練圖像分類模型。

舉例來說，一幅大小為 320×240 的 RGB 圖像，僅僅一個神經元就會需要 230,400（320×240×3）個權重。由於網路層毫無疑問地會需要好幾個神經元來辨別複雜的問題，模型很可能因為可訓練參數的數量過多而產生**過度擬合**的問題。

在以前，資料科學家會採用特徵工程技術從圖像中擷取一組少量的優良特徵。然而，該方法在選取特徵時便遇到了瓶頸，因為它既耗時且會根據特定領域而有不同作法。

隨著 CNN 的興起，由於**卷積層**使特徵擷取成為了問題學習的一部分，視覺辨識作業得以大幅改善。

在需要處理圖像問題的前提下，並受到動物視覺皮質的生物處理程序的啟發，卷積層借用了在圖像處理中廣泛使用的卷積運算子以建立一組可學習的特徵。

卷積運算子的執行過程類似於其他圖像處理程序：於整張輸入圖像上滑動一個窗（或稱過濾器或核心），並在其權重和底層像素之間進入點乘積，如下圖：

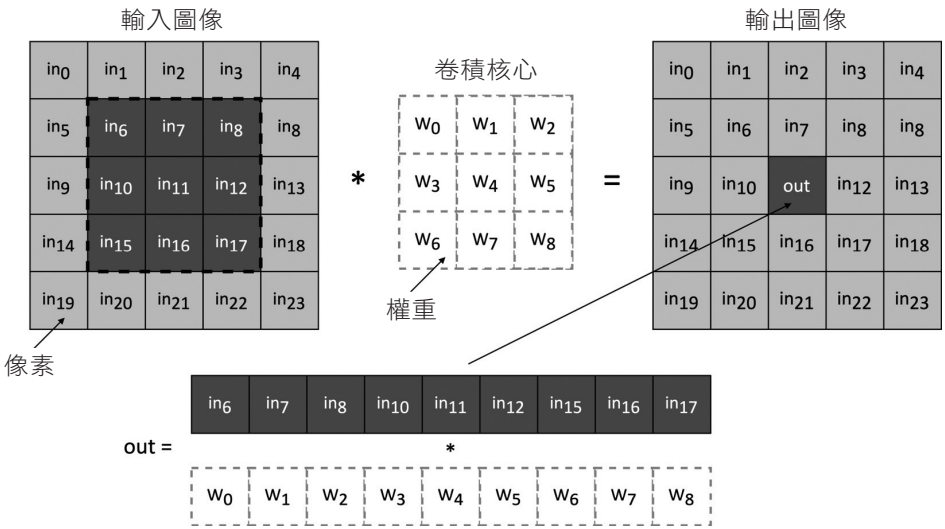

圖 1-6 卷積運算子

這個方法帶來了兩個顯著的好處：

- 無須人工介入便可自動擷取相關特徵。
- 大幅減少了每一個神經元的輸入訊號數量。

比方說，在之前的 RGB 圖像上使用 3×3 的過濾器，所需權重數量便大幅減少到只剩 27 個（3×3×3）。

如同全連接神經網路，卷積層會需要多個卷積核心以盡量學習更多特徵。因此，卷積層的輸出通常會是一組圖像（**特徵圖**），保存在稱為**張量**的多維記憶體物件中。

在為視覺辨識作業設計 CNN 時，通常會在網路的末端配置全連接層以執行預測。由於卷積層的輸出是一組圖像，通常我們會採用分階抽樣策略以減少透過網路傳輸的訊息量，並且在送入全連接層時降低過度擬合的風險。

