POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Domain Adaptation for Remaining Useful Life Estimation of Lithium-Ion Batteries

Tesi di Laurea Magistrale in
Computer Science and Engineering - Ingegneria Informatica

Authors: **Giuseppe Meli, Luca Pasquarelli**

Student IDs: 992313 102200
Advisor: Prof. Francesco Amigoni
Co-advisors: Prof. Loredana Cristaldi, Dott. Davide Azzalini, Dott. Luca Martiri
Academic Year: 2022-2023

Abstract

Nowadays there are more electronic devices than human beings on Earth. These devices are usually powered using lithium-ion batteries. Each of these batteries can assume a completely different behavior from its peers based on usage, charging, and many other factors, leading to potential harm and many major potential issues depending on the importance and purpose of the given device being powered. It stems from here the importance of being able to accurately predict the Remaining Useful Life (RUL) for a given set of batteries, and most importantly creating a model that is capable of generalizing across different sets of batteries.

This thesis introduces a Domain Adversarial Neural Network (DANN) to align feature representations across different domains, effectively harmonizing feature distributions. The DANN's unsupervised learning approach is in this thesis compared with the more traditional Transfer Learning and Fine-Tuning approaches, in the Direct Estimation of RUL. The DANN methodology as we'll cover in our thesis, offers a distinct advantage over traditional transfer learning methods, as it does not rely on explicit labels from source or target domains, making it particularly valuable in scenarios with limited labeled data availability. Additionally, its flexibility allows seamless integration into various neural architectures, as in the Convolutional-LSTM neural network improved with the use of an attention mechanism, that we used in our thesis. This makes the DANN an attractive choice for domain adaptation tasks.

This research also highlights the benefits of traditional data augmentation techniques, particularly when tailored to the dataset's specific characteristics. These techniques contribute to improved model performance, enhancing the predictions' accuracy.

We evaluated our work with the MIT-Toyota 2019 collaboration dataset, which is the largest lithium-ion batteries dataset publicly available.

Keywords: domain adaptation, domain adversarial learning, remaining useful life estimation, lithium-ion batteries, deep learning

Abstract in lingua italiana

Oggi ci sono più dispositivi elettronici che esseri umani sulla Terra. Questi dispositivi sono solitamente alimentati da batterie agli ioni di litio. Ciascuna di queste batterie può assumere un comportamento completamente diverso dalle altre in base all'uso, alla carica e a molti altri fattori, con conseguenti potenziali danni e molti altri potenziali problemi a seconda dell'importanza e dello scopo del dispositivo alimentato. Da qui l'importanza di essere in grado di prevedere con precisione la vita utile residua (RUL) per un determinato insieme di batterie e, soprattutto, di creare un modello che sia in grado di generalizzare tra diversi insiemi di batterie.

Questa tesi introduce una Domain Adversarial Neural Network (DANN) per allineare le rappresentazioni delle caratteristiche in diversi domini, armonizzando efficacemente le distribuzioni delle caratteristiche. L'approccio di apprendimento non supervisionato della DANN viene confrontato in questa tesi con i più tradizionali approcci di Transfer Learning e Fine-Tuning, nella stima diretta della RUL. La metodologia DANN, come illustreremo nella nostra tesi, offre un netto vantaggio rispetto ai metodi tradizionali di apprendimento per trasferimento, in quanto non si basa su etichette esplicite provenienti dai domini di origine o di destinazione, il che la rende particolarmente preziosa in scenari con una disponibilità limitata di dati etichettati. Inoltre, la sua flessibilità consente una perfetta integrazione in varie architetture neurali, come nella rete neurale convoluzionale-LSTM migliorata con l'uso di un meccanismo di attenzione, che abbiamo utilizzato nella nostra tesi.

Ciò rende la DANN una scelta interessante per i compiti di adattamento al dominio.

Questa ricerca evidenzia anche i vantaggi delle tecniche tradizionali di data augmentation, in particolare se adattate alle caratteristiche specifiche del set di dati. Queste tecniche contribuiscono a migliorare le prestazioni del modello, aumentando l'accuratezza delle previsioni.

Abbiamo valutato il nostro lavoro con il dataset della collaborazione MIT-Toyota 2019, che è il più grande dataset di batterie agli ioni di litio disponibile pubblicamente.

Keywords: adattamento al dominio, apprendimento avversario del dominio, stima della vita utile residua, batterie agli ioni di litio, deep learning

Contents

1 | Introduction

Lithium-ion batteries are widely employed due to their numerous advantages, such as high output power, extended cycle life, high energy density, and eco-friendliness [46]. These batteries find applications in diverse industries, including electronics, aerospace, and military. However, prolonged usage can also cause a rapid deterioration of lithium-ion batteries' discharge capacity due to a variety of microphenomena that occur inside the battery and whose entanglement creates a strongly non-linear and scarcely moldable degradation.

To ensure the safe and effective operation of batteries, the implementation of a Battery Management System (BMS) is essential [24]. BMS's play a critical role in intelligently managing each unit. This strategic management is crucial not only to enhance service life but also to guarantee safety. State Of Health (SOH) and the prediction of Remaining Useful Life (RUL) are integral aspects of the BMS framework.

In recent years, the field has witnessed a surge in the use of data-driven methods for fault diagnosis and prediction [38]. This trend has been driven by advances in computing power and data processing capacity. The integration of these sophisticated methods within BMS's has significantly contributed to the improvement of battery management and overall performance in various applications.

Our research aims to contribute to expedited deployment and cost reduction in the production phase of batteries by building one model trained on one specific dataset of batteries, that is capable of delivering good performances also on a different set of batteries, while accounting for an unavoidable loss due to unpredictability and diversity in battery behavior. We must remember that accurate RUL estimation is not merely a technical requirement but holds paramount importance in averting potential unreliability, mitigating risks of harm.

We employ a sophisticated architecture, specifically utilizing a Convolutional Long Short-Term Memory Neural Network (LSTM) with an Attention Layer, supported by a Domain Adversarial Neural Network (DANN) in the training phase [41]. The DANN serves as a crucial tool for aligning feature representations across diverse domains, thereby harmo-

nizing feature distributions. To assess the effectiveness and versatility of our proposed methodology, we utilized the MIT-Toyota 2019 collaboration dataset. This dataset, one of the most extensive publicly available collections, comprises a diverse set of batches of lithium-ion battery data.

The practical relevance of our research is underscored by its potential to significantly advance RUL estimation methodologies for lithium-ion batteries. Notably, our emphasis on addressing challenges associated with domain adaptation introduces a pioneering approach in the field, as domain adversarial learning has not been extensively explored in the context of RUL estimation. The use of DANN offers a distinctive advantage, as its unsupervised learning approach eliminates the dependency on explicit labels from source or target domains. This is particularly valuable in scenarios where labeled data is limited or costly, facilitating practical implementation in real-world applications.

