國立臺灣大學生物資源暨農學院生物機電工程學系

碩士論文

Department of Biomechatronics Engineering

College of Bioresources and Agriculture

National Taiwan University

Master's Thesis

利用服役數據和機器學習模型評估混合動力汽車鎳氫電池的健康狀態

State of health estimation of nickel metal hybrid batteries for hybrid electric vehicles with on-road service data and machine learning models

王贊棠

Wang, Zan-Tang

指導教授：陳洵毅 博士

Advisor: Hsun-Yi Chen, Ph.D.

中華民國 113 年 01 月

January, 2024

doi:10.6342/NTU202400743

致謝

在完成這篇碩士論文的過程中，有許多人給予了我無私的支持與幫助，使得這項工作得以順利完成。首先，我要衷心感謝我的指導教授，陳洵毅教授。感謝您不僅在學術上給予我耐心的指導和寶貴的建議，督促我展開研究，教導正確的研究心態，在獲得成就時給予鼓勵，出現過錯時協助善後並促使我反省。您的專業知識和豐富經驗讓我受益良多，是我學術道路上的良師益友。

其次，我要感謝我的博士班學姊 Cindy，和已離職的前研究助理潘楷睿，在我進入研究領域的初期給予寶貴的建議和指引，在每一次進度報告時認真地聽取簡報、給予豐富而真實的回饋，點出我的錯誤，使我得以精進。在此特別感謝 Cindy，從我入學以來一直待到我畢業，在口試時列席旁聽，記錄口委們的意見，為我整理口委提出的問題，成為我修正論文的很重要的資源，節省了我很大的精力，使我得以順利地通過口試。你們的幫助讓我更快地融入了研究環境，並且在學術上取得了更大的進步。我會永遠感激你們的慷慨和支持。

接著，感謝現任的研究助理 Ken、新加入的博士班學姊林育秀，Ken 借給我桌機，幫助我跑程式，節省了等待結果的時間，讓我得以花更多的精神設計其他的研究細節和修正論文；林育秀是隔壁的室友，協助我解決疑難雜症或雜事，並在我當助教時協助我處理學生事情。

最後，一定要感謝我的父母。是你們一直以來的支持和鼓勵讓我堅持走到了今天。無論是在生活上還是在學業上，你們都給予了我無微不至的關愛和支持，願意理解無法準時畢業的我面臨的處境，給予無比的信任，支持我繼續休學、在台北生活，讓我做研究時無後顧之憂。你們的辛勤付出和無私奉獻是我不斷努力的動力和源泉，我會永遠感激你們的愛和支持。

在此，我要衷心感謝以上所有人的幫助和支持，是你們的關懷和鼓勵讓我度過了這段寶貴的學術時光。我將永遠珍惜這段美好的回憶，並將努力不懈地追求更高的學術目標。

doi:10.6342/NTU202400743

摘要

隨著全球氣侯變遷加劇及環保意識抬頭，油電混合車及電動車的銷量與日俱增，其中油電混合車於上世紀末商業化，至今已有逾千萬台的油電車於全球運行；而鎳氫電池作為其主要的儲電元件，未來全球將有大量的汰役電池需要被妥善處理。過往處理電池需要對電池做完全充放電，知道電池實際的充放電能力後，再依照量測結果判斷電池是否適合繼續服役，抑或是廢棄淘汰，但是完全充放電時間太長，不利於應付未來龐大的電池回收潮。此外也希望能夠預測電池性能未來的使用狀況和變化程度，依照電池老化狀況安排合適的使用方式，延長服役時間以節省製造電池的成本和資源，為此我們急需快速且準確的手段，檢測電池實際性能並進階地預測到未來的電池狀態變化。

由於科技的普及與資料科學之快速進展，人工智慧(artificial intelligence, AI)已經被應用於許多產業以增進生產效率和社會福祉。過去已有文獻將其應用於預測電池之健康狀態(state of health, SOH)，能準確地甄別電池狀態並取得優秀的成效，但是訓練樣本通常不大，僅限於幾百甚至幾十個，樣本數過小可能導致訓練出現偏差；再者，文獻中的電池大多是處於實驗室特定參數設置下規律地操作，與大多數上路電池經歷的操作歷程相當不同，訓練出的模型是否能處理使用歷程複雜的電池值得懷疑；最後，大部分的文獻關注探討鋰電池的性能和應用，近年少有針對鎳氫電池之健康狀態的預測和研究。但參考市場機構的調查，鎳氫電池以其突出的安全性依舊能在新能源交通工具中扮演重大的角色，因此發展出一套系統性判斷電池健康狀態，以利廠商分級回收再利用是很重要的事情。

我們也將應用人工智慧進行本研究。首先，我們透過詳盡的鎳氫電池文獻回顧，了解鎳氫電池的運作機制和衰退成因，以及其健康狀態的定義及量測方法，決定用恆流充放電(galvanostatic charge/discharge, GCD)檢測電池性能。我們

doi:10.6342/NTU202400743

認為充電曲線對電池性能高度相關，並據此設計並選擇了 5 個特徵。之後，歸納常見的機器學習模型並選擇貝葉斯山脊型迴歸(Bayesian ridge regression, BRR)、前饋神經網路(feedforward neural network, FNN)和隨機森林(Random forest, RF)進行訓練、測試，以很小的誤差準確預測電池之性能。為了確認特徵的泛化性能(generalizability)，我們執行了兩組老化實驗，驗證選擇的特徵在不同環境下產生的數據上依然適用，最後輔以遷移學習(transfer learning)，縮小不同數據中特徵的差異，進一步地縮小預測誤差。

關鍵字：鎳氫電池、電池健康狀態、機器學習、老化實驗、遷移學習

doi:10.6342/NTU202400743

Abstract

Along with the aggravating climate change and rising environmental awareness, sales of hybrid electric vehicles (HEVs) and electric vehicles (EVs) are increasing. The HEVs in particular, are commercialized by the end of last century and more than 10 million of them are operating around the world. Since nickel metal hydride (NiMH) batteries are the core energy storage component on board of HEVs, a significant amount of retired NiMH batteries will flood the world, requiring proper treatment. In the past, battery testing required a complete charge and discharge process to measure the actual charging and discharging capacity of the battery, and determine theirs application, to be on service or be discarded afterward. However, the complete charge and discharge process is time-consuming, not suitable for dealing with the upcoming massive battery recycling wave. Additionally, there is a desire to predict the future usage conditions and changes in battery performance, based on the battery's aging status, to arrange appropriate usage methods and extend the service life to save the cost and resources of battery manufacturing. Therefore, a fast and accurate method to obtain the actual performance of batteries and predict their future state changes is desired.

Due to the popularization of technology and rapid advancements in data science, artificial intelligence (AI) has been applied in various industries to improve production efficiency and social welfare. Previous studies have applied AI to predict the state of health (SOH) of batteries, achieving accurate identification of battery conditions and excellent results. However, the training samples are usually small, a few hundreds or even dozens. The small sample size may lead to bias in training. Moreover, the batteries in the literature are mostly operated under specific laboratory parameters, which are quite different from those on-road batteries, doubtful whether the trained models can

doi:10.6342/NTU202400743

handle batteries with complex usage history. Lastly, most of the literature focuses on the performance and application of lithium batteries, with limited research on the prediction of the state of health of nickel-hydrogen batteries, which is however, still play a significant role in new energy vehicles due to their outstanding safety. Therefore, it is important to develop a systematic approach to assess the health status of batteries, allowing manufacturers to classify and recycle them for reuse.

We also applied AI techniques in this study. Firstly, through a comprehensive literature review on NiMH, we gained an understanding of their operating mechanisms, degradation factors, and the definition and measurement methods of their SOH. We used galvanostatic charge/discharge (GCD) to assess battery performance. Recognizing the high correlation between charge curves and battery performance, we designed and selected five features. Subsequently, we summarized common machine learning models and chose Bayesian ridge regression (BRR), feedforward neural network (FNN), and random forest (RF) for training and testing. These models accurately predict battery performance with minimal errors. To confirm the generalizability of features, we conducted two aging experiments, validating that the selected features remain applicable in different environments. Finally, employing transfer learning helped narrow feature differences across various datasets, further reducing prediction errors.

Keywords: NiMH, SOH, machine learning, aging experiment, transfer learning

doi:10.6342/NTU202400743

目次

doi:10.6342/NTU202400743

doi:10.6342/NTU202400743

圖次

doi:10.6342/NTU202400743

doi:10.6342/NTU202400743

表次

doi:10.6342/NTU202400743

第一章 研究目的

氣候變遷產生的災變促使世界各國更積極地推動環保相關政策。研究發現溫室氣體的排放是造成氣候變遷的主因之一，其中在運輸過程中產生的溫室氣體可能高達總排放的 1/3 (Aminzadegan, et al., 2022) ，其為移動式排放源，較難集中處理，因此各國的減碳政策大多要求提高燃油車的燃油效率，推升了對油電混合動力汽車 (HEV)及電動車 (EVs)的需求。其中油電混合車自上世紀末被商業化，迄今 30 餘年內銷量年年增加，估計平均每日有一千萬台油電混合車 (Mohammadi, et al, 2023) 正上路運行；與之相應的，老舊車內部衰退的鎳氫電池 (Nickel Metal Hydride battery, NiMH battery) 預計持續增長，每年可能產生數萬的廢電池，如何妥善地回收和處理廢電池成為潛在問題。

鎳氫電池係以金屬儲氫合金與氫氧化鎳為兩電極所製成的二次電池，其利用氫氧根離子在電解液中移動造成電荷轉移、產生電能，過程中不消耗電極與電解液的成分，使得電池內部結構保持穩定，拉長循環壽命。其能量密度比鋰電池低，不過由於穩定性高且循環壽命長，適合支援引擎、滿足不需要持續輸出動力的電力需求，因此在以燃油車主宰的時代中率先受到重視，累積了許多分析鎳氫電池的電化學特徵與電池衰退現象的研究，以及判斷電池性能與老化速度的文獻，是技術相當成熟的電池。

然而，判斷電池性能的研究有很大的改善空間，因為多數發表過的方法是在實驗室裡研究學術問題，有的方式需要花費長時間取得相關資料，有的則步驟繁瑣、不易操作，這些方法在未來處理龐大電池時可能降低運作效率，甚至是準確度。目前台灣鮮少有業者針對汰役鎳氫電池發展再生利用或相關技術，因此我們認為這是絕佳的時機，對已退役的電池做研究，以因應即將到來的廢電池退役潮，為此，我們與豐田(Toyota)在台灣的代理商皓恆科技股份有限公司(POWEREIGN TECHNOLOGIES INC.) 合作，協請該公司提供商業化電池

doi:10.6342/NTU202400743

的數據，而我們從中找出判斷電池性能的最佳方法，為蓄勢待發的回收產業提高使用效率及解決方案，同時降低對環境的衝擊。

本研究之目的為研發出一套快速、有效的方法判別鎳氫電池的性能狀態，依據判別結果對汰役電池做再生、重組，重新安裝到油電車上做再利用。我們採取非破壞性檢測方法，只以電池的充放電數據預測電池的容量及健康狀態，以達成在生產線上快速選別可再利用之鎳氫電池模組。我們幫助合作業者判斷退役電池的性能，以利再生回收或重新組裝成整新電池後再安裝到車上，節省製造新電池的成本，也減少因製作及廢電池後處理產生的環境汙染，期盼讓經濟利益與環保雙贏。

doi:10.6342/NTU202400743

第二章 文獻探討

2.1 鎳氫電池的充電曲線和化學反應

充放電過程中，鎳氫電池的電化學反應式為

$$(\text{正極})\ NiOOH + H_2O + e^- \underset{\text{charge}}{\overset{\text{discharge}}{\rightleftharpoons}} Ni(OH)_2 + OH^- \quad (1)$$

$$(\text{負極})\ MH + OH^- \underset{\text{charge}}{\overset{\text{discharge}}{\rightleftharpoons}} M + H_2O + e^- \quad (2)$$

並且伴隨著各式副反應。首先，在充電過程末段及過充電的狀況，由於 $Ni(OH)_2$ 即將或已經消耗完畢，在充電末端和過充電下，正極會析出氧氣，之後擴散到負極與氫結合形成水:

$$4OH^- \rightarrow 2H_2O + O_2 + 4e^- \quad (3)$$

$$O_2 + 4MH \rightarrow 4M + 2H_2O \quad (4)$$

而在放電過程末段及過放電下，正極會析出氫氣，之後擴散到負極上被氧化為水：

$$2H_2O + 2e^- \rightarrow H_2 + 2OH^- \quad (5)$$

$$H_2 + 2OH^- \rightarrow 2H_2O + 2e^- \quad (6)$$

從以上的反應式可以得知，在理想的充放電循環中，電解液的數量和濃度沒有淨變化，可以防止電解液損失 (Shukla, Venugopalan et al. 2001)。隨著充電時間持續，持續產出的 NiOOH 由於熱力性質不穩定，會再與電解液發生反應，與主反應同時發生並瓜分外部電源供給的電能並產生氧氣，之後擴散到電池負極被金屬氫化物消除，防止氣體積累在電池內傷害電池:

$$2NiOOH + H_2O \rightarrow 2Ni(OH)_2 + \frac{1}{2}O_2 \quad (7)$$

鎳氫電池典型的充電曲線如下圖 2-1 和圖 2-2。圖 2-1 為不同充電速率(C

doi:10.6342/NTU202400743

rates)下量測到的電壓曲線和溫度曲線，其中可觀察到電壓及溫度的曲線隨著充電速率增加而提高。低速率的充放電(0.1C、0.3C)能讓反應物質盡可能地完全反應，維持低溫與低壓避免損害電池結構，但是耗時長、不利於研究和產線利用，過高的充電速率(高於 1.0 C)省時但是會拉高反應溫度、阻止完全反應而損害電池性能，所以本研究以中間值 1.0 C 作為實驗速率，在便利性及有效性中間取得折衷，同時這也是被眾多文獻推薦的充電速率。

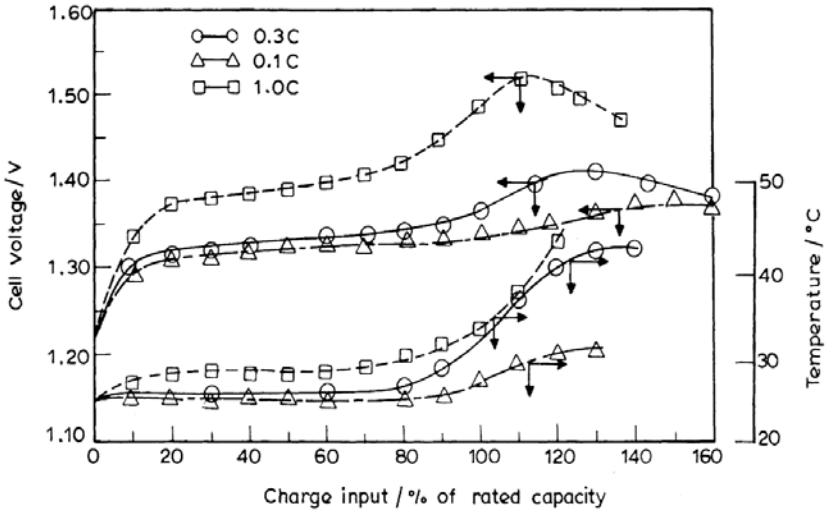


圖 2-1 不同 C rates 下電池的電壓、溫度的變化(Shukla et al., 2001)

本研究中分析的電池，其全新狀態下的 1C 充電曲線如圖 2-2 所示。假設電極過程受到電子轉移步驟和液相傳質步驟混合控制，$j_d > j_c \approx j_a \gg j^0$，電池電壓為電動勢、歐姆過電位、濃差過電位和電化學過電位四者之和:

$$V=E+\eta_{anode}+\eta_{cathode}+IR_S \tag{8}$$

其中:

$$\eta_{anode} = \frac{RT}{\overleftarrow{\alpha}F}\ln\frac{j_a}{j^0} + \frac{RT}{\overleftarrow{\alpha}F}\ln\frac{j_d}{j_d - j_a} \tag{9}$$

$$\eta_{cathode} = \frac{RT}{\vec{\alpha}F}\ln\frac{j_c}{j^0} + \frac{RT}{\vec{\alpha}F}\ln\frac{j_d}{j_d - j_c} \tag{10}$$

(η_{anode}、$\eta_{cathode}$: 陽極、陰極過電位；$\overleftarrow{\alpha}$、$\vec{\alpha}$: 氧化反應、還原反應傳遞係數；j^0: 交換電流密度；j_d: 極限擴散電流密度；j_a、j_c: 陽極、陰極電流密度)

由上面兩個式子可知，過電位與電流密度有關，所以由電流密度的變化就可以推估電壓曲線走勢，右式第一項是電化學極化，第二項是濃差極化。

doi:10.6342/NTU202400743

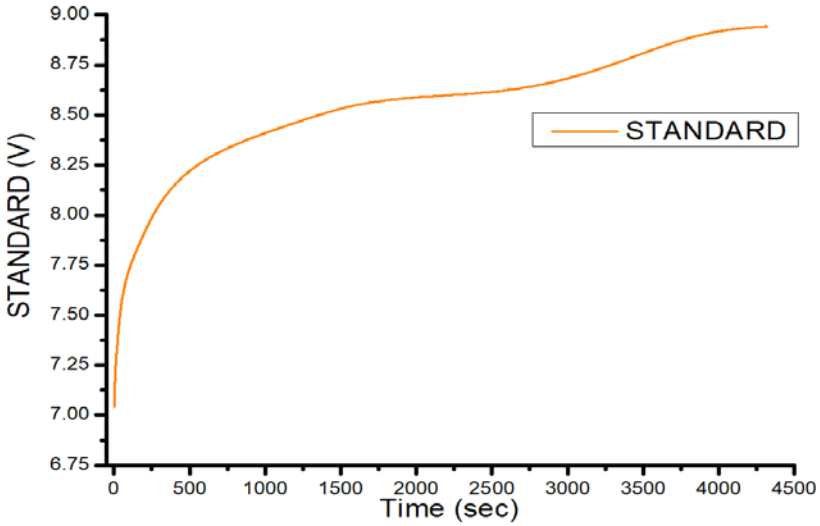


圖 2-2 新鎳氫電池在 1.0 C 下的充電曲線

充電一開始時正極電位從平衡向正偏移、負極電位向負偏移，形成陽極極化和陰極極化，電極電流密度(j_a 、j_c，以下統稱j)逐漸增加，過電位受到電化學極化($\frac{RT}{\alpha F}\ln\frac{j}{j^0}$)影響最深，因為自然對數的特性充電初期電壓陡升，不過高過電位使電極反應加快，提高電流密度並開始去極化，所以電壓增速由陡升變成緩升。到了充電中後期，正極的$Ni(OH)_2$減少，改進行副反應: $4OH^- \rightarrow O_2 + 2H_2O + 4e^-(E^0 = +0.4)$氧氣、繼續提高電壓；另一方面，濃差極化項($\frac{RT}{\alpha F}\ln\frac{j_d}{j_d - j}$)因為正極被生成物 NiOOH 包覆，反應面積變小，使j增加、濃差極化項變大，兩個因素作用下趨緩的電壓增速再次增加。最後電池變成純內阻，充電主要產生熱能，加熱電池使電池內阻(IR)降低，所以電壓趨於平緩並降低。因為鎳氫電池在充電最末段有上述電壓下降、溫度上升、內部壓力上升的特性，在決定鎳氫電池充滿電的策略通常為檢測電壓變化或是設定截止溫度及壓力，本研究中則參考此特性，發現電壓在一秒內降低 3 mV 時即終止充電。

2.2 鎳氫電池性能與容量衰退成因

電池運作時內部元件會隨著使用時間拉長鈍化和衰老，降低電池的儲能能力、容量降低。鎳氫電池性能與容量衰退分為可逆損失和不可逆損失 (Young

doi:10.6342/NTU202400743

and Yasuoka 2016)，可逆的損失又稱為自放電現象，指的是放電容量隨著存放時間拉長而減少，但再充電後便可回復原先的容量。相比之下，不可逆的容量無法由充電回復，通常是因為電池內部構造在運作時毀損和老化造成，是電池性能衰退的主因，因此本研究聚焦在探討不可逆的容量損失，找出與其相對應的性能參數評估電池的健康狀態。

在一般的充放電循環下，電池衰退主要由正負極的損害造成。循環過程中正極活性物質剝離、膨脹，以及添加劑(鈷)降解。電極降解減少儲存的容量(Sastry, Choi et al. 1998)，負極活性物質氧化和剝離 (Chen, Xu et al. 2002)，其中負極容量衰退較快，失去處理氧氣和儲存氫氣的能力，限制化學反應的效率，進而加速電極衰退，形成惡性循環。電池的外部條件也嚴重影響電池的性能，常見的是溫度和充放電速率，高溫(≥ 45℃)下長期存放或操作會加速反應速率，包括加快儲氫合金表面氧化、加快自放電現象、隔離膜降解，並造成嚴重的極化現象；高速充放電(> 2C)下會生成大量氣體，使電極表面產生剝離，且氣體擴散速率又受限於儲氫合金表面固相氫擴散能力而累積在一處，造成熱的累積，因此升高電池溫度，造成類似高溫操作的效果並加劇衰退。鎳氫電池的衰退現象整理如下圖 2-3 所示。

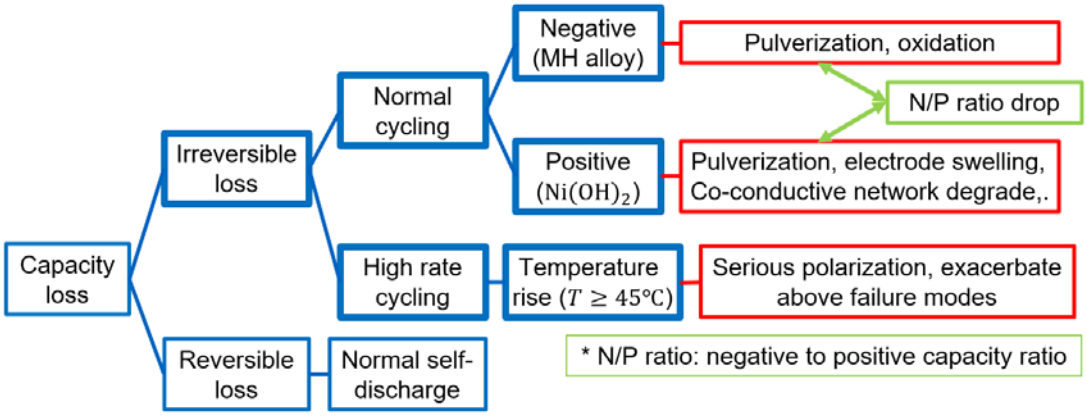


圖 2-3 鎳氫電池的衰退現象和機制

doi:10.6342/NTU202400743

2.3 充電曲線上的特徵

2.3.1 反曲點

從衰退成因當中，我們發現兩個影響鎳氫電池性能的重要因素:氧氣和極化現象。結合充電時發生的電化學反應推論，氧氣反應會在反應物快用完時發生，是充電的副反應，產生高過電位並讓持續減小的電壓增速轉向遞增，在數學上稱電壓增速由負轉正或由正轉負的點「反曲點」(Inflection point)，若反曲點發生時間提早代表電池反應物不足或不完全，反映電池性能衰退嚴重。許多文獻或專利都有研發尋找反曲點的方法，例如 (Park, Miwa et al. 2008)對原曲線做一次微分後，用非零的值之間的時間差距判斷反曲點、(Nicolai and Wuidart 1995)同樣對原曲線做一次微分，但是加入平滑處理並以平滑後的曲線最高值當作反曲點、 (Jones and Reoch 1985)則同時利用二次曲線和一次曲線做判斷。上述研究成果豐富，但大多數是針對新電池開發的方法，尚未針對回收電池做研究，且有電池溫度作為輔助，對設備要求更高。我們將借鑒文獻成果，針對本研究中的電池設計出合適的反曲點分析法。

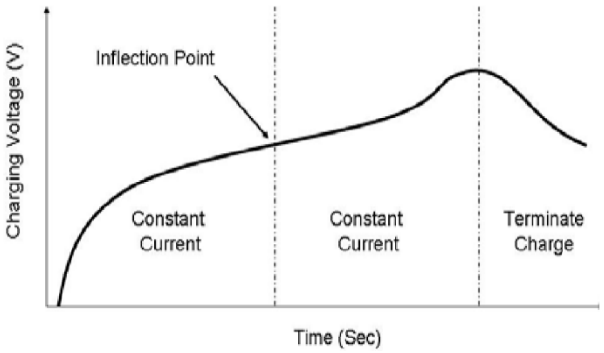


圖 2-4 充電曲線與電壓反曲點 (Park, Miwa et al. 2008)

2.3.2 弛豫現象

極化消退又被稱為「弛豫現象」(Relaxation)，是電池從工作狀態中穩定下來，轉換成平衡狀態的過程，平衡過程中會把因為過電位而產生的濃差極化和電化學極化給消除 (张志刚, 张涛 et al. 2022)，達成平衡所需時間稱為弛豫時間，更短的時間表示電池極化程度小，能夠很快地達成平衡，代表極化不嚴

doi:10.6342/NTU202400743

重、電池性能尚優，是電池恢復能力的量度。弛豫現象是電池運行時必然發生的現象，但大多數研究都是針對鋰電池做分析 (汤依伟, 艾亮 et al. 2015)，對鎳氫電池的弛豫現象很少，雖然鎳氫電池和鋰電池的成分不同，但弛豫行為係過電位於充電完成後消退的現象，也就表示兩種電池都會發生弛豫行為，因此我們將弛豫現象視為判斷鎳氫電池性能的可能參數。我們將借鑒鋰電池的研究方式，配合鎳氫電池的反應機制，以電壓變化衡量弛豫現象。

除了確認弛豫行為與電池性能之間的相關性，我們進一步探討不同種類的極化對電池性能的影響力，並細分出電池性能衰退的來源和表現的形式。歐姆極化源於電池內電阻，被視為斷電時的瞬時電壓變化，由於容易量測而常被用來預測健康狀態 (Schneider, Oliveira et al. 2014, Mu, Agrawal et al. 2022)；歐姆極化消退之後是電化學極化和濃差極化混合作用，其中以濃差極化為主，被視為電池極化中最重要的指標 (Zhu, Zhou et al. 2019)，但文獻並未提及濃差極化的確切的持續時間，無從判定哪一個時間尺度適合代表電池性能。受限於充放電程序的時間，我們無法等待電池電壓完全收斂，因此我們決定在有限的休息時間中，找到最適合衡量電池極化程度的弛豫時間。有文獻指出 1C 放電過程中，濃差極化約在第 6 分鐘達到極大值並收斂 (Paxton and Newman 1997)，而在 Raffaele, et al. 的研究中對電池進行短時間高速率的充放電，發現休息 2.5 分鐘已足以讓電壓收斂 (Bornatico, Storti et al. 2007)，於是推測代表極化程度的弛豫時間應該在以上兩種時間尺度左右。我們會先衡量充放電之間的休息時間下的弛豫現象對 SOH 的相關性，若發現高度相關性再將其分割成不同時間段，衡量不同弛豫時間下的電壓變化對 SOH 的相關性後，選出兼顧高度相關和省時的時間尺度。

doi:10.6342/NTU202400743

2.3.3 其他特徵變數

由於本研究旨在於油電混合動力汽車中將鎳氫電池模組再利用，故不拆解電池模組，只使用非破壞性的檢測方法進行量測。非破壞性檢測方法大部分是基於特別設計的充放電實驗以及交流阻抗實驗，例如 Laure Le Guenne 和 Patrick Bernard(2002)將一個高頻和低頻的 RC 等效電路擬合交流阻抗頻譜，以低頻的實部阻抗作為負極儲氫合金氧化程度的指標；(Cheng et. al. 1998) 以全充放電循環實驗來模擬電池老化，得出循環圈數和放電容量的關係圖，再以等效電路擬合不同區段的健康狀態；(Chelab et. al. 2006) 以高電流短時間的充放電使電池老化，在固定循環圈數時測量電池整體容量以及交流阻抗頻譜，得出電阻、電容與放電容量的關係；(Yang et. al. 2015) 設計特別的充放電步驟，使用充放電時電池開路到充電的瞬間電壓差當作電池放電容量的指標，並整理出電壓差與電池放電容量的關係式。

在圖 2-3 鎳氫電池的衰退現象和機制圖 2-3 中整理出衰退現象中，正負極破碎(pulverization)是常見的現象，且和正負極的電極容量比例(N/P ratio)會相互作用，因此找到能反映電極破碎和 N/P ratio 的特徵就很重要。N/P ratio 通常會維持在 1.35~1.5 之間以確保電池壽命，兩端電極將以參考電極為基準測量其電極容量在相除後得到 N/P ratio (Minjie et, al. 2008)。影響 n/p ratio 的因素眾多，很難逐一且細緻地考慮每一種因素的影響力，因此篩選出較簡單且直接的方法，比如用兩電極的重量比代替容量比(Kasnatscheew, Placke et al. 2017)、開路電位(Mu, Agrawal et al. 2022, Kim, Jeong et al. 2015), 電壓曲線(Mu, Agrawal et al. 2022, Kim, Jeong et al. 2015, Chen, Yang et al. 2022)和歐姆壓降(Mu, 2022)。

另一方面，電極的破碎程度需要拆解電池、使用 XRD 分析電極表面狀態，或者使用 PCT 遲滯系統(pressure-concentration-temperature hysteresis system)在局部電極通入氫氣後令電極釋出氫氣，觀察氣體釋出之氣壓和輸入

doi:10.6342/NTU202400743

氣體之氣壓之比例，以其高低判斷電極是否破碎，比例越小表示電極能良好地儲存氫氣、結構尚且完整(Young, 2014, Young, 2009)。由於需要拆解電池，我們將不考慮採行文獻中的方式，並尋找替代方法。

由上述段落可知，電壓相關的參數很適合做為篩選好壞電池的條件。受到弛豫行為的啟發，電壓高低和過電位應有緊密的關係，因此將充放電電壓納入判斷電池性能的可能參數。不少文獻是擷取充電曲線的一小段當作訓練資料(Kim, Jeong et al. 2015, Roman, Saxena et al. 2021, Chen, Yang et al. 2022)，而截止電壓也常常被用來作為變數 (Fetcenko, Koch et al. 2015)(Schneider, Oliveira et al.2014)。電壓曲線的分析比較涉及多個資料點，會加重運算負荷，因此在本研究中以某一時刻的電壓代替曲線，節省分析的精力，而截止電壓為充電結束當下之電壓。我們將上述電壓的特徵變數也納入分析，從中選出具代表性的參數做進一步的研究。

2.4 性能狀態指標

在衡量及評估電池性能狀態時，通常以電量狀態(state of charge, SOC)及電池健康狀態(state of health, SOH)做描述。文獻中多數的描述類似 (Galeotti, Giammanco et al. 2015)，將 SOC 和 SOH 分別定義為「某一時刻下，電池儲存的電量佔當下的最大容量的比值」及「單次充放電循環中，電池最大容量與剛出廠時的最大容量的比值」，前者衡量電池「當下的電量有多少」，後者衡量電池「當下最多能充/放多少電」，皆以百分比的形式呈現。 (Semanjski and Gautama 2016)將上述比喻成容器和盛裝物: 容器可以被填滿甚至外溢(SOC≥100%)，但在使用過程中會逐漸毀損導致容積縮小(SOH<100%)，即便被填滿總量也比一開始少，代表容器的性能下降(電池衰退)。因此 SOC ≥ 0%，數值高低取決於電池內的電量多寡，而 SOH 介於 0%~100%，數值高低取決於電池毀損程度，使用越

doi:10.6342/NTU202400743

久數值越低。本研究旨在為業界提供預測電池性能的方法，而非追蹤、預測電量，因此以預測 SOH 為目標。

SOH 無法直接測量，因為在電池運行時會遭遇許多不確定因素，比如電化學和熱力學反應的不確定性、開路電位、溫度、電池老化、整體效率，導致電池產生非線性的表現，無法精準地評估電池當下狀態及最佳性能 (Xuan, Shi et al. 2020, Cui, Wang et al. 2022)。所以研究上採取一個簡單、具代表性的算式算出一個數值代表 SOH，然後用各種方法測量電池性能數據，採取不同方法讓獲得之數據逼近該假設值。

學界對預測 SOH 的共識是將使用過後的電池的最佳表現拿去和最初狀態相比，用新舊狀態下的最佳表現的差異反映電池的衰退/老化程度，並視其為 SOH。最佳表現通常以最大容量或內電阻代表，絕大多數的研究採用最大容量，將電池進行完整的充放電，測出電池的最大容量(actual capacity)後除以最初的容量(initial capacity, 又稱額定容量)作為 SOH (Yang, Qiu et al. 2015):

$$\mathrm{SOH} = \frac{\text{actual capacity}}{\text{initial capacity}} \times 100\% \tag{11}$$

其中，最大容量法過程簡單、直接，被大多數研究採納，但也被質疑容量不足以代表電池的性能狀態，特別是安裝在交通工具上的電池，更會受到工作環境干擾，使得運行的條件不一致(inconsistency)，失去統一的基礎衡量電池的衰退程度 (Zhang, Li et al. 2021)，因此催生出許多種測量及演算方式。 (Pradhan and Chakraborty 2022)彙整了許多測量 SOH 的方法及其利弊，針對不同的需求和研究目的可以選用不同的技術，如圖 2-5 所示。

縱然以容量代表健康狀態有欠週全，我們仍然採用這個方式，原因是它與電池能量儲存能力和電池的剩餘使用時間直接相關，此方式經歷長時間研究已經被確認過有效 (Roman, Saxena et al. 2021)。此外，本研究中的資料中都有容

doi:10.6342/NTU202400743

量、充放電時間的詳細記錄，因此採用容量測試是最簡便有效的，在本研究中 SOH 為某一時刻電池完全充電後、放電至 6.0 V 的容量，除以該電池的額定容量 (6.5 Ah):

$$\text{SOH} = \frac{\text{完全充電後，放電至 6.0 V 時減少的容量}}{\text{額定容量(6.5Ah)}} \times 100\% \quad (12)$$

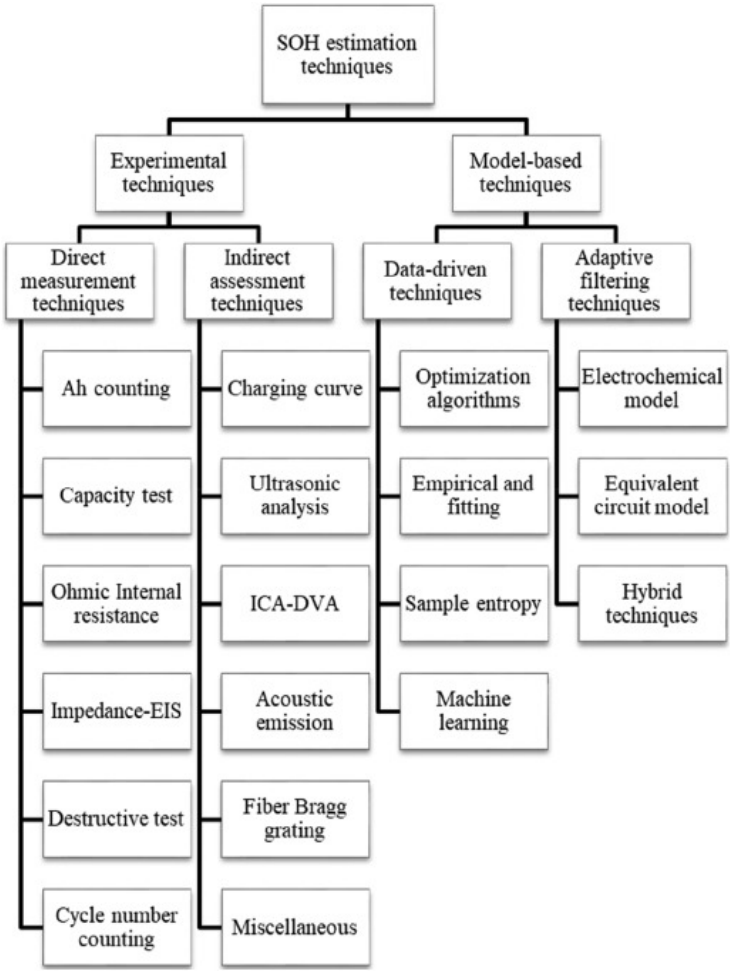


圖 2-5 測量 SOH 的方法 (Pradhan and Chakraborty 2022)

doi:10.6342/NTU202400743

2.5 SOH 預測方法文獻回顧

預測SOH已行之有年，隨著時代的演變，預測SOH的特徵和模型雙雙發生變化，形成了如圖2-5中琳瑯滿目的預測方式。較常見的有如(Micea, M. V., et al. 2011) 即借助電池管理系統(BMS, battery management system)提供數據，用多項式和最小平方法擬合電壓曲線和容量衰退曲線協助預測SOH；(Galeotti, M., et al. 2015)量測電池在不同檢測頻率下的阻抗，結合等效電路模型模擬電池的衰退情況，只要知道特定位置的阻抗就能反推該電池進行了幾圈的老化實驗；(Yang, H., et al. 2015)將開始充電瞬間電壓的上升幅度視為電池衰退的特徵，用曲線擬合、最小平方法擬合了容量衰退曲線，並分析每種SOH下預測的精準度如何；(Telmoudi, A. J., et al. 2020)發現等效電阻模型存在缺陷，於是引入歐氏粒子群優化算法(euclidean particle swarm optimization)和模糊迴歸模型(Fuzzy c-regression model)做優化，使之能夠更好地擬合衰退曲線、預測SOH。