常見的分階抽樣方法有兩種：

- 跳過某些輸入像素的卷積運算子。這麼一來卷積層輸出的空間維度便會少於輸入。
- 使用**池化層**等分階抽樣函式。

下圖為常見的 CNN 架構，其中池化層降低了空間維度，而全連接層負責執行分類：

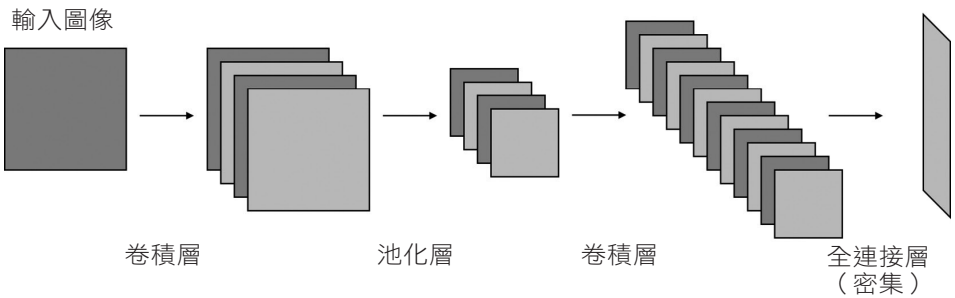

圖 1-7 常見 CNN 架構，使用池化層以降低空間維度

模型大小是在為 TinyML 部署 DL 網路時需要考慮的關鍵之一，也就是保存權重所需的記憶體容量。

由於這些迷你平台的記憶體有限，因此模組需要盡量簡潔小巧才能裝進目標裝置中。

然而，記憶體限制並不是在微控制器上部署模型時可能會遇到的唯一難題。儘管經過訓練的模型一般會以浮點精度進行算術運算，但微控制器的 CPU 無法為其提供硬體加速。

因此，**量化**便是克服上述限制不可或缺的一項技術。

◉ 量化

量化是指在較低位元精度上執行神經網路計算的過程。這項廣泛地被使用在微控制器上的技術會在訓練後執行，並將 32 位元浮點數權重轉換為 8 位

元的整數值。此技術可以將模型大小縮小 4 倍，並顯著地改善延遲時間但同時又能儘量保有原本的準確率。

DL 對於能否做出智慧決策的應用程式來說至關重要。然而，對以電池供電的應用程式來說，最關鍵的還是低功率裝置。截至目前為止，我們只是很籠統地提到了功率和能量，但接下來讓我們來好好地認識一下各自所代表的真正含義。

認識功率與能量的差異

功率大小對 TinyML 來說很重要，而我們的目標是**毫瓦（mW）**或更低，這意味著它的效率會比傳統桌上型機器好上數千倍。

雖然在一些情況下可能會使用太陽能板等**獵能（energy harvesting）**方案，但由於成本和尺寸的關係，未必可行。

話說回來，功率和能量究竟是什麼？我們就從控制電路的基本物理量來理解這兩個專有名詞吧！

◉ 電壓與電流

電流讓電路得以運作，它是在特定時間內通過導體表面 A 的電荷流動，如下圖：

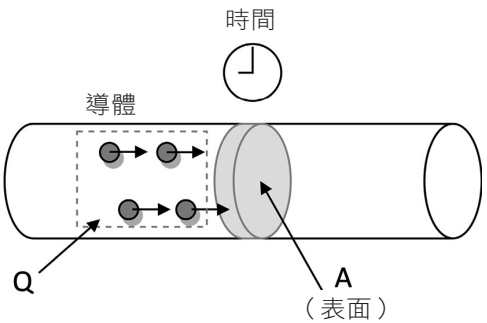

圖 1-8 電流為在特定時間內通過表面 A 的電荷流動

電流的定義如下：

$$I = \frac{Q}{t}$$

公式說明如下：

- I：電流，單位為**安培（A）**
- Q：特定時間內通過表面 A 的電荷，單位為**庫倫（C）**
- t：時間，單位為**秒（s）**

電流在滿足以下條件時會於電路中流動：

- 具有可以讓電荷流動的**導電材料**（像是銅線）
- 一條無中斷，可以讓電流連續流動的**閉合電路**
- 具有**潛在的電位差**，即**電壓**，定義如下：

$$V = V^+ - V^-$$

電壓的單位是**伏特（V）**，會產生電場讓電荷在電路中流動。USB 連接埠和電池都是一種電位差。

下圖為電源的符號表示：

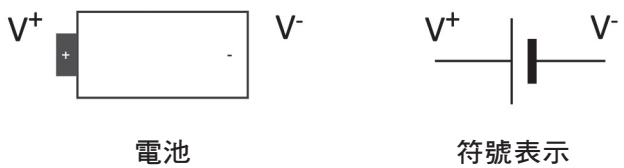

圖 1-9 電池的符號表示

為了避免一直看到 V+ 和 V-，按照慣例在此將電池的負極定義為 **0 V（GND）**。

歐姆定律涉及了電壓和電流，透過以下公式可知通過導體的電流會與電阻上的電壓成正比：

電阻是一種用來降低電流的電子元件。此元件具有以歐姆為單位的阻抗，標示為字母 R。

電阻的符號表示如下：

圖 1-10 電阻的符號表示
（https://openclipart.org/detail/276048/47k-ohm-resistor）

電阻是所有電路必不可少的重要元件之一，本書所使用的電阻會透過元件上的色碼來顯示其電阻值。標準電阻可能有 4、5 甚至 6 條色碼，其顏色代表了電阻值，如下圖：

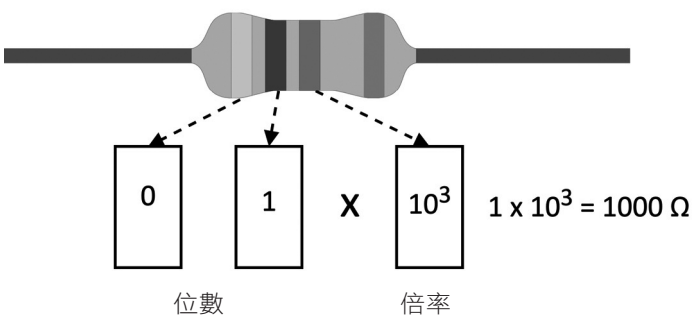

圖 1-11 一個有 4 條色碼的電阻

你可以利用 Digi-Key 輕鬆解讀這些色碼的意思：

https://www.digikey.com/en/resources/conversion-calculators/conversion-calculator-resistor-color-code

了解控制電路的主要物理量後，就可以來認識功率與能量之間的區別了。