Furthermore, the incorporation of traditional data augmentation techniques enhances the overall model performance, contributing to more accurate predictions. Our study's systematic method for optimizing hyperparameters in the convolutional LSTM model not only increases prediction accuracy but also reduces the time investment compared to earlier manual methods.

This research, therefore, holds practical significance in advancing the reliability and efficiency of lithium-ion batteries, which serve as indispensable components in the electronic devices that permeate our daily lives.

Our experiments showed that utilizing a DANN holds promise for RUL estimation, highlighting its distinctive unsupervised learning paradigm and adaptability compared to conventional transfer learning methods. However, in instances of limited data, as evidenced by our study involving only eight battery cells' data, our experiments showed that fine-tuning outperformed domain adversarial learning. The effectiveness of fine-tuning was evident in its ability to leverage labeled data for target-specific insights, aligning closely with the unique attributes of the target domain and surpassing the domain invariance goal of adversarial learning. Furthermore, our experiments demonstrated good performance improvements through the incorporation of traditional data augmentation techniques, such as jittering [21], especially when manually adjusted to align with the specific characteristics of our dataset.

1.1. Outline of the thesis

This thesis is structured as follows. Chapter 2 provides an overview of the background of this work, incorporating the definition of lithium-ion batteries' features, remaining useful life, and the significance of accurate RUL estimations. Additionally, Chapter 2 includes a discussion on the basics of domain adaptation. Chapter 3 delves into various approaches to battery RUL estimation, with a particular focus on domain adaptation. It also provides a broad overview of general methodologies for domain adaptation. Chapter 4 offers a detailed description of the method proposed in this thesis, encompassing its input features, model architecture, and the implementation specifics of the Domain Adversarial Neural Network (DANN). This chapter serves to elucidate the unique aspects of our approach in the context of both RUL estimation and domain adaptation. Chapter 5 presents the datasets used to validate our work, elucidates how training is conducted on these different datasets, and presents the results obtained. Finally, Chapter 6 concludes the thesis, summarizing key findings and proposing possible future directions for research and development.

2 | Background

In the realm of modern technological advancements, lithium-ion batteries stand as a testament to innovation. Their longevity and reliability are paramount, and predicting their operational lifespan has become a pressing necessity. This background section aims to provide a holistic understanding of the challenges and methodologies associated with Remaining Useful Life (RUL) estimation, especially when confronted with limited and diverse data.

2.1. Introduction to Lithium-Ion Batteries

A lithium-ion battery often denoted as Li-ion, stands as one of the most prevalent rechargeable batteries, powering a vast array of electronic devices, ranging from everyday gadgets like cell phones and laptops to more demanding applications such as electric vehicles. The cornerstone of its energy storage mechanism is the movement of lithium ions, which shift between the positive and negative electrodes of the battery.

One of the most interesting features of lithium-ion batteries is their high energy density. This characteristic allows them to store a significant amount of energy relative to their physical size and weight, making them an ideal choice for applications where compactness and lightness are crucial. Additionally, they boast a low self-discharge rate, ensuring that they retain their charge for extended periods when not in active use. However, these batteries, if subjected to inappropriate usage or charging conditions, can be susceptible to an impressive degeneration in capacity or even damage, especially when exposed to extreme temperatures.

To further understand lithium-ion batteries, let us characterize them by several parameters:

- **Physical size and weight**

  These attributes can vary considerably based on the battery's intended application and its capacity. For instance, a battery designed for a cell phone will differ in size and weight from one intended for an electric vehicle.

- **Voltage**

  Defined as the potential difference between the battery's positive and negative electrodes, the voltage for a typical single lithium-ion cell hovers around 3.6-3.7 volts.

- **Capacity**

  Denoted as Q, the capacity of a battery provides a measure of its energy storage capability. It is commonly expressed in units of milliampere-hour (mAh). Depending on the battery's design and purpose, its capacity can span from a few hundred mAh, for smartphones, to several thousand mAh for larger applications.

- **Discharge rate**

  This parameter, often measured in ampere (A) or milliampere (mA), signifies the rate at which the battery can dispense its stored energy. Influenced by factors such as temperature, battery age, and its inherent design, the discharge rate is pivotal in determining how swiftly a device can draw power from the battery.

In essence, lithium-ion batteries, with their myriad advantages and few caveats, have revolutionized the landscape of portable energy storage, offering both efficiency and compactness. Still, they present a set of challenges since their degradation in performance is not linear and due to many factors.

2.2. Charge-Discharge Curves of Li-Ion Cells

As in Figure 2.1, a lithium-ion battery's charge curve is a plot where various variables may be seen, including charging time, voltage, charging current, and charged capacity as a dotted black line expressed in terms of State of Charge. The cells must be charged using a constant current and constant voltage (CC-CV) technique.

This approach is made of various phases:

- **Constant Current (CC)**

  The battery is charged at a constant current until the voltage reaches the full charge cut-off voltage. There is a drop in the charging current.

- **Constant Voltage (CV)**

  The battery is charged at a constant voltage and the current keeps decreasing until the cells are fully charged.

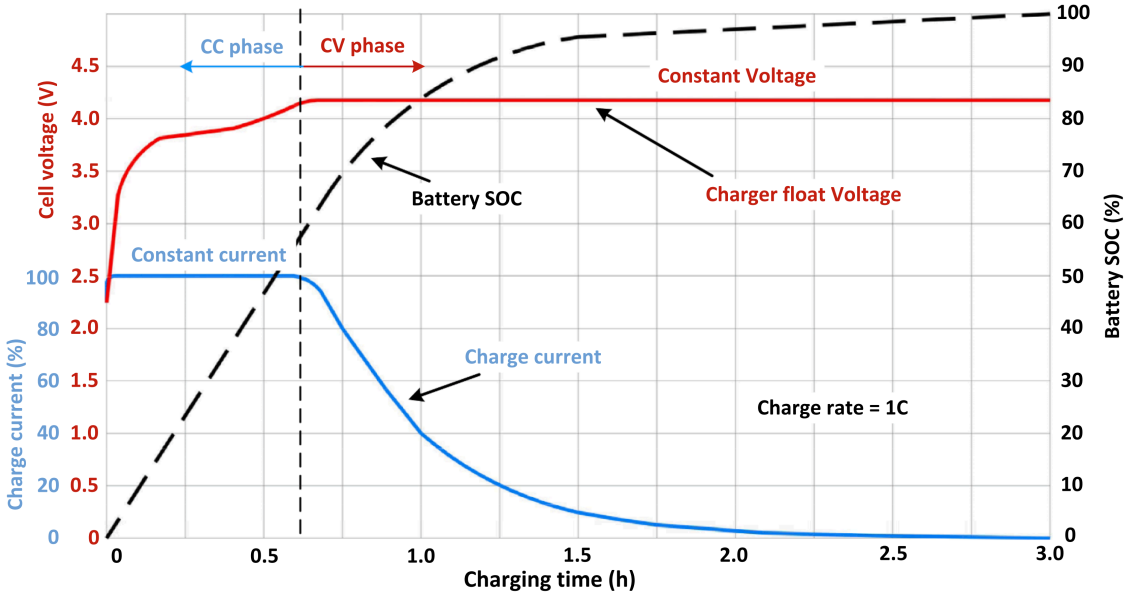


Figure 2.1: Charge-Discharge Curves of Li-ion cells [40].