在往後，機器學習、深度學習領域崛起，且可以分析的數據變多，有大量的文獻點出前人方法的不足，比如完全充放電、獲得容量衰退曲線過於耗費時間；等效電路模擬需要同時了解電化學和頻繁調整模擬電路，效果卻並不穩定；測量阻抗、EIS的方法僅限線下測量使用，難以引進BMS；利用數學工具擬合的結果，可能無法用於複雜的狀況。以機器學習為首的方法以其高精準、低耗時、應用場景多聞名，因此成為近年來預測SOH的主流，所以我們將使用此方法進行本研究。

doi:10.6342/NTU202400743

2.6 機器學習法(ML)

得益於人工智慧的突破和大數據的發展，越來越多的研究以資料驅動方法(data-driven approach)取代模型方法(model approach)分析或預測電池健康狀態和電量狀態，其中以機器學習法(machine learning, ML)備受關注，其將數據做非線性的處理並根據結果做修正和迭代，在長年的研究下已經發展出針對不同的研究目的或數據的來源的方法，幫助使用者解決問題。

機器學習過程包含前處理、建立模型、優化和驗證模型成效。前處理是篩選放入輸入層的變數、處理有缺漏的資料，及對樣本做分組的步驟。放入輸入層的變數被稱為健康指標(health indicator, health factor)，被視為和電池健康狀態高度相關的數據，選擇適合放入模型的變數的過程稱為特徵工程(feature engineering)，有助於提升訓練成果、減少誤差，是最重要也最耗時間的環節。篩選完變數後若發現樣本缺少該變數或變數的值不合理，需要替換成合理的數值，比如將數值設為平均數或 0，或者把該樣本排除。確認完所有數據為正常後，將所有數據分割成至少兩組，即訓練組(train set)和測試組(test set)，將訓練組的資料用來訓練模型，用測試組的資料衡量模型成效。

選擇和建立模型步驟包括釐清研究目的、審視擁有的資料種類和特色，選出適合的模型進行訓練。模型分為將數據正確地分類(classification，分類模型)，或者預測某項數據的數值(regression，迴歸模型)，共 4 大類、13 分支，詳見表 2-1 (Vidal, Malysz et al. 2020, Roman, Saxena et al. 2021, Ng, Zhao et al. 2020) ，依序說明如下:

1. 神經網路(neural network, NN):

神經網路是一種受到生物神經系統啟發的計算模型，由大量的單元(類神經元)相互聯結，形成單層或多層的結構，針對欲預測的目標進行估計或近似，具備學習資料特性的功能，是目前備受矚目，且被公認是預測效果

doi:10.6342/NTU202400743

佳、應用性廣泛的方法。一般來說，網路分為輸入層、隱藏層和輸出層，輸入層、輸出層分別為特徵、預測目標，各為一層，中間的隱藏層數目不定，一層隱藏層的網路被稱為簡單神經網路，兩層以上或更多的會被稱作深度學習網路，但認定的標準不一。數據自輸入層進入隱藏層後，每層的神經元包含一次矩陣運算和一個激活函數(activation function)，上一層的輸入經過矩陣映射後輸出，並經過激活函數過濾掉不合理、不重要的數字再輸出，重複此步驟直到抵達輸出層，稱為訓練「一輪」(epoch)。每輪預測的誤差會被用來校正每一層內的神經元，再進行下一輪的預測，直到完成預定的 epoch 或者其他適度擬合的條件。自輸入層單向映射至輸出層的這個過程稱為前饋(feedforward)，用上一輪誤差校正下一輪網路權重的機制稱為反向傳播(back propagation)，因此神經網路又別名前饋神經網路(feedforward neural network, FNN)或多層感知器(multiple perception, MLP)

在(Luo, Y.-F. and K.-Y. Lu, 2022)中，作者將電化學阻抗圖譜(electrochemical impedance spectroscopy, EIS)特定的頻率區間當特徵，用僅有一個隱藏層的 NN 預測 SOH，誤差最低可達 0.21 %； 在(Roman, Saxena et al. 2021)中，作者將 NN、貝葉斯方法、集成學習方法結合，由許多個 2 層 NN 共同預測 SOH，並且提供該預測的概率分布情況，稱為DNNe (deep learning neural network ensemble)，在所有電池中的預測誤差未必是最強的，但不確定性通常最低；(Li, W., et al. 2021)想要預測 SOH 以及剩餘壽命，因此選擇有利於分析時間序列，又有能力線上部署的長短期記憶網路(long-short term memory, LSTM)，誤差僅 2 %。

2. 支援向量機器(support vector machine, SVM):

SVM 被視為對分類問題、非線性問題、特徵數目多的問題時，最佳的演算法，目標是對一群分布狀況不同的數據做最佳的區分，透過計算數據之間的距離，在一空間中找到一平面或合適的函數，將數據區隔開來。以

doi:10.6342/NTU202400743

二元線性分類為例，將數據置於一空間中，並找到兩個超平面，盡可能地區分數據，接著找出和兩個超平面距離等距的超平面，稱為最大間隔超平面，能夠分類最佳化，且保有一定的空間如容忍噪音和新數據。超平面和點、任意兩個超平面之間的距離由向量計算而得，而和最大間隔超平面距離最近的數據點稱為「支援向量」，只有該數據點能夠影響分類結果，非支援向量不行，所以可有可無，此方法由此得名。

在更複雜、非線性的分類問題中，還會引入核函數，將原始數據投影到高維空間中再做預測，在分類問題中要找最大間隔超平面，在迴歸問題中則是找最佳擬合曲線，無論是分類問題還是迴歸問題都能勝任。常見的是徑向基函數(radial basis function, RBF)，衡量某個點(x)到中心(c)的距離，以距離長短或在一定距離下準確區分數據的能力作為模型成效，RBF 運算式子(13)見下方。在(Kheirkhah-Rad, E., M. Moeini-Aghtaie, 2021) 中，作者拿 SVM 和兩層 NN 做比較，發現 SVM 在 21 個電池的容量衰退曲線預測中，有 12 個達到最佳預測，遠超過只贏 6 個的 NN，不過消耗的時間比 NN 多上不少； (Hu, X., et al. 2015) 把 SVM 作為基準，驗證 Sparse Bayesian predictive modeling (SBPM) 的有效性，跟結構較簡單的模型相比，SVM 的誤差僅略低於 SBPM，差距僅 0.1 %，誤差只有 1.48 %。

$$\phi(\mathrm{x},\mathrm{c}) = \phi(\|x - c\|) \tag{13}$$

3. 高斯過程(Gaussian process, GP)/貝氏方法(Bayesian method):

這兩個方法認為數據分布可以用機率去描述、量化，在訓練數據有限的情況下提高預測新數據的能力，減小預測的不確定性。高斯過程包含兩個概念: 高斯分布、隨機過程，其假設樣本出現的機率呈現高斯分布，但不同環境下的分布會有形狀的差異，能用數據平均值(μ)和協方差(σ^2)描述不同環境下數據分布的特性，如式子(14)所示，然後再把不同數據分布重疊，得到完整的分布情況，據此進行預測。代表的模型為高斯過 (14)

doi:10.6342/NTU202400743

程迴歸(Gaussian process regression, GPR)

$$p(x) = N\big(\mu(x), \sigma^2(x)\big)$$

貝氏方法是一種基於貝葉斯統計理論的方法，跟 GP 一樣用概率分佈來描述不確定性，差別是具有隨著數據的更新，變更原本假設的分布參數的機制。其更新方式如式子(15)所示，該式子被稱為貝葉斯公式，在知道新數據的分布特性後，更新先驗概率分佈以獲得後驗概率分佈，再進行預測。代表模型為貝葉斯嶺迴歸(Bayesian ridge regression, BRR):

$$P(\theta|X) = \frac{P(X|\theta) \cdot P(\theta)}{P(X)} \tag{15}$$

$P(\theta|X)$: 後驗概率，在觀察到數據X後對參數θ的信念

$P(\theta)$: 先驗概率，對參數θ的初始信念、$P(X)$: 樣本數據的概率

$P(X|\theta)$: 似然函數，在參數θ下觀察到數據X的可能性

在(Roman, Saxena et al. 2021)中，作者重視預測中的不確定性，因此同時使用 GPR 和貝氏方法的 BRR 預測電池的容量衰退曲線，發現 BRR 在恆流放電電池中有高的準確度，不過 GPR 並不佳。而(Xue, Y., et al. 2022) 結合 NN 和 GPR 預測鋰電池的使用壽命，誤差都小於 1.5 %，(Zhang, Y., et al. 2022)更顯示 GPR 甚至能超越神經網路，得到很低的誤差，不過同時揭露了 GPR 訓練時間較長、很容易出現極值，這是建立模型前要考慮的事。

4. 決策樹:

此方法建立一個樹狀結構，對數據集的特徵進行遞歸分割，在每個葉子節點(nodes)上依照所選特徵區分數據，目標是讓一個節點上的數據都屬於同一類別（分類）或具有相似的數值（回歸）。決策樹具有直觀的解釋性，決策過程能被可視化，但是單一決策樹容易導致過度擬合，因此常見的使用方式是使用集成學習方法(ensemble)，如隨機森林(random forest, RF)、梯度提升樹(gradient boosting tree)來改進性能。

doi:10.6342/NTU202400743

隨機森林透過建立多個獨立的決策樹，將所有樹的預測結果取平均當作擬合值，通常是用於大型數據集。首先進行樣本抽樣，從原始數據集中隨機抽樣，形成多個子樣本(Bootstrap 樣本)，接著在每個子樣本中隨機選擇特徵，形成不同的特徵子集，然後對於每個子樣本構建一個決策樹，決策樹們彼此獨立，最後將每個決策樹的預測結果進行投票(分類)或平均(迴歸)，形成最終的模型預測。前面多次提過的(Roman, Saxena et al. 2021)也有用隨機森林，發現他在恆流-恆壓和快充數據上的誤差最低，但是不確定性較高，會對預測過度樂觀。(Zhang, Y., et al. 2022) 把隨機森林和神經網路、高斯過程迴歸和支持向量迴歸，在三種公開數據上的預測誤差和耗費時間做比較，隨機森林沒有出現極端值，在一些數據上可以取得最優或次優的成果，僅次於神經網路，值得關注。

梯度提升樹透過迭代地組合多個決策樹，在每一次迭代時優化預測結果。首先設立一個決策樹並將它初始化，接著訓練該樹、計算殘差(實際值與模型預測值之間的誤差)，然後新增一個弱學習器(通常是淺層決策樹)擬合這些殘差，並通過優化方法計算弱學習器的權重，使它將殘差最小化。再來更新模型，將已優化的學習器的預測值乘以權重，將乘積加到初始決策樹的預測值上，最後重複迭代直到誤差最小化，每一步都專注於補償前一步模型未能準確預測的部分。此模型之後更衍生出了「極端梯度提升樹」(extreme gradient boosting, XGBoost)，新增了正則化和自動停止功能，防止過度擬合，並且在新增弱學習器時隨機抽取特徵建造決策樹，提高泛化性能。(Jafari, S., et al. 2022)就利用XGBoost，擊敗了用深度學習方法預測 SOH 的兩份文獻，誤差低至 0.002 %，是備受矚目的模型之一。

根據以上對各類模型的回顧，我們選擇三個模型: 貝葉斯嶺迴歸(BRR)、隨機森林法(RF)和前饋神經網路(FNN)。BRR 屬於機率分布模型，將每一個資料點視為一個鐘形曲線概率分布圖，將訓練組資料點轉換成概率分布後，全部疊

doi:10.6342/NTU202400743

加起來形成新的概率分布圖，測試組的預測值則由該數據對應到的機率決定。此演算法常被應用於資料集規模小、數據不足的情況，通過概率分布的情況彌補訓練集小的缺點。GPR 也是機率分布模型，但是由於會出現極端值，運算時間較長，我們決定選擇 BRR 預測 SOH。

RF 是以決策樹(decision tree)為底的演算法，因其具有高準確度且不易過度擬合，近年以來備受推崇。另外一支 XGBoost 也是強大且新興的演算法，不過其運算方式和 FNN 相似，比如透過誤差更新原預測值，就跟神經網路中以誤差更新神經元的權重的機制相似，因此選擇 RF 做為決策樹方法的代表。

FNN 藉由讓輸入值經過多次非線性的運算以及誤差最佳化，被證實能達到高準確度，且能跟其他類模型結合，有很大的發揮空間。提高 FNN 中的隱藏層數目有助於降低誤差，但容易出現過度擬合並提高訓練時間，因此本研究採用一般的神經網路，調整層數觀察預測誤差的變化，找到適合的隱藏層數目。

表 2-1 機器學習種類和分支

種類	衍生方法
Neural network	Feedforward neural network (FNN) Deep neural network ensemble (DNNe) Recurrent neural network (RNN) Hamming network
Support vector machine	Radial basis function (RBF) Support vector machine (SVM) Support vector regressor (SVR)
Gaussian / Bayesian	Bayesian ridge regression (BRR) Gaussian probability regression (GPR) Naive Bayesian algorithm (NB) Sparse Bayesian predictive modeling (SBPM)
Decision tree	Random forest (RF) Gradient boosting

doi:10.6342/NTU202400743

2.7 加速老化實驗

廠商提供的數據多為二手模組，已經服役過一些時間，SOH 大多偏低，僅針對這些電池做出的研究成果可能很難用於所有模組。我們期待在低 SOH 的模組上面做的研究，尤其是提取的特徵，在高 SOH 模組上面也具有代表性，因此我們規劃了加速老化實驗，獲取最新的模組進行循環實驗，企圖獲取 SOH 較高時電池的充放電資料，增加研究成果的應用範圍和有效性。

常見的老化實驗參數包括環境溫度(℃)、充放電速率(C rate)、放電深度(depth of discharge, DOD)和電量狀態變化區間(SOC swing)，其中以溫度和充放電速率對電池性能影響最深(Chen, et al., 2021) 。我們將建立一個老化實驗矩陣(aging test matrix) ，進行兩組老化實驗，一組在高溫環境以中速進行充放電循環，另一組則在常溫下用高速率進行充放電循環，經過固定循環次數後採集特徵，進行分析和建模預測，並觀察在極端溫度和極端速率下的結果有何差異。Hosen 等人 (Hosen, Youssef et al. 2021) 使用分類模型預測老化趨勢，我們預計使用迴歸模型進行預測，並比較不同模型的預測結果，篩選最適合的模型。

doi:10.6342/NTU202400743

2.8 遷移學習

遷移學習（Transfer Learning）是一種機器學習的方法，將從一個任務（source task, 源任務，或稱源域）中學到的知識應用於另一個相關但不同的任務（target task, 目標任務，或稱目標域），目的是通過在一個任務上學到的知識，改進在另一個任務上的性能，尤其是在目標任務的數據較為有限或缺乏的情況下，常見的應用範圍包括圖像識別、自然語言處理、語音識別，近年來也衍生出不同的應用和針對新問題而開發的技術。在預測SOH方面，分為三種方法(Shen et al., 2023).

1. 微調(fine-tune):

   是遷移學習中最簡單且最流行的策略。首先在源域中訓練一個基礎模型(base model)，然後凍結預訓練模型中某些層的權重，再於目標域中重新訓練，如此一來模型同時保有預測和區分源域(被凍結的層)和目標域(未被凍結、經過二次訓練的層)的能力，不需要因為數據集更新而重新訓練，或者因為數據不足而無法建模。

   各個研究中，只有最後一層是不被凍結的，差異在於使用的特徵、處理特徵的方法，以及改造模型。(Shu, X et al. 2021)利用充電曲線固定電壓區間內的電壓曲線，及容量衰退曲線當輸入，訓練並微調一個長短期記憶網路(long short term memory, LSTM)，誤差在 3 %內；(Kim, S et al. 2021) 用容量衰退曲線中，早期的衰退曲線當輸入，並改造 LSTM、加入 Monte-Carlo Dropout (蒙特卡羅 dropout) 避免不確定性，預測往後的衰退走勢，誤差大約 2 %；(Yao, L et al. 2022)認為老化實驗早期的放電曲線，和未來衰退走勢高度相關，因此利用容量增量分析法 (incremental capacity analysis, ICA)獲取數據，並使用離散小波變換 (discrete wavelet transform, DWT)平滑 ICA 曲線、用灰色關聯度分析 (grey relation analysis, GRA)對曲線做歸一化，誤差少於 2 %；(Sahoo, S

doi:10.6342/NTU202400743

et al. 2022)則將電池視為電阻，測量每次實驗後通過該電池的電壓下降數值，將其當作特徵放入多層感知器(Multilayer Perceptron, MLP)做預測，誤差少於 2 %。

2. 度量學習(metric learning):

此方法通過計算源域、目標域的特徵的分布差異，並透過將原始特徵投影到不定維度的空間中，試圖將此差異最小化，以提升預測的精準度和預測不同域的數據的可能性。自 2022 年起開始看到數篇文獻使用此方法預測 SOH，方法有: 最大均值差異 (maximum mean discrepancy, MMD)、多核 MMD (multi-kernel MMD, MK-MMD)和相關對齊 (CORAL)，多以 MMD 為核心做運算和改造進行研究。以 MMD 為例，MMD 是廣泛使用的非監督特徵對齊度量。它測量在再生核希爾伯特空間(RKHS)的單位球中源域和目標域之間的均值差異。此方法不需要標記的目標數據，使它們適用於實時模型更新。MMD 式子如下:

$$\mathrm{MMD}(x^S, x^T) = \left\| \frac{1}{N^S}\sum_{i=1}^{N^S}\phi\left(x_i^S\right) - \frac{1}{N^T}\sum_{i=1}^{N^T}\phi(x_i^T) \right\|_{\mathcal{H}}^2$$

$$= \mathrm{K}(x^S, x^S) - 2\mathrm{K}(x^S, x^T) + \mathrm{K}(x^T, x^T) \tag{16}$$

$$\mathrm{K}(\mathrm{A}, \mathrm{B}) = \frac{1}{N^A N^B}\sum_{i=1}^{N^A}\sum_{j=1}^{N^B} k(A_i, B_j) \tag{17}$$

$$k\left(A_i, B_j\right) = \mathrm{e}^{\left(-\|A_i - B_i\|^2/2\gamma^2\right)} \tag{18}$$

其中，x^S, x^T分別為源域和目標域的數據、N^S, N^T是這些數據的數目、$\|\cdot\|_H^2$指在 RKHS 計算兩個特徵向量的距離(內積)、ϕ是 RKHS 中的映射函數、K是核平均、k是核函數，內含參數γ控制核函數的寬度。核函數 k 計算每一對樣本之間的距離，將全部的計算結果加總、平均後，用計算結果度量兩組樣本之間的相似性。

MMD 具有和 RMSE、MAE 類似的性質，是衡量特徵在不同域的差異的度量，因此可以被用來將模型參數最佳化的因子，比如 Ma, G et al.

doi:10.6342/NTU202400743

(2022) 運用深度學習的卷積神經網路 (Convolution neural network, CNN) 提取特徵並進行預測，他們將 MMD 視為損失的一部份，所以將 MMD 和預測 SOH 的誤差相加，形成總損失，用來調整 CNN；Han, T., et al. (2022) 甚至將總損失拆成三塊: 源域的預測損失、目標域的損失，再加上 MMD，且為了減低單一核函數的參數對特徵的影響，他們用了 5 個不同參數的核函數做映射，取其總合當作 MMD。

3. 域對抗自適應 (adversarial adaptation methods):

   此方法為 MMD 的衍生方法。在模型中加入一個判別器 (discriminator)，在計算 MMD 並且調整映射的特徵時，會把映射特徵送入判別網路中，評估其是否能夠區別目標域的特徵和映射的特徵，最終目標是要讓判別器無法判斷映射特徵究竟處於哪個領域，以此進一步降低特徵在不同欲之間的差異。Su, S., et al. (2022) 將判別網路、MMD、深度學習網路合用，利用等效電阻模型 (equivalent circuit model, ECM) 生成虛擬數據並加入到原本的數據中協助預測，預測誤差降低至最低 3.7 % (0.0941 Ah / 2.5 Ah)

在以上三個方法中，採用微調和度量學習的研究居多，是較普遍且有效的方法。在我們有大量數據支持研究，但數據種類多的情況下，很有可能出現源域和目標域特徵分布不一致的狀況，因此將採取度量學習法，以 MMD 為主進行遷移學習，降低預測誤差。

doi:10.6342/NTU202400743

2.9 研究缺口與文獻回顧總結

執行文獻回顧後，我們發現了幾個大類的研究缺口，分別是對象、數據、特徵和模型。首先是研究對象，幾乎所有的文獻都是針對鋰電池，少有針對鎳氫電池的研究，在目前的市場上，鋰電池確實擔任市場領頭羊的腳色，是潛力巨大、尚待研究的電池，而鎳氫電池是一個商業化數十年的電池，對其電池機制的瞭解已經很深，但適逢人工智慧和電動車同時迅速發展、受到更多注意力，因此少有運用 AI 預測鎳氫電池的性能的研究。有鑑於未來將出現的電池退役潮，且有大量服役數據支持，應該增加預測鎳氫電池性能的研究。

接著是數據的部分，我們不想被廠商提供的數據限制模型的泛化能力，而進行老化實驗，獲取更多 SOH 下的數據將模型最佳化。我方取得的數據全部是二手電池的充放電數據，且絕大多數都是 70 % SOH 以下，沒有高 SOH 的性能資料。這些低 SOH 過去被認為缺乏穩定性，不適合服役，若僅針對低 SOH 建模，用途將被受限，縱使有機率分布模型和遷移學習等解套方式，仍需要一定量的數據支持，因此我們進行老化實驗，且將停止時機設為 50 % SOH，遠低於文獻普遍的停止時間(70 %~ 80 % SOH)，試圖建造一個模型，無論好壞電池都能得到很好的預測結果。

再來是特徵的部分，我們決定採用人工方式選取特徵，包括使用曲線平滑、多項式擬合等數學工具找出反曲點，曾被提及具有預測性能的潛力，但過去尚未有文獻使用反曲點預測 SOH；根據電化學特性，找了若干個電壓、極化現象相關的特徵，這相對常見，多數是取歐姆極化、在電壓曲線上的特定點，這些也被我們囊括近來，並且將更進一步地分析各個特徵的重要性。

最後是模型，我們依照機器學習模型的性質和文獻中的表現，依據模型原理、誤差大小、耗時、不確定性來篩選模型，並搭配遷移學習的 MMD 提升預測效果。目前看過用各類型的深度神經網路搭配 MMD，或者機器學習、深度網路兼用進行的研究，尚無機器學習搭配 MMD 的組合，值得一試。

doi:10.6342/NTU202400743

第三章 研究材料和方法

3.1 研究架構與流程

本研究分析的電池為 Toyota 油電車車款採用的鎳氫電池，由 Panasonic 和 Toyota 合資公司 Primearth EV Energy 所生產，負極以$MnNi_5$(AB_5型)的吸氫合金體系以及經過親水處理的聚丙烯隔膜(hydrophilic - treated polypropylene)做成，正極由 NiOOH 與$Ni(OH)_2$做成，電解液為 30wt%的 KOH 溶液。以一個 Camry 用的鎳氫電池組(pack)為例，內含 34 個模組(module)，如圖 3-1 所示，表 3-1 是模組的基本規格。一個鎳氫電池模組包覆 6 顆額定電壓 1.2V 的電池(cell)，模組之間彼此串聯成一組，為了方便再生後重組再利用，模組一經組裝即不再拆解以避免電池損壞，因此本研究中電池的最小操作單位是模組，只對其進行「非破壞性檢測」(Non-invasive inspection)，採用恆流充放電(galvanostatic charge/discharge, GCD)評估 SOH。

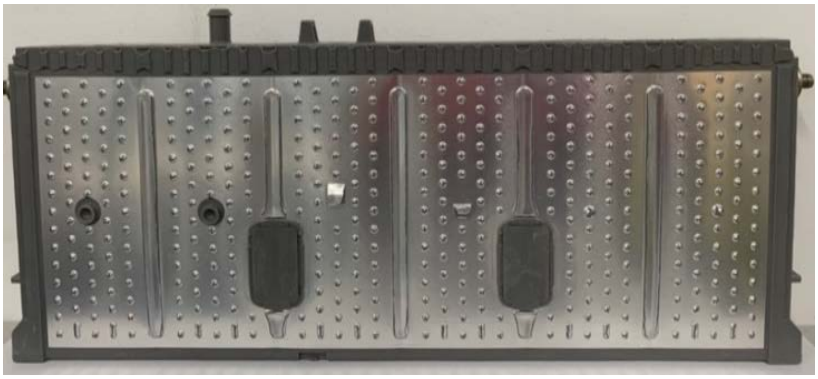


圖 3-1 鎳請電池模組(module)示意圖

表 3-1 鎳氫電池模組基本規格

Nominal Capacity	6.5 Ah
Working Voltage	7.2 V
Cut-off Discharge Voltage	6 V

doi:10.6342/NTU202400743

研究過程依序分為四個部分: 決定對樣本進行的充放電程序、設計量化特徵的步驟、篩選特徵變數、訓練特徵變數與分析結果。首先，依據欲量化的特徵性質跟廠商的要求設計合適的充放電程序；下一步是闡述如何用電化學知識將特徵量化，並畫出流程圖；接著將各個變數對電池的 SOH 做線性迴歸分析，篩選出高度相關的變數，或重新設計量化流程使變數更具代表性；最後選擇欲訓練的機器模型訓練變數，調整模型的超參數以優化結果。

3.2 實驗方法和數據

本研究中的數據有兩個來源，服役數據和老化數據，前者是皓恆將退回工廠的服役模組，經過充放電測試後依放電時間多寡分級重組的模組；後者是跟皓恆取得全新和二手模組，在實驗室以固定的實驗流程操作而產生的數據。依照測試方法的不同又再細分為以下 4 種，見表 3-2:

(1) 服役前數據:

皓恆接收因故障退回的電池組，對其做充放電測試後，將放電時間超過 30 分鐘的模組挑選出來，重新組成一個電池組，再對其做充放電測試後得到的數據。得到數據後，這些模組即上車服役。充放電流程圖詳見圖 3-2，在本研究稱其為 DCD35 (Discharge, charge 35 min, and discharge)。

(2) 服役後數據:

數據(1)因故障退回的電池組，對其做充放電測試後得到的數據。充放電流程圖同服役前數據，詳見圖 3-2。

模組服役時，車上的 ECU (electrical control unit)會將兩個模組串聯後的電壓跟相鄰的兩個串聯模組相比，相差超過 1V 會告知損壞，被送回車廠檢修。此外我們很早就獲得服役數據，發現服役後模組之間的 SOH 和檢測結果有巨大的差異，因此決定從服役後模組找特徵，並在老化實驗中證明其合理且有效。

doi:10.6342/NTU202400743

(3) 高溫老化數據:

將 6 個全新模組(SOH=100%)組成一個電池組，遵循圖 3-4 的流程操作。其流程包含: 檢測 SOH (紫色步驟)、預充電前處理 (綠色步驟)、高溫環境下(45℃常溫烘箱)以 1C、6.5 A 的電流循環充放電 50 次 (橘區步驟)、回歸室溫靜置並在測 SOH 前做前處理(藍色步驟, 即 DCD35)，重複以上流程直至其 SOH 衰退至 50 %。組成的電池組見圖 3-5，使用的充放電機器為皓恆提供的 TBTS (Toyota battery test system)，見圖 3-6(右)，實驗自 2/19 起實施至 11/15。在 4/10 發現之前數據在檢測 SOH 時未被充飽，故修正實驗流程；另外，在 6/20 因充放電機器狀況不穩，不適合長時間運行，為避免外界干擾而取消 DCD35，改為休息 4hr 後再開始檢測 SOH。數據範圍則從 4/10 開始取自結束日 11/15。

(4) 高速老化數據:

將 1 個全新模組(SOH=100%)與 5 個服役過、SOH 各不同的模組 (SOH: 80~95%) 組成一個電池組，遵循圖 3-7 的流程操作。其流程包含: 檢測 SOH、預充電、常溫下(25℃實驗室溫度) 以 3C、19.5 A 循環充放電 10 次、靜置回歸室溫、檢測 SOH，重複以上步驟直至其 SOH 衰退至 50 %。模組的起始容量見，電池組見圖 3-8，使用的充放電機器為新威電子有限公司的 BTS8.0 (Battery test system) ，見圖 3-9。實驗自 2023 年 6/21 開始，執行至同年 10/11，耗時約四個月。

流程圖中出現的「PVD」(Peak voltage determination)是用來判定模組達到充電截止條件的機制。若在充電時電壓於一定時間內停滯、不再增加，或者降低一定的電壓(ΔV)，將判定模組已充飽，終止充電，具體條件見表 3-2、示意圖見圖 3-3。服役數據的 PVD 因為儀器存在量測誤差，故有許多模組未精準依照預定條件停止充電，尤其是ΔV，所以會採取一定範圍內的ΔV使用而非定值。

doi:10.6342/NTU202400743

表 3-2 本研究中的 4 種數據

數據來源	服役數據		老化數據	
數據種類	服役前數據	服役後數據	高溫老化數據	高速老化數據
實驗方法	DCD35	DCD35	1C 循環 + 完全充放電	3C 循環 + 完全充放電
PVD 條件	ΔV: 3 mV、電壓在 2 min 內不再增加			
備註	機器存在檢測誤差，PVD 範圍和預定條件有所差距		於 4/10、6/20 更改實驗流程	

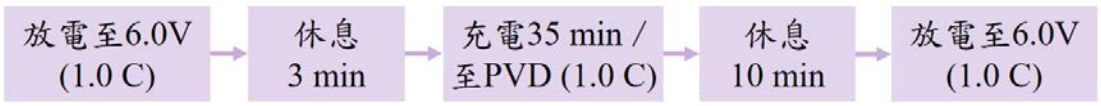


圖 3-2 服役前、服役後數據的充放電檢測流程 (DCD35)

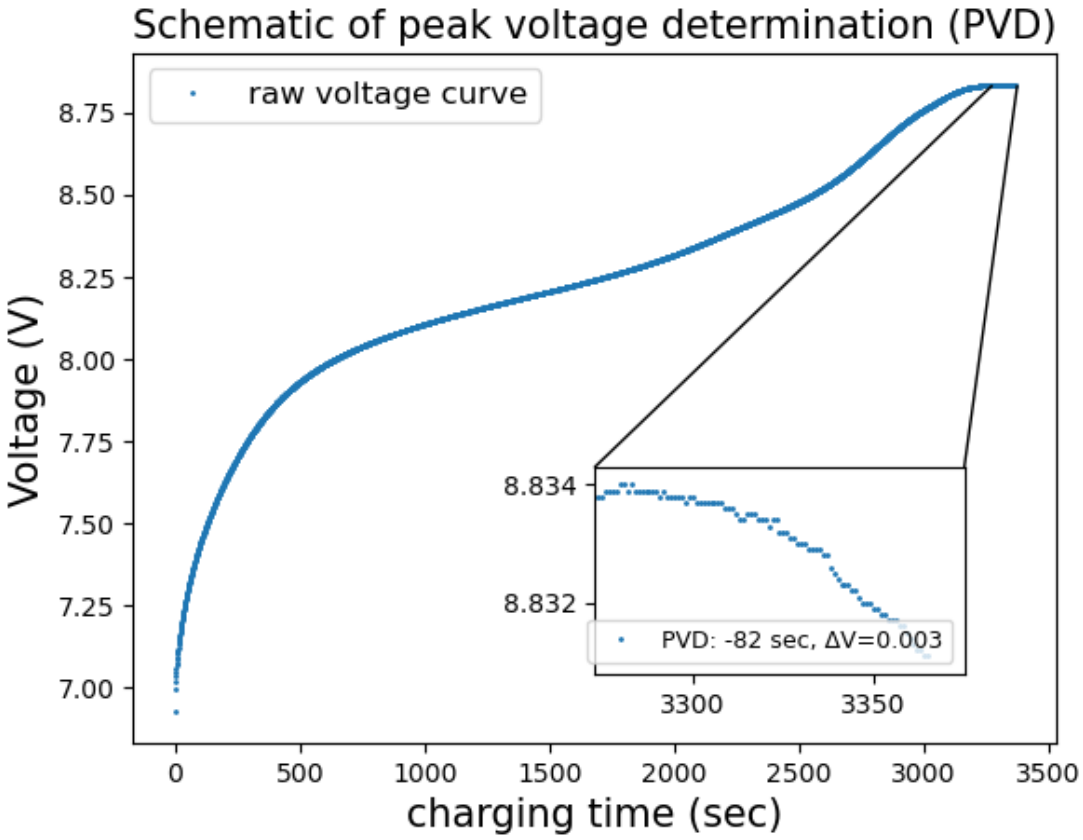


圖 3-3 PVD 示意圖

doi:10.6342/NTU202400743

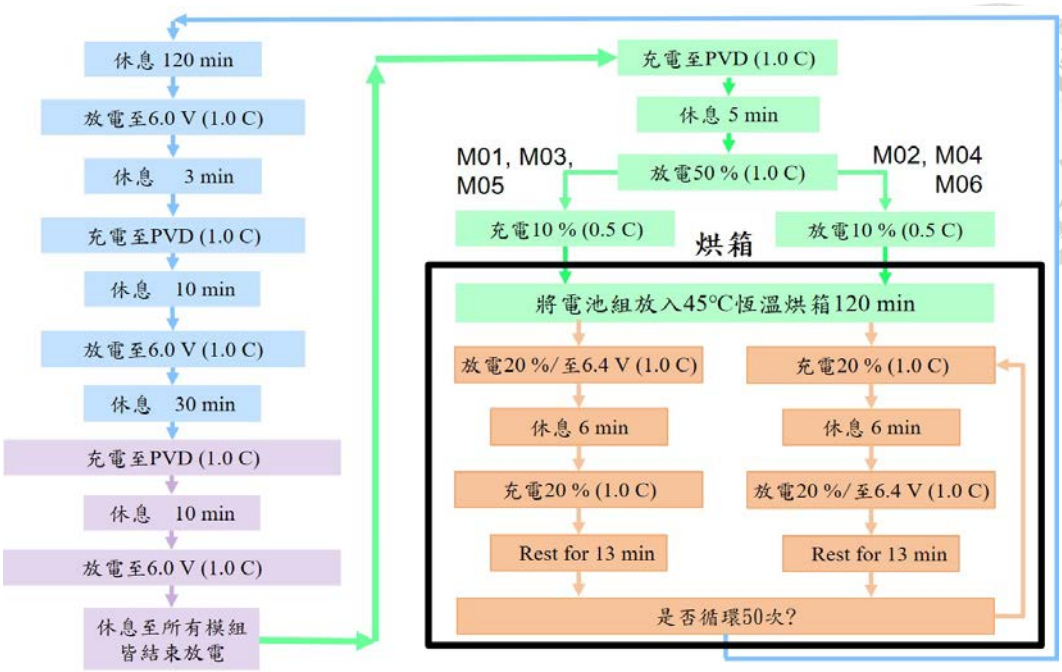


圖 3-4 高溫老化實驗流程

圖 3-5 高溫老化實驗電池組 (內含 6 個模組)