- **Trickle Charge**

  The battery reaching its full charge voltage at this stage does not mean that it is 100% charged. Trickle charge mode consists of charging the battery at a very low current to reach 100%.

- **Conclusion**

  Once every cell has been balanced and has reached its full charge voltage, the battery pack is really 100% charged.

The industry standard is to provide 80% fast charge, then the charging current is reduced to reach 100% charge. This is done to preserve the battery's lifespan.

2.2.1. Discharge Curve and Discharge Capacity

The discharge curve is a plot of voltage against the percentage of capacity discharged. A flat discharge curve is desirable, as this means that the voltage remains constant as the battery is used up. But a flat discharge curve also means the battery might not deliver close to 100% Depth of Discharge (DoD) because the battery cuts off if one of the cells reaches its lower cut-off voltage, we can see an example in Figure 2.2.

On the other hand, discharge capacity, often expressed in units such as ampere-hours (Ah) or milliampere-hours (mAh), is a fundamental metric that characterizes a battery's ability to deliver electrical energy. It represents the total amount of energy a battery can

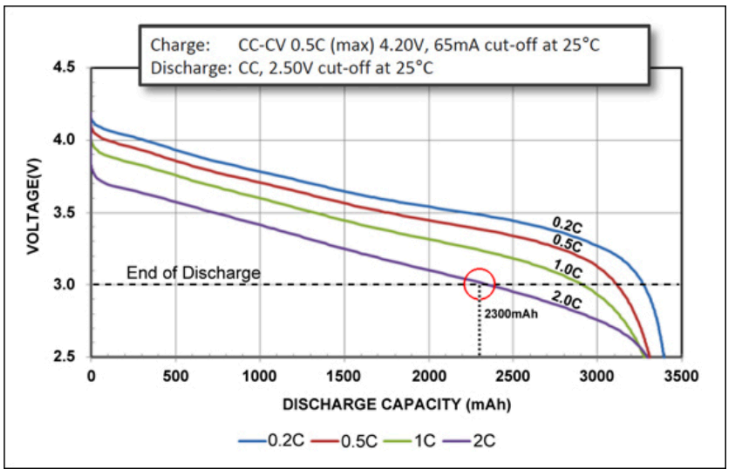


Figure 2.2: Discharge Curves of Li-ion cells [1].

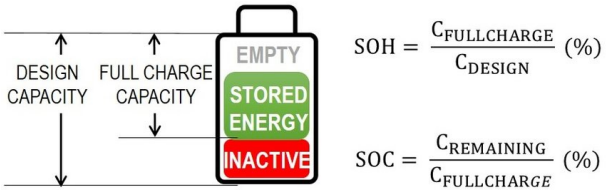


Figure 2.3: Batteries metrics [2].

provide when it is discharged from a fully charged state to a specified lower voltage or state of charge. Discharge capacity is a critical parameter, as it directly influences the runtime and performance of devices or systems powered by the battery. Batteries with higher discharge capacities can deliver more energy before reaching a predetermined lower voltage or state of charge, ensuring longer operational periods and enhanced utility. In contrast, batteries with lower discharge capacities exhibit shorter runtimes and energy delivery capabilities, which are significant considerations when selecting the right battery for specific applications. Understanding discharge capacity is essential for effective battery management and optimizing the performance of electronic devices, portable equipment, and various power systems.

2.3. State of Health (SoH) and State of Charge (SoC)

2.3.1. State of Health (SoH)

The State of Health (SoH) of a battery is a measure of its long-term condition and overall performance relative to its initial state. SoH is an essential metric for assessing a battery's health and predicting its remaining lifespan. It takes into account various factors that

influence a battery's condition. These include its capacity, internal resistance, and aging, which can be affected by factors such as temperature, charge-discharge cycles, rate of charge and discharge, and depth of discharge [46].
SoH is typically expressed as a percentage, with 100% representing a battery in its new and optimal state. A lower SoH percentage indicates a battery's reduced capacity and overall health.

2.3.2. State of Charge (SoC)

The State of Charge (SoC) of a battery represents its immediate charge level at a specific moment, usually expressed as a percentage of the full charge of the battery. SoC provides real-time information about how much energy is available in the battery, with 0% indicating a fully discharged battery and 100% indicating a fully charged battery.

SoC is crucial for day-to-day usage, helping users determine when to recharge their devices or vehicles to prevent over-discharge or overcharging, which can harm the battery and shorten its lifespan.

2.3.3. Differences between SoH and SoC

While SoH and SoC are both vital battery metrics, they serve different purposes, as shown in Figure 2.3 and focus on distinct aspects of a battery's performance:

- **Focus:** SoH evaluates a battery's long-term health and overall condition, considering factors such as capacity, resistance, and aging, while SoC reflects the immediate charge level.
- **Measurement:** SoH is measured over the entire lifespan of a battery, providing insights into its health and degradation trends. SoC is assessed in real time, indicating the current charge level.
- **Units:** SoH is typically expressed as a percentage relative to the battery's original capacity, decreasing over time. SoC is also expressed as a percentage, in this case, relative to the battery's full capacity, but represents the battery's current charge level, fluctuating with usage.
- **Importance:** SoH is crucial for understanding a battery's longevity and predicting its remaining lifespan. SoC is essential for determining when to recharge the battery to avoid over-discharge or overcharging and ensuring its optimal daily operation.

2.4. Remaining Useful Life and Li-Ion Batteries

The Remaining Useful Life (RUL) of a system or component refers to the duration for which it can continue to operate efficiently before it reaches the end of its serviceable life. In essence, RUL provides a predictive measure, offering insights into the expected operational lifespan of a device from a given point in time [44]. This predictive capability is especially crucial for systems where reliability and safety are paramount.

- **General Importance of RUL**

  Estimating RUL has broad applications across various industries, from aerospace and automotive to energy and electronics. By accurately predicting when a component might fail or require maintenance, organizations can optimize operational efficiency, reduce downtime, and enhance safety. Furthermore, RUL estimation aids in making informed decisions regarding maintenance schedules, inventory management, and long-term planning.

- **RUL in the Context of Lithium-Ion Batteries**

  Given the widespread adoption of lithium-ion batteries in numerous applications, understanding their RUL becomes particularly significant. As these batteries degrade over time due to factors like charge-discharge cycles, temperature variations, and depth of discharge, their capacity to store and deliver energy diminishes. Predicting this degradation trajectory allows users to anticipate when a battery might no longer meet its performance criteria. For applications like electric vehicles or critical medical devices, such predictions can be the difference between seamless operation and unexpected failure.

- **Challenges in RUL Estimation for Lithium-Ion Batteries**

  Estimating the RUL of lithium-ion batteries is not straightforward. Their degradation patterns can be intricate, influenced by a myriad of internal and external factors. Traditional methods, which often rely on physical and chemical degradation models, might struggle to capture these complexities adequately. Hence, advanced predictive modeling techniques, including machine learning and neural networks (NN), have been explored to improve the accuracy of RUL predictions for these batteries.

- **Benefits of Accurate RUL Estimation for Lithium-Ion Batteries**

  With precise RUL predictions, users can optimize battery usage, ensuring they extract maximum value while minimizing risks. For instance, electric vehicle owners can plan long trips with confidence, knowing their battery's state. Similarly, energy

Dataset Name	Battery Type	Test Method	Test Conditions
NASA Battery Capacity Dataset	Li-Ion	Constant-Current Discharge	Room Temperature
JPL Battery Aging Dataset	Li-Ion	Cycling	Room Temperature
CALCE Battery Dataset	Various	Cycling	Room Temperature
Battery 500 Project	Li-Ion	Cycling	Room Temperature
NREL Battery Thermal Dataset	Li-Ion	Cycling and Thermal Stress	Various Temperatures and Rates
Sandia Dataset	Li-Ion	Cycling and Thermal Stress	Various Temperatures and Rates

Table 2.1: Battery Dataset Information.

storage systems can be managed more efficiently, ensuring consistent power delivery. Moreover, accurate RUL predictions can lead to better recycling strategies, contributing to sustainability.

While RUL estimation is a critical aspect across various domains, its relevance for lithium-ion batteries is particularly pronounced. Given the pivotal role these batteries play in modern technology, ensuring their reliability through accurate RUL predictions is of paramount importance.

2.5. Prior Research on RUL Estimation

Battery lifespan estimation was previously conducted by reviewing the performance of some cells, in a laboratory specialized in conducting these tests. This was enough to get a rough estimation of similar batteries' expected performance. To assess battery lifespan, various methods are employed depending on the type of battery and its intended use. Common techniques include temperature cycling and charge-discharge cycles. During these assessments, the battery's voltage, current, and temperature are constantly monitored to gauge its performance. Testing conditions and methodologies differ across datasets. Some of these are mentioned in Table 2.1. These data are the base of our research, being vital for a correct estimation of RUL via neural networks or other data-driven models.

The dataset utilized for this research, as detailed in [35], offers a comprehensive insight into

the behavior of commercial lithium-ion batteries under fast-charging conditions. These cells, specifically the Lithium Iron Phosphate (LFP)/graphite type from A123 Systems, underwent rigorous testing, providing invaluable data for RUL estimation.

2.6. Domain Adaptation and Transfer Learning in RUL Estimation

In the realm of RUL estimation, the concepts of domain adaptation and transfer learning emerge as pivotal solutions to bridge the gap between available data resources and the need for accurate predictive models. The overarching objective remains constant: to have a model trained on a comprehensive dataset and enable it to perform effectively on a more limited target dataset. However, the choice between domain adaptation and transfer learning depends on the specific characteristics of the target dataset, particularly with regard to the availability of labeled data. A similar approach has been used also in [11] on some Turbofan degradation datasets, and by [9] on a training set of 6 rolling bearings that were operated in three different conditions. But to the best of our knowledge, we are the first to focus on unsupervised domain adaptation for lithium-ion batteries.

2.6.1. Target Dataset Labeling Conditions

The decision to apply either domain adaptation or transfer learning hinges on the labeling conditions of the target dataset. These conditions can be broadly categorized into two scenarios:

- **All data is labeled (Transfer Learning)**

  When the entire target dataset is labeled, transfer learning is typically the preferred approach. This method leverages the annotated data to adapt the model to the new domain efficiently.

- **No labeled data (Domain Adaptation)**

  If no labeled data is available for the target dataset, domain adaptation comes to the forefront. This approach excels in scenarios where direct annotation of the data is infeasible or costly.

2.6.2. Domain Adaptation

Domain adaptation involves training a model on a source domain, which possesses a substantial dataset, while incorporating a subset of the target domain data into the training process, enabling the model to learn and adapt to the distinctive features and patterns unique to the target domain. This process operates on the fundamental assumptions that the source and target domains only differ in terms of the underlying data distribution, and that there exists a single hypothesis with low error that can be applied to both domains. By facilitating a seamless transition between domains, domain adaptation empowers the model to generalize its knowledge effectively, ensuring accurate RUL predictions in the target domain.

2.6.3. Transfer Learning

Transfer learning, in contrast, is a strategy that primarily focuses on reusing pre-trained models or their components. The approach involves transferring knowledge from a source task to a target task. Fine-tuning, a key component of transfer learning, enables the model to adapt to the specifics of the target domain without entirely retraining it from scratch. This efficient transfer of knowledge is especially beneficial in situations where some labeled data is available for the target domain.

In the context of RUL estimation, domain adaptation, and transfer learning prove invaluable when adapting NN models trained on one set of battery data to predict RUL for batteries operating under different conditions or from various manufacturers. These approaches stand out by their potential to save valuable time and resources that would otherwise be expended in collecting extensive new data for every unique battery type or operating condition. In essence, domain adaptation and transfer learning provide vital tools to enhance the flexibility and efficiency of RUL estimation models, catering to a diverse range of data scenarios and ensuring robust predictive performance.

2.7. Rationale and Process of Developing the Research Question

The motivation behind focusing on domain adaptation and transfer learning for RUL estimation stems from the observed discrepancies between laboratory and real-world battery performance. While NN models have shown high accuracy in controlled settings, their performance can be inconsistent when applied to different domains. This inconsistency poses a challenge, especially when considering the vast diversity of lithium-ion battery

applications.

Given the potential benefits of accurate RUL estimation and the limitations of current methods in diverse domains, the research question was developed to explore how domain adaptation techniques can be employed to improve NN models for RUL estimation.

For researchers, this topic presents an opportunity to push the boundaries of current RUL estimation techniques, making them more versatile and applicable in diverse real-world scenarios. For potential readers, especially those in industries relying on lithium-ion batteries, accurate RUL estimation can lead to better maintenance schedules, longer battery lifespans, and reduced costs. Furthermore, it can enhance safety by predicting potential battery failures in advance.