圖 3-6 高溫老化實驗用到的烘箱(左)和充放電機器(右), TBTS

doi:10.6342/NTU202400743

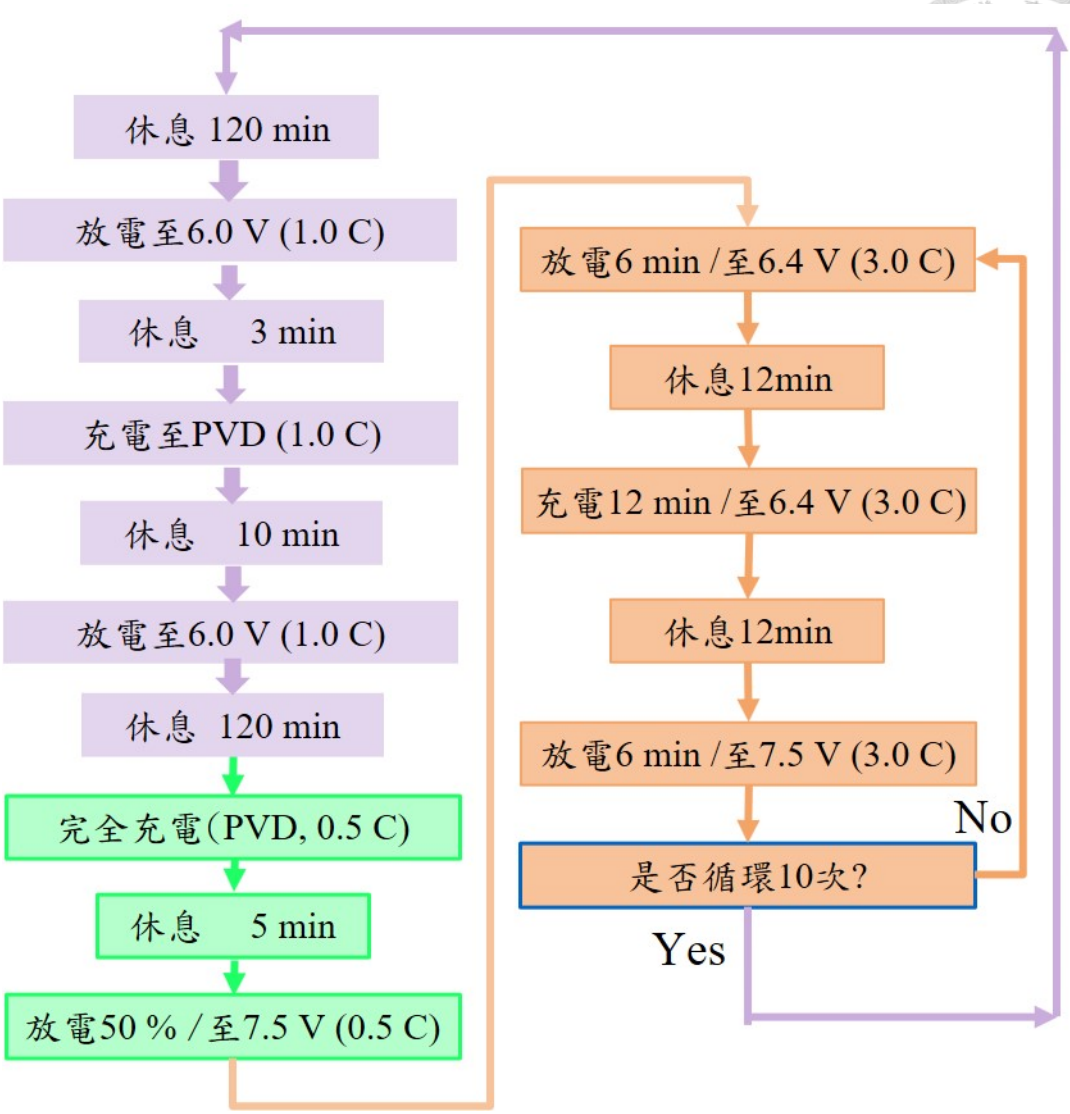


圖 3-7 高速老化實驗流程

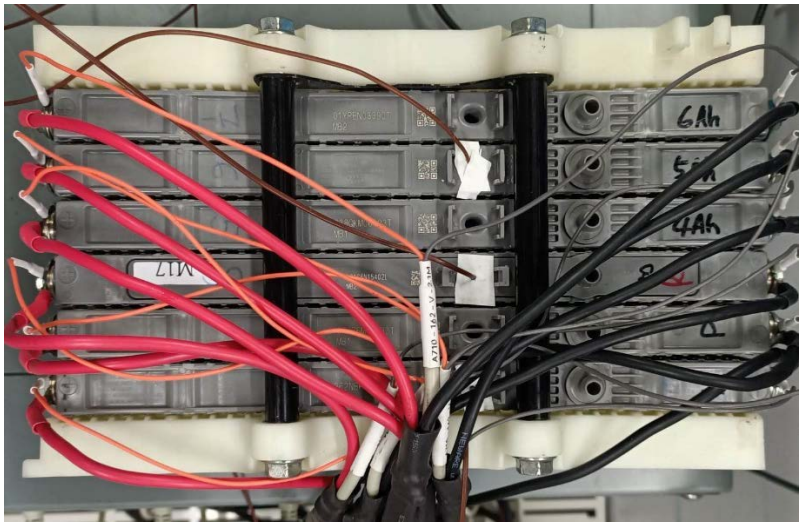


圖 3-8 高速老化實驗電池組

doi:10.6342/NTU202400743

表 3-3 高速老化實驗之電池組，各個模組的起始容量

模組編號	充電時間 (時:分:秒)	放電時間 (分:秒)	起始 SOH (%)
K	54:04	48:44	81.22
P	59:45	49:40	82.77
4Ah	1:03:28	53:13	88.69
5Ah	1:01:07	52:25	87.36
6Ah	1:07:12	56:50	94.72
M17	1:10:00	58:58	98.27

圖 3-9 BTS8.0

doi:10.6342/NTU202400743

3.3 反曲點分析

如 2.3.1 節所述，我們想要找到充電曲線圖上的反曲點（如圖 2-4）。此點為電壓曲線由平緩走向陡升的位置，數學意義上代表電壓加速度由負值轉正(電壓曲線二階微分數值)，同時電壓、電壓增速(電壓曲線一階微分數值)也是正值。根據這個現象我們選將電壓對時間做二次微分以搜尋反曲點。首先對原始資料做前處理，採用高階多項式函數對曲線做擬合(fitting)、消除雜訊。接著對擬合曲線做二次微分，尋找電壓斜率出現反向變化的時間點，如圖 3-10。在此我們發現，不同階數的多項式擬合同一條電壓曲線，會產生多餘的反曲點，然而一條曲線中通常只有一個明顯的出現電壓增速加快的時間點，意即氧氣副反應發生的時間點，因此最合理的情況是只產生兩個反曲點，因此我們選擇四次微分曲線擬合電壓曲線，將反曲點數量降低到最多兩個，第一個點表示電壓開始加速，第二個電表示電壓開始減速，然後選擇第一個點代表氧氣副反應的開始時間，符合鎳氫電池充電時的電壓特性，即電壓增速從趨緩轉向上揚。

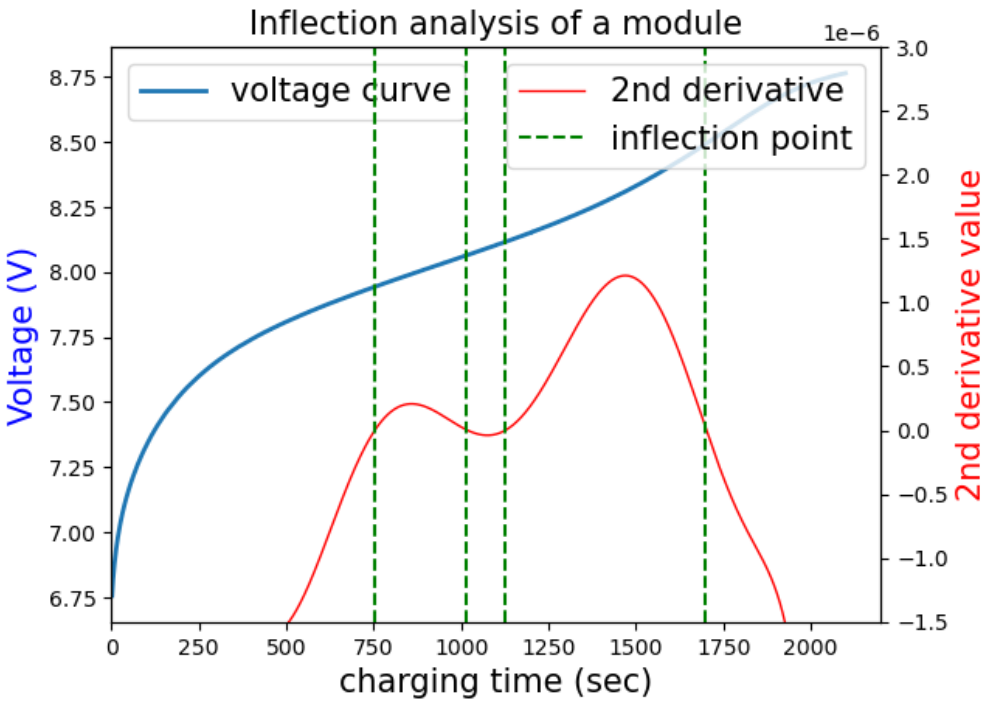


圖 3-10 樣本之反曲點示意圖

doi:10.6342/NTU202400743

為了驗證四次擬合曲線的有效性，我們對同一條電壓曲線，以 3~6 次多項式進行擬合，結果如下圖 3-11，圖 3-11 - (b) 是用 4 次多項式擬合的結果，正好出現兩個反曲點，即圖片上虛線、圖標為 inflection point 的兩個點，且無論是擬合曲線還是反曲點位置都對應到原始曲線開始加速和減速的點，我們從中選取的一個點為想要的反曲點，圖標為 wanted inflection point，以綠色實心點標示在粉紅色的二次微分曲線上。圖 3-11 - (a) 3 次擬合曲線只出現一個點，，且該點位置和原始曲線開始加速的點還大許多，3-(c)、3-(d)與 4 次擬合曲線產生的點位置相當，但是出現多餘的點，恐怕有過度擬合的問題，因此最終決定用 4 次多項式來擬合原始電壓曲線。

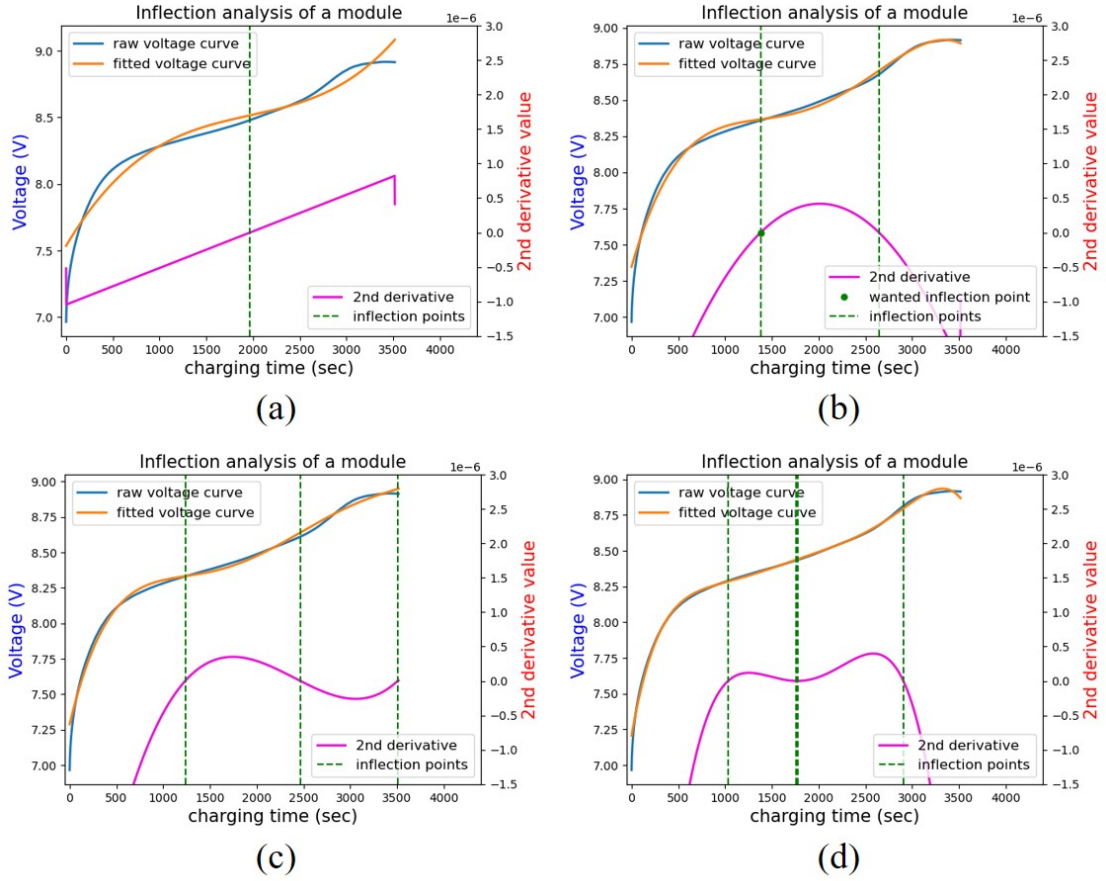
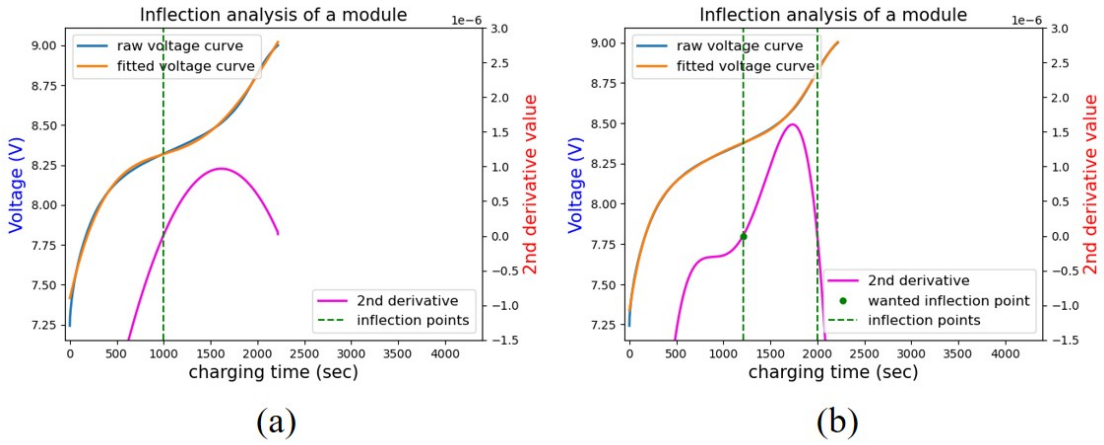


圖 3-11 不同冪次的多項式的擬合曲線，以及產生反曲點的數量 (a) 3 次多項式 (b) 4 次多項式、(c) 5 次多項式、(d) 6 次多項式

doi:10.6342/NTU202400743

如果沒有在 4 次多項式下產生兩個反曲點，我們會將該模組視為有問題的模組並排除它，如下圖 3-12。該模組為高速老化實驗中的一筆充電曲線數據，在 4 次多項式擬合下只產生一個反曲點，本該有的第二個反曲點沒有出現，改用 6 次多項式才擬合出兩個反曲點。究其原因，從它的充電曲線中發現電壓還在上升，並沒有出現 PVD 的條件，因此判斷該電池並未充飽，不應該納入分析。發現並分析反例出現的原因，再次說明使用 4 次微分曲線的合理性。



(a) (b)

圖 3-12 充電過程出現瑕疵的模組的多項式擬合結果 (a) 4 次多項式、(b) 6 次多項式

上述步驟以下列流程圖 3-13 表示:

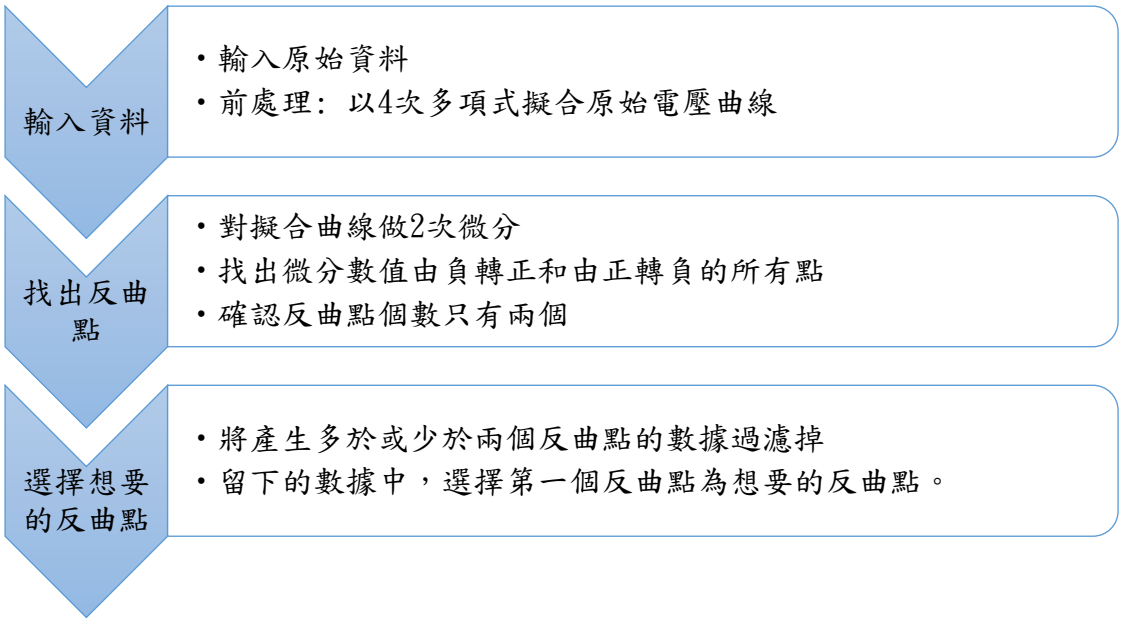


圖 3-13 決定反曲點的工作流程

doi:10.6342/NTU202400743

3.4 弛豫現象

根據文獻，觀察弛豫時間可以衡量電池性能。在充放電程序中有兩段休息時間，3 分鐘和 10 分鐘，我們以兩段時間內的電壓變化代表弛豫現象，數值越大表示平衡電位越低，即電池性能較差。充放電機在切換工作狀態後會將每一道程序的進行時間從 0 開始計算，因此每一階段的時間段落為該階段第 0 秒到下一階段第 0 秒的前一刻，但是考量到電池斷電那一刻會產生歐姆極化，我們改以休息階段開始的前一秒的電壓，和休息階段最後一秒的電壓相減，所得之值當作弛豫現象。詳細流程如下圖 3-14 所示:

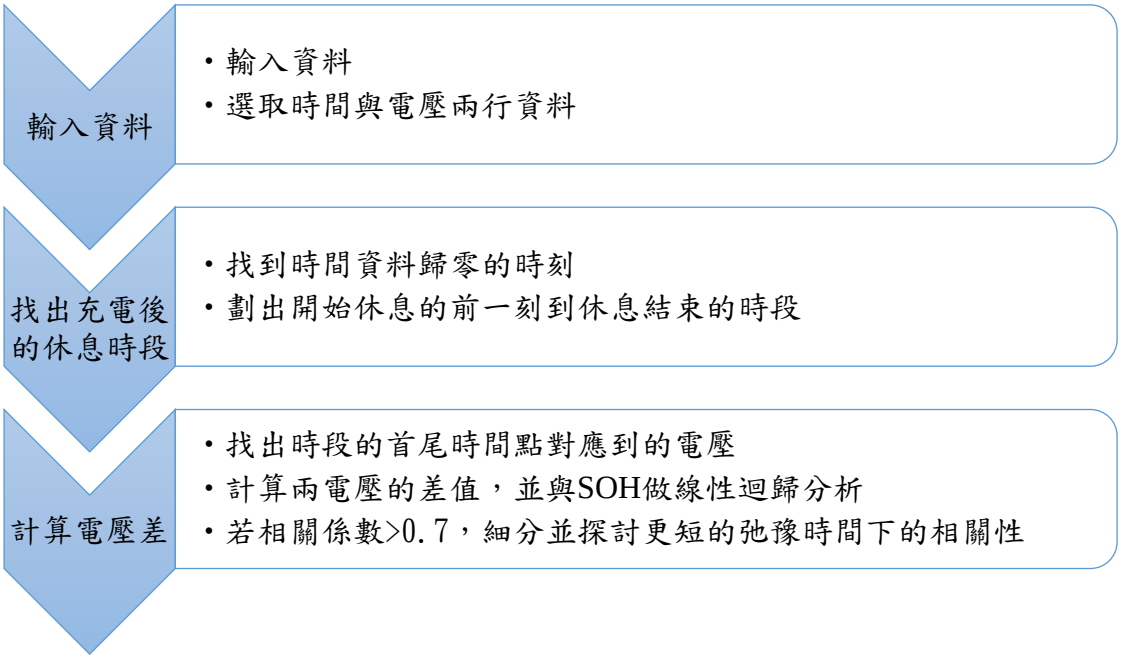


圖 3-14 弛豫行為工作流程

如果發現弛豫現象和 SOH 的相關係數大於 0.7，我們將探討更短的弛豫時間內的成效。我們將時間分為:斷電後 1 秒內、10 秒內、30 秒內、1 分鐘、3 分鐘、5 分鐘、7 分鐘，和 10 分鐘，第一次的休息時間最長到 3 分鐘，所以只細分到 3 分鐘。將每一段時間內起始點和終點的電壓差代表弛豫現象，衡量他們的相關性後再做篩選。

doi:10.6342/NTU202400743

3.5 其他特徵變數

我們將皓恆公司取回電池模組時固定進行量測的變數一同納入分析，見表3-4。這些變數是在充放電程序進行時一併量測的，大多是某一時刻的電壓，以其大小推測電池的極化程度進而判斷模組衰退程度；此外，根據判斷模組故障的標準，即每兩個模組串聯後的電壓和相鄰的串聯模組電壓相差 1V 以上，我們將統一電池的電量狀態所需的放電時間 DIS0_Time 當作變數之一，因為我們認為 1V 的差異表示模組容量之間出現嚴重落差，而 DCD35 中第一次的放電時間是把模組的剩餘容量歸零的過程，所以其放電時間相當於模組的剩餘容量，反映故障時模組的狀態，可以視為電池性能的一部份。

表 3-4 皓恆公司取得之電壓變數

變數	含意
Charging time	故障模組在 DCD35 中的充電時間
DIS0_Time	故障模組在 DCD35 中的第一次放電時間
V15	故障模組充電 15 分鐘時的電壓
V20	故障模組充電 20 分鐘時的電壓
V25	故障模組充電 25 分鐘時的電壓
V30	故障模組充電 30 分鐘時的電壓
V35	故障模組充電 35 分鐘時的電壓
VPVD	故障模組充電時的最高電壓
VEOC	故障模組充電結束時的電壓

從 DCD35 取得特徵變數後進行數據清理(data cleaning)，並將沒量測到數值或數值不合理的模組剔除，剃除項目有:

1. 任何一個變數沒被量測到
2. 放電時間大於充電時間
3. 弛豫行為中前後電壓變化>0
4. 充電時間 ≥ 2219 秒 (即部分充放電、沒有 SOH 的模組)

doi:10.6342/NTU202400743

清理數據後，以 spearman 相關係數計算各個變數對 SOH 的相關性，根據統計學上區分相關性高低的經驗，將相關性(R)的絕對值高於 0.7 的變數留下。該程式碼放在 A.4 。

3.3~3.5 節提取特徵的程式碼，放在 A.1、A.2，在讀取數據時就提取特徵，圖 3-12 的程式碼被獨立出來放在 A.3 。

doi:10.6342/NTU202400743

3.6 建立機器學習模型

本研究使用 windows 10 的桌機，裝備 Intel i5 CPU 和 16 GB RAM，在 Python 3.8 上執行機器學習。其中 BRR 和 RF 使用公開套件scikit-learn 中的 linear_model. BayesianRidge 和 RandomForestRegressor 做訓練，而 FNN 則使用nn. sequential，程式碼置於 A.5、A.6。三種模型之示意見圖 3-13。

建立模型後，我們針對幾個常見的、對於預測結果影響較大的超參數(hyper-parameter，由人為設置的模型參數)做網格搜尋(gridsearch)，搭配 5 折交叉驗證法(5-fold cross-validation, 5-fold CV)降低特定的切割數據的方式造成的偏差。首先將清理過的數據，以 8:2 的比例分成訓練組和測試組，訓練組共 17120 筆數據，測試組共 4281 筆數據，用訓練組進行 gridsearch 和 CV，找出平均 RMSE 最低的超參數組合，再用測試組做測試。BRR 和 RF 用GridSearchCV套件尋找最佳參數，FNN 則是自行編碼執行網格搜尋。

對 BRR 而言，控制機率分布的超參數對模型影響巨大，因此誤差分布(alpha_1, alpha_2, 以下通稱 alpha) 和能控制分布寬窄的正則化係數(lambda_1, lambda_2, 以下通稱 lambda) 成為搜尋的對象。對 RF 而言，主要受到樹木多寡($n\_estimators$)、每棵樹的每個節點分割時可使用的特徵數量(max_features)、樹分類的次數(樹的深度)的影響。出於盡可能地擬合數據、以及認可集成學習降低噪音的能力，我們不改變其他的關於樹的結構的超參數，僅調整樹木多寡和max_features。FNN 有隱藏層數量(layers)、每層的神經元個數(neurons)、學習率(learning rate, lr)和正則化係數(lambda)，共計四個超參數。其中 layers 和 neurons 控制模型的複雜度，lr 控制模型更新權重的幅度，會影響模型訓練的快慢，以及是否能接近全局最小值(global minimum)，Lambda 是正則化項，防止過度擬合。Layers 設為兩層，Lambda 採取預設值 0.0001，並從學習率中衍生出兩個超參數: factor 和 patience，讓學習率也根據誤差大小進行調整，讓誤差能持續降低、快速收斂。Factor 是降低 lr 的倍率，在 0~1 之間；patience 表示能容

doi:10.6342/NTU202400743

許多少次迭代時誤差沒有降低，此值大於 0，例如 factor = 0.5、patience = 10 表示若連續迭代 10 次內誤差沒有降低，則將當次迭代時的學習率乘以 0.5，若在截止前觸發了 3 次，則結束時的學習率只會是原來的 1/8 倍 (0.5*0.5*0.5)。

三個模型中納入網格搜尋的超參數，以及該超參數的數值列於下表 3-5、表 3-6、表 3-7，數值由套件的預設值為基準，並參考文獻中提及的數值設立網格搜尋的參數。BRR 中 alpha 和 lambda 的預設值皆為$1 * 10^{-6}$，將此設為基準並乘以 10 和 100 倍得到搜尋範圍；RF 的 $n\_estimators$ 和 max $\_features$ 預設值分別為 100 和 1，$n\_estimators$ 以首項為 100，公差為 50，末項為 2000 的數列、max $\_features$ 則根據數據清理後得到的特徵數目，安排最少一個、最多全部的特徵用來訓練；FNN 共 5 個超參數: H1 (第一個隱藏層的神經元數目)、H2 (第二個隱藏層的神經元數目)、lr_init (初始學習率)、factor 和 patience。H1 和 H2 以 100 的因數: 10, 25, 50, 100 構成，且 H1 被刻意設計的比 H2 要大，以避免過度擬合；lr_init 以 0.001 為基準，分別乘、除 10 倍得到搜尋範圍的上、下界，再插入 0.0003、0.002、0.003，探索下界到基準中間(0.0003)、基準值周圍(0.002)，和基準到上界中間(0.003)，盡可能地找出最佳組合。

表 3-5 BRR 的 gridsearch 範圍

n_iter	alpha_1	alpha_2	lambda_1	lambda_2
100	$1 * 10^{-6}$	$1 * 10^{-4}$	$1 * 10^{-4}$	$1 * 10^{-6}$
200	$1 * 10^{-5}$	$1 * 10^{-5}$	$1 * 10^{-5}$	$1 * 10^{-5}$
300	$1 * 10^{-4}$	$1 * 10^{-6}$	$1 * 10^{-6}$	$1 * 10^{-4}$

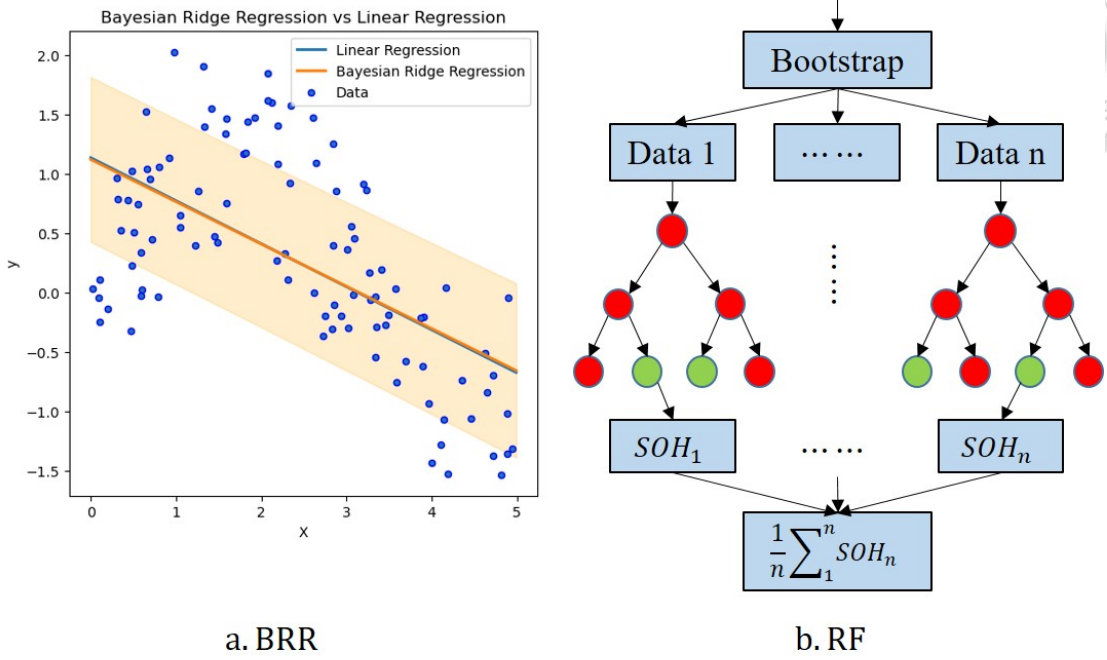
表 3-6 RF 的 gridsearch 範圍

$n\_estimators$	100, 150, 200…2000
max $\_features$	1 ~ 數據清理後剩下的所有特徵

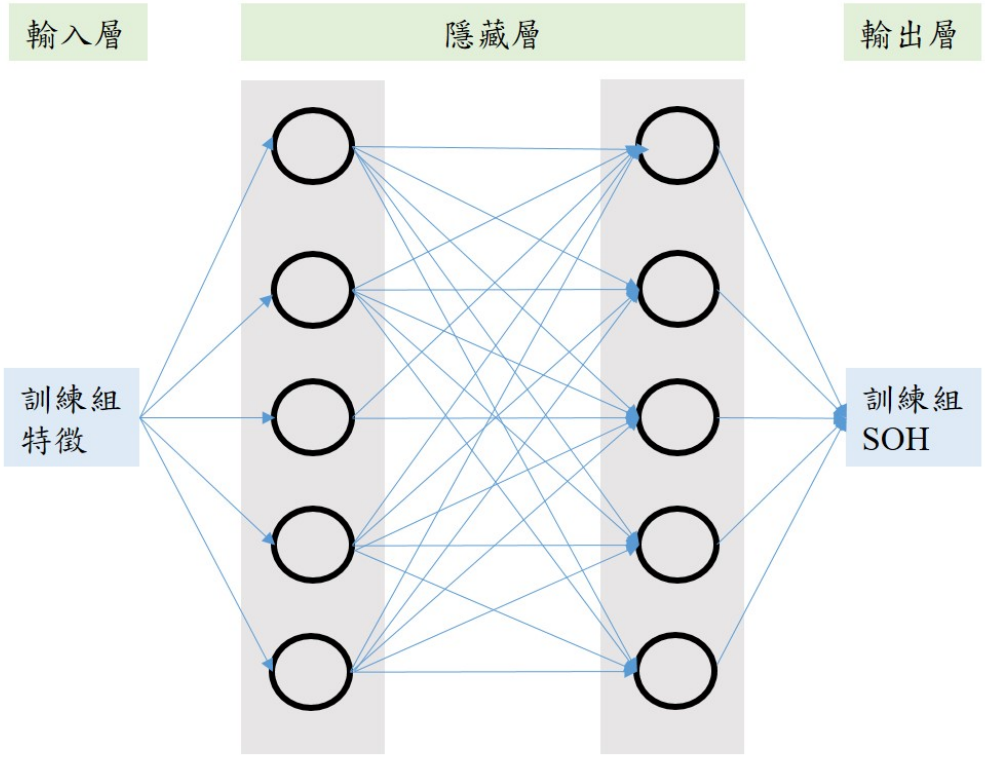
表 3-7 FNN 的 gridsearch 範圍

H1	H2	lr_initt	factor	patience
25	10	0.0001	0.2	5
50	25	0.0003	0.5	10
100	50	0.001		20
		0.002		
		0.003		
		0.01		

doi:10.6342/NTU202400743



a. BRR　　b. RF



c. FNN

圖 3-15 選擇的三種機器學習模型: a. BRR、b. RF、c. FNN

doi:10.6342/NTU202400743

3.7 衡量機器學習模型成效

機器學習模型的成效，取決於預測值和真實值之間的差異(殘差)，以及預測的不確定性(uncertainty)，前者將殘差量化成損失函數(cost function)，數值越低者模型預測能力越優秀，後者會觀察殘差的分布狀況，判斷模型產生偏差值的頻率和大小，若出現極大或較多的偏差代表模型預測能力不穩、較不可靠。在此研究中使用 5 個度量來衡量模型成效。

首先我們考慮模型是否具有有效的預測能力，用的是決定係數，R^2:

1. R^2(coefficient of determination): R^2表示模型的預測能力，越高越好。此方式將模型產生之殘差和目標之平均值做比較，如果模型預測能力越好，產生之殘差一定遠小於猜平均值下產生之誤差，計算後R^2會越接近 1

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(SOH_{pred,i} - SOH_{real,i})^2}{\sum_{i=1}^{n}(SOH_{mean,i} - SOH_{real,i})^2} \tag{17}$$

損失函數以下列兩者做代表:

2. MAE (mean absolute error): MAE 代表模型預測時出現的殘差的平均值，越小越好，此方式將每一個資料點之殘差取絕對值後做平均。

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|SOH_{pred,i} - SOH_{real,i}| \tag{18}$$

3. RMSE (root mean square error): RMSE 與 MAE 類似，但是放大了離群值(outliers)的影響力，越小越好，此方式將殘差做平方和後再開根號，經過平方處理後離群值將大幅上升，影響最後的計算結果，一般情況下 RMSE≧MAE

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(SOH_{pred,i} - SOH_{real,i})^2} \tag{19}$$

doi:10.6342/NTU202400743

不確定性則用以下兩個度量:

4. Confidence interval: 我們對殘差進行統計分析，計算殘差之標準差並劃出信賴區間，以區間之大小衡量模型的精度和穩定性，越小越好。首先將計算殘差之平均值和標準差，再以平均值為中心增加、減少兩個標準差，畫出信賴區間。在此區間下將涵蓋 95 % 離平均值最近的殘差
5. Percentage errors (PE): 殘差的百分比誤差的標準差(standard deviation of percentage error, PE)衡量模型的精度和穩定性，越小越好。此方式考將各個資料點的殘差除以其真實值，計算偏離真實值的幅度。與第一到第三個損失函數計算殘差之絕對值不同，此方式想要確保模型對不同的資料點能夠保持一致地預測能力，不因為資料點之間存在的差異影響預測結果。

$$PE(\%) = \frac{SOH_{pred,i} - SOH_{real,i}}{SOH_{real,i}} \tag{19}$$

以上的五個衡量模型成效的程式碼中，損失函數的計算已被包含在 ML 程式碼中，作圖的代碼以及五個度量的計算則在 A.7。

.

doi:10.6342/NTU202400743

3.8 遷移學習

在取得老化數據後，我們將其和服役數據結合，使用度量學習法讓特徵維度增加、縮小不同域的 MMD，流程如下圖 3-16，程式碼在 A.8。首先，由於服役數據規模遠大於老化實驗，且存在嚴重的機器誤差，因此縮緊 PVD 的條件篩選服役數據以降低預測誤差，才將所有數據合併、做正規化。接著以 8:2 比例分成訓練組和測試組，將訓練組分為高溫老化數據(1C Aging)、高速老化數據(3C Aging)和服役數據(On-road)，然後讓各類數據通過一個高斯徑向基函數(Gaussian radial basis function, RBF) 映射到不同維度的空間，RBF(φ)式子如下:

$$\varphi(x^S, x^T) = e^{(-\|x^S - x^T\|^2 / 2\gamma^2)} \tag{20}$$

該 RBF 的大小為γ、映射後的特徵維度為m，接著計算兩兩之間的 MMD，包含MMD_1、MMD_2、MMD_3，將其加總後成為MMD_{ALL}，找出讓MMD_{ALL}最小的一組γ、m，使用該組特徵建立一個神經網路模型，最後由測試組透過同樣γ、m 更換特徵後測試該模型，將結果和不用 MMD 時的預測結果做對照。

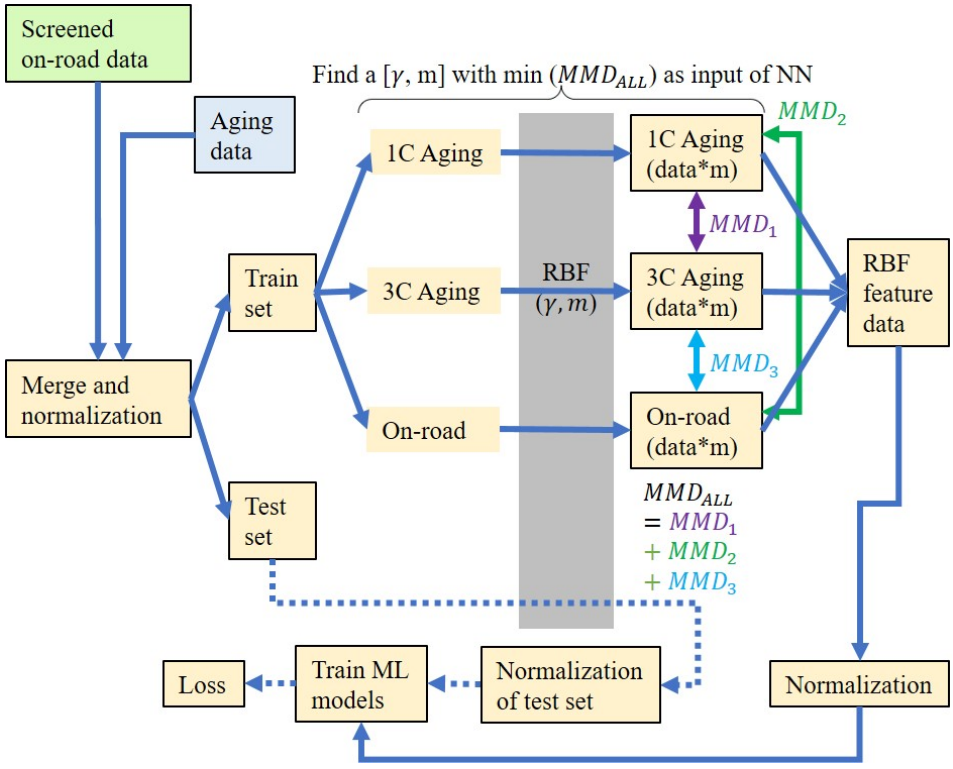


圖 3-16 進行度量學習法的流程圖

doi:10.6342/NTU202400743

3.9 作業流程

整體作業流程如下圖 3-17，分為三大塊: 首先是率先利用從皓恆取得的服役後模組，從這些模組的充放電數據中提取特徵、建立模型、預測 SOH。為了驗證提取得特徵具有泛化性能，我們進行了老化實驗，獲取鎳氫電池在高 SOH 時的充放電資料，從中提取了和服役數據中一樣的特徵。待老化實驗結束後，將老化數據和部分服役數據結合，使用 MMD 改變特徵，再建立另一個 FNN 預測 SOH，試圖讓誤差最小化、提高預測能力。

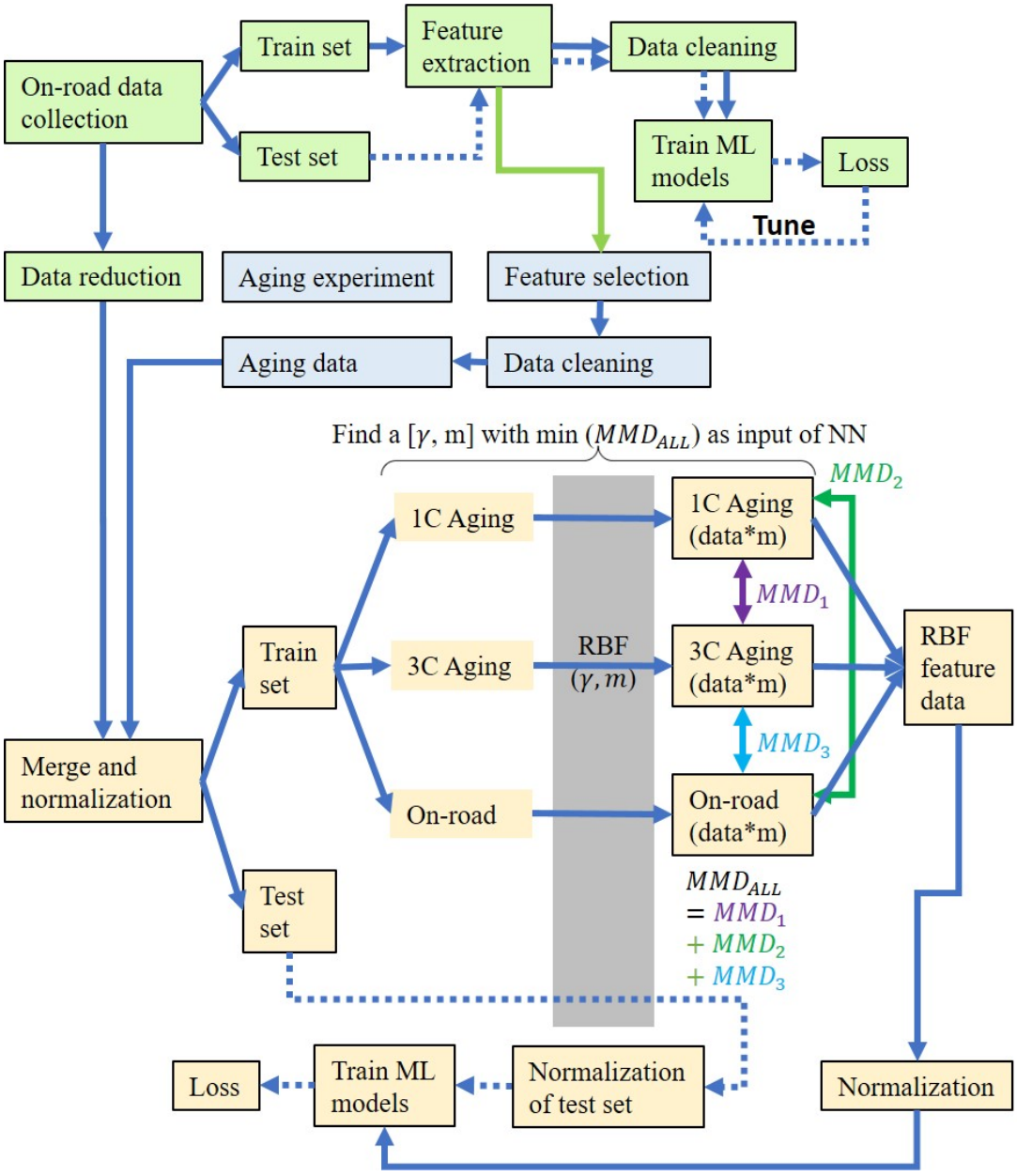


圖 3-17 研究流程示意圖

doi:10.6342/NTU202400743

第四章 結果

4.1 特徵工程

所有特徵變數對 SOH 相關性的結果呈現如下表 4-1

表 4-1 所有變數的有效樣本數及其和 SOH 之相關性

特徵變數	合格模組數目	剔除模組數目	相關係數(R)
反曲點(ift)	22541	4523	0.955
弛豫行為(充電前)	26932	79	0.686
弛豫行為(充電後)	26932	132	0.879
Charging time	26988	76	0.988
DIS0_Time	26988	76	0.967
V15	25908	1156	- 0.966
V20	23203	3861	- 0.978
V25	18631	8433	- 0.958
V30	12009	15055	- 0.791
V35	4591	22473	- 0.328
VPVD	26988	76	- 0.861
VEOC	26988	76	- 0.858

由上表可知，11 個變數中僅有兩個變數，V35 和弛豫行為(充電前)的相關性不滿 0.7，故將兩者剔除。在特定時間下的電壓中，V20 是相關性最高的特徵變數，但是被剔除的樣本高達四千多個，多數是充電時間不滿 20 分鐘而無數據。我們認為樣本過少不利於訓練通用的模型，因此改採相關性略低但剔除的樣本數較少的 V15，並剔除 V20、V25 和 V30。最後，VPVD 和 VEOC 意思相近，而 VPVD 的相關係數略勝於 VEOC，所以剔除 VEOC。弛豫行為(充電後)的相關性高於 0.7，因此我們針對弛豫行為(充電後)再做細分，衡量不同弛豫時間對電池性能的關係，結果如下表 4-2

doi:10.6342/NTU202400743

表 4-2 不同時間尺度下，充電後的弛豫行為預測 SOH 能力之比較

特徵變數	相關係數(R)	代表的極化種類
弛豫行為(1 sec)	0.556	歐姆極化
弛豫行為(10 sec)	0.877	歐姆極化和電化學極化
弛豫行為(30 sec)	0.884	歐姆極化、電化學極化和濃差極化
弛豫行為(1 min)	0.892	濃差極化
弛豫行為(3 min)	0.895	濃差極化
弛豫行為(5 min)	0.882	電壓收斂，極化現象不再主導
弛豫行為(7 min)	0.881	電壓收斂，極化現象不再主導
弛豫行為(10 min)	0.879	電壓收斂，極化現象不再主導

當休息時間到達 3 分鐘時，相關性到達極大值，更長的時間相關性逐漸走低。更長的休息時間會讓電池更穩定，電壓仍會持續降低，但降低幅度已趨緩並逐漸收斂，此時除了濃差極化外應包含其他電化學反應使電壓下降，例如氣體擴散、電池降溫等等，不屬於電池性能的一部份，因此我們就選取 3 分鐘內的弛豫現象作為電池健康狀態的特徵變數。綜上所述，我們最終篩選出 6 個訓練模型的特徵，共有:第一次的放電時長(Dis0_Time)、充電時間(Chr_Time)、充電時期第 15 分鐘之電壓(V15min)、充電曲線上之反曲點 (inflection point)、充電時期出現之最高電壓(VPVD)、充電結 3 分鐘內之弛豫行為(3min relax)。數據總數為 21401 筆，其和 SOH 之相關性如圖 4-1 所示

doi:10.6342/NTU202400743

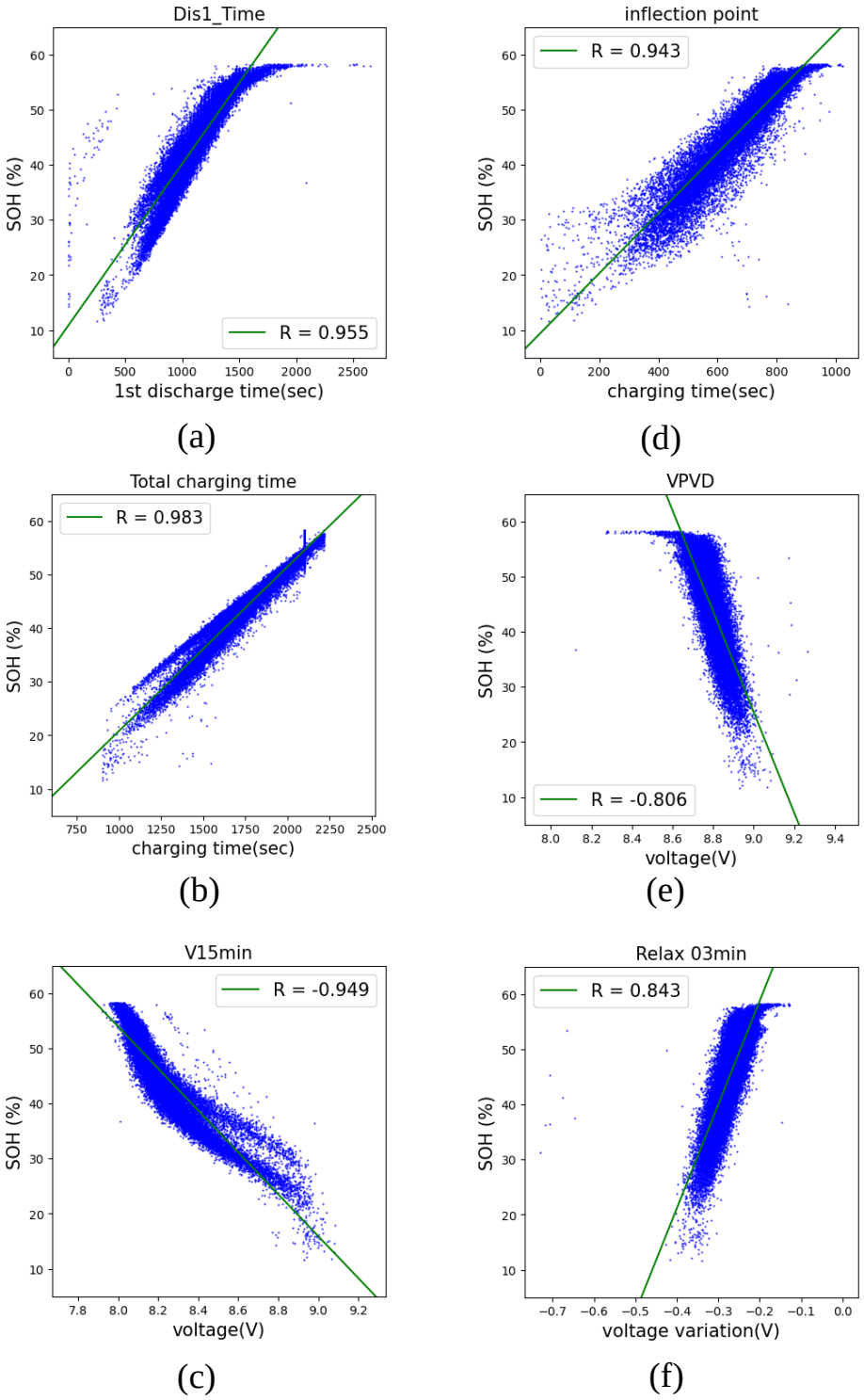


圖 4-1 六種特徵對 SOH 之關係圖。(a) 前處理的放電時間 (Dis1_Time) 、(b) 充電時間、(c) 充電階段第 15 分鐘之電壓(V15min) 、(d) 充電曲線上之反曲點 (inflection point) 、(e) 充電時期出現之最高電壓 (VPVD)、(f) 充電結束 3 分鐘內之弛豫行為 (3min relax) 。

doi:10.6342/NTU202400743

4.2 機器學習模型

4.2.1 網格搜尋的結果

數據以 8:2 的比例劃分為訓練組和測試組，分別有 17120 筆和 4821 筆數據，三個模型經過 gridsearch 後找到的最佳組合列於表 4-3、表 4-4、表 4-5，在表中將最佳化組合改為粗體黑字並用該組合建立模型，全部的網格化搜尋結果放在附錄 A9、A10。

表 4-3 中，BRR 無論是哪一組參數，都產生一樣的結果，看似是模型產生缺陷，但更可能的原因是擬合結果和實際值相似，且數據量龐大，跟 RF 和 FNN 相比雖有較大的誤差和離群值，多數預測值是和實際值相似，並且相當集中，如圖 4-2 (a)顯示的一樣，因此調整各項超參數幾乎不會改變預測結果，因此決定以預設值建立 BRR 模型。

表 4-3 BRR 的 gridsearch 結果

n_iter	alpha_1	alpha_2	lambda_1	lambda_2	CV loss (%)	Test loss (%)
100	$1*10^{-6}$	$1*10^{-4}$	$1*10^{-4}$	$1*10^{-6}$	1.635	1.654
200	$1*10^{-5}$	$1*10^{-5}$	$1*10^{-5}$	$1*10^{-5}$		
300	$1*10^{-4}$	$1*10^{-6}$	$1*10^{-6}$	$1*10^{-4}$		

表 4-4 中顯示 RF 的每個 max$_features$ 中，讓交叉驗證的誤差(CV loss)最低的$n_estimators$，及用該參數組合進行預測後得到的誤差 (Test loss)。max$_features = 2$, n_estimators $= 1950$ 時效果最佳，誤差僅 1.34 %，不過各個組合之間差距不大，最大差距僅 0.06 %，只要建構上千個決策樹的 RF 就足以做出可靠且強韌的預測。

表 4-4 RF 的 gridsearch 結果

max$_features$	$n\_estimators$	CV loss (%)	Test loss (%)
1	1800	1.443	1.405
2	**1950**	**1.424**	**1.34**
3	2000	1.427	1.393
4	1600	1.435	1.397
5	1050	1.442	1.399
6	1250	1.451	1.404

doi:10.6342/NTU202400743

表 4-5 中，共計 6 個超參數組合出現最小值，它們的 H1、H2、lr_init 一樣，factor 和 patience 則涵蓋了預設的全部的搜尋範圍，推測可能是因為採用交叉驗證的方式，消去了數據差異造成的波動，因此我們測試了這 6 組數據，以全部的訓練組資料 (17120 筆資料) 調整每一個組合建構的 FNN 模型，然後用測試組 (4281 筆資料) 測試該組合之 FNN，將最後一個 epoch 的誤差 (train loss) 和測試組的誤差 (test loss) 記於下表 4-6，最低誤差為 1.384 %，對應到的參數組合 [H1, H2, lr_init, factor, patience] = [100, 50, 0.002, 0.2, 20]

表 4-5 FNN 的 gridsearch 結果

H1	H2	lr_init	factor	patience	CV loss (%)	Test loss (%)	Last lr
100	50	0.002	0.5	5	1.4073	1.4256	0.002
100	50	0.002	0.5	10	1.4073	1.4256	0.002
100	50	0.002	0.5	20	1.4073	1.4256	0.002
100	50	0.002	0.2	5	1.4073	1.4256	0.002
100	50	0.002	0.2	10	1.4073	1.4256	0.002
100	50	0.002	0.2	20	1.4073	1.4256	0.002

表 4-6 不同的 factor 和 patience 產生的誤差，以及最佳的 FNN 的超參數組合

factor	patience	train loss (%)	test loss (%)
0.2	5	1.451	1.436
	10	1.419	1.407
	20	**1.395**	**1.384**
0.5	5	1.648	1.67
	10	1.416	1.397
	20	1.397	1.385

總結 gridsearch 的結果，BRR 並未因為超參數不同而有不同的測試結果，因此就用預設值做測試；RF 的最佳超參數組合為$\max_features = 2$, $n\_estimators = 1950$；FNN 的最佳組合為[H1, H2, lr_init, factor, patience] = [100, 50, 0.002, 0.2, 20]。我們就用這些超參數組合建立模型，進行預測。

doi:10.6342/NTU202400743

4.2.2 模型成效

測試組的預測結果如下圖 4-2 所示，擬合圖中除了有預測值和實際值的擬合結果，還有以 SOH 預測值(SOH_p)為中心，增加、減少兩個殘差標準差(σ)，畫出的一個綠色的範圍，圖標為 $SOH_p \pm 2\sigma$。在綠色範圍內的點表示殘差比標準差要低，預測較精準，反之綠色範圍外的點表示預測失準，方便我們觀察模型對不同 SOH 的數據的預測能力如何，間接判斷該模型的預測能力，甚至是模型特色。

FNN 和 RF 訓練成效，無論以何種方式衡量模型成效，皆比 BRR 還要高，且 BRR 的擬合圖中低 SOH 的區域資料點出現分散、超出綠色範圍很多的情況，而 FNN 和 RF 沒有。這個現象跟採用的模型有關，BRR 是由一維線性模型演變而來的，只是加入了貝葉斯統計和高斯機率分布的概念，能夠補足資料點不足產生偏差的情況，在數據充足的情況下 BRR 更接近於一般的線性回歸模型，其輸出受到特徵影響，而充電時長是和 SOH 最相關的特徵 ($R = 0.983$)，因此會對 BRR 的輸出有最大的影響力，其和 SOH 的擬合關係也顯現在 BRR 上，在低 SOH 的地方出現資料點分散的情況。相較之下，FNN 和 RF 不屬於線性模型，特徵和 SOH 的個別關係訓練模型時不會直接影響結果，事實也是如此，兩圖形外觀相當一致，RF 的 RMSE 大於 FNN 但僅差距 0.004 %，MAE 只比 FNN 低 0.007 %，說明 RF 和 FNN 在損失函數上的表現相當良好且相近。

殘差分布圖及殘差百分比誤差(PE)如下圖 4-3、圖 4-4 所示，觀察重點在於信賴區間之大小和分布圖形。三張圖片的分布狀況皆類似正態分布，但 BRR 圖形左側出現微微隆起，相較於 FNN 和 RF 的對稱分布有明顯差距，信賴區間顯著地高於 FNN 和 RF。另一方面，FNN 的信賴區間比 RF 小 ($5.43 < 5.447$)，FNN 的離群值略小於 RF，顯示殘差分布比 RF 更集中，可以映證圖 4-2 中損失函數比 RF 低的事實。不過 FNN 的 PE 比 RF 要大一些，顯示出 FNN 還是有發生過擬合的可能。

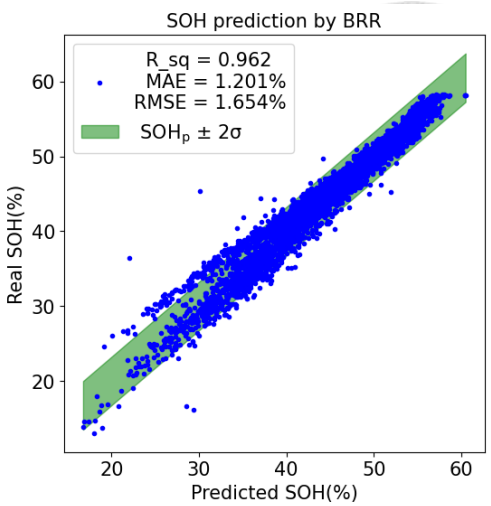
doi:10.6342/NTU202400743

Code package:

- linear_model. BayesianRidge

Hyper-parameters:

- Alpha_1: $1 * 10^{-6}$
- Alpha_2: $1 * 10^{-6}$
- Lambda_1: $1 * 10^{-6}$
- Lambda_2: $1 * 10^{-6}$



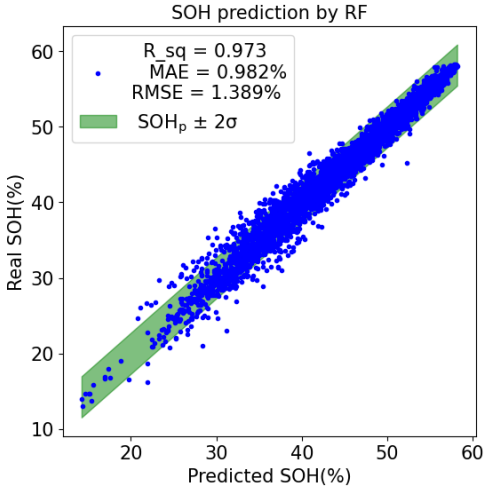
(a)

Code package:

- RandomForestRegressor

Hyper-parameters:

- The number of the decision tree: 1950
- Maximum features used in each nodes: 2
- Random state: 100



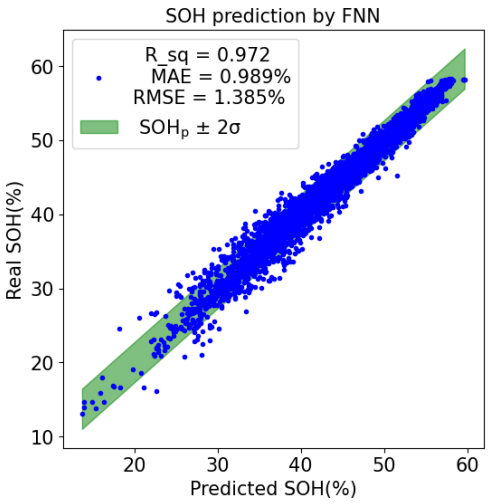
(b)

Code package:

- nn. sequential
- Adam optimizer

Hyper-parameters:

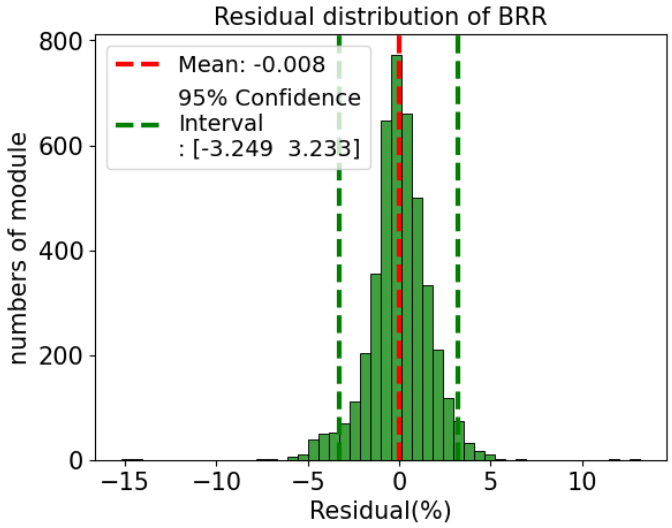
- Hidden layers: 2
- Neurons: (100, 50)
- lr＝0.002 (factor = 0.2, patience = 20)
- Lambda＝0.0001
- Epoch = 1000



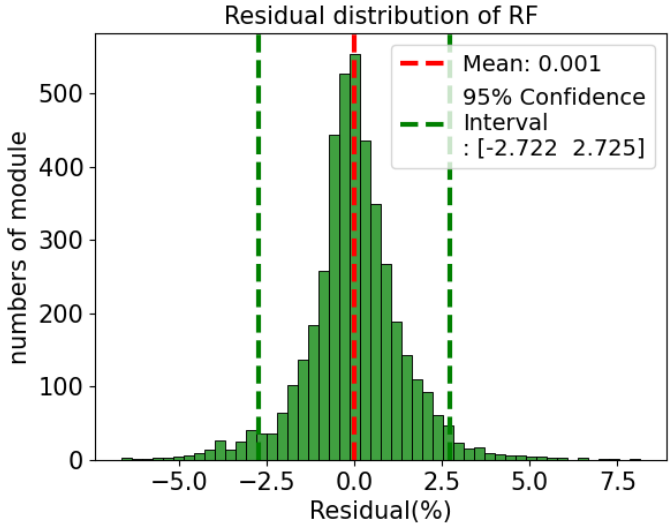
(c)

圖 4-2 三種模型預測 SOH 的結果 (a) BRR、(b) RF、(c) FNN

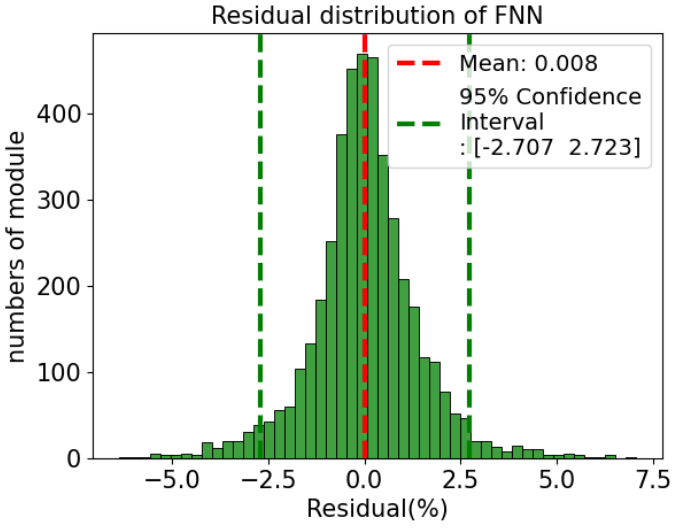
doi:10.6342/NTU202400743



(a)



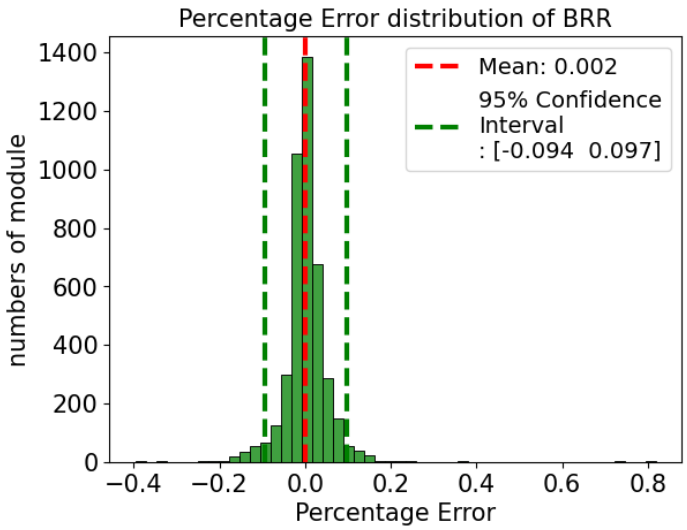
(b)



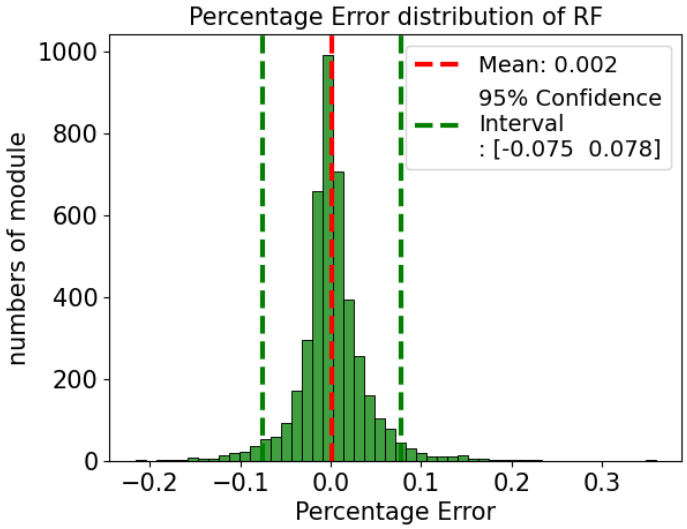
(c)

圖 4-3 三種模型的殘差分布圖 (a) BRR、(b) RF、(c) FNN

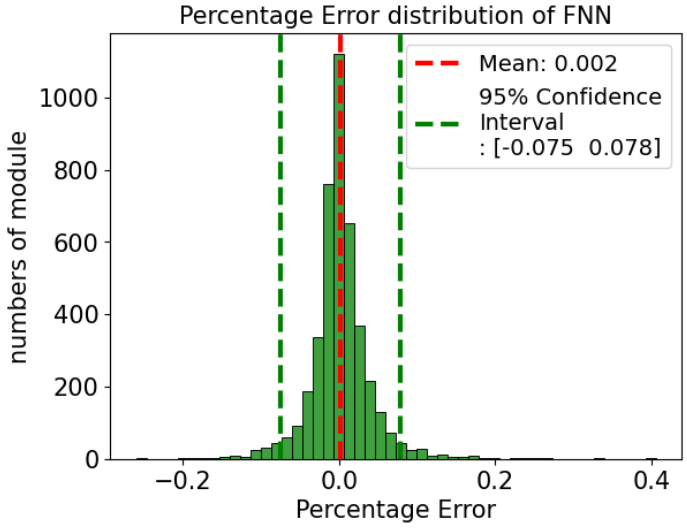
doi:10.6342/NTU202400743



(a)



(b)



(c)

圖 4-4 三模型的殘差百分比(PE)分布圖 (a) BRR、(b) RF、(c) FNN

doi:10.6342/NTU202400743

4.3 特徵重要性

在 BRR 中，我們知道特定的特徵會對模型效能有很大的影響力，在線性迴歸模型中可能可以較輕易地判斷每個特徵的對模型的影響力，但如果是在非線性模型中就不大容易看出。以本研究來說，非線性模型的 RF 和 FNN 預測效果明顯較好，所以知道哪個特徵對非線性模型影響較大很重要，替我們找出重要的特徵。我們建立出了一套判斷特徵重要性的方法，方式為從 6 個特徵中刪除任一個特徵，用 4.2.1 節找出的最佳超參數組合和同樣的訓練組和測試組，測試模型效能，以損失函數和不確性度量的變化幅度，綜合評估特徵的重要性。每個 feature 對 loss 和 uncertainty 的影響如表 4-7、表 4-8、表 4-9 所示:

表 4-7 在 BRR 中移除不同特徵後，模型的損失提高的幅度

Features be removed.	Coefficient of determination (r2)	MAE (%)	RMSE (%)	Standard deviation of residuals	Standard deviation of PE
充電時間	0.941	1.444	2.057	2.057	0.057
3 分鐘弛豫現象	0.961	1.229	1.667	1.667	0.049
反曲點	0.962	1.206	1.652	1.652	0.049
充電第 15 分鐘電壓	0.962	1.216	1.653	1.653	0.049
最高電壓	0.96	1.241	1.692	1.692	0.05
前處理的放電時間	0.962	1.208	1.653	1.653	0.049

doi:10.6342/NTU202400743

在 RF 中，每個 feature 對 loss 和 uncertainty 的影響如下表所示:

表 4-8 在 RF 中移除不同特徵後，模型的損失提高的幅度

Features be removed.	Coefficient of determination (r2)	MAE (%)	RMSE (%)	Standard deviation of residuals	Standard deviation of PE
充電時間	0.97	1.065	1.472	1.472	0.041
3 分鐘弛豫現象	0.969	1.049	1.474	1.474	0.041
反曲點	0.969	1.046	1.475	1.475	0.042
充電第 15 分鐘電壓	0.972	0.999	1.412	1.412	0.04
最高電壓	0.972	0.991	1.404	1.404	0.039
前處理的放電時間	0.972	1.002	1.419	1.419	0.04

在 FNN 中，每個 feature 對 loss 和 uncertainty 的影響如下表所示:

表 4-9 在 FNN 中移除不同特徵後，模型的損失提高的幅度

Features be removed.	Coefficient of determination (r2)	MAE (%)	RMSE (%)	Standard deviation of residuals	Standard deviation of PE
充電時間	0.967	1.093	1.515	1.515	0.043
3 分鐘弛豫現象	0.97	1.054	1.441	1.441	0.041
反曲點	0.967	1.099	1.515	1.515	0.043
充電第 15 分鐘電壓	0.97	1.051	1.44	1.44	0.041
最高電壓	0.97	1.019	1.441	1.441	0.043
前處理的放電時間	0.971	1.013	1.415	1.415	0.04

doi:10.6342/NTU202400743

上述表格圖像化如下:

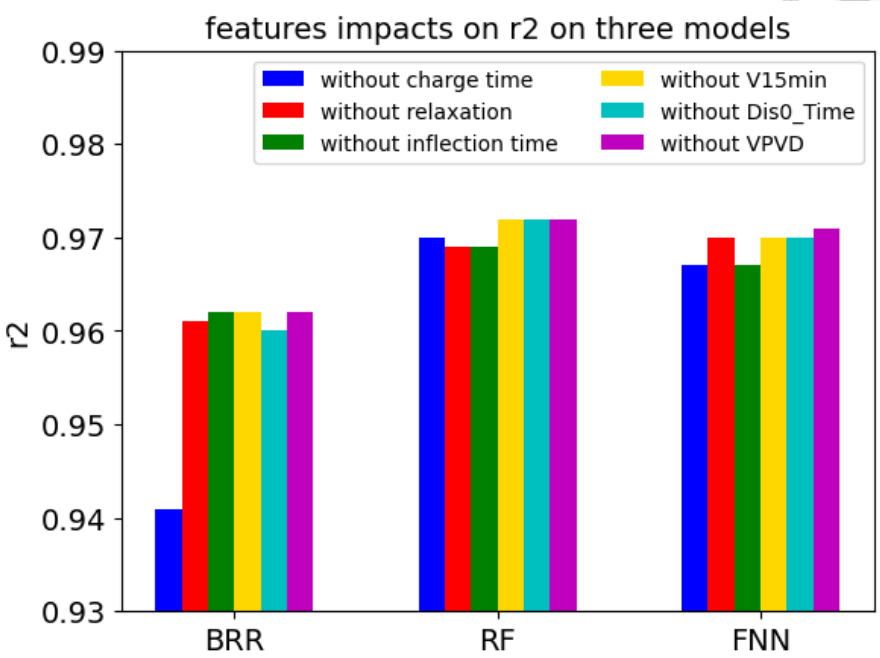


圖 4-5 移除不同特徵後剩下的 r2 大小

r2 表示模型的預測能力，越高越好，若在缺少了一個特徵下 r2 提升地越高，表示該特徵對模型的預測能力影響越弱，反之越強。圖中看到最能影響 BRR 的特徵是充電時長(without charge time)，最弱特徵是反曲點(without inflection time)，但在 RF 和 FNN 中缺少反曲點和充電時長都會拉低模型效能

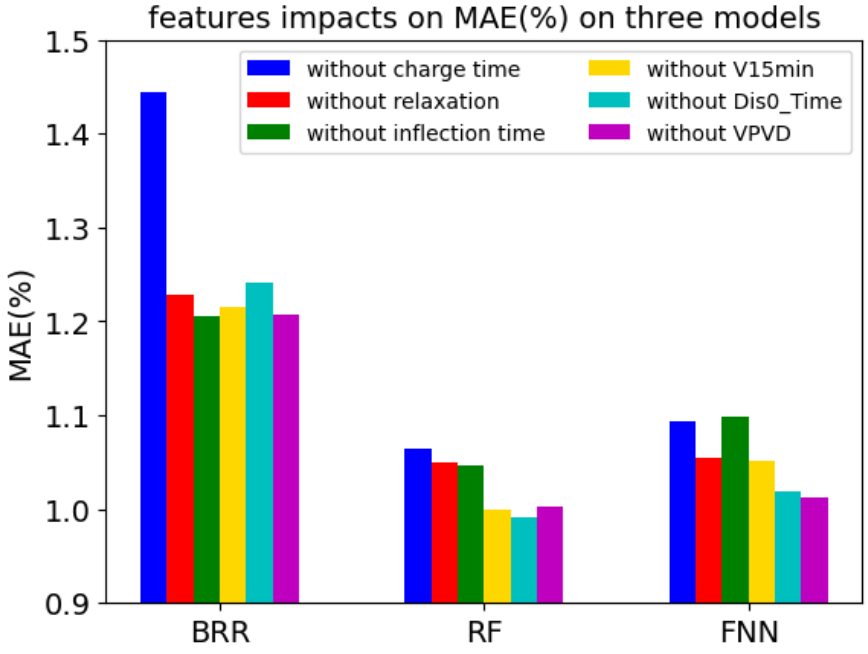


圖 4-6 移除不同特徵後對 MAE(%)的影響

doi:10.6342/NTU202400743

MAE 衡量模型預測時出現的殘差的平均值，越高越差，若在缺少了一個特徵下 MAE 提升地越高，表示該特徵對模型的預測能力影響越強，反之越弱。BRR 的 MAE 都遠高於 RF 和 FNN，且最能影響 BRR 的效果的特徵依舊是充電總時長(no charge time)，最弱的特徵是反曲點(no inflection time)，在 RF 和 FNN 中缺少反曲點和充電時長也會拉低模型效能，結論類似 r2 看到的現象，不過 3 分鐘弛豫現象也對 RF 有高度影響，。

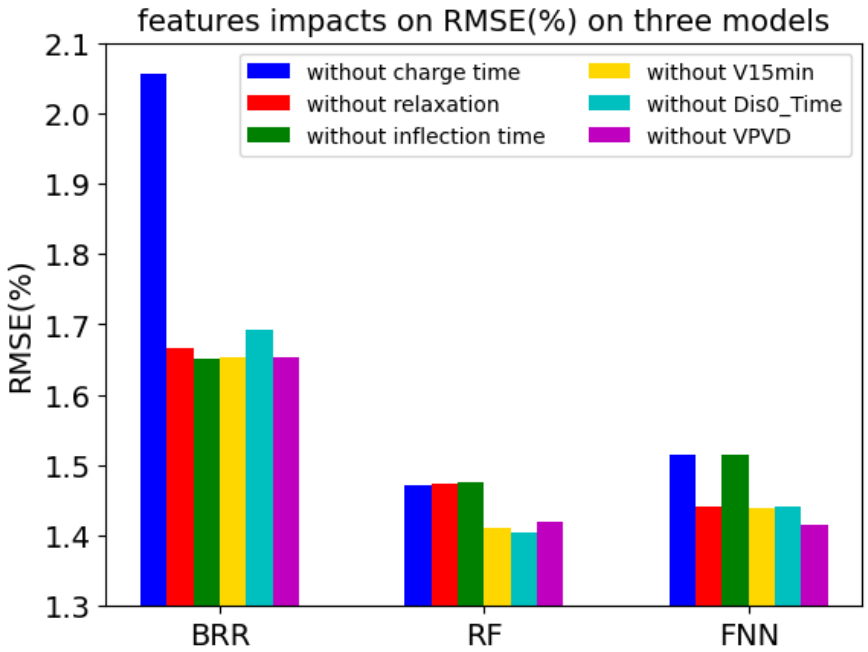


圖 4-7 移除不同特徵後對 RMSE(%)的影響

RMSE 衡量模型預測時離群值(outliers)的影響力，越高越差，若在缺少了一個特徵下 RMSE 提升地越高，表示該特徵對模型的預測能力影響越強，反之越弱。特徵對模型的影響力和 MAE 看到的情況一樣。

doi:10.6342/NTU202400743

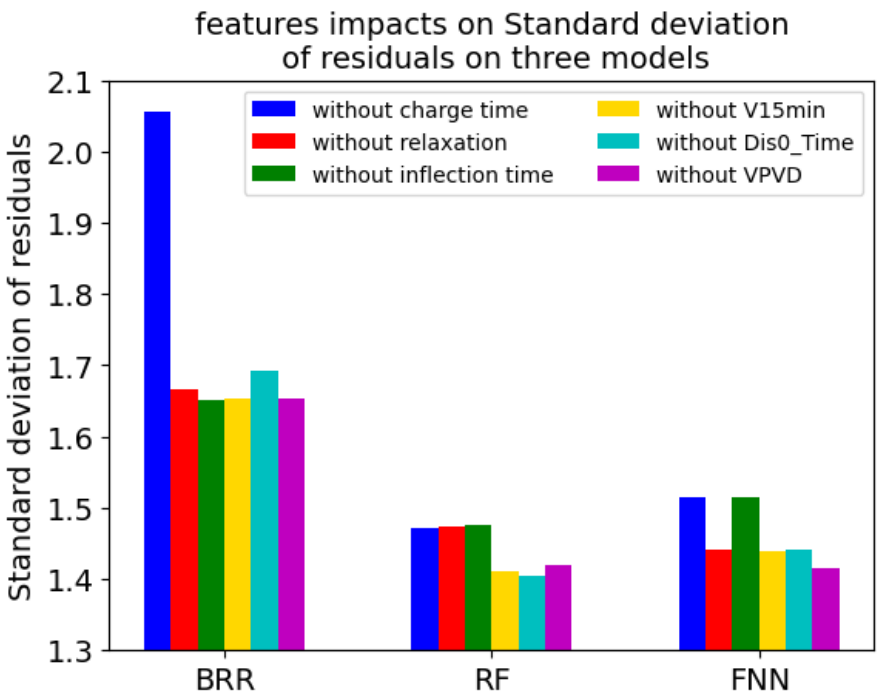


圖 4-8 移除不同特徵後對殘差分布程度的影響

殘差的標準差(std of residuals)衡量模型的精度和穩定性，越高越差，若在缺少了一個特徵下 RMSE 提升地越高，表示該特徵對模型的預測能力影響越強，反之越弱。特徵對模型的影響力和 RMSE、MAE 看到的情況一致。

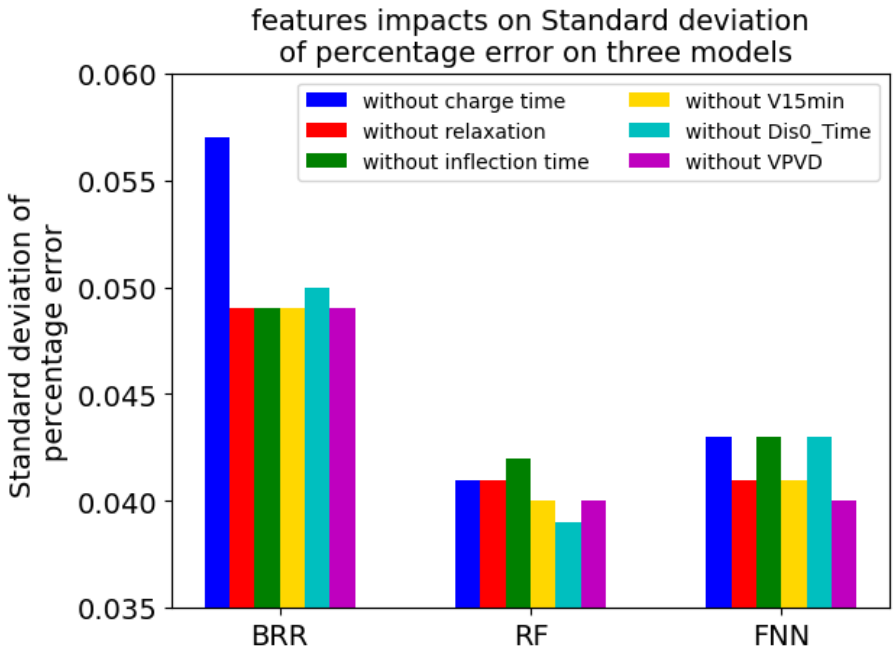


圖 4-9 移除不同特徵後對殘差誤差百分比的影響

殘差的百分比誤差的標準差 (standard deviation of percentage error, PE) 衡量

doi:10.6342/NTU202400743

模型的精度和穩定性，越高越差，若在缺少了一個特徵下 PE 提升地越高，表示該特徵對模型的預測能力影響越強，反之越弱。特徵對模型的影響力和 std of residuals、RMSE、MAE 看到的情況類似，但缺少反曲點對 RF 的 PE 影響最大，成為對 RF 來說最重要的特徵。

結論: 缺少充電時長皆會降低三個模型的效能，以 BRR 最為明顯，但是 BRR 的訓練成效不如 FNN 和 RF。相對地，在 FNN 和 RF 中，反曲點比充電時長更能提升模型性能，另外弛豫現象在 RF 中也是很重要的特徵。

doi:10.6342/NTU202400743

4.4 老化數據視覺化

老化實驗所得數據: 高溫老化數據、高速老化數據，以及所有老化數據的充放電曲線，數據各有 440 筆、331 筆，合計 771 筆，見下方圖 4-10。圖中可以

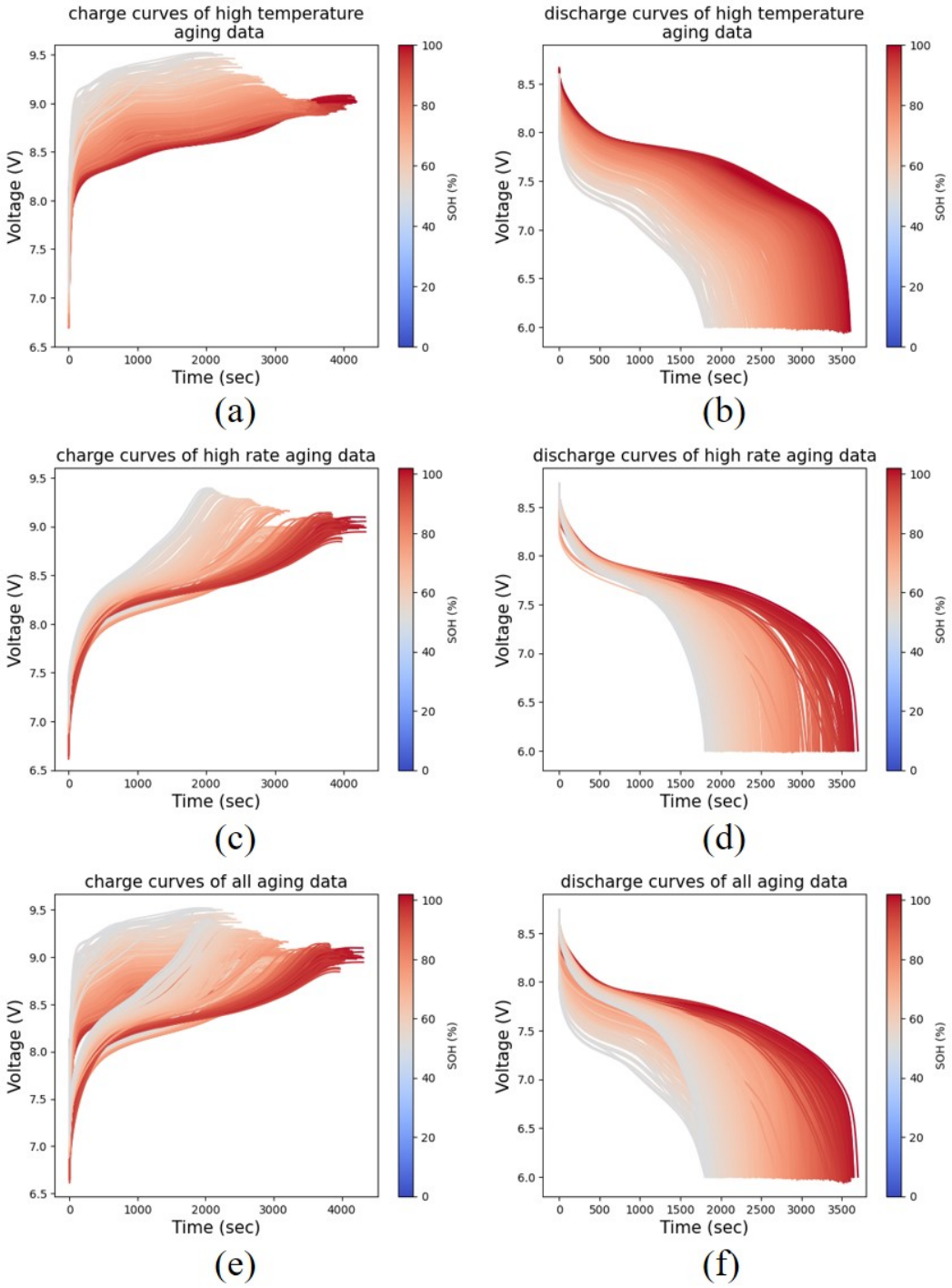


圖 4-10 老化數據電壓曲線圖 (a), (b) 高溫老化數據的充電、放電曲線 (440 筆)、
(c), (d) 高速老化數據的充電、放電曲線 (331 筆)、
(e), (f) 所有老化數據的充電、放電曲線 (771 筆)。

doi:10.6342/NTU202400743

見到各組曲線皆由長變短、由右向左移、曲線越來越高的趨勢，可以明確地看出來模組正在衰退，且伴隨著衰退發生的現象。

不過，不同組的數據表現出的特徵是不同的: 高溫老化模組的電壓曲線，在程序剛開始時都會急遽地變動，且越是衰退電壓變動的越多，推測是高溫且長時間的循環下，對電極產生較強的破壞力，使電極產生破碎、有效反映面積減少，放大了電化學極化的作用，使電壓在一開始就快速上升，之後一路緩緩上升直至完全充電。高速老化數據中，儘管每顆模組的起始 SOH 不同，循環實驗結果仍然相似，可以判斷出高低 SOH 的曲線的特色。將兩組數據合在一起觀察時，可以明顯地看出有兩群的曲線，一群在充電一開始時電壓竄高，另一群較低，直到充電中後期時兩群開始重合。這些數據提高樣本的多樣性，可以幫助模型更好地擬合充電曲線。

doi:10.6342/NTU202400743

4.5 遷移學習效果

4.5.1 數據集和特徵分布

由於服役數據有嚴重的雜訊，因此重設 PVD 的條件，使其和老化數據所用條件相近以減少測量時造成的雜訊:

PVD: ΔV = 2.5~3.5 mV 或電壓停滯 57 ~ 60 秒

另外，我們也限制模組的放電時間，讓數據的型態跟老化數據相似。服役前數據中選取充電 35 分鐘內(簡寫為 C35)、放電 30 分鐘以上(簡寫為 D30)的模組，服役後數據中選取充電 37 分鐘內(簡寫為 C37)、放電 30 分鐘以上(簡寫為 D30)的模組。服役後數據的充電條件多 2 分鐘，乃因廠商將 PVD 中電壓遲滯的時間納入考慮，因此多出了充電 35~37 分鐘內完全充電的模組。判定完全充放電模組的完整條件整理如下:

服役前: C35、D30、 ΔV=2.5~3.5mV 或停滯 57~60 秒

服役後: C37、D30、 ΔV=2.5~3.5mV 或停滯 57~60 秒

通過新 PVD 的數據，服役前數據有 492 個，服役後數據有 1515 個，如圖 4-11 (c), (d)、圖 4-12 (c), (d)所示。服役前模組是經過一次充放電測試後，挑選放電時間超過 30 分鐘者重組成一個電池組的，因此當中多數模組性能都很相似，只代表一個特定的、極小的 SOH 區間，由新 PVD 選出來的模組們更說明這件事，因此我們認為這一類的模組不需要太多，僅以其中的一部份模組代表這一群數據為佳。服役後模組的曲線走勢多樣。

doi:10.6342/NTU202400743

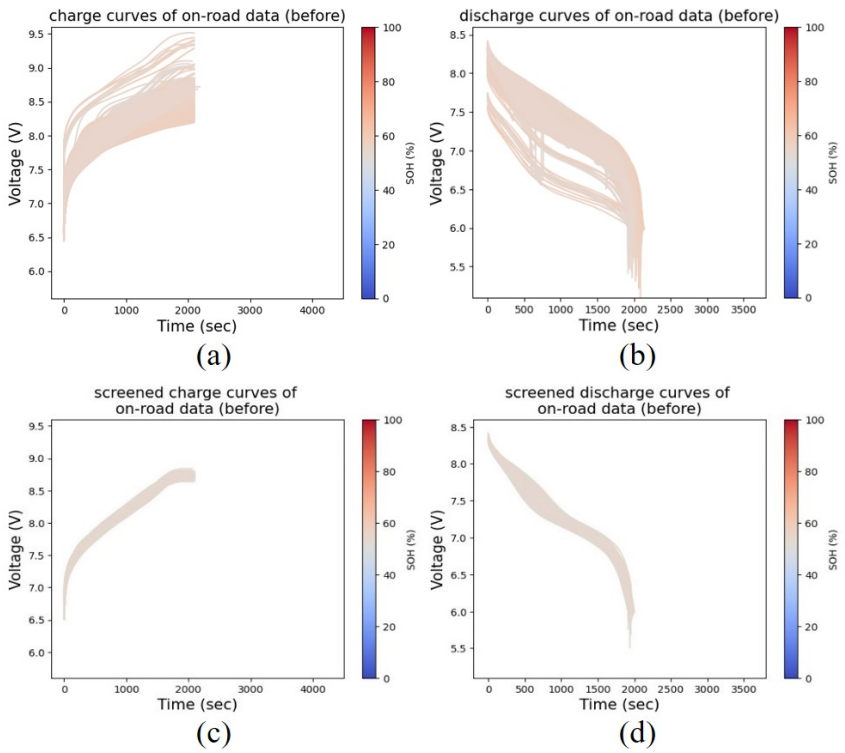


圖 4-11 服役前數據。(a) 全部數據的充電曲線 (b) 全部數據的放電曲線 (c) 篩選後的充電曲線 (d) 篩選後的放電曲線

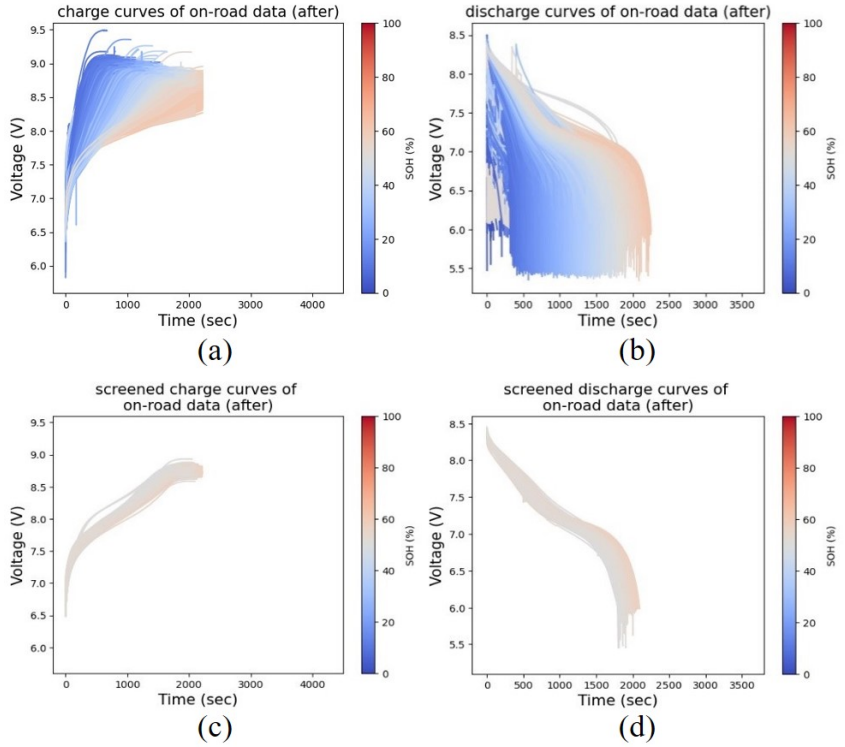


圖 4-12 服役後數據。(a) 全部數據的充電曲線 (b) 全部數據的放電曲線 (c) 篩選後的充電曲線 (d) 篩選後的放電曲線

doi:10.6342/NTU202400743

選出的服役數據和老化數據合併後共 2778 筆數據，在各個特徵中的分布狀況如下圖 4-13 所示。我們刪去了前處理的放電時間(Dis0_Time)，因為在老化實驗中該值受循環步驟影響較深，不適合代表模組的 SOH。將剩下的 5 個特徵對放電時間作圖，並觀察該特徵在不同數據中的分布方式。除了充電時間，其他特徵都依照數據種類分群，適合使用 MMD 對特徵做處理、縮小間距。

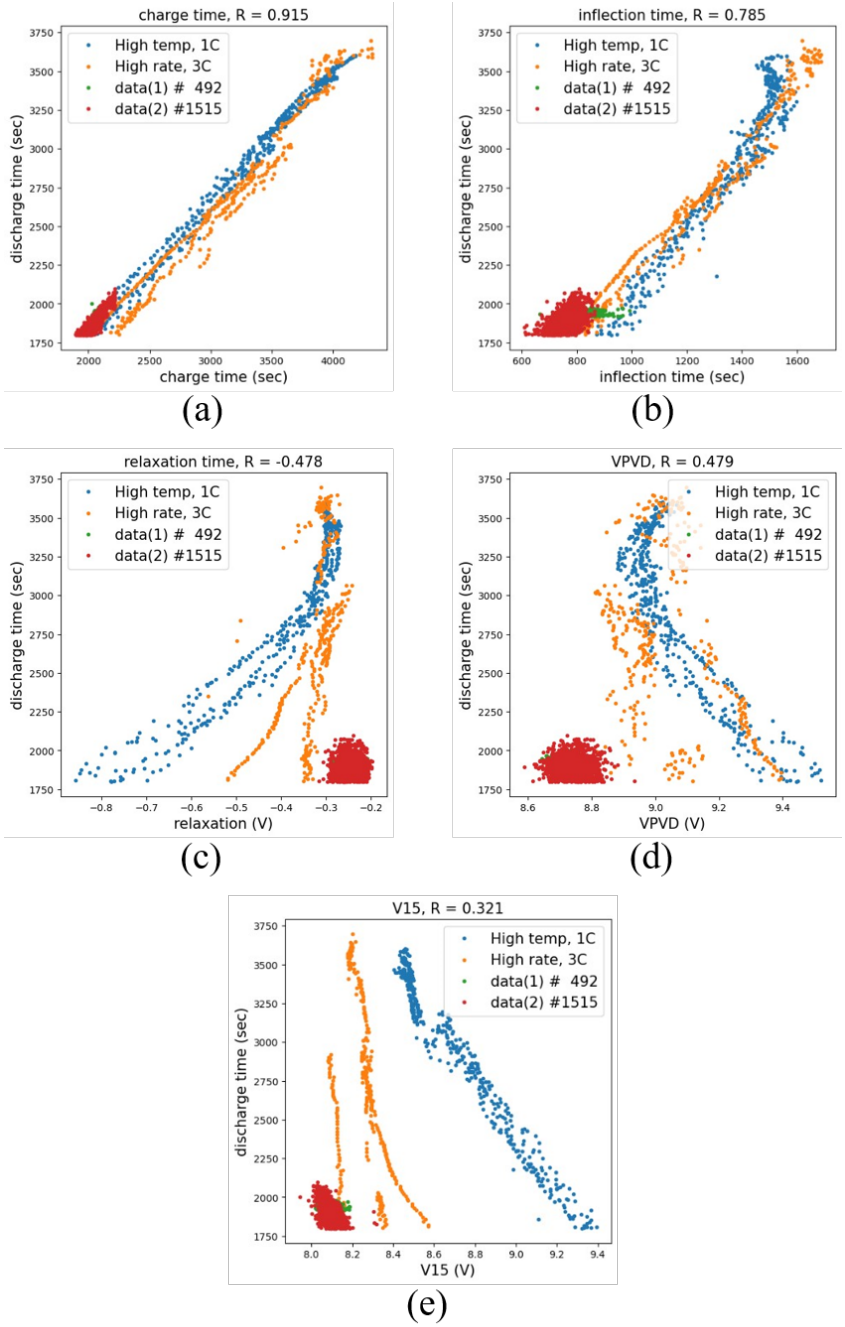


圖 4-13 不同數據在各個特徵中的分布狀況 (a) 充電時間、(b) 3 min 弛豫時間、(c) 反曲點、(d) 充電曲線的最高電壓、(e) 充電第 15 分鐘的電壓

doi:10.6342/NTU202400743

計算所有特徵在不同數據中和 SOH 的相關性，結果見表 4-10，可以見到烘箱老化實驗和所有特徵的相關性都很高，反映其在規律實驗操作下，比較容易在特徵中看到一致的變化；高速老化實驗、服役後數據在充電時間和 V15 和 SOH 較有相關性，在另外三個特徵中比較不一致；最糟的是服役前數據，經分析得知服役前數據的 SOH 範圍較小且經過前處理，各個特徵和 SOH 的相關性並不高。考慮到服役前數據的特殊性，在 MMD 時我們將它跟服役後數據合併，成為第三個數據集，服役數據，並和兩組老化數據計算 MMD 並選擇適合的模型參數。

表 4-10 五個特徵在各個數據別中，和 SOH 之相關性(R)

	充電時間	3 min 弛豫	反曲點	V15	VPVD
服役前數據 (492)	0.481	0.013	0.119	−0.067	−0.053
服役後數據 (1515)	0.891	0.121	0.511	−0.532	−0.103
烘箱老化實驗 SOH	0.996	0.919	0.92	−0.981	−0.704
高速老化實驗 SOH	0.99	0.665	0.988	−0.53	−0.253

doi:10.6342/NTU202400743

4.5.2 測試結果

在測試之前，再次執行 gridsearch 決定新的 FNN 超參數，用來最佳化地擬合合併數據，gridsearch 的項目和範圍列於下表 4-11。 H1、H2 沿用 4.2 節中最佳化組合的 100、50，讓模型的規模和複雜度一樣；將 lr_init 的範圍縮減至 0.001~0.003，刪去明顯過低的 lr_init；factor 和 patience 保持不變。訓練組為合併數據的 80 %，共 2222 筆數據(2778*0.8)，測試組為剩下的 20 %，共 556 筆(2778*0.2)。在此範圍下找到最佳的組合為 lr_init = 0.003、factor = 0.5、patience = 10。由於 lr_init 是本次搜尋範圍的上限，因此再新增一個 lr_init: 0.004，確認找到的最佳組合是否恰當。新增的 lr_init = 0.004 的誤差並未比其他組低，說明 lr_init = 0.003 這一組已經是最佳化的表現，被用來建立模型。

表 4-11 將合併數據的 SOH 預測最佳化的 FNN 超參數項目、範圍和結果

lr_init	factor	patience	Test loss (%)
0.001	0.2	5	0.9392
		10	
		20	
	0.5	5	
		10	
		20	
0.002	0.2	5	2.7243
		10	0.9295
		20	0.9154
	0.5	5	0.9469
		10	0.9126
		20	0.9054
0.003	0.2	5	0.9379
		10	0.9328
		20	0.9391
	0.5	5	1.0774
		10	**0.8957**
		20	0.9391
0.004	0.2	5	1.0945
		10	0.9487
		20	0.9254
	0.5	5	1.0525
		10	0.9064
		20	0.9102

doi:10.6342/NTU202400743

RBF 的兩個超參數: 每一組 γ 和 output 的特徵數量(m)產生的MMD_{ALL}及預測誤差，列於下表 4-12。γ從 1 開始測試，每一次調整加 1，在發現$\gamma = 5$的最小值後將γ設定為 1~10；m則固定為 1~30。下表由左至右分別列出γ、該γ之下讓 MMD 最小的m、MMD_{ALL}，和 FNN 預測的 RMSE。RBF使用 scikit − learn 的 RBFSampler，FNN 的超參數為前一頁 gridsearch 找到的最佳組合，訓練組和測試組亦同前一頁，分別為 2222 筆和 556 筆。

在不使用 MMD 下，靠原始數據進行預測 RMSE 為 0.8957 %，MMD_{ALL} 為 0.1918，使用 MMD 後發現$\gamma = 5, m = 20$ 時 RMSE 是所有組合中最小的一個，只有 0.757 %，符合我們對 MMD 的功能的預期，確實可以降低預測誤差。有趣的是該組產生的MMD_{ALL}並非所有組合中最小的一個，顯示MMD_{ALL}並非影響預測效能的最大因素，RBF 的 γ 和 m 也是影響預測效能的原因，且無一定規則。γ 決定了 RBF 的寬度，γ 較小時核函數較寬、支援向量較多，模型可能欠擬合，數據分布差異較小，反之過大會過度擬合，分布差異較大；m 決定投影空間的維度，越大擬合效果越好，但分布差異可能會更大，會直接影響MMD_{ALL}的大小。兩參數和MMD_{ALL} 之間無絕對關係，需要探索各種組合。

表 4-12 各種 RBF 的MMD_{ALL}和 RMSE

不使用 MMD			
MMD_{ALL}		RMSE (%)	
0.1918		0.8957	
使用 MMD			
γ	m	MMD_{ALL}	RMSE (%)
1	28	0.044	1.0321
2	3	0.053	1.5368
3	28	0.054	1.0323
4	22	0.062	0.7965
5	**20**	**0.063**	**0.757**
6	20	0.066	0.7739
7	20	0.067	0.8102
8	26	0.069	0.799
9	26	0.069	0.7811
10	26	0.07	0.7767

doi:10.6342/NTU202400743

$\gamma = 5, \mathrm{m} = 20$ 的擬合結果見下圖 4-14，不確性度量分布圖見圖 4-15。使用 MMD 的預測效果(右) 比沒用 MMD (左) RMSE 要低 0.1 %，兩張圖長得很類似，不過右圖的殘差標準差比左圖略低，表示有比較少的離群值或殘差分布比較窄，這點從圖 4-15 (a), (b) 殘差分布圖中得到證實，而且還發現使用 MMD 後信賴區間的長度下降了不少，由 [-1.784 ~ 1.726] 縮小到 [-1.629 ~ 1.236]。除了殘差分布圖以外，從圖 4-15 (c)、(d) 百分比誤差分布圖中也可以見到，使用 MMD 後 PE 的分布範圍和信賴區間比沒有用 MMD 時略小，從各種層面上證實使用 MMD 後能全面提升預測效果。