By focusing on domain adaptation, the research present in this thesis hopes to make significant strides in improving the accuracy and reliability of RUL estimation for these crucial energy storage devices.

3 | Related Work

3.1. Problem Description

Classical machine learning operates under the assumption that training and testing data are sourced from similar environments. This suggests that a model trained on one dataset should perform effectively on a similar dataset. Yet, this is not always true in practical scenarios. Often, training and testing data differ due to diverse origins or changes over time, potentially leading to suboptimal model performance on new data sets. Domain adaptation in machine learning addresses this discrepancy. It primarily deals with scenarios where labels are available only in the training (source) dataset. This field is challenging, as it involves ensuring a model's ability to adapt to and function with varied data types, a problem that has proven difficult in numerous studies [12]. Also in the field of Remaining Useful Life (RUL) estimation, domain adaptation plays a pivotal role, even though not has always been taken into account in numerous studies. In the realm of batteries, addressing the issue is contingent upon the manner in which pertinent battery-related data is acquired. Access to sufficient battery aging data is challenging due to both the complicated battery operations and the significantly laborious and time-consuming battery aging experiments [38]. The laboratory resources frequently place a cap on the range of these modes due to a lack of available resources and costs. It is a matter of choices and trade-offs. Since we are left with data sets with few batteries, and hence forced to train a model with little data, we are going to face the difficulties of training and testing models on data sets with different data distributions. Designing large-scale lab tests where several batteries are evaluated with hundreds or thousands of similar cycles is the most cutting-edge and simple way to collect aging data. However, the laboratory tests connected to aging are quite time-consuming and labor-intensive. One of the most cited papers on RUL estimation, by Lipu et al. [25], clearly demonstrates that RUL estimation using neural networks, particularly Long Short-Term Memory (LSTM) networks, outperforms other mathematical methods in accuracy, namely Particle Filter and Probability Density function among others, but still doesn't address the challenges of Domain Adaptation. A notable study by Ye et al. [48], on the other hand, shows promising results

in RUL estimation using domain adaptation techniques. They provided an overview of classical transfer learning methods, including simple transfer learning (TL), fine-tuning, and advanced Domain-Adversarial Neural Network (DANN) based transfer learning techniques. As anticipated, these methods improved performance in scenarios where battery behaviors were similar, or they didn't deal with batteries at all, but with other devices. The effectiveness of these techniques remains to be validated in situations where data distributions across different domains are more divergent, and this is instead the direction where our thesis pushes forward.

3.2. Statistical Matching Approaches

Until very recently, the majority of the papers and studies on the topic of domain adaptation were based on statistical approaches such as in [27] and in [47].
Some major contributions to this approach were based on the idea of aligning distributions of the source and target domains by minimizing the distance between statistical moments, such as the ones proposed by Zhang et al. [49].
The objective of distribution alignment can be thought of as the opposite of classification. Classification seeks a representation that differentiates between two distributions, whereas alignment aims to merge or align two distributions. The underlying principle of this method is that if the source and target domain distributions are closely aligned or similar, a model trained on the source domain is likely to exhibit good performance on the target domain.
The most common statistical approaches involved the use of the following metrics to compute distances:

- Maximum Mean Discrepancy (MMD)
- Correlation Alignment (CORAL)
- Central Moment Discrepancy (CMD)

A different way of addressing the problem involved weighting the instances in the target domain based on their similarity to the instances in the source domain [31].
It is implicit that if the instances in the target domain are similar to those in the source domain, they should be weighted more than the instances that are not similar. However since statistical matching is a complex operation that requires specific technical expertise and raises several methodological issues and also since in literature [32] it has been shown that transfer learning and domain adversarial learning usually outperform statistical matching approaches, in our work we directly decided to focus on these last

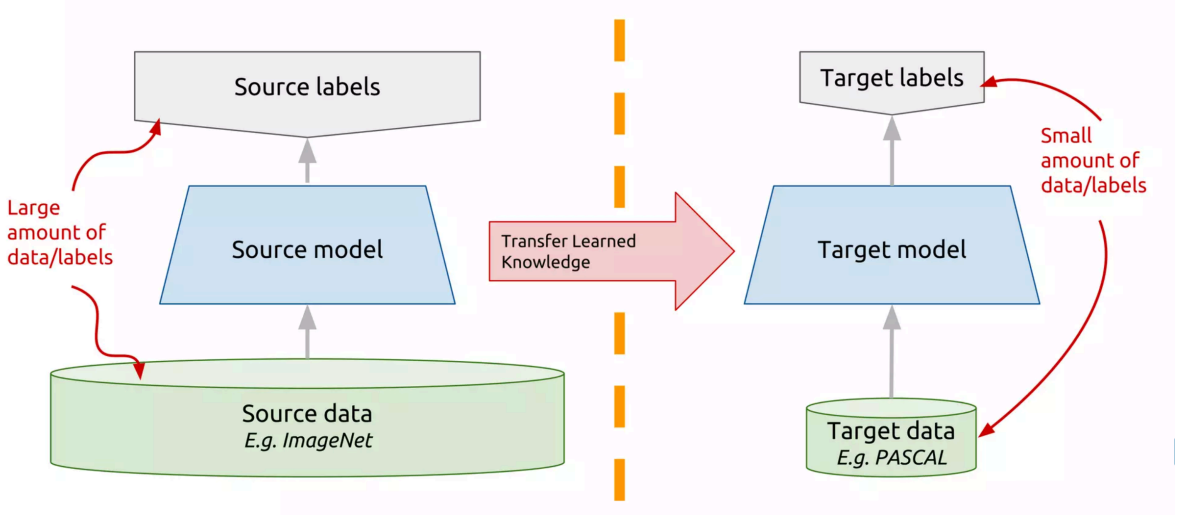


Figure 3.1: Transfer learning, a graphical representation [20].

state-of-the-art approaches.

3.3. Transfer Learning

Transfer learning is a broader concept than domain adaptation per se. It encompasses various approaches to using knowledge gained from one task or domain to improve the performance of another task or domain. In essence, transfer learning involves utilizing pre-trained models (or *knowledge*) to enhance prediction models, irrespective of the source and target domains' distribution differences [33].

The training dataset is usually larger and more diverse than the target dataset, it serves as a starting point. The idea is to transfer the knowledge encoded in the pre-trained model to a new, possibly related task, as can be seen in Figure 3.1. To get a broader view it is important to remember that in contrast, domain adaptation, focuses on addressing the challenges posed by a domain shift when the source and target domains have related but non-identical characteristics.

3.3.1. Challenges of Transfer Learning in RUL Prediction

While literature indicates that traditional transfer learning and fine-tuning generally outperform most other methods, they have inherent limitations, especially when applied to battery RUL estimation:

- **Domain Adaptation**

  Batteries can exhibit various degradation patterns, and different battery types may

have distinct characteristics. Traditional transfer learning methods might struggle to adapt to these diverse domains, especially if the source domain is significantly different from the target domain regarding battery chemistry, usage, or environmental conditions.

- **Limited Data**

  RUL estimation in batteries often requires a significant amount of historical data, including battery discharge profiles, aging conditions, and failure events. Fine-tuning a pre-trained model might not be effective when there is insufficient labeled data available for the specific battery type or condition of interest.

- **Feature Engineering**

  Traditional transfer learning often relies on feature extraction from pre-trained models. In battery RUL estimation, domain-specific feature engineering is crucial, as the features characterizing battery health and degradation may not be effectively captured by pre-trained models.

  Fine-tuning typically involves modifying the top layers of a pre-trained model, such as a neural network, to adapt it to a new task. However, the architecture of the pre-trained model might not be suitable for capturing the intricate features and patterns relevant to RUL estimation in batteries.

Some methodologies, like the one proposed by Amogne et al. [3], employ transfer learning in RUL estimation to leverage knowledge from a data-rich source domain to benefit a target domain with scarce but labelled data. The study addresses the challenge of accurate prediction in intelligent Battery Management Systems (BMS). The proposed method, incorporating a transfer learning approach, outperforms other LSTM-based models in Li-ion battery Remaining Useful Life (RUL) prediction. Relative error values for the proposed method on three target batteries are notably low, showcasing its effectiveness.
The transfer learning approach, transferring weights from trained source batteries to target batteries, significantly improves State of Health (SOH) prediction compared to models without transfer learning.
This method not only yields superior prediction results but also proves efficient for online RUL prediction, saving a substantial amount of training time, but ti doesn't address the problem of lack of labelled data or completely different distribution of features.

Other works instead went further by fine-tuning the offline trained models on the data in target batteries, as described by Chou et al. [10], this research offers better results and saves considerable training time, making it suitable for online prediction. On the other

hand, it still encounters the same limitations that the previously mentioned work and it also doesn't use relevant metrics of evaluation such as mean absolute error or root mean squared error to assess the quality of their predictions.

Another intriguing approach by Shao et al. [36] involves directly converting raw signals into graphical images as input and fine-tuning pre-trained models from ImageNet, an image database with hundreds of thousands of pictures. However, this method did not achieve the same level of performance as more classical approaches, despite introducing a new perspective on addressing the problem.

3.4. Hyperparameter Optimization

This section deals with the optimization of a given model architecture through a recent automatic approach. The optimization of neural networks, particularly deep learning models, is crucial to achieve robust and accurate results. One challenge with deep learning models, especially when faced with limited data like in our case, is selecting the right hyperparameters. The choice of hyperparameters can drastically influence the model's performance, making the optimization process pivotal.

3.4.1. Traditional Optimization Approaches

Traditionally, optimization of neural network hyperparameters has been achieved through methods such as grid search or random search. While these methods are straightforward and widely used, they have recognized limitations. For instance, grid search can be computationally expensive and inefficient, especially when exploring a high-dimensional hyperparameter space. Random search, although sometimes more efficient than grid search, can be unpredictable and may miss the optimal hyperparameters [7].

3.4.2. Advent of KerasTuner

To overcome the limitations of traditional methods, tools like KerasTuner have emerged. KerasTuner is an easy-to-use, distributable hyperparameter optimization framework that solves the challenges of finding the right hyperparameter configurations that provide good performance. Unlike traditional methods, KerasTuner employs techniques like Bayesian optimization, which iteratively refines the search based on past evaluations, making the search process more efficient and precise even in situations like ours as described in Banna et al. [5].

Several studies have illustrated the efficacy of KerasTuner in various deep-learning tasks.

For example, Joshi et al. [22] used KerasTuner to optimize CNN models for image recognition tasks, demonstrating a noticeable improvement in accuracy compared to models optimized using traditional methods.

3.4.3. KerasTuner in RUL Estimation

While KerasTuner has been used in various domains, its application in RUL estimation, especially for lithium-ion batteries, represents a novel approach. Given the intricate nature of RUL estimation, where capturing the intricate degradation patterns is paramount, an optimized model can substantially enhance prediction accuracy. Leveraging KerasTuner in this domain suggests a promising avenue to achieve better performance and accelerate the optimization process.

3.5. Feature-Based Adaptation

In our work, we focused on feature-based adaptation, which is a recent approach for time series data, [19], since it is more commonly found in classification tasks for images and similar tasks [50].
Feature-based adaptation approaches aim to map the source data into the target data by learning a transformation that extracts invariant feature representation across domains. They usually create a new feature representation by transforming the original features into a new feature space and then minimizing the gap between domains in the new representation space in an optimization procedure, while preserving the underlying structure of the original data.
In this thesis domain adversarial learning was the selected approach as it is explained in the following sections.

3.5.1. Domain Adversarial Neural Networks

The challenge of domain adaptation, notably when there is a difference in source and target domain distributions, has garnered considerable interest in the machine learning field. Domain Adversarial Neural Networks (DANN) have emerged as a promising approach to address this challenge. In this section, we review seminal and recent works in the field of domain adversarial methods, emphasizing their contributions, methodologies, and implications.

3.5.2. Foundation of Domain Adversarial Techniques

In the foundational work by Ajakan et al. [14], the Domain-Adversarial Neural Networks (DANN) algorithm was proposed, drawing inspiration from the domain adaptation theory put forth by Ben-David et al. [6]. The primary goal of DANN was to design a network's hidden layer to learn representations predictive of the source labels while being neutral about whether the data is from the source or target domain. Experimental results, especially on the Amazon reviews sentiment analysis dataset, confirmed the efficacy of this strategy. Particularly, combining DANN with the marginalized stacked denoising autoencoders (mSDA) of Chen et al. [8] led to state-of-the-art performances, emphasizing that these approaches to representation learning are complementary. Ajakan et al. also envisaged the adaptability of the domain adaptation regularizer from DANN to a plethora of other learning algorithms, hinting at potential applications in deeper network architectures, multi-source adaptation challenges, and a variety of learning tasks beyond basic binary classification [14].

Following this pioneering work, Ganin et al. [15] introduced the domain adversarial approach, which involves training a neural network to accomplish two objectives. The first objective is to predict labels in the source domain, while the second objective, which is adversarial, involves tricking a domain classifier and thus ensuring that the feature representations are not biased towards any particular domain. Through this adversarial training, the learned features become domain-invariant, improving the model's ability to generalize to the target domain. We can see a graphical representation of the Ganin proposed architecture in Figure 3.2

3.5.3. Expanding the Domain Adversarial Paradigm

Following the initial work, various modifications and extensions to DANN were proposed. Tzeng et al. [39] introduced ADDA (Adversarial Discriminative Domain Adaptation), refining the adversarial adaptation technique by employing a discriminative model for the target domain, which further enhanced feature alignment.