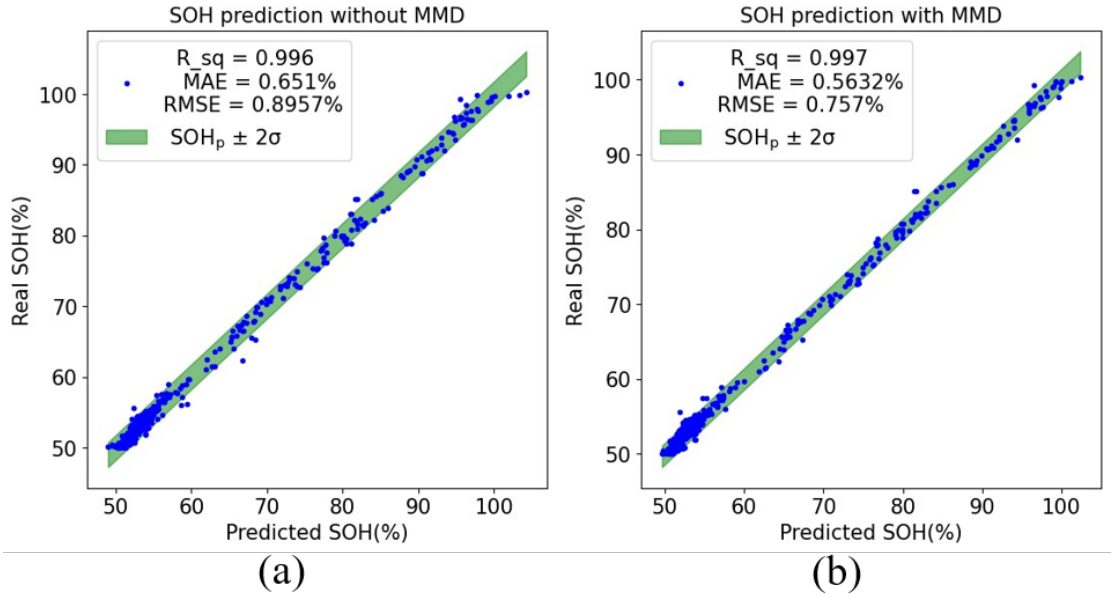


圖 4-14 用合併數據預測 SOH 的結果 (a) 未使用 MMD、(b) 使用 MMD

doi:10.6342/NTU202400743

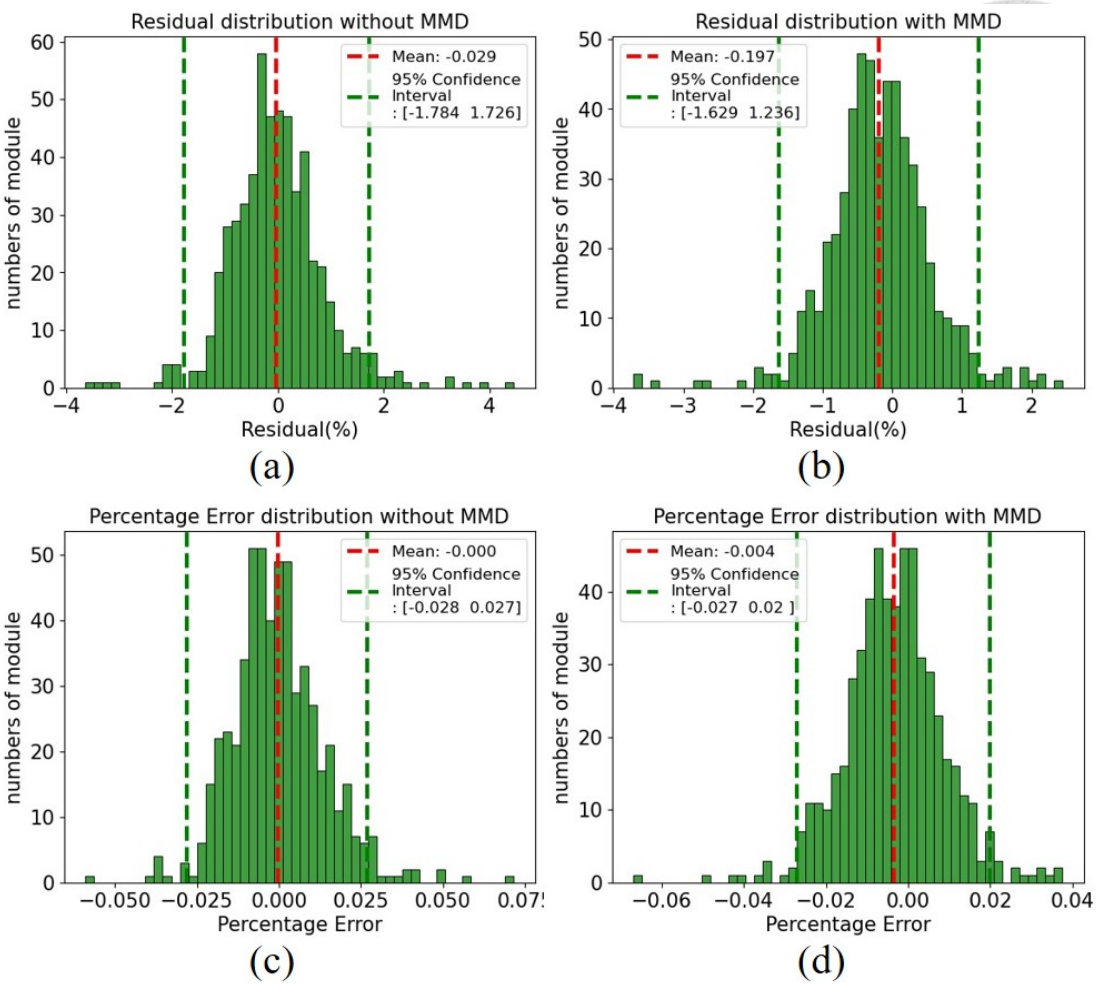


圖 4-15 用合併數據預測 SOH 的不確定性度量 (a) 未使用 MMD 的殘差分布、(b) 使用 MMD 的殘差分布、(c) 未使用 MMD 的殘差百分比(PE)分布、(b) 使用 MMD 的殘差百分比(PE)分布

doi:10.6342/NTU202400743

4.6 完整流程圖

本研究涉及的數據種類繁多，有很多處理步驟，因此於分析結果後，將每一個環節執行後產生的數據，以括號方式附註於各個環節名稱下方，方便閱覽、對照，見圖 4-16。括號內第一個數字是數據的筆數，第二個數字是該數據的維度，包含 SOH 以及特徵，如綠色框框內寫著「Data reduction (2007*6)」的

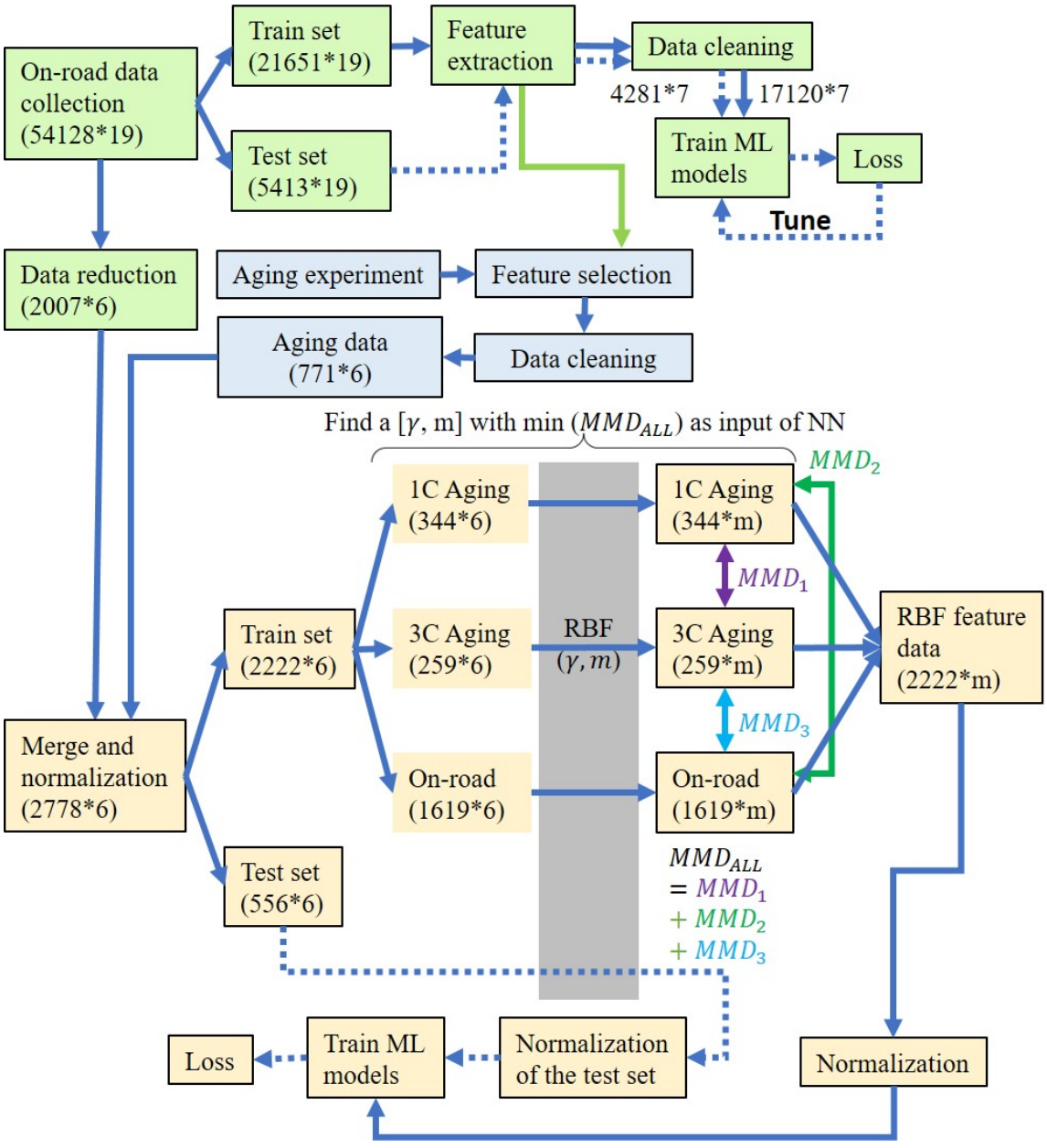


圖 4-16 加註數據量的研究流程示意圖

意思是縮緊 PVD 篩選數據後，剩餘 2007 筆，有 6 個維度 (1 個 SOH，5 個特徵)；「Train set (2222*6)」指訓練組的資料共 2222 筆，有 6 個維度(1 個 SOH，5

doi:10.6342/NTU202400743

個特徵)，其他以此類推。另外在綠色的 Data cleaning 格子下方，兩條分別代表訓練組和測試組的箭頭旁邊也有數字標示，該數字是兩組數據在清理後剩下的數量，訓練組剩下 17120 筆、測試組剩下 4281 筆。

doi:10.6342/NTU202400743

第五章 總結

本研究透過機器學習法研究鎳氫電池的電化學特性，從中提取特徵預測 SOH，並執行加速老化實驗，獲取高 SOH 的充放電資料，驗證特徵在高低 SOH 中皆可發揮一定的預測能力，最後透過遷移學習的 MMD，降低特徵在不同域的差異，進一步提高預測的精準度，補足了 **2.9** 提到的研究缺口。在研究對象上，我們仿照研究鋰電池性能預測的文獻，將 ML 應用在鎳氫電池上，可以將預測誤差縮小到約 1.384 %；在數據方面，我們執行了以高溫和高速讓電池老化的循環實驗，結果發現兩組數據有明顯的差異，獲得的高 SOH 數據確實協助我方強化模型的泛化能力；在特徵方面，我們學習了鎳氫電池運作機制，觀察充電曲線的走勢和電化學理論，找出了五個特徵，不僅在服役後數據有效，在老化數據上也同樣看到不錯的效果，尤其是反曲點和弛豫現象，是繼充電時間之後可以跨域預測 SOH 的特徵；最後是模型，我們結合機器學習模型和遷移學習，能夠預測極大的 SOH 範圍並將誤差降低至 0.757 %，能跟文獻的預測成效匹敵，且還有優化、持續探索的空間，包括增加服役數據量、改變遷移學習流程等等。

當然，獲得這些研究成果也並非一帆風順。首先是數據來源，我們努力取得皓恆授權，不僅給我方服役數據，也贈送我方全新模組和測試機台，在機器有問題時能夠協助排除、檢修，使實驗持續進行，否則高溫老化實驗只能進行到 6/20；獲得高度相關性的反曲點因為涉及許多數學工具，該如何設定參數，找出在多種實驗環境下都有效的反曲點、過濾雜訊...等等的問題花費了許多時間，最大的變動是將反曲點由數學定義的「二次微分值由負轉正/由正轉負」，改成「二次微分值為正的時段的中點」，配合著鎳氫電池的運作機制做修改，以期待它能更好地擬合 SOH。最後模型的部分，建立模型時花費很多的時間調整超參數，尤其是神經網路，每次調整需要耗費近一小時等待結果，MMD 時樣

doi:10.6342/NTU202400743

本較小，但也需要 15 分鐘左右進行預測，需要分神處理別的事情以度過該等待時光；MMD 不論在理解上還是應用上都耗費心力，包括理解核函數、映射的概念，調整 RBF 和決定最佳預測參數，以及如何將映射後的特徵用來建立 FNN。不久前剛發現 RBF 映射的特徵會和原本特徵的分布相似，但是數據中有負數，在建立 FNN 時會因為不符合激活條件而無效化，使模型預測失敗。知道這一點後將 RBF 後的特徵做歸一化，將數據範圍壓縮到 0 和 1 之間，就解決問題並給出圖??的成果。這些經歷促成研究成型，成為寶貴的經驗或者教訓，期待之後需要深入研究優化時去蕪存菁，拓展研究成果。

在能夠預測精準的情況下，我們可以省略掉完全充放電步驟中，最後一步的放電步驟，可以為工作人員節省至少 1/3~1/2 的操作時間，處理更多的退役電池，如圖 5-1 所示，確認利用機器學習並借鑑遷移學習的技術，在面對不同狀況的模組時可以發揮更強的預測效果，預期未來能夠導入生產線上，提高生產力並降低檢測電池所消耗的能源。

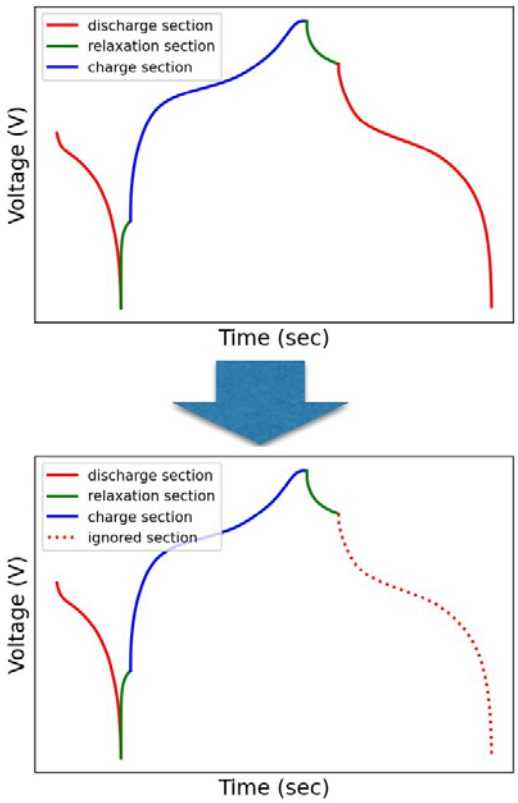


圖 5-1 導入機器學習後，能夠節省近一半的檢測時間

doi:10.6342/NTU202400743

第六章 未來工作

雖然取得了不錯的成果，未來仍有一些課題需要完成，或者以其他方式進行研究。首先是針對本研究的優化措施，包括數據清理，確保數據來源是可靠的，避免機器量測產生的誤差影響到模型的可靠性；優化 MMD 方法，持續尋找可以使預測效果最佳化的 RBF 參數和輸出特徵的維度，以及放寬篩選服役數據的條件，將更多的、個體差異變大的數據納入遷移學習，觀察 MMD 方法的效果如何；最後是研究方法的選擇，目前是使用機器學習習，針對電池的電化學特性提取特徵，需要該領域的知識協助，往後可以嘗試深度神經網路提取特徵、預測 SOH 預測，如在遷移學習中看到使用卷積神經網路(convolution neural network, CNN)從電壓曲線中提取特徵並進行預測，或者連接多種深度模型，讓準確度和運算時間持續優化，都是可以嘗試的方向。期待未來此方法可以不限於線下預測、簡化廠商工作流程，還可以連接雲端、在模組服役時能實時地根據模組的運行數據，如電壓、電流、溫度等推估模組的 SOH。

doi:10.6342/NTU202400743

參考文獻

汤依伟, 艾亮, 程昀, 王安安, 李书国 and 贾明 (2015). "锂离子动力电池高倍率充放电过程中弛豫行为的仿真." 物理学报 **64**(24): 248201.

张志刚, 张涛, 汤爱华, 姚疆 and 蒋依汗 (2022). "车用锂电池健康状态下快充方法研究综述." 西南大学学报 (自然科学版) **44**(2): 194-206

Aminzadegan, S., Shahriari, M., Mehranfar, F., & Abramović, B. (2022). Factors affecting the emission of pollutants in different types of transportation: A literature review. Energy Reports, 8, 2508-2529.

Bornatico, R., A. Storti, L. Mandrioli, A. Zappavigna, Y. Guezennec and G. Rizzoni (2007). NiMH battery characterization and state-of-charge estimation for HEV applications. ASME International Mechanical Engineering Congress and Exposition.

Chen, W., Z. Xu and J. Tu (2002). "Electrochemical investigations of activation and degradation of hydrogen storage alloy electrodes in sealed Ni/MH battery." International journal of hydrogen energy **27**(4): 439-444.

Chen, X., A. Chu, D. Li, Y. Yuan, X. Fan and Y. Deng (2021). "Development of the cycling life model of Ni-MH power batteries for hybrid electric vehicles based on real-world operating conditions." Journal of Energy Storage **34**: 101999.

Chen, Y., L. Yang, F. Guo, D. Liu, H. Wang, J. Lu, J. Zheng, X. Yu and H. Li (2022). "Mechanical-electrochemical modeling of silicon-graphite composite anode for lithium-ion batteries." Journal of Power Sources **527**: 231178.

Chehab, Z., L. Serrao, Y. G. Guezennec and G. Rizzoni (2006). Aging Characterization of Nickel: Metal Hydride Batteries Using Electrochemical Impedance Spectroscopy. ASME International Mechanical Engineering Congress and Exposition.

Cheng, S., J. Zhang, H. Liu, Y. Leng, A. Yuan and C. Cao (1998). "Study of early cycling deterioration of a Ni/MH battery by electrochemical impedance spectroscopy." Journal of power sources **74**(1): 155-157.

Cui, Z., L. Wang, Q. Li and K. Wang (2022). "A comprehensive review on the state of charge estimation for lithium-ion battery based on neural network." International Journal of Energy Research **46**(5): 5423-5440.

Fetcenko, M., J. Koch and M. Zelinsky (2015). Nickel–metal hydride and nickel–zinc batteries for hybrid electric vehicles and battery electric vehicles. Advances in battery

doi:10.6342/NTU202400743

technologies for electric vehicles, Elsevier**:** 103-126.

Galeotti, M., C. Giammanco, L. Cinà, S. Cordiner and A. Di Carlo (2015). "Synthetic methods for the evaluation of the State of Health (SOH) of nickel-metal hydride (NiMH) batteries." Energy conversion and management **92**: 1-9.

Gao, D., M. Huang and J. Xie (2017). "A novel indirect health indicator extraction based on charging data for lithium-ion batteries remaining useful life prognostics." SAE International Journal of Alternative Powertrains **6**(2): 183-193.

Han, T., et al. (2022). "End-to-end capacity estimation of Lithium-ion batteries with an enhanced long short-term memory network considering domain adaptation." Journal of Power Sources 520: 230823.

Hosen, M. S., R. Youssef, T. Kalogiannis, J. Van Mierlo and M. Berecibar (2021). "Battery cycle life study through relaxation and forecasting the lifetime via machine learning." Journal of Energy Storage **40**: 102726.

Hu, X., et al. (2015). "Battery health prognosis for electric vehicles using sample entropy and sparse Bayesian predictive modeling." IEEE Transactions on Industrial Electronics 63(4): 2645-2656.

Jafari, S., et al. (2022). "Lithium-Ion Battery Health Prediction on Hybrid Vehicles Using Machine Learning Approach." Energies 15(13): 4753.)

Jones, R. and W. Reoch (1985). Charging system for nickel-zing batteries.

Kasnatscheew, J., T. Placke, B. Streipert, S. Rothermel, R. Wagner, P. Meister, I. C. Laskovic and M. Winter (2017). "A tutorial into practical capacity and mass balancing of lithium ion batteries." Journal of the electrochemical society **164**(12): A2479.

Khaleghi, S., D. Karimi, S. H. Beheshti, M. S. Hosen, H. Behi, M. Berecibar and J. Van Mierlo (2021). "Online health diagnosis of lithium-ion batteries based on nonlinear autoregressive neural network." Applied Energy **282**: 116159.

Khaleghi, S., M. S. Hosen, D. Karimi, H. Behi, S. H. Beheshti, J. Van Mierlo and M. Berecibar (2022). "Developing an online data-driven approach for prognostics and health management of lithium-ion batteries." Applied Energy **308**: 118348.

Kheirkhah-Rad, E. and M. Moeini-Aghtaie (2021). A novel data-driven SOH prediction model for lithium-ion batteries. 2021 31st Australasian Universities Power Engineering Conference (AUPEC), IEEE.

Kim, C.-S., K. M. Jeong, K. Kim and C.-W. Yi (2015). "Effects of capacity ratios between anode and cathode on electrochemical properties for lithium polymer

doi:10.6342/NTU202400743

batteries." Electrochimica Acta **155**: 431-436.

Kim, S., et al. (2021). "Forecasting state-of-health of lithium-ion batteries using variational long short-term memory with transfer learning." Journal of Energy Storage 41: 102893.

Le Guenne, L. and P. Bernard (2002). "Life duration of Ni–MH cells for high power applications." Journal of power sources **105**(2): 134-138.

Li, W., et al. (2021). "Online capacity estimation of lithium-ion batteries with deep long short-term memory networks." Journal of Power Sources 482: 228863.

Luo, Y.-F. and K.-Y. Lu (2022). "An online state of health estimation technique for lithium-ion battery using artificial neural network and linear interpolation." Journal of Energy Storage 52: 105062.

Ma, G., et al. (2022). "A transfer learning-based method for personalized state of health estimation of lithium-ion batteries." IEEE Transactions on Neural Networks and Learning Systems.

Micea, M. V., et al. (2011). "Online State-of-Health Assessment for Battery Management Systems." Ieee Transactions on Instrumentation and Measurement 60(6): 1997-2006.

Minjie, Y. A. N. G., Junmin, N. A. N., Xianlu, H. O. U., & Weishan, L. I. (2008). Preparation and Electrochemical Performances of Nickel Metal Hydride Batteries with High Specific Volume Capacity. Chinese Journal of Chemical Engineering, 16(6), 944-948.

Mohammadi, F., & Saif, M. (2023). A Comprehensive Overview of Electric Vehicle Batteries Market. e-Prime-Advances in Electrical Engineering, Electronics and Energy, 100127.

Mu, G., S. Agrawal, P. Sittisomwong and P. Bai (2022). "Impacts of negative to positive capacities ratios on the performance of next-generation lithium-ion batteries." Electrochimica Acta **406**: 139878.

Nicolai, J. and L. Wuidart (1995). From nickel-cadmium to nickel-hybride fast battery charger. Proceedings of 1995 International Conference on Power Electronics and Drive Systems. PEDS 95, IEEE.

Ng, M.-F., J. Zhao, Q. Yan, G. J. Conduit and Z. W. Seh (2020). "Predicting the state of charge and health of batteries using data-driven machine learning." Nature Machine Intelligence **2**(3): 161-170.

Noura, N., L. Boulon and S. Jemeï (2020). "A review of battery state of health

doi:10.6342/NTU202400743

estimation methods: Hybrid electric vehicle challenges." World Electric Vehicle Journal **11**(4): 66.

Park, S.-Y., H. Miwa, B. T. Clark, D. S. Ditzler, G. Malone, N. S. D'souza and J.-S. Lai (2008). A universal battery charging algorithm for Ni-Cd, Ni-MH, SLA, and Li-Ion for wide range voltage in portable applications. 2008 IEEE Power Electronics Specialists Conference, IEEE.

Paxton, B. and J. Newman (1997). "Modeling of nickel/metal hydride batteries." Journal of the Electrochemical Society **144**(11): 3818.

Pradhan, S. K. and B. Chakraborty (2022). "Battery management strategies: An essential review for battery state of health monitoring techniques." Journal of Energy Storage **51**: 104427.

Roman, D., S. Saxena, V. Robu, M. Pecht and D. Flynn (2021). "Machine learning pipeline for battery state-of-health estimation." Nature Machine Intelligence **3**(5): 447-456.

Sahoo, S., et al. (2022). "Transfer learning based generalized framework for state of health estimation of Li-ion cells." Scientific Reports 12(1): 13173.

Sastry, A., S. Choi and X. Cheng (1998). "Damage in composite NiMH positive electrodes."

Schneider, E., C. Oliveira, R. Brito and C. Malfatti (2014). "Classification of discarded NiMH and Li-Ion batteries and reuse of the cells still in operational conditions in prototypes." Journal of Power Sources **262**: 1-9.

Semanjski, I. and S. Gautama (2016). "Forecasting the state of health of electric vehicle batteries to evaluate the viability of car sharing practices." Energies **9**(12): 1025.

Shen, L., Li, J., Meng, L., Zhu, L., & Shen, H. T. (2023). Transfer Learning-based State of Charge and State of Health Estimation for Li-ion Batteries: A Review. IEEE Transactions on Transportation Electrification.

Shu, X., et al. (2021). "A flexible state-of-health prediction scheme for lithium-ion battery packs with long short-term memory network and transfer learning." IEEE Transactions on Transportation Electrification 7(4): 2238-2248.

Shukla, A., S. Venugopalan and B. Hariprakash (2001). "Nickel-based rechargeable batteries." Journal of Power Sources **100**(1-2): 125-148.

Su, S., et al. (2022). "A hybrid battery equivalent circuit model, deep learning, and transfer learning for battery state monitoring." IEEE Transactions on Transportation

doi:10.6342/NTU202400743

Electrification 9(1): 1113-1127.

Telmoudi, A. J., et al. (2020). "Modeling and state of health estimation of nickel-metal hydride battery using an EPSO-based fuzzy c-regression model." Soft Computing **24**(10): 7265-7279.

Vidal, C., P. Malysz, P. Kollmeyer and A. Emadi (2020). "Machine learning applied to electrified vehicle battery state of charge and state of health estimation: State-of-the-art." IEEE Access **8**: 52796-52814.

Xuan, D.-J., Z. Shi, J. Chen, C. Zhang and Y.-X. Wang (2020). "Real-time estimation of state-of-charge in lithium-ion batteries using improved central difference transform method." Journal of Cleaner Production **252**: 119787.

Xue, Y., et al. (2022). Battery Degradation Modelling and Prediction with Combination of Machine Learning and Semi-empirical Methods. 2022 12th International Conference on Power, Energy and Electrical Engineering (CPEEE), IEEE.

Yang, H., Y. Qiu and X. Guo (2015). "Prediction of state-of-health for nickel-metal hydride batteries by a curve model based on charge-discharge tests." Energies **8**(11): 12474-12487.

Yao, L., et al. (2022). "State of Health Estimation Based on the Long Short-Term Memory Network Using Incremental Capacity and Transfer Learning." Sensors 22(20): 7835.

Young, K.-h. and S. Yasuoka (2016). "Capacity degradation mechanisms in nickel/metal hydride batteries." Batteries **2**(1): 3.

Young, K., T. Ouchi, W. Mays, B. Reichman and M. Fetcenko (2009). "Pressure–composition–temperature hysteresis in C14 Laves phase alloys: Part 2. Applications in NiMH batteries." Journal of alloys and compounds **480**(2): 434-439.

Young, K., D. Wong, S. Yasuoka, J. Ishida, J. Nei and J. Koch (2014). "Different failure modes for V-containing and V-free AB2 metal hydride alloys." Journal of Power Sources **251**: 170-177.

Zhang, Q., X. Li, Z. Du and Q. Liao (2021). "Aging performance characterization and state-of-health assessment of retired lithium-ion battery modules." Journal of Energy Storage **40**: 102743.

Zhang, Y., et al. (2022). "A machine learning-based framework for online prediction of battery ageing trajectory and lifetime using histogram data." Journal of Power Sources 526: 231110.

doi:10.6342/NTU202400743

Zhu, D., W. Zhou, Z. Tang, Y. Heng, K. Liu, J. Li, L. Xie and Y. Chen (2019). "SOC-dependent high-rate dischargeability of AB5-type metal hydride anode: Mechanism linking phase transition to electrochemical H-desorption kinetics." International Journal of Hydrogen Energy **44**(29): 15278-15286.

doi:10.6342/NTU202400743

附錄

A.1 讀取服役數據、高溫老化數據，提取特徵的程式碼

```
import os
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy.signal import savgol_filter
import warnings
import time

t1 = time.monotonic()
starttime = time.ctime();
starttime = starttime.split()

Dis0_q, Char_q, V15_q, VPV_q, inflect_q, rex03_q, dis_5_q, chr_7_q, dis_9_q = [], [], [], [], [], [], [], [], [];
Dis0_t, Char_t, V15_t, VPV_t, inflect_t, rex03_t, dis_5_t, chr_7_t, dis_9_t = [], [], [], [], [], [], [], [], [];

for dirpath, dirnames, filenames in os.walk(r"C:\Users\AMRG\Documents\B&A to NTU"):
    for f in filenames:
        a = os.path.join(dirpath,f)
        if os.path.isfile(a):
            if 'csv' in a:
                try:
                    # print(a)
                    data = np.genfromtxt(a, dtype = None, delimiter = ',', skip_header = 0, filling_values = 0, usecols=np.arange(0,4), invalid_raise = False, encoding = 'latin-1')
                    [Row,Column] = data.shape
                    dat_r = np.round(data[4:,2].astype(np.float32),1)
                # Extract features
                    a_c = np.where(dat_r == 6.5)[0]; a_c = np.insert(a_c, [0, 0], [a_c[0]-2, a_c[0]-1])
```


doi:10.6342/NTU202400743

```
                    Cha = data[4:,0][a_c].astype(np.float32);
                    Cha = Cha - Cha[0]
                    vot = data[4:,1][a_c].astype(np.float32);              # Voltage
                    a_d = np.where(dat_r == -6.5)[0];
                    dis = data[4:,0][a_d].astype(np.float32);       # Discharge time
                    vot_d = data[4:,1][a_d].astype(np.float32);            # Voltage
                    sec1_end = np.where(vot_d[1:]-vot_d[0:-1] > 0.5)[0]
# # ------------------------------------------------------------------------------# #
                # 修正充電時間和電壓 #
                    if len(Cha) < int(Cha[-1]):
                        firstrow_v = np.zeros((1, int(Cha[-1])+1), dtype='float32')
                        firstrow_t = np.zeros((1, int(Cha[-1])+1), dtype='float32')
                        for i in range(len(Cha)):
                            order = int(Cha[i])
                            firstrow_v[0, order] = float(vot[i])
                            firstrow_t[0, order] = int(Cha[i])
                        whereis0 = np.where(firstrow_v[0] == 0)[0]; whereis0 = np.appe
nd(whereis0, int(whereis0[-1])+10);
                    #找連續缺 0 的部分
                        manyzero = np.where(whereis0[1:] - whereis0[0:-
1] > 1)[0]; manyzero = np.insert(manyzero, 0, -1)
                    #要插入數字的位置、插入的個數、數字的大小
                        inputvalue_s = []
                        for i in range(1, len(manyzero)):
                            j = int(manyzero[i])
                            inputvalue = np.round(0.5*firstrow_v[0][whereis0[j]-
1] + 0.5*firstrow_v[0][whereis0[j]+1], 4)
                            if inputvalue > 6:
                                inputvalue_s.append(inputvalue)
                            else:
                                length = manyzero[i] - manyzero[i-1]
                                end = firstrow_v[0][whereis0[j]+1]; start = firstrow_v
[0][whereis0[j]-length];
                                try:
                                    injectarry = np.arange(start, end, (end-
start)/length).tolist()
                                except:
                                    injectarry = [start]
```


doi:10.6342/NTU202400743

```
                                inputvalue_s = inputvalue_s + injectarry
                        row1st_v_list = firstrow_v[0].tolist()
                        for m in range(len(inputvalue_s)):
                            row1st_v_list[whereis0[m]] = inputvalue_s[m]
                        row1st_v_list = np.round(row1st_v_list, 4)
                    # Effective charge curve #
                        vot = row1st_v_list
                        Cha = np.arange(len(vot))
# # ----------------------------------------------------------------------------# #
                # extract features from charging section
                    sec3_end_tm = len(vot);                              ### Char_Time
                    try:
                        sec3_end_v15 = vot[900];       ### V15
                    except:
                        sec3_end_v15 = 0
                    sec3_end_vpv = np.max(vot)                      ### VPVD
                # extract features from discharge section
                    sec1_end_tm = len(dis[sec1_end[0]+1:]);          ### Dis0_Time
                    if int(sec1_end_tm) > int(sec3_end_tm):
                        sec1_end_tm = data[4:,0][-
1].astype(np.float32) - data[4:,0][a_c[-1]+601].astype(np.float32)
                        if int(sec1_end_tm) > int(sec3_end_tm):
                            print(f'{len(dis_9_q)}th : ' + a + f' Dis vs. Char :{[sec1
_end_tm, sec3_end_tm]}')
                ## Inflection analysis ##
                    b = np.polyfit(Cha, vot, 5);
                    y = np.poly1d(b); v = y(Cha);
                    dv1 = np.gradient(v, Cha);
                    dv2 = np.gradient(dv1, Cha);
                    b2 = np.polyfit(Cha, dv2, 3);
                    y2 = np.poly1d(b2); v2 = y2(Cha);
                    pts = np.where(v2[1:]*v2[0:-1]<0)[0];
#-------------------------------------------------------------------------------#
                    if len(pts) <= 1:
                        v = savgol_filter(vot, len(Cha)//8, 2, mode='nearest')
                        dv1 = np.gradient(v, Cha);
                        dv15 = savgol_filter(dv1, len(Cha)//8, 2, mode='nearest');
                        dv2 = np.gradient(dv15, Cha);
```


doi:10.6342/NTU202400743

```
                        dv25 = savgol_filter(dv2, len(Cha)//4, 2, mode='nearest');
                        dv30 = savgol_filter(dv25, len(Cha)//2, 4, mode='nearest');
                        pts = np.where(dv30[1:]*dv30[0:-1]<0)[0];
                        v2 = dv30
                    pyd1 = dv1[pts];
                    n = math.ceil(len(pts)*0.5);
                    pyt = np.empty((1,n), dtype = object);
#-------------------------------------------------------------------------------#
                    if len(pts) <= 1:           # 算入 0,1 的狀況    : 直接移除
                        ift = 0
                    elif (len(pts)%2) == 1:     # 算入 3,5...的狀況 : 讓它變 2,4 再計算
                        pyt = np.empty((1,n-1), dtype = object);
                        ptss = np.delete(pts,-1)
                        for i in range(n-1):
                            pyt[0,i] = pyd1[i*2+1]-pyd1[i*2]
                        pos = np.argmax(pyt);
                        ift_1 = ptss[pos*2]; ift_2 = ptss[pos*2+1];
                        accea = v2[ift_1:ift_2]
                        umost = np.argmax(accea)
                        ift = umost+ift_1
                    else:                       # 算入 2,4...的狀況 : 直接計算
                        ptss = pts
                        for ii in range(n):
                            pyt[0,ii] = pyd1[ii*2+1]-pyd1[ii*2]
                        pos = np.argmax(pyt);
                        ift_1 = pts[pos*2]; ift_2 = pts[pos*2+1];
                        accea = v2[ift_1:ift_2]
                        umost = np.argmax(accea)
                        ift = umost+ift_1
# -------------------------------------------------------------------------------#
                    # 畫圖
                    if (int(a[-6:-4])%4) < 0:
                        print('pts = ', [ift, pts])
                        fig, ax1 = plt.subplots()
                        ax1.set_xlabel('charging time (sec)', fontsize=15)
                        ax1.set_ylabel('Voltage (V)', color='blue', fontsize=15)
                        plt.title('Inflection analysis of a module', fontsize=15);
                        ax1.plot(Cha,vot, linewidth=2, label='raw voltage curve')
```


doi:10.6342/NTU202400743

```
                        ax1.plot(Cha,v, linewidth=2, label='fitted voltage curve')
                        plt.legend(fontsize=12,loc=2)
                        # Adding Twin Axes
                        ax2 = ax1.twinx()
                        try:
                            ax2.plot(Cha,dv30, color='#ED0DD9', linewidth=2, label=f'2nd derivative')
                        except:
                            ax2.plot(Cha,v2, color='r', linewidth=1, label=f'2nd derivative')
                        ax2.plot(ift,v2[ift], color='g', marker='o', markersize=5, linestyle='None', label=f'inflection point')   #1st
                        for j in range(len(pts)):
                            if j == 0:
                                plt.axvline(pts[j], color='g', linestyle='--', label='acceleration period')
                            else:
                                plt.axvline(pts[j], color='g', linestyle='--')
                        # Add label & Show plot
                        plt.ylabel('2nd derivative value', color='r', fontsize=15);
                        plt.legend(fontsize=12,loc=4);
                        plt.show()
# ------------------------------------------------------------------------------#
                # relaxation 3 minutes
                    a_r = np.where(dat_r[1:]-dat_r[0:-1] < -0.1)[0];
                    gap = a_r[1:]-a_r[0:-1]
                    gap_one = np.where(gap == 1)[0];
                    a_r_cl = np.delete(a_r, gap_one)
                    rex = data[a_r_cl[0]+4:a_r_cl[1]+5,0].astype(np.float64)-data[a_r_cl[0]+5,0].astype(np.float64);                    # Relaxation time;
                    vot_r = data[a_r_cl[0]+4:a_r_cl[1]+5,1].astype(np.float64);
                    try:
                        vot_r_180 = np.where(rex == 180)[0][0];
                    except:
                        vot_r_180 = np.where(rex == 179)[0][0];
                    sec4_end_rex = vot_r[vot_r_180]-vot_r[0]     ## 03min relaxation
                except:
                    print(a + ' , 充電時間 : ' + str(len(a_c)))
```