Sicilia et al. [37] delved deeper into the nuances of applying DANN for domain generalization. Their insights highlighted the intricate relationship between source domain variability and domain alignment, leading to a novel extension of DANN that displayed improvements consistent with theoretical predictions.

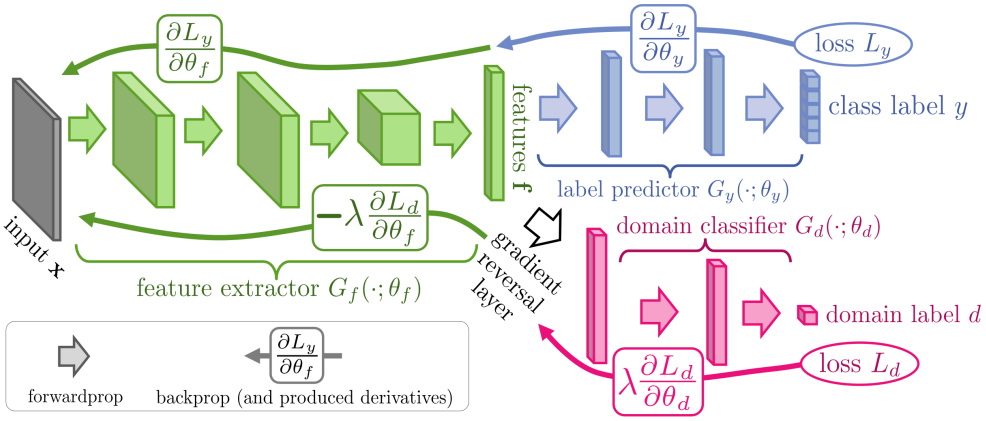


Figure 3.2: The suggested architecture integrates a deep feature extractor, depicted in **green**, with a deep label predictor, shown in **blue**, creating a conventional feed-forward structure. To facilitate unsupervised domain adaptation, a domain classifier, colored in **red**, is incorporated. This classifier connects to the feature extractor through a gradient reversal layer, which modifies the gradient by multiplying it with a specific negative constant during backpropagation training. Adapted from [16].

3.5.4. Applications of Domain Adversarial Methods

DANN and its variants have been successfully applied in various domains. In computer vision, these methods have been utilized for tasks such as digit classification in diverse datasets [15] and semantic segmentation under different environmental conditions [45]. Beyond vision, domain adversarial techniques have shown promising natural language processing, where the distributional shift between training and test data can significantly degrade model performance.

Regarding RUL prediction, some pioneering work has been done by [26] where it is clearly stated that while advanced deep learning and machine learning methods have achieved excellent results in the prediction of the Remaining Useful Life of the battery, the superiority of these methods is obtained by training under the same charge and discharge condition data. This paper introduces a lithium-ion battery capacity estimation method using transformer-adversarial discriminative domain adaptation (T-ADDA). The T-ADDA model, validated with NASA lithium-ion battery data, demonstrates high accuracy. Despite its effectiveness in cross-domain estimation, the T-ADDA model does encounter limitations, with CNN and LSTM models displaying higher accuracy under the same working conditions. Apart from this last work, in the literature, there is a limited exploration of the Domain Adversarial Learning paradigm applied to the Remaining Useful Life (RUL) estimation problem for batteries. Our work addresses this gap by concen-

trating on constructing a simpler yet effective model, contrasting with the approach in [26], we aimed at building a model that can be more easily deployed in a variety of cases without losing accuracy, stemming from the DANN implementation description present in the most relevant paper about it [15].

4 | Methodology

4.1. Proposed Set of Solutions

This chapter delineates the methodologies employed in this thesis, providing a comprehensive insight into the rationale, processes, and techniques adopted. By detailing these methods, this chapter not only aims to offer transparency in the research process but also provides a roadmap for replicating or building upon this study in future endeavors.

4.2. Architecture Redefinition

After having analyzed the convolutional-LSTM neural network improved with the use of an attention mechanism developed by Luca Martiri [29], we realized the need to tailor the architecture to better suit the nuances of our problem. Given the limited dataset available to us, a comprehensive overhaul of the neural network was deemed necessary. We drew inspiration from Martiri's work and identified three crucial components, each of which is here explained:

- **Convolutional Layers**

  Fundamental in processing high-dimensionality data, these layers play a pivotal role in extracting features from input data. The simultaneous decrease in dimensionality and increase in feature number gives the network the capability to learn complex features that were not easily deducible from the original data. Their utility in signal processing makes them an essential component of many deep learning architectures, including ours. The features under our consideration, namely actual cycle numbers, temperature, and discharge capacity, can be aptly perceived as signals, underscoring the relevance of convolutional layers in our model.

- **LSTM Layers with Attention**

  Long Short-Term Memory (LSTM) units are a type of recurrent neural network (RNN) architecture. Their capability to remember patterns over long durations

makes them well-suited for sequential data. The inclusion of attention mechanisms further refines the process by enabling the model to focus on specific parts of the sequence, enhancing its performance in various sorts of tasks [4]. In our case, is particularly useful for the neural network remembering sequence-based insights.

- **Fully Connected Layers for Regression**

  These layers are designed to process the features extracted by the previous layers and produce a final output. In our context, they perform regression tasks, aiming to predict continuous values based on the processed input.

4.3. Hyperparameter Optimization

The selection of appropriate hyperparameters is fundamental to the performance of deep learning models. Manual tuning can be cumbersome, and time-consuming, and may not always lead to good results. Automated tuning tools aim to alleviate these challenges. Among these tools, KerasTuner stands out as an advanced method specifically tailored for the hyperparameter tuning of deep learning models implemented in Keras.

There were several compelling reasons behind our decision to use KerasTuner:

- **Integration with Keras**

  Given that our models were implemented using Keras, the seamless integration of KerasTuner with this framework was a natural fit, ensuring smooth functionality without extensive modification to our existing codebase.

- **Efficiency**

  KerasTuner deploys a range of search algorithms, such as Random Search, Hyperband, and Bayesian Optimization. These algorithms are designed to rapidly hone in on optimal hyperparameter configurations, thus shortening the time and resources required for tuning.

- **Flexibility**

  KerasTuner provides the capability to customize the tuning process and tunable parameters, allowing us to define the search space and the optimization objectives according to our requirements. This was crucial since our model is pretty complex and tuning all hyperparameters meant running the tuner for multiple weeks.