doi:10.6342/NTU202400743

```
                    sec1_end_tm, sec3_end_vpv, sec3_end_v15, ift, sec1_end_tm, sec4_end_rex = 0, 0, 0, 0, 0, 0
                    sec3_end_tm = len(Cha)
                    if len(sec1_end) == 0:
                        sec1_end_tm == 0
                    else:
                        sec1_end_tm = len(dis[sec1_end[0]+1:]);        ### Dis0_Time
            # features
                if 'Q' in a:
                    Dis0_q.append(sec1_end_tm); VPV_q.append(sec3_end_vpv); inflect_q.append(ift); dis_9_q.append(sec1_end_tm)
                    Char_q.append(sec3_end_tm); V15_q.append(sec3_end_v15); rex03_q.append(np.round(sec4_end_rex,4));
                elif 'Q' not in a:
                    Dis0_t.append(sec1_end_tm); VPV_t.append(sec3_end_vpv); inflect_t.append(ift); dis_9_t.append(sec1_end_tm)
                    Char_t.append(sec3_end_tm); V15_t.append(sec3_end_v15); rex03_t.append(np.round(sec4_end_rex,4));

t2 = time.monotonic()
print("start time:", starttime[1:4])
print("time elapsed: " + str(np.round(t2-t1,3)) + " seconds")
```


doi:10.6342/NTU202400743

A.2 讀取高速老化數據、提取特徵的程式碼

```
import os
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
from scipy.signal import savgol_filter
import time

t1 = time.monotonic()
starttime = time.ctime();
starttime = starttime.split()

featurec30 = np.zeros((1,7), dtype = object);
voltagec30 = []

for dirpath, dirnames, filenames in os.walk(r"C:\Users\AMRG\Documents\Aging 0901 起_3C07\2023 0921-1011 第 07"):
    for f in filenames:
        a = os.path.join(dirpath,f)
        if os.path.isfile(a):     #判斷是否是檔案,是檔案才打開 and '.xlsx'
            if '-1-' in a:
                print('import data link =', a)
                if '~$' in a:
                    continue
                data = pd.read_excel(a, sheet_name=3, engine="openpyxl")
                data1 = np.hstack((data.values[:,0:1], np.round((data.values[:,5:6]/1000).astype(np.float32),2),data.values[:,6:7], np.round(data.values[:,7:8].astype(np.float64),7), np.round(data.values[:,8:9].astype(np.float64),5)))
# :,[0,5,6,7,8]
            # extract features from discharge section
                a_d = np.where(data1[:,1] == -6.5)[0];
                sec1_end = np.where(a_d[1:]-a_d[0:-1] > 5)[0];
                sec2_end = np.insert(sec1_end, [0,len(sec1_end)], [-2,len(a_d)-1])
```


doi:10.6342/NTU202400743

```
                    dischargetime = sec2_end[1:]-sec2_end[0:-1]-2
                    ds0 = np.where(dischargetime[0:-1]<dischargetime[1:])[0]
                    ft5_ds0 = dischargetime[ds0]
                    soh = np.where(dischargetime[0:-1]>dischargetime[1:])[0]
                    tar_soh = np.append(dischargetime[soh], dischargetime[-1])
                    tar_soh = tar_soh[tar_soh > 1700]
                # extract features from charge section
                    a_c = np.where(data1[:,1] == 6.5)[0];
                    sec3_end = np.where(a_c[1:]-a_c[0:-1] > 5)[0];
                    sec4_end = np.insert(sec3_end, 0,-2)
                    ft1_cha = sec4_end[1:]-sec4_end[0:-1]-2
                    sec5_end = np.insert(a_c[sec4_end[1:]], 0, a_c[0])
                    v_c = data1[:,2]
                    v_t = data1[:,0]
                    sec6_end = sec3_end+1
                    chargebegin = np.delete(np.insert(a_c[sec6_end],0,a_c[0]),-
1)
                    ft4_v15 = v_c[chargebegin+900]
                    ft6_pvd, ft2_3ax, ft3_flc, ftx_eoc = [], [], [], [];
                    for i in range(len(chargebegin)):
                        # ft6_pvd
                        chargevolt = v_c[chargebegin[i]:sec5_end[i+1]].tolist();
                        chargetime = v_t[chargebegin[i]:sec5_end[i+1]].tolist();
                        ft6_pvd.append(np.max(chargevolt))
                        ftx_eoc.append(chargevolt[-1])
                        voltagec30.append(chargevolt)
                        # ft2_3mn
                        time___after3min = sec5_end[1:]+180
                        voltageafter3min = v_c[time___after3min]
                        voltage_attheend = v_c[sec5_end[1:]]
                        ft2_3mn = voltageafter3min.astype(float)-
voltage_attheend.astype(float)
                        # ft3_inflection point
                        for deg in range(2,6,1):
                            with warnings.catch_warnings():
                                warnings.filterwarnings('error')
                                try:
                                    b = np.polyfit(chargetime, chargevolt, deg);
```


doi:10.6342/NTU202400743

```
                                    y = np.poly1d(b); v = y(chargetime);
                                    dv1 = np.gradient(v, chargetime);
                                    dv2 = np.gradient(dv1, chargetime);
                                    for deg2 in range(2,6,1):
                                        with warnings.catch_warnings():
                                            warnings.filterwarnings('error')
                                            try:
                                                b2 = np.polyfit(chargetime, dv2, deg2);
                                            except np.RankWarning:
                                                break
                                    except np.RankWarning:
                                        break
                            y2 = np.poly1d(b2); v2 = y2(chargetime);
                            pts = np.where(v2[1:]*v2[0:-1]<0)[0];
    #-------------------------------------------------------------------------#
                            if len(pts) <= 4:
                                v = savgol_filter(chargevolt, len(chargetime)//8, 2, mode='nearest')
                                dv1 = np.gradient(v, chargetime);
                                dv15 = savgol_filter(dv1, len(chargetime)//8, 2, mode='nearest');
                                dv2 = np.gradient(dv15, chargetime);
                                dv25 = savgol_filter(dv2, len(chargetime)//4, 2, mode='nearest');
                                dv30 = savgol_filter(dv25, len(chargetime)//2, 4, mode='nearest');
                                pts = np.where(dv30[1:]*dv30[0:-1]<0)[0];
                                v2 = dv30
                            pyd1 = dv1[pts];
                            n = math.ceil(len(pts)*0.5);
                            pyt = np.empty((1,n), dtype = object);
    #-------------------------------------------------------------------------#
                            if len(pts) <= 1:
                                ift = pts
                            elif (len(pts)%2) == 1:    # 遇 3,5 : 讓它變 2,4 再計算
                                pyt = np.empty((1,n-1), dtype = object);
                                ptss = np.delete(pts,-1);
```


doi:10.6342/NTU202400743

```
                            for ij in range(n-1):
                                pyt[0,ij] = pyd1[ij*2+1]-pyd1[ij*2]
                            pos = np.argmax(pyt);
                            ift_1 = ptss[pos*2]; ift_2 = ptss[pos*2+1];
                            accea = v2[ift_1:ift_2]
                            umost = np.argmax(accea)
                            ift = umost+ift_1
                        else:                       # 遇 2,4 : 直接計算
                            ptss = pts
                            for ii in range(n):
                                pyt[0,ii] = pyd1[ii*2+1]-pyd1[ii*2]
                            pos = np.argmax(pyt);
                            ift_1 = pts[pos*2]; ift_2 = pts[pos*2+1];
                            accea = v2[ift_1:ift_2]
                            umost = np.argmax(accea)
                            ift = umost+ift_1
                        ft3_flc.append(ift)
    # ------------------------------------------------------------ #
                    # 畫圖
                        if (i%2) == 0:
                            print('points =', np.insert(pts,0,ift))
                            date = a[-32:-23]
                            fig, ax1 = plt.subplots()
                            ax1.set_xlabel('charging time (sec)', fontsize=15)
                            ax1.set_ylabel('Voltage (V)', color='blue', fontsize
   =15)
                            plt.title(f'charge curve of m0{a[-
9]}, '+ date +f', #{len(ft3_flc)}', fontsize=15)
                            ax1.plot(chargetime, chargevolt, linewidth=2, label=
'raw voltage curve')
                            ax1.plot(chargetime, v, linewidth=2, label='fitted v
oltage curve')
                            plt.legend(fontsize=12,loc=2)
                        # Adding Twin Axes
                            ax2 = ax1.twinx()
                            try:
                                ax2.plot(chargetime,dv30, color='#ED0DD9', linew
idth=2, label='2nd derivative')
```


doi:10.6342/NTU202400743

```
                        except:
                            ax2.plot(chargetime,v2, color='r', linewidth=1, label='2nd derivative')
                        ax2.plot(min(chargetime)+ift,v2[ift], color='g', marker='o', markersize=5, linestyle='None', label='inflection point')   #1st
                        for j in range(len(pts)):
                            if j == 0:
                                plt.axvline(min(chargetime)+pts[j], color='g', linestyle='--', label='acceleration period')
                            else:
                                plt.axvline(min(chargetime)+pts[j], color='g', linestyle='--')
                        if int(a[-9:-8]) < 7:
                            plt.xlim(min(chargetime)-100, min(chargetime)+3700);
                        else:
                            plt.xlim(min(chargetime)-100, min(chargetime)+4400);
                    # Add label & Show plot
                        plt.ylabel('2nd derivative value', color='r', fontsize=15);     #2nd
                        plt.legend(fontsize=12,loc=4);
                        plt.show()
# ------------------------------------------------------#
                produce = np.vstack((tar_soh, ft1_cha, np.round(ft2_3mn,4), ft3_flc, ft4_v15, ft5_ds0, ft6_pvd)).T
                featurec30 = np.vstack((featurec30, produce))

t2 = time.monotonic()
print("\nstart time:", starttime[1:4])
print("time elapsed: " + str(np.round(t2-t1,3)) + " seconds")
```


doi:10.6342/NTU202400743

A.3 找出合適的弛豫時間的程式碼

```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

err_Chrtime = [119, 934, 1062, 1341, 1354, 1467, 1858, 2097, 2220, 2227, 2243, 2479, 3376, 3484, 3485, 3750, 4294, 4375, 4383, 4873, 5324, 5766, 6918, 6935, 7119, 7611, 8363, 8955, 9120, 9515, 10055, 10083, 10177, 10191, 10342, 10390, 11527, 12168, 12849, 13317, 13557, 13853, 14121, 14222, 14618, 16676, 17079, 18105, 18193, 18584, 18626, 18814, 19038, 19107, 19287, 19527, 19663, 19669, 20254, 20742, 21440, 21614, 21719, 21804, 21961, 21963, 23175, 23348, 23948, 24152, 24388, 24397, 24410, 25981, 26001]

txt = "反曲點 2+弛豫.csv"
data = np.genfromtxt(txt, dtype = None, delimiter = ',', skip_header = 0, filling_values = 0, invalid_raise = False, encoding = 'latin-1')
[Row,Column] = data.shape
variable = [0,1,2,3,4,5,6,7,10,11,12,13,14,15,16,17,18,19];      # 出廠前:[8,13,16,17,18,19,20,21,22,23,30,32,33,34,35]；回廠後:[42,47,50,51,52,53,54,55,56,57,64,66,67,68,69]
Variable = ['01 sec(前)','10 sec(前)','30 sec(前)','01 min(前)','03 min(前)','05 min(前)','07 min(前)','10 min(前)','01 sec(後)','10 sec(後)','30 sec(後)','01 min(後)','03 min(後)','05 min(後)','07 min(後)','10 min(後)','ift(前)','ift(後)']
item = []; sample = []; remove = []; R_feature = [];
for i in range(len(variable)):
    feature = data[1:,variable[i]]; soh = data[1:,9]; row = len(feature)
    if i<16:
        err_feature = [w for w,x in enumerate(feature) if x >= 0]
    else:
        err_feature = [w for w,x in enumerate(feature) if x==0 or x==2500]
 # 刪掉無數據(nan)或數據=0 的
    err = sorted(set(err_Chrtime+err_feature))
    feature_ok = np.delete(feature[:], err).astype(np.float32); soh_ok = np.delete(soh[:], err).astype(np.float32);
```


doi:10.6342/NTU202400743

```
    R = np.corrcoef(feature_ok, soh_ok)[0,1]; R_sq = np.round(R*R,3)     # 相關性分析
    item.append(Variable[i]); sample.append(len(soh_ok)); R_feature.append(R_sq);
    removed = len(soh_ok)-27064; remove.append(removed)     # 算有效數據占比

file = np.asarray([item,sample,remove,R_feature]).T
#np.savetxt(r'C:\Users\AMRG\Documents\Master_NiMH 研究+代碼\儲存分析結果(數據)\進出廠 ift+relax 分析結果.csv',file,fmt='%s',delimiter=',')
Table = pd.DataFrame(file, columns = ['變數','樣本數','剔除的樣本','相關性(R^2)'])
Table.style.set_properties(**{'text-align': 'middle'})
```

A.4 找出缺失值、異常值的程式碼

```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

txt = r"C:\Users\AMRG\Documents\Master_NiMH 研究+代碼\儲存分析結果(數據)\合併 796 檔案(raw)_補上 VPVD+VEOC.csv"
data = np.genfromtxt(txt, dtype = None, delimiter = ',', skip_header = 0, filling_values = 0, invalid_raise = False, encoding = 'latin-1')
[Row,Column] = data.shape
# print(data[1:,2]); print(type(data[1:,2]))
variable = [42,47,50,51,52,53,54,55,56,57,66,67,68,69];     # 出廠前:[8,13,16,17,18,19,20,21,22,23,30,32,33,34,35]；回廠後:[42,47,50,51,52,53,54,55,56,57,64,66,67,68,69]
Variable = ['DIS0_wh','CHR_Time','V15','V20','V25','V30','V35','DIS0_Time','RV10','VDT','VPVD','TptE','VEOC','VDmax']
item = []; sample = []; portion = []; R_feature = [];
for i in range(len(variable)):
    feature = data[1:,variable[i]]; soh = data[1:,48]; chrtime = data[1:,47]; row = len(feature)
    no_nan = np.empty((1,row), dtype = object);
    for h in range(row):          # 把 feature 中有 nan 的都換成 0
```


doi:10.6342/NTU202400743

```
17.         no_nan[0,h] = feature[h].replace('nan', '0')
18.     number = np.empty((1,row), dtype = object); SOH = np.empty((1,row), dtype = object); Chrtime = np.empty((1,row), dtype = object);
19.     for j in range(row):          # 把字串全部轉成數字
20.         number[0,j] = float(no_nan[0][j]); SOH[0,j] = np.round(float(soh[j])*86400/156,1); Chrtime[0,j] = np.round(float(chrtime[j])*86400,1)
21.     err_Chrtime = [u for u,v in enumerate(Chrtime[0]) if v <= 60]     # 刪掉充電時間過短的(<50 秒)
22.     err_number = [w for w,x in enumerate(number[0]) if x==0]        # 刪掉無數據(nan)或數據=0 的
23.     err_over = np.where(Chrtime[:]<SOH[:])[1].tolist()              # 放電比充電長的
24.     err = sorted(set(err_Chrtime+err_number))
      # +err_over
25.     number_ok = np.delete(number[:], err).astype(np.float32); SOH_ok = np.delete(SOH[:], err).astype(np.float32);
26.     R = np.corrcoef(number_ok, SOH_ok)[0,1]; R_sq = np.round(R*R,3)     # 相關性分析
27.     item.append(Variable[i]); sample.append(len(SOH_ok)); R_feature.append(R_sq);
28.     used = str(27064-len(SOH_ok)); portion.append(used)     # 算無效數據量；used = str(np.round(len(SOH_ok)*100/27064,1)) + '%'; portion.append(used)  # 算有效數據占比
29.
30. print("err_over\n", err_over)
31. print("err_Chrtime = \n",err_Chrtime); print('共有'+ str(len(err_Chrtime)) +'個modules 有充電問題')
32. file = np.asarray([item,sample,portion,R_feature]).T
33. np.savetxt(r'C:\Users\AMRG\Documents\Master_NiMH 研究+代碼\儲存分析結果(數據)\回廠 796 分析結果.csv',file,fmt='%s',delimiter=',')
34. Table = pd.DataFrame(file, columns = ['變數','樣本數','無效數據量','相關性(R^2)'])
35. Table.style.set_properties(**{'text-align': 'middle'})
```


doi:10.6342/NTU202400743

A.5 進行機器學習之前，進行數據清理

```
'選來做 ML 的模組'
import os
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import torch
from torch import nn, optim
from sklearn.model_selection import train_test_split
from sklearn import ensemble
from sklearn.metrics import mean_squared_error, mean_absolute_error, explained_variance_score, r2_score
import pylab as plot

a = '改掉 flect 的資料集.csv'
DIS0 = np.genfromtxt(a, dtype = None, delimiter = ',', skip_header = 0, filling_values = 0, invalid_raise = False, encoding = 'latin-1')[:, 3:4]
[Row,Column] = DIS0.shape
print(DIS0[0:2,:])
b = r'C:\Users\AMRG\Documents\aa_c35 預測 full\皓恆補特徵_27064.csv'
dat = np.genfromtxt(b, dtype = None, delimiter = ',', skip_header = 0, filling_values = 0, invalid_raise = False, encoding = 'latin-1')[:, 14:20]
[Row,Column] = dat.shape
print(dat[0:2,:])
data = np.hstack((dat, DIS0))

nolen = np.where(data[:, 1] >= 2219)[0]; print('len(nolen) =', len(nolen))
notim = np.where(data[:, 0] >= data[:, 1])[0]; print('len(notim) =', len(notim))
norax = np.where(data[:, 2] >= -0.01)[0]; print('len(norax) =', len(norax))
noinf = np.where(data[:, 3] == 0)[0]; print('len(noinf) =', len(noinf))
nov15 = np.where(data[:, 4] == 0)[0]; print('len(nov15) =', len(nov15))
novad = sorted(set(np.hstack((nolen, notim)))); print('novad =', len(novad))
novad = sorted(set(np.hstack((novad, norax)))); print('novad =', len(novad))
novad = sorted(set(np.hstack((novad, noinf)))); print('novad =', len(novad))
```


doi:10.6342/NTU202400743

```
32.   novad = sorted(set(np.hstack((novad, nov15)))); print('novad =', len(novad))
33.   novad = sorted(set(np.hstack((novad, np.asarray(err_Chrtime))))); print('nov
ad =', len(novad))
34.
35.   data[:, 0:2] = np.round(100*data[:, 0:2]/3600, 3)
36.   data = np.delete(data, novad, 0)
37.
38.   features_train2, features_test2, soh_train2, soh_test2 = train_test_split(da
ta[:,1:7], data[:,0:1].ravel(), test_size=0.2, random_state=100)
39.   print(f"features_train2 ({len(features_train2)}/{len(data)})\n", features_tr
ain2)
40.   print(f"soh_train2 ({len(soh_train2)}/{len(data)})\n", soh_train2)     #檢查
格式是否正確,才不會讓 ML 程式垮掉    #.ravel()
```

A.6 BRR、RF、FNN 程式碼

```
1.    from sklearn import linear_model     # BayesianRidge
2.    import time
3.    t1 = time.monotonic()
4.
5.    params_Baye = {"n_iter":700 , "tol":0.01, "alpha_1":1e-04, "alpha_2":1e-
06, "lambda_1":1e-06, "lambda_2":1 "lambda_2":1e-06}
6.    model = linear_model.BayesianRidge(tol=0.0001, fit_intercept=True, compute_s
core=True)   #
7.    model.set_params(**params_Baye)
8.    model.fit(features_train2, soh_train2)
9.    prediction = model.predict(features_test2)
10.   RMSE = np.round(math.sqrt(mean_squared_error(soh_test2, prediction)),3)
11.   MAE = np.round(mean_absolute_error(soh_test2, prediction),3)
12.   R_sq = np.round(r2_score(soh_test2, prediction),3)
13.   Residual = prediction-soh_test2          #.reshape((len(soh_test2),1))
14.   print("Residual", Residual)
15.
16.   t2 = time.monotonic()
17.   print("time elapsed:" + str(np.round(t2-t1,3)) + " seconds")
```

```
1.  import time
```


doi:10.6342/NTU202400743

```
t1 = time.monotonic()

rmseoos = []
ntreelist = range(50, 2050, 50)
# ntreelist = range(3, 19)
for itrees in ntreelist:
    depth = None
    maxft = 3
    model = ensemble.RandomForestRegressor(n_estimators=itrees, max_depth=depth, max_features=maxft, oob_score=False, random_state=100)
    model.fit(features_train2, soh_train2)
    predictions = model.predict(features_test2)
    rmseoos.append(math.sqrt(mean_squared_error(soh_test2, predictions)))
    print(itrees, end = " ")
R_sq = np.round(model.fit(features_train2, soh_train2).score(features_test2, soh_test2),3)
MAE = np.round(mean_absolute_error(predictions,soh_test2),4)
RMSE = np.round(np.sqrt(mean_squared_error(predictions,soh_test2)),4)
Residuals = predictions-soh_test2
print("Residuals =", Residuals)    #print("Best RMSE(%) = ", min(rmseoos))

t2 = time.monotonic()
print("time elapsed: " + str(np.round(t2-t1,3)) + " seconds")
```

```
  'FNN'
  import torch
  from torch import nn, optim
  from IPython import display
  import matplotlib.pyplot as plt
  from sklearn import preprocessing as pp

  scaler = pp.MinMaxScaler().fit(data[:,[1,2,3,4,5,6]])     # 默認數據壓縮范圍為[0,1]
  data_var = scaler.transform(data[:,[1,2,3,4,5,6]])
  data_SOH = data[:,0:1]
  data_all = np.hstack((data_SOH, data_var))     # 製作 normalized features+未調整 SOH 數據 1
  print("data_all\n", data_all[0])
```


doi:10.6342/NTU202400743

```
13.
14.   features_train, features_test, soh_train, soh_test = train_test_split(data_a
ll[:,[1,2,3,4,5,6]], data_all[:,0:1], test_size=0.2, random_state=100)
15.   print(f"features_train ({len(features_train)}/{len(data_all)})\n", features_
train)
16.   print(f"soh_train ({len(features_test)}/{len(data_all)})\n", soh_train)
#檢查格式是否正確,才不會讓 ML 程式垮掉
17.
18.   X_train = torch.from_numpy(features_train.astype(np.float64))
19.   y_train = torch.from_numpy(soh_train.astype(np.float64))
20.   X_test = torch.from_numpy(features_test.astype(np.float64))
21.   y_test = torch.from_numpy(soh_test.astype(np.float64))
22.
23.   D = 6  # dimensions
24.   C = 1  # num_classes
25.   H1 = 50  # num_hidden_units_1
26.   H2 = 10  # num_hidden_units_2
27.
28.   learning_rate = 0.03
29.   lambda_l2 = 0.0001
30.   n_networks = 1     # Number of networks
31.   criterion = torch.nn.MSELoss()     # nn package also has different loss func
tions. We use MSE for a regression task
32.   totalerr_train = []; totalerr_test = []; total_corr = [];
33.   train_set = []; test_set = []; eveloss = [];
34.   models = list(); y_pretrain = list()
35.   device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")
36.
37.   seed = 0
38.   torch.manual_seed(seed)     #固定 random states
39.
40.   for mod in range(n_networks):
41.       model = nn.Sequential(
42.           nn.Linear(D, H1),
43.           nn.ReLU(),
44.           nn.Linear(H1, H2),
45.           nn.ReLU(),
46.           nn.Linear(H2, C))
```


doi:10.6342/NTU202400743

```
47.       model.to(device)
48.     # Append models
49.       models.append(model)
50.     # Use the optim package to apply ADAM for our parameter updates
51.       optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate, weigh
t_decay=lambda_l2)     # built-in L2
52.       learning_rate = optim.lr_scheduler.StepLR(optimizer, 500, 0.9, last_epoc
h=-1)
53.       for t in range(30000):
54.           y_pred = model(X_train.float())
55.           if t == 0:
56.               y_pretrain.append(y_pred.detach())
57.         # Compute the loss and accuracy
58.           loss = torch.sqrt(criterion(y_pred.float(), y_train.float()))
59.           eveloss.append(np.round(loss.item(),6))
60.           print("start time: ", starttime[1:4])
61.           print(f"[EPOCH]: {t}, [LOSS]: {loss.item():.6f}")
62.           display.clear_output(wait=True)
63.         # zero the gradients before running the backward pass.
64.           optimizer.zero_grad()
65.         # Backward pass to compute the gradient of loss w.r.t our learnable pa
rams.
66.           loss.backward()
67.         # Update params
68.           optimizer.step()
69.           if t < 20001:
70.               learning_rate.step()
71.           if t == 29999:
72.               totalerr_train.append(np.round(loss.item(),6))
73.             ## Testing
74.               y_test_pred = model(X_test.float())
75.             # Compute the loss and accuracy
76.               loss_test = torch.sqrt(criterion(y_test_pred.float(), y_test.flo
at()))
77.               y_test_array = []; y_test_pred_array = [];
78.               for ii in range(len(y_test)):
79.                   y_test_array.append(y_test.detach().numpy()[ii][0].astype(np
.float64))     # [ii]後面有時候要加[0]有時不用,使用時請注意資料集種類
```


doi:10.6342/NTU202400743

```
                    y_test_pred_array.append(y_test_pred.detach().numpy()[ii][0].astype(np.float64))
                R_sq = np.round(r2_score(y_test_pred_array, y_test_array),3)
                MAE = np.round(mean_absolute_error(y_test_pred_array, y_test_array),3)
                RMSE = np.round(np.sqrt(mean_squared_error(y_test_pred_array,y_test_array)),3)
                Residualss = np.array(y_test_pred_array)-np.array(y_test_array)
                totalerr_test.append(np.round(loss_test.item(),6))
                total_corr.append(np.round(R_sq,3))
```

A.7 畫 ML 你和曲線圖、殘差分布圖、殘差百分比分布圖的程式碼

```
import seaborn as sns
import matplotlib
import scipy.stats as st
matplotlib.rcParams.update({'font.size': 15})

# predicted expect and calculte confidence interval
low_ci_bound, high_ci_bound = st.t.interval(0.95, len(prediction),
                                            loc=prediction,scale=Residual.std())
print("low_ci_bound\n", low_ci_bound)
print("high_ci_bound\n", high_ci_bound)
print("std of the SOH\n  = ", np.round(st.sem(prediction),3))

# plot confidence interval
plt.figure(figsize=(6,6), dpi = 100)   #
plt.plot(prediction,soh_test2, color='blue', linestyle='None' , marker='.', label=f'  R_sq = {R_sq}\n  MAE = {MAE}%\nRMSE = {RMSE}%')
plt.fill_between(sorted(prediction), sorted(low_ci_bound), sorted(high_ci_bound), color='green', alpha=0.5, label=' μ+2σ')
plt.xlabel("Predicted SOH(%)", fontsize=15)
plt.ylabel("Real SOH(%)", fontsize=15)
```


doi:10.6342/NTU202400743

```
plt.tick_params(labelsize=15)
plt.legend(fontsize=15)
plt.title("SOH prediction by BRR", fontsize=15)   #_No relaxation
#plt.grid()
plt.show()
```

```
'''''Residuals'''
from scipy import stats

# 取得 X_mean 的 mean, std, 95% 信賴區間
m = Residual.mean()                  # X_bar 的平均
se = Residual.std()                  # 標準誤
ci = np.round(stats.norm.interval(0.95, m, se),3)   # return [m - 1.96 std, m + 1.96 std]

print (f'  Sampling Mean: {m:.3f}')
print (f'Sampling StdErr: {se:.3f}')
print (f'95% Confidence Interval: {ci}')

sns.histplot(Residual, color='green', bins=50, kde=False)
plt.axvline(m, color='red', linestyle='dashed', linewidth=3, label=f'Mean: {m:.3f}')
plt.axvline(ci[0], color='green', linestyle='dashed', linewidth=3, label=f'95% Confidence\nInterval\n: {ci}')
plt.axvline(ci[1], color='green', linestyle='dashed', linewidth=3)
plt.xlabel('Residual(%)', fontsize=15)
plt.ylabel('numbers of module', fontsize=15)
plt.tick_params(labelsize=15)
plt.title('Residual distribution of BRR', fontsize=15)   #_No relaxation
plt.legend(fontsize=14);
#plt.grid();
plt.show()
```

```
'''''Percentage Error'''
pep_brr = Residual/soh_test2

# 取得 X_mean 的 mean, std, 95% 信賴區間
m1 = pep_brr.mean()                  # X_bar 的平均
```


doi:10.6342/NTU202400743

```
se1 = pep_brr.std()                    # 標準誤
ci1 = np.round(stats.norm.interval(0.95, m1, se1),3)   # return [m - 1.96 std, m + 1.96 std]

print (f'  Sampling Mean: {m1:.3f}')
print (f'Sampling StdErr: {se1:.3f}')
print (f'95% Confidence Interval: {ci1}')

sns.histplot(pep_brr, color='green', bins=50, kde=False)
plt.axvline(m1, color='red', linestyle='dashed', linewidth=3, label=f'Mean: {m1:.3f}')
plt.axvline(ci1[0], color='green', linestyle='dashed', linewidth=3, label=f'95% Confidence\nInterval\n: {ci1}')
plt.axvline(ci1[1], color='green', linestyle='dashed', linewidth=3)
plt.xlabel('Percentage Error', fontsize=15)
plt.ylabel('numbers of module', fontsize=15)
plt.tick_params(labelsize=15)
plt.title('Percentage Error distribution of BRR', fontsize=15)   #_No relaxation
plt.legend(fontsize=14);
#plt.grid();
plt.show()
```

A.8 MMD 的程式碼以及進行遷移學習的流程

```
    # 匯入服役數據
    link_f = r'C:\Users\AMRG\Documents\aa_c35 預測 full\皓恆補特徵_27064.csv'
    data_f = np.genfromtxt(link_f, dtype = 'float32', delimiter = ',', skip_header = 0, filling_values = 0, invalid_raise = False, encoding = 'latin-1')
    [Row,Column] = data_f.shape
    print('data_f =\n', data_f[[0,1,-1], :])

    # 列出 data1 完全充放電的條件、選擇要合併的數據
    d1_f0 = [w for w, x in enumerate(data_f[:, 4]) if x >= 1800]
    d1_f1 = [w for w, x in enumerate(data_f[:, 5]) if x > 2098]
    d1_f2 = [w for w, x in enumerate(data_f[:, 11]) if x > 0.0025 and x < 0.0035]
```


doi:10.6342/NTU202400743

```
d1_f3 = [w for w, x in enumerate(data_f[:, 12]) if x > 56 and x < 61]
d1_f4 = [w for w, x in enumerate(data_f[:, 24]) if x >= 6]
d1_fn = [w for w, x in enumerate(data_f[:, 12]) if x > 120]
d1_f2f3 = sorted(set(d1_f2 + d1_f3))
d1_f1fn = sorted(set(d1_f1 + d1_f4 + d1_fn + [5214, 12274, 12319]))
d1_f = []
for m in d1_f0:
    if m in d1_f2f3:
        d1_f.append(m)
print(f'1st: d1_f ({len(d1_f)})=\n', d1_f)
for n in d1_f1fn:
    if n in d1_f:
        d1_f.remove(n)
d1_f = [int(d1_f[i]) for i in range(len(d1_f))]
print(f'2nd: d1_f ({len(d1_f)})=\n', d1_f)

# 選需要合併數據的 data2
d2_f0 = [w for w, x in enumerate(data_f[:, 14]) if x >= 1800]
d2_f4 = [w for w, x in enumerate(data_f[:, 15]) if x > 2218]
d2_f1 = [w for w, x in enumerate(data_f[:, 21]) if x > 0.0025 and x < 0.0035
]
d2_f2 = [w for w, x in enumerate(data_f[:, 22]) if x > 56 and x < 61]
d2_f3 = [w for w, x in enumerate(data_f[:, 26]) if x >= 6]
d2_fn = [w for w, x in enumerate(data_f[:, 12]) if x > 120]
d2_f1f2 = sorted(set(d2_f1 + d2_f2))
d2_f3fn = sorted(set(d2_f3 + d2_f4 + d2_fn + [6917, 13282, 14720, 14734, 168
50, 18119, 19572, 20814]));
d2_f = []
for m in d2_f0:
    if m in d2_f1f2:
        d2_f.append(m)
print(f'1st: d2_f ({len(d2_f)})=\n', d2_f)
for n in d2_f3fn:
    if n in d2_f:
        d2_f.remove(n)
d2_f = [int(d2_f[i]) for i in range(len(d2_f))]
print(f'2nd: d2_f ({len(d2_f)})=\n', d2_f)

```


doi:10.6342/NTU202400743

```
47.   # 匯入老化數據
48.   rawf = r'C:\Users\AMRG\Desktop\0928 研究\預測(1)的 soh 和特徵\合併後再 norm\原
feature_27064.csv'
49.   mn_rawf = np.genfromtxt(rawf, dtype = None, delimiter = ',', skip_header = 0
, filling_values = 0, invalid_raise = False, encoding = 'latin-1')
50.   [Row,Column] = mn_rawf.shape
51.   print(f'mn_rawf ({len(mn_rawf)})=\n', mn_rawf[[0,-1], :])
52.
53.   # 合併選取的老化數據和服役數據
54.   aggf = mn_rawf[:771, :]
55.   dat1f = data_f[d1_f, 4:10]
56.   dat2f = data_f[d2_f, 14:20]
57.   mergef0 = np.vstack((aggf, dat1f))
58.   mergef = np.vstack((mergef0, dat2f))
59.   scaler_mergef = pp.MinMaxScaler().fit(mergef[:, :])
60.   scaler_mergef_nor = scaler_mergef.transform(mergef[:, :])
61.
62.   # 將數據 tensor 化(一)
63.   tor_feat = torch.zeros((len(scaler_mergef_nor), len(scaler_mergef_nor[0])))
64.   for i in range(len(tor_feat)):
65.       for j in range(len(scaler_mergef_nor[1,:])):
66.           tor_feat[i, j] = torch.tensor(scaler_mergef_nor[i,j])
67.           tor_4feat = tor_feat
68.   tor_volt = tor_feat[:, 1:]   # 特徵
69.
70.   # 依數據種類區分目標域和源域
71.   # Source 1
72.   v0_tra1 = tor_4volt[:441, curve]; print(f'v0_tra1 ({len(v0_tra1)})=\n', v0_t
ra1[[0, -1], :])
73.   f0_tra1 = tor_4feat[:441, 0:1]; print(f'f0_tra1 ({len(f0_tra1)})=\n', f0_tra
1[[0, -1], :])
74.   # Source 2
75.   v0_tra2 = tor_4volt[441:771, curve]; print(f'v0_tra2 ({len(v0_tra2)})=\n', v
0_tra2[[0, -1], :])
76.   f0_tra2 = tor_4feat[441:771, 0:1]; print(f'f0_tra2 ({len(f0_tra2)})=\n', f0_
tra2[[0, -1], :])
77.   # Target
```


doi:10.6342/NTU202400743

```
78.   v0_tes = tor_4volt[771:, curve]; print(f'v0_tes ({len(v0_tes)})=\n', v0_tes[
[0, -1], :])
79.   f0_tes = tor_4feat[771:, 0:1]; print(f'f0_tes ({len(f0_tes)})=\n', f0_tes[[0
, -1], :])
80.
81.   # MMD #
82.   from sklearn.datasets import make_classification
83.   from sklearn.svm import SVR
84.   from sklearn.metrics.pairwise import pairwise_kernels
85.   from sklearn.kernel_approximation import RBFSampler
86.   from sklearn import preprocessing as pp
87.
88.   def max_median_discrepancy(X, Y, kernel='rbf'):
89.       K_XX = pairwise_kernels(X, X, metric=kernel)
90.       K_XY = pairwise_kernels(X, Y, metric=kernel)
91.       K_YY = pairwise_kernels(Y, Y, metric=kernel)
92.       m = X.shape[0]
93.       n = Y.shape[0]
94.       mmd = np.sum(K_XX) / (m * (m - 1)) + np.sum(K_YY) / (n * (n - 1)) - 2 *
np.sum(K_XY) / (m * n)
95.       return mmd
96.
97.   mmds, RMSEs = [], []
98.   for i in range(1, 30):
99.       # 將樣本投影到高維空間（這裡使用 RBF 核函數的隨機映射）
100.      rbf_sampler = RBFSampler(gamma=10, n_components=i, random_state=0)
101.      v0_tra1_high_dim = rbf_sampler.fit_transform(v0_tra1)
102.      v0_tra2_high_dim = rbf_sampler.fit_transform(v0_tra2)
103.      v0_test_high_dim = rbf_sampler.transform(v0_tes)
104.      # 計算源領域和目標領域的 MMD
105.      mmd_s1s2 = max_median_discrepancy (v0_tra1_high_dim, v0_tra2_high_dim)
106.      mmd_s1ta = max_median_discrepancy (v0_tra1_high_dim, v0_test_high_dim)
107.      mmd_s2ta = max_median_discrepancy (v0_tra2_high_dim, v0_test_high_dim)
108.      mmd_value = mmd_s1s2/3 + mmd_s1ta/3 + mmd_s2ta/3
109.      mmds.append(float('%.3f'%mmd_value))
110.  print(' mmds =\n', mmds)
111.
112.  # 把 mmd 後的特徵併起來、分組 #
```


doi:10.6342/NTU202400743

```
unione_v = np.vstack((v0_tra1_high_dim, v0_tra2_high_dim))
unione_v = np.vstack((unione_v, v0_test_high_dim))
scal_uni_v = pp.MinMaxScaler().fit(unione_v[:, :])
scal_uni_v_nor = scal_uni_v.transform(unione_v[:, :])
unione_f = np.vstack((f0_tra1, f0_tra2))
unione_f = np.vstack((unione_f, f0_tes))

indice = np.arange(2785)
v0_train, v0_test, f0_train, f0_test, ind_train, ind_test = train_test_split(scal_uni_v_nor[:, :], unione_f[:, 0:1], indice, test_size=0.2, random_state=100)
tor_v0_train = [(v0_train[ii,:], f0_train[ii,:]) for ii in range(len(v0_train))]
tor_v0_test = [(v0_test[ii,:], f0_test[ii,:]) for ii in range(len(v0_test))]

# 建立 FNN #
torch.manual_seed(1)

ly01 = nn.Linear(19,25)
ly02 = nn.Linear(25,25)
ly03 = nn.Linear(25,5)
ly04 = nn.Linear(5,1)
class BaseMode2(nn.Module):
    def __init__(self):
        super().__init__()
        self.model = nn.Sequential(
            ly01,
            nn.ReLU(),
            ly02,
            nn.ReLU(),
            ly03,
            nn.ReLU(),
            ly04)
    def forward(self, x):
        output = self.model(x)
        return output

base_mode2 = BaseMode2()
criterion = nn.MSELoss()
```


doi:10.6342/NTU202400743

```
optimizer_2 = optim.Adam(base_mode2.parameters(), lr=0.02)
lr_2 = optim.lr_scheduler.StepLR(optimizer_2, 500, 0.9, last_epoch=-1)

# 訓練 FNN #
rmse02 = []
num_epochs = 30000
for epoch in range(num_epochs):
    for input_3, label_3 in DataLoader(tor_v0_train, batch_size=len(tor_v0_train), shuffle=True):
        optimizer_2.zero_grad()
        output_3 = base_mode2(input_3)
        loss_3 = criterion(output_3, label_3)
        loss_3.backward()
        optimizer_2.step()
        lr_2.step()
        loss_3 = math.sqrt(loss_3)
        loss_3 = "%.4f"%loss_3
        rmse02.append(float(loss_3))

# 驗證 FNN #
for input_3t, label_3t in DataLoader(tor_v0_test, batch_size=len(tor_v0_test), shuffle=True):
    output_3t = base_mode2(input_3t)
    loss_3t = criterion(output_3t, label_3t)
    loss_3t = math.sqrt(loss_3t)
    loss_3t = "%.4f"%loss_3t

# 畫出 FNN 預測結果 #
rev_output3 = output_3t.detach().numpy()
rev_label3 = label_3t.detach().numpy()

labels = ['discharge capacity', 'charge time', '3min relaxation','inflection time', 'V15', 'VPVD']
for ii in range(len(rev_output3[0])):
    R_sq = "%.3f"%np.corrcoef(rev_label3[:, ii].tolist(), rev_output3[:, ii].tolist())[0,1]
    _MSE = criterion(torch.tensor(rev_output3), torch.tensor(rev_label3))
    RMSE = 100*math.sqrt(_MSE)
```


doi:10.6342/NTU202400743

```
183.      RMSE = "%.2f"%RMSE
184.      plt.figure(figsize=(7,7), dpi = 100)
185.      plt.plot(rev_output3[:, ii], rev_label3[:, ii], "."
186.               , label=f'{labels[ii]}\nRMSE = {RMSE}%\n R_sq = {R_sq}')
187.      plt.plot(rev_label3[:, ii], rev_label3[:, ii], color = 'red', label = 'y
=x')
188.      plt.xlabel(f'predicted {labels[ii]}', fontsize=16)
189.      plt.ylabel(f'real {labels[ii]}', fontsize=16)
190.      plt.legend(fontsize=16)
191.      plt.show()
```


doi:10.6342/NTU202400743

A9 對隨機森林演算法做 Gridsearch 的結果

max_features	n_estimators	CV loss (%)	Test loss (%)
100	1	1.449	1.4194
100	2	1.435	1.3976
100	3	1.433	1.3996
100	4	1.444	1.3991
100	5	1.448	1.4067
100	6	1.456	1.4127
150	1	1.447	1.4150
150	2	1.433	1.3977
150	3	1.43	1.4002
150	4	1.441	1.3974
150	5	1.446	1.4053
150	6	1.455	1.4102
200	1	1.447	1.4135
200	2	1.429	1.3963
200	3	1.429	1.3983
200	4	1.44	1.3989
200	5	1.446	1.4026
200	6	1.455	1.4086
250	1	1.445	1.4114
250	2	1.428	1.3953
250	3	1.429	1.3970
250	4	1.439	1.3983
250	5	1.445	1.4037
250	6	1.454	1.4079
300	1	1.444	1.4096
300	2	1.428	1.3934
300	3	1.429	1.3969
300	4	1.438	1.3984
300	5	1.444	1.4032
300	6	1.454	1.4075
350	1	1.444	1.4083
350	2	1.427	1.3933
350	3	1.43	1.3961
350	4	1.438	1.3984
350	5	1.444	1.4017
350	6	1.453	1.4068
400	1	1.444	1.4077
400	2	1.427	1.3933
400	3	1.429	1.3956

doi:10.6342/NTU202400743

400	4	1.437	1.3983
400	5	1.444	1.4011
400	6	1.453	1.4065
450	1	1.444	1.4066
450	2	1.426	1.3925
450	3	1.429	1.3959
450	4	1.437	1.3975
450	5	1.443	1.4008
450	6	1.453	1.4054
500	1	1.444	1.4062
500	2	1.426	1.3930
500	3	1.429	1.3951
500	4	1.436	1.3973
500	5	1.443	1.4010
500	6	1.452	1.4054
550	1	1.444	1.4070
550	2	1.425	1.3927
550	3	1.428	1.3944
550	4	1.436	1.3979
550	5	1.442	1.4001
550	6	1.452	1.4055
600	1	1.443	1.4062
600	2	1.426	1.3924
600	3	1.428	1.3943
600	4	1.436	1.3979
600	5	1.442	1.4000
600	6	1.452	1.4054
650	1	1.443	1.4063
650	2	1.426	1.3915
650	3	1.428	1.3934
650	4	1.436	1.3979
650	5	1.442	1.4001
650	6	1.451	1.4055
700	1	1.442	1.4067
700	2	1.425	1.3910
700	3	1.429	1.3935
700	4	1.436	1.3977
700	5	1.442	1.4003
700	6	1.451	1.4053
750	1	1.442	1.4060
750	2	1.425	1.3905
750	3	1.428	1.3937

doi:10.6342/NTU202400743

750	4	1.436	1.3976
750	5	1.442	1.4001
750	6	1.451	1.4053
800	1	1.442	1.4055
800	2	1.425	1.3906
800	3	1.428	1.3937
800	4	1.435	1.3978
800	5	1.442	1.3999
800	6	1.451	1.4051
850	1	1.442	1.4059
850	2	1.425	1.3904
850	3	1.428	1.3935
850	4	1.435	1.3975
850	5	1.442	1.3997
850	6	1.451	1.4047
900	1	1.442	1.4060
900	2	1.425	1.3901
900	3	1.428	1.3930
900	4	1.435	1.3971
900	5	1.442	1.3995
900	6	1.451	1.4047
950	1	1.442	1.4058
950	2	1.425	1.3899
950	3	1.428	1.3932
950	4	1.435	1.3974
950	5	1.442	1.3997
950	6	1.451	1.4047
1000	1	1.443	1.4059
1000	2	1.425	1.3906
1000	3	1.428	1.3932
1000	4	1.435	1.3972
1000	5	1.442	1.3995
1000	6	1.451	1.4050
1050	1	1.443	1.4062
1050	2	1.425	1.3906
1050	3	1.428	1.3930
1050	4	1.435	1.3973
1050	5	1.442	1.3992
1050	6	1.451	1.4045
1100	1	1.442	1.4065
1100	2	1.425	1.3903
1100	3	1.428	1.3932

doi:10.6342/NTU202400743

1100	4	1.435	1.3976
1100	5	1.442	1.3996
1100	6	1.451	1.4046
1150	1	1.442	1.4063
1150	2	1.425	1.3901
1150	3	1.428	1.3933
1150	4	1.435	1.3974
1150	5	1.442	1.3995
1150	6	1.451	1.4044
1200	1	1.442	1.4058
1200	2	1.425	1.3901
1200	3	1.428	1.3934
1200	4	1.435	1.3975
1200	5	1.442	1.3997
1200	6	1.451	1.4045
1250	1	1.442	1.4055
1250	2	1.425	1.3901
1250	3	1.428	1.3936
1250	4	1.435	1.3975
1250	5	1.442	1.3997
1250	6	1.451	1.4042
1300	1	1.442	1.4056
1300	2	1.425	1.3901
1300	3	1.428	1.3936
1300	4	1.435	1.3974
1300	5	1.442	1.3995
1300	6	1.451	1.4047
1350	1	1.442	1.4058
1350	2	1.425	1.3903
1350	3	1.428	1.3938
1350	4	1.435	1.3974
1350	5	1.442	1.3996
1350	6	1.451	1.4047
1400	1	1.442	1.4053
1400	2	1.425	1.3903
1400	3	1.428	1.3937
1400	4	1.435	1.3973
1400	5	1.442	1.3998
1400	6	1.451	1.4046
1450	1	1.442	1.4053
1450	2	1.425	1.3902
1450	3	1.428	1.3934

doi:10.6342/NTU202400743

1450	4	1.435	1.3974
1450	5	1.442	1.4000
1450	6	1.451	1.4045
1500	1	1.442	1.4054
1500	2	1.425	1.3903
1500	3	1.428	1.3935
1500	4	1.435	1.3971
1500	5	1.442	1.4002
1500	6	1.451	1.4046
1550	1	1.442	1.4053
1550	2	1.424	1.3905
1550	3	1.428	1.3933
1550	4	1.435	1.3970
1550	5	1.442	1.4002
1550	6	1.451	1.4047
1600	1	1.442	1.4052
1600	2	1.424	1.3906
1600	3	1.428	1.3933
1600	4	1.435	1.3969
1600	5	1.442	1.4002
1600	6	1.451	1.4045
1650	1	1.442	1.4052
1650	2	1.425	1.3903
1650	3	1.428	1.3932
1650	4	1.434	1.3969
1650	5	1.442	1.4002
1650	6	1.451	1.4044
1700	1	1.442	1.4052
1700	2	1.424	1.3905
1700	3	1.428	1.3933
1700	4	1.434	1.3971
1700	5	1.442	1.4002
1700	6	1.451	1.4043
1750	1	1.443	1.4048
1750	2	1.425	1.3905
1750	3	1.428	1.3934
1750	4	1.434	1.3973
1750	5	1.442	1.4001
1750	6	1.451	1.4044
1800	1	1.443	1.4047
1800	2	1.425	1.3904
1800	3	1.427	1.3930

doi:10.6342/NTU202400743

1800	4	1.434	1.3972
1800	5	1.442	1.4000
1800	6	1.451	1.4043
1850	1	1.443	1.4050
1850	2	1.425	1.3902
1850	3	1.427	1.3930
1850	4	1.434	1.3971
1850	5	1.442	1.4000
1850	6	1.451	1.4043
1900	1	1.443	1.4052
1900	2	1.425	1.3900
1900	3	1.427	1.3931
1900	4	1.434	1.3972
1900	5	1.442	1.3998
1900	6	1.451	1.4044
1950	1	1.443	1.4052
1950	**2**	**1.424**	**1.3899**
1950	3	1.427	1.3930
1950	4	1.434	1.3974
1950	5	1.442	1.3999
1950	6	1.451	1.4045
2000	1	1.443	1.4051
2000	2	1.424	1.3900
2000	3	1.427	1.3930
2000	4	1.434	1.3975
2000	5	1.442	1.4001
2000	6	1.451	1.4044

doi:10.6342/NTU202400743

A10 對前饋神經網路做 Gridsearch 的結果

H1	H2	lr_init	factor	patience	CV loss	Test loss	Last lr
25	10	0.0000	0.5	5	2.1056	2.0889	0.0001
25	10	0.0001	0.5	10	2.1056	2.0889	0.0001
25	10	0.0001	0.5	20	2.1056	2.0889	0.0001
25	10	0.0001	0.2	5	2.1056	2.0889	0.0001
25	10	0.0001	0.2	10	2.1056	2.0889	0.0001
25	10	0.0001	0.2	20	2.1056	2.0889	0.0001
25	10	0.0001	0.5	5	1.6356	1.6397	0.0003
25	10	0.0003	0.5	10	1.6356	1.6397	0.0003
25	10	0.0003	0.5	20	1.6356	1.6397	0.0003
25	10	0.0003	0.2	5	1.6356	1.6397	0.0003
25	10	0.0003	0.2	10	1.6356	1.6397	0.0003
25	10	0.0003	0.2	20	1.6356	1.6397	0.0003
25	10	0.0003	0.5	5	1.5279	1.5378	0.001
25	10	0.001	0.5	10	1.5279	1.5378	0.001
25	10	0.001	0.5	20	1.5279	1.5378	0.001
25	10	0.001	0.2	5	1.5279	1.5378	0.001
25	10	0.001	0.2	10	1.5279	1.5378	0.001
25	10	0.001	0.2	20	1.5279	1.5378	0.001
25	10	0.001	0.5	5	1.5143	1.5221	0.002
25	10	0.002	0.5	10	1.5143	1.5221	0.002
25	10	0.002	0.5	20	1.5143	1.5221	0.002
25	10	0.002	0.2	5	1.5143	1.5221	0.002
25	10	0.002	0.2	10	1.5143	1.5221	0.002
25	10	0.002	0.2	20	1.5143	1.5221	0.002
25	10	0.002	0.5	5	1.4890	1.5046	0.003
25	10	0.003	0.5	10	1.4890	1.5046	0.003
25	10	0.003	0.5	20	1.4890	1.5046	0.003
25	10	0.003	0.2	5	1.4890	1.5046	0.003
25	10	0.003	0.2	10	1.4890	1.5046	0.003
25	10	0.003	0.2	20	1.4890	1.5046	0.003
25	10	0.003	0.5	5	1.5167	1.5294	0.005
25	10	0.01	0.5	10	1.5348	1.5369	0.005
25	10	0.01	0.5	20	1.5403	1.5392	0.005
25	10	0.01	0.2	5	1.5181	1.5270	0.002
25	10	0.01	0.2	10	1.5483	1.5511	0.002
25	10	0.01	0.2	20	1.5497	1.5536	0.002
25	25	0.01	0.5	5	1.7390	1.7410	0.0001
25	25	0.0001	0.5	10	1.7390	1.7410	0.0001

doi:10.6342/NTU202400743

25	25	0.0001	0.5	20	1.7390	1.7410	0.0001
25	25	0.0001	0.2	5	1.7390	1.7410	0.0001
25	25	0.0001	0.2	10	1.7390	1.7410	0.0001
25	25	0.0001	0.2	20	1.7390	1.7410	0.0001
25	25	0.0001	0.5	5	1.5584	1.5603	0.0003
25	25	0.0003	0.5	10	1.5584	1.5603	0.0003
25	25	0.0003	0.5	20	1.5584	1.5603	0.0003
25	25	0.0003	0.2	5	1.5584	1.5603	0.0003
25	25	0.0003	0.2	10	1.5584	1.5603	0.0003
25	25	0.0003	0.2	20	1.5584	1.5603	0.0003
25	25	0.0003	0.5	5	1.4911	1.4986	0.001
25	25	0.001	0.5	10	1.4911	1.4986	0.001
25	25	0.001	0.5	20	1.4911	1.4986	0.001
25	25	0.001	0.2	5	1.4911	1.4986	0.001
25	25	0.001	0.2	10	1.4911	1.4986	0.001
25	25	0.001	0.2	20	1.4911	1.4986	0.001
25	25	0.001	0.5	5	1.4704	1.4756	0.002
25	25	0.002	0.5	10	1.4704	1.4756	0.002
25	25	0.002	0.5	20	1.4704	1.4756	0.002
25	25	0.002	0.2	5	1.4704	1.4756	0.002
25	25	0.002	0.2	10	1.4704	1.4756	0.002
25	25	0.002	0.2	20	1.4704	1.4756	0.002
25	25	0.002	0.5	5	1.4646	1.4835	0.003
25	25	0.003	0.5	10	1.4646	1.4835	0.003
25	25	0.003	0.5	20	1.4646	1.4835	0.003
25	25	0.003	0.2	5	1.4646	1.4835	0.003
25	25	0.003	0.2	10	1.4646	1.4835	0.003
25	25	0.003	0.2	20	1.4646	1.4835	0.003
25	25	0.003	0.5	5	1.5111	1.5126	0.005
25	25	0.01	0.5	10	1.5134	1.5185	0.005
25	25	0.01	0.5	20	1.5058	1.5097	0.005
25	25	0.01	0.2	5	1.5175	1.5276	0.002
25	25	0.01	0.2	10	1.5373	1.5456	0.002
25	25	0.01	0.2	20	1.5363	1.5421	0.002
25	50	0.01	0.5	5	1.6837	1.6781	0.0001
25	50	0.0001	0.5	10	1.6837	1.6781	0.0001
25	50	0.0001	0.5	20	1.6837	1.6781	0.0001
25	50	0.0001	0.2	5	1.6837	1.6781	0.0001
25	50	0.0001	0.2	10	1.6837	1.6781	0.0001
25	50	0.0001	0.2	20	1.6837	1.6781	0.0001
25	50	0.0001	0.5	5	1.5266	1.5317	0.0003
25	50	0.0003	0.5	10	1.5266	1.5317	0.0003

doi:10.6342/NTU202400743

25	50	0.0003	0.5	20	1.5266	1.5317	0.0003
25	50	0.0003	0.2	5	1.5266	1.5317	0.0003
25	50	0.0003	0.2	10	1.5266	1.5317	0.0003
25	50	0.0003	0.2	20	1.5266	1.5317	0.0003
25	50	0.0003	0.5	5	1.4623	1.4741	0.001
25	50	0.001	0.5	10	1.4623	1.4741	0.001
25	50	0.001	0.5	20	1.4623	1.4741	0.001
25	50	0.001	0.2	5	1.4623	1.4741	0.001
25	50	0.001	0.2	10	1.4623	1.4741	0.001
25	50	0.001	0.2	20	1.4623	1.4741	0.001
25	50	0.001	0.5	5	1.4843	1.4933	0.002
25	50	0.002	0.5	10	1.4843	1.4933	0.002
25	50	0.002	0.5	20	1.4843	1.4933	0.002
25	50	0.002	0.2	5	1.4843	1.4933	0.002
25	50	0.002	0.2	10	1.4843	1.4933	0.002
25	50	0.002	0.2	20	1.4843	1.4933	0.002
25	50	0.002	0.5	5	1.4715	1.4862	0.003
25	50	0.003	0.5	10	1.4715	1.4862	0.003
25	50	0.003	0.5	20	1.4715	1.4862	0.003
25	50	0.003	0.2	5	1.4715	1.4862	0.003
25	50	0.003	0.2	10	1.4715	1.4862	0.003
25	50	0.003	0.2	20	1.4715	1.4862	0.003
25	50	0.003	0.5	5	1.4996	1.5220	0.005
25	50	0.01	0.5	10	1.5142	1.5284	0.005
25	50	0.01	0.5	20	1.5105	1.5183	0.005
25	50	0.01	0.2	5	1.5097	1.5196	0.002
25	50	0.01	0.2	10	1.5309	1.5362	0.002
25	50	0.01	0.2	20	1.5527	1.5586	0.002
50	10	0.01	0.5	5	1.7976	1.7995	0.0001
50	10	0.0001	0.5	10	1.7976	1.7995	0.0001
50	10	0.0001	0.5	20	1.7976	1.7995	0.0001
50	10	0.0001	0.2	5	1.7976	1.7995	0.0001
50	10	0.0001	0.2	10	1.7976	1.7995	0.0001
50	10	0.0001	0.2	20	1.7976	1.7995	0.0001
50	10	0.0001	0.5	5	1.5548	1.5673	0.0003
50	10	0.0003	0.5	10	1.5548	1.5673	0.0003
50	10	0.0003	0.5	20	1.5548	1.5673	0.0003
50	10	0.0003	0.2	5	1.5548	1.5673	0.0003
50	10	0.0003	0.2	10	1.5548	1.5673	0.0003
50	10	0.0003	0.2	20	1.5548	1.5673	0.0003
50	10	0.0003	0.5	5	1.4689	1.4840	0.001
50	10	0.001	0.5	10	1.4689	1.4840	0.001

doi:10.6342/NTU202400743

50	10	0.001	0.5	20	1.4689	1.4840	0.001
50	10	0.001	0.2	5	1.4689	1.4840	0.001
50	10	0.001	0.2	10	1.4689	1.4840	0.001
50	10	0.001	0.2	20	1.4689	1.4840	0.001
50	10	0.001	0.5	5	1.4561	1.4707	0.002
50	10	0.002	0.5	10	1.4561	1.4707	0.002
50	10	0.002	0.5	20	1.4561	1.4707	0.002
50	10	0.002	0.2	5	1.4561	1.4707	0.002
50	10	0.002	0.2	10	1.4561	1.4707	0.002
50	10	0.002	0.2	20	1.4561	1.4707	0.002
50	10	0.002	0.5	5	1.4923	1.5021	0.003
50	10	0.003	0.5	10	1.4923	1.5021	0.003
50	10	0.003	0.5	20	1.4923	1.5021	0.003
50	10	0.003	0.2	5	1.4923	1.5021	0.003
50	10	0.003	0.2	10	1.4923	1.5021	0.003
50	10	0.003	0.2	20	1.4923	1.5021	0.003
50	10	0.003	0.5	5	1.4831	1.5004	0.005
50	10	0.01	0.5	10	1.4948	1.5040	0.005
50	10	0.01	0.5	20	1.5098	1.5106	0.005
50	10	0.01	0.2	5	1.4737	1.4809	0.002
50	10	0.01	0.2	10	1.5032	1.5059	0.002
50	10	0.01	0.2	20	1.5147	1.5185	0.002
50	25	0.01	0.5	5	1.6616	1.6686	0.0001
50	25	0.0001	0.5	10	1.6616	1.6686	0.0001
50	25	0.0001	0.5	20	1.6616	1.6686	0.0001
50	25	0.0001	0.2	5	1.6616	1.6686	0.0001
50	25	0.0001	0.2	10	1.6616	1.6686	0.0001
50	25	0.0001	0.2	20	1.6616	1.6686	0.0001
50	25	0.0001	0.5	5	1.4910	1.4983	0.0003
50	25	0.0003	0.5	10	1.4910	1.4983	0.0003
50	25	0.0003	0.5	20	1.4910	1.4983	0.0003
50	25	0.0003	0.2	5	1.4910	1.4983	0.0003
50	25	0.0003	0.2	10	1.4910	1.4983	0.0003
50	25	0.0003	0.2	20	1.4910	1.4983	0.0003
50	25	0.0003	0.5	5	1.4509	1.4648	0.001
50	25	0.001	0.5	10	1.4509	1.4648	0.001
50	25	0.001	0.5	20	1.4509	1.4648	0.001
50	25	0.001	0.2	5	1.4509	1.4648	0.001
50	25	0.001	0.2	10	1.4509	1.4648	0.001
50	25	0.001	0.2	20	1.4509	1.4648	0.001
50	25	0.001	0.5	5	1.4404	1.4571	0.002
50	25	0.002	0.5	10	1.4404	1.4571	0.002

doi:10.6342/NTU202400743

50	25	0.002	0.5	20	1.4404	1.4571	0.002
50	25	0.002	0.2	5	1.4404	1.4571	0.002
50	25	0.002	0.2	10	1.4404	1.4571	0.002
50	25	0.002	0.2	20	1.4404	1.4571	0.002
50	25	0.002	0.5	5	1.4562	1.4579	0.003
50	25	0.003	0.5	10	1.4562	1.4579	0.003
50	25	0.003	0.5	20	1.4562	1.4579	0.003
50	25	0.003	0.2	5	1.4562	1.4579	0.003
50	25	0.003	0.2	10	1.4562	1.4579	0.003
50	25	0.003	0.2	20	1.4562	1.4579	0.003
50	25	0.003	0.5	5	1.5254	1.5547	0.005
50	25	0.01	0.5	10	1.5099	1.5103	0.005
50	25	0.01	0.5	20	1.4853	1.4868	0.005
50	25	0.01	0.2	5	1.4817	1.4972	0.002
50	25	0.01	0.2	10	1.4948	1.5032	0.002
50	25	0.01	0.2	20	1.5045	1.5108	0.002
50	50	0.01	0.5	5	1.6155	1.6258	0.0001
50	50	0.0001	0.5	10	1.6155	1.6258	0.0001
50	50	0.0001	0.5	20	1.6155	1.6258	0.0001
50	50	0.0001	0.2	5	1.6155	1.6258	0.0001
50	50	0.0001	0.2	10	1.6155	1.6258	0.0001
50	50	0.0001	0.2	20	1.6155	1.6258	0.0001
50	50	0.0001	0.5	5	1.4696	1.4799	0.0003
50	50	0.0003	0.5	10	1.4696	1.4799	0.0003
50	50	0.0003	0.5	20	1.4696	1.4799	0.0003
50	50	0.0003	0.2	5	1.4696	1.4799	0.0003
50	50	0.0003	0.2	10	1.4696	1.4799	0.0003
50	50	0.0003	0.2	20	1.4696	1.4799	0.0003
50	50	0.0003	0.5	5	1.4208	1.4369	0.001
50	50	0.001	0.5	10	1.4208	1.4369	0.001
50	50	0.001	0.5	20	1.4208	1.4369	0.001
50	50	0.001	0.2	5	1.4208	1.4369	0.001
50	50	0.001	0.2	10	1.4208	1.4369	0.001
50	50	0.001	0.2	20	1.4208	1.4369	0.001
50	50	0.001	0.5	5	1.4407	1.4552	0.002
50	50	0.002	0.5	10	1.4407	1.4552	0.002
50	50	0.002	0.5	20	1.4407	1.4552	0.002
50	50	0.002	0.2	5	1.4407	1.4552	0.002
50	50	0.002	0.2	10	1.4407	1.4552	0.002
50	50	0.002	0.2	20	1.4407	1.4552	0.002
50	50	0.002	0.5	5	1.4411	1.4561	0.003
50	50	0.003	0.5	10	1.4411	1.4561	0.003

doi:10.6342/NTU202400743

50	50	0.003	0.5	20	1.4411	1.4561	0.003
50	50	0.003	0.2	5	1.4411	1.4561	0.003
50	50	0.003	0.2	10	1.4411	1.4561	0.003
50	50	0.003	0.2	20	1.4411	1.4561	0.003
50	50	0.003	0.5	5	1.5422	1.4919	0.005
50	50	0.01	0.5	10	1.4609	1.4612	0.005
50	50	0.01	0.5	20	1.4832	1.4881	0.005
50	50	0.01	0.2	5	1.4720	1.4734	0.002
50	50	0.01	0.2	10	1.4704	1.4742	0.002
50	50	0.01	0.2	20	1.4857	1.4971	0.002
100	10	0.0001	0.5	5	1.6411	1.6491	0.0001
100	10	0.0001	0.5	10	1.6411	1.6491	0.0001
100	10	0.0001	0.5	20	1.6411	1.6491	0.0001
100	10	0.0001	0.2	5	1.6411	1.6491	0.0001
100	10	0.0001	0.2	10	1.6411	1.6491	0.0001
100	10	0.0001	0.2	20	1.6411	1.6491	0.0001
100	10	0.0003	0.5	5	1.5047	1.5133	0.0003
100	10	0.0003	0.5	10	1.5047	1.5133	0.0003
100	10	0.0003	0.5	20	1.5047	1.5133	0.0003
100	10	0.0003	0.2	5	1.5047	1.5133	0.0003
100	10	0.0003	0.2	10	1.5047	1.5133	0.0003
100	10	0.0003	0.2	20	1.5047	1.5133	0.0003
100	10	0.001	0.5	5	1.4443	1.4612	0.001
100	10	0.001	0.5	10	1.4443	1.4612	0.001
100	10	0.001	0.5	20	1.4443	1.4612	0.001
100	10	0.001	0.2	5	1.4443	1.4612	0.001
100	10	0.001	0.2	10	1.4443	1.4612	0.001
100	10	0.001	0.2	20	1.4443	1.4612	0.001
100	10	0.002	0.5	5	1.4324	1.4485	0.002
100	10	0.002	0.5	10	1.4324	1.4485	0.002
100	10	0.002	0.5	20	1.4324	1.4485	0.002
100	10	0.002	0.2	5	1.4324	1.4485	0.002
100	10	0.002	0.2	10	1.4324	1.4485	0.002
100	10	0.002	0.2	20	1.4324	1.4485	0.002
100	10	0.003	0.5	5	1.4657	1.4759	0.003
100	10	0.003	0.5	10	1.4657	1.4759	0.003
100	10	0.003	0.5	20	1.4657	1.4759	0.003
100	10	0.003	0.2	5	1.4657	1.4759	0.003
100	10	0.003	0.2	10	1.4657	1.4759	0.003
100	10	0.003	0.2	20	1.4657	1.4759	0.003
100	10	0.01	0.5	5	1.4559	1.4680	0.005
100	10	0.01	0.5	10	1.4729	1.4810	0.005

doi:10.6342/NTU202400743

100	10	0.01	0.5	20	1.4739	1.4928	0.005
100	10	0.01	0.2	5	1.4551	1.4737	0.002
100	10	0.01	0.2	10	1.4731	1.4826	0.002
100	10	0.01	0.2	20	1.4779	1.4887	0.002
100	25	0.0001	0.5	5	1.5664	1.5721	0.0001
100	25	0.0001	0.5	10	1.5664	1.5721	0.0001
100	25	0.0001	0.5	20	1.5664	1.5721	0.0001
100	25	0.0001	0.2	5	1.5664	1.5721	0.0001
100	25	0.0001	0.2	10	1.5664	1.5721	0.0001
100	25	0.0001	0.2	20	1.5664	1.5721	0.0001
100	25	0.0003	0.5	5	1.4535	1.4597	0.0003
100	25	0.0003	0.5	10	1.4535	1.4597	0.0003
100	25	0.0003	0.5	20	1.4535	1.4597	0.0003
100	25	0.0003	0.2	5	1.4535	1.4597	0.0003
100	25	0.0003	0.2	10	1.4535	1.4597	0.0003
100	25	0.0003	0.2	20	1.4535	1.4597	0.0003
100	25	0.001	0.5	5	1.4231	1.4352	0.001
100	25	0.001	0.5	10	1.4231	1.4352	0.001
100	25	0.001	0.5	20	1.4231	1.4352	0.001
100	25	0.001	0.2	5	1.4231	1.4352	0.001
100	25	0.001	0.2	10	1.4231	1.4352	0.001
100	25	0.001	0.2	20	1.4231	1.4352	0.001
100	25	0.002	0.5	5	1.4295	1.4458	0.002
100	25	0.002	0.5	10	1.4295	1.4458	0.002
100	25	0.002	0.5	20	1.4295	1.4458	0.002
100	25	0.002	0.2	5	1.4295	1.4458	0.002
100	25	0.002	0.2	10	1.4295	1.4458	0.002
100	25	0.002	0.2	20	1.4295	1.4458	0.002
100	25	0.003	0.5	5	1.4574	1.4729	0.003
100	25	0.003	0.5	10	1.4574	1.4729	0.003
100	25	0.003	0.5	20	1.4574	1.4729	0.003
100	25	0.003	0.2	5	1.4574	1.4729	0.003
100	25	0.003	0.2	10	1.4574	1.4729	0.003
100	25	0.003	0.2	20	1.4574	1.4729	0.003
100	25	0.01	0.5	5	1.4807	1.4978	0.005
100	25	0.01	0.5	10	1.4803	1.4939	0.005
100	25	0.01	0.5	20	1.4925	1.5094	0.005
100	25	0.01	0.2	5	1.4487	1.4637	0.002
100	25	0.01	0.2	10	1.4830	1.4984	0.002
100	25	0.01	0.2	20	1.4817	1.4970	0.002
100	50	0.0001	0.5	5	1.5348	1.5436	0.0001
100	50	0.0001	0.5	10	1.5348	1.5436	0.0001

doi:10.6342/NTU202400743

100	50	0.0001	0.5	20	1.5348	1.5436	0.0001
100	50	0.0001	0.2	5	1.5348	1.5436	0.0001
100	50	0.0001	0.2	10	1.5348	1.5436	0.0001
100	50	0.0001	0.2	20	1.5348	1.5436	0.0001
100	50	0.0003	0.5	5	1.4358	1.4511	0.0003
100	50	0.0003	0.5	10	1.4358	1.4511	0.0003
100	50	0.0003	0.5	20	1.4358	1.4511	0.0003
100	50	0.0003	0.2	5	1.4358	1.4511	0.0003
100	50	0.0003	0.2	10	1.4358	1.4511	0.0003
100	50	0.0003	0.2	20	1.4358	1.4511	0.0003
100	50	0.001	0.5	5	1.4211	1.4386	0.001
100	50	0.001	0.5	10	1.4211	1.4386	0.001
100	50	0.001	0.5	20	1.4211	1.4386	0.001
100	50	0.001	0.2	5	1.4211	1.4386	0.001
100	50	0.001	0.2	10	1.4212	1.4386	0.001
100	50	0.001	0.2	20	1.4212	1.4386	0.001
100	**50**	**0.002**	**0.5**	**5**	**1.4073**	**1.4256**	**0.002**
100	**50**	**0.002**	**0.5**	**10**	**1.4073**	**1.4256**	**0.002**
100	**50**	**0.002**	**0.5**	**20**	**1.4073**	**1.4256**	**0.002**
100	**50**	**0.002**	**0.2**	**5**	**1.4073**	**1.4256**	**0.002**
100	**50**	**0.002**	**0.2**	**10**	**1.4073**	**1.4256**	**0.002**
100	**50**	**0.002**	**0.2**	**20**	**1.4073**	**1.4256**	**0.002**
100	50	0.003	0.5	5	1.4497	1.4619	0.003
100	50	0.003	0.5	10	1.4497	1.4619	0.003
100	50	0.003	0.5	20	1.4497	1.4619	0.003
100	50	0.003	0.2	5	1.4497	1.4619	0.003
100	50	0.003	0.2	10	1.4497	1.4619	0.003
100	50	0.003	0.2	20	1.4497	1.4619	0.003
100	50	0.01	0.5	5	1.4855	1.4992	0.005
100	50	0.01	0.5	10	1.4661	1.4843	0.005
100	50	0.01	0.5	20	1.4871	1.4986	0.005
100	50	0.01	0.2	5	1.4425	1.4594	0.002
100	50	0.01	0.2	10	1.4682	1.4798	0.002
100	50	0.01	0.2	20	1.4594	1.4740	0.002

doi:10.6342/NTU202400743