- **Visualization**

The tool offers visualization utilities to review and analyze the tuning results, granting us insights into the optimization process and aiding in refining our model further.

Given the nature and complexity of our project, it was imperative to leverage a tool that could offer both efficiency and precision in the hyperparameter tuning process. KerasTuner, with its array of features and its alignment with the Keras ecosystem, emerged as the logical choice for ensuring our model reached its optimal performance.

4.4. Feature Selection

Achieving broad applicability in model development and deployment often necessitates a trade-off between the richness of the data and the universality of the model. In light of this, we prioritized features that were ubiquitously available across multiple datasets to ensure the widest possible relevance of our model. Specifically, we zeroed in on three key features: *Cycle*, *Discharge Q*, and *Average Temperature*. How these features were collected in our specific case and how they are measured is going to be presented in depth in Chapter 5 1.1.
Their widespread presence in numerous online datasets made them the most viable candidates for a generalized model.
The most informative one is the *Discharge Q* feature since it is the one that relates directly to the battery's state of health.
In most of the other works [26], the most relevant feature adopted was the cycle-to-cycle evolution of the discharge voltage curve, expressed as a function of voltage for the given cycle. The discharge voltage curve of a battery demonstrates the variation in voltage throughout the discharge process. It depicts the relationship between the battery's voltage and its state of charge (SOC). Especially the difference over cycles of this feature, denoted as $\Delta Q(V)$ is of particular interest because voltage curves and their derivatives are a rich data source that is effective in degradation diagnosis.
The problem with this feature is that it can be used only under the assumption that all the batteries are discharged under the same conditions, which is preferable, or charged under the same conditions. This makes it less appealing for domain adaptation or generalization to other datasets, which is our thesis's main focus.
The discharge capacity curve of a battery depicts instead the variation in capacity throughout a discharge cycle. It showcases the relationship between the remaining capacity of the battery and its state of charge (SOC) during the discharge process. In literature, using the discharge capacity curve as a substitute feature for the discharge voltage curve led to poorer performance and less accurate results in general, we are speaking about a loss

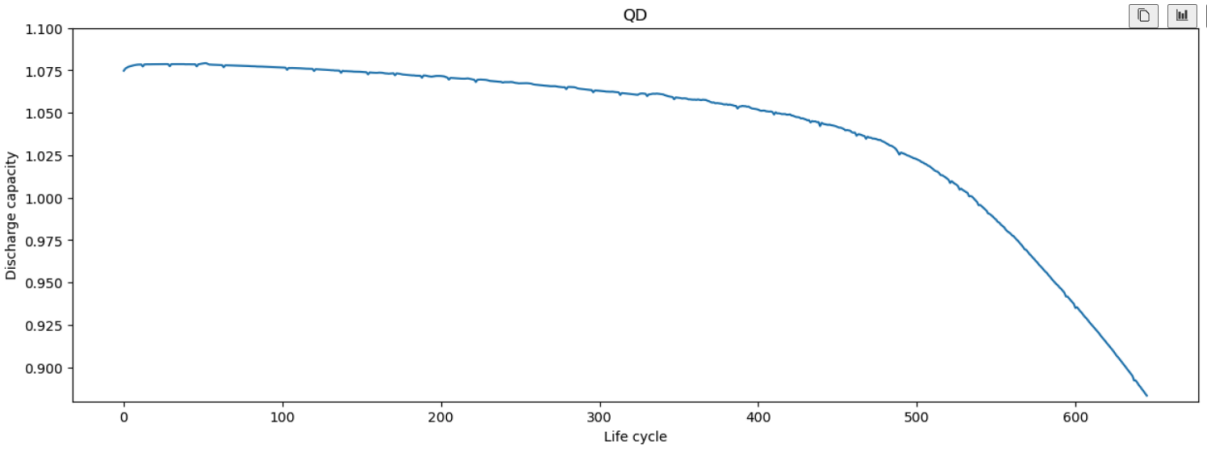


Figure 4.1: Capacity fade curve of a battery.

from 30% to 40% in accuracy [35].
This highlights the difficulty of prediction without using voltage features.
In the present work, we were able to obtain comparable results to previous models found in the literature that used the discharge voltage curve, surprisingly using a similar size dataset for training.
We can see an example of the discharge capacity curve in Figure 4.1.

We also give a short description of the used features for completeness and a better understanding of the work:

- **Cycle**

  The cycle number k of the battery. It is a discrete feature that represents the number of times the battery has been charged and discharged. It is just an integer number.

$$c_k = k \tag{4.1}$$

- **Average temperature**

  The average temperature of the battery during each cycle k is a continuous feature that represents the average temperature of the battery during the cycle, where n_k is the number of temperature values collected for each cycle. It is measured in degrees Celsius (°C)

$$T_k = \frac{1}{n_k} \sum_{i=1}^{n_k} T_{k,i} \tag{4.2}$$

- **Discharge Q**

The discharge capacity of the battery. It is a continuous feature that represents the amount of charge that the battery can store at the given cycle. It is measured in ampere-hours (Ah)

$$Q_k = Q_{k,n_k} \tag{4.3}$$

4.5. Ad-Hoc Data Augmentation

The size and consistency of the training datasets have a significant impact on the performance of deep neural networks used to process time series. In the real world, these traits are typically scarce and frequently subject to requirements that must be guaranteed. Consequently, utilizing Data Augmentation techniques, either by adding noise or permutations and by generating new synthetic data, is an efficient way to augment the amount of data [21].

Many traditional data augmentation algorithms cannot be directly applied to time series [21] due to the nature of the data. For instance, while rotating images to augment a dataset is feasible, the same approach is not directly applicable in the time series domain. If a time series sample is divided into segments and rearranged using linear interpolation, the outcome may lack validity as it disrupts the inherent data trends. Given the varied nature of time series data, not all augmentation techniques are universally applicable. While few algorithms for data augmentation from computer vision can be adapted to the time series domain, there are instances where new, specialized algorithms need to be devised for effective treatment of time series data.

There is always the risk of producing an invalid result, where the tendency of the data would be destroyed. Not all strategies can be used on every dataset because of the variability of time series data.

When using data augmentation in the time series domain, particularly in signal processing, it is vital to keep in mind that data modification may overly distort the signal and result in incorrect training.

There are two main types of data augmentation techniques [43]: *data augmentation from existing data, by adding noise* and *data augmentation by generating whole new synthetic data*. In the former algorithms, the transformations are applied directly to the data, while in the latter the objective is to learn the probability distribution of the data to generate completely new samples trying to imitate the data distribution.

We focused on the first approach because it was the simplest and most effective for our purposes, and also because our main focus was improving the performance of the domain adaptation, generating too much new synthetic data for one domain would not have made much sense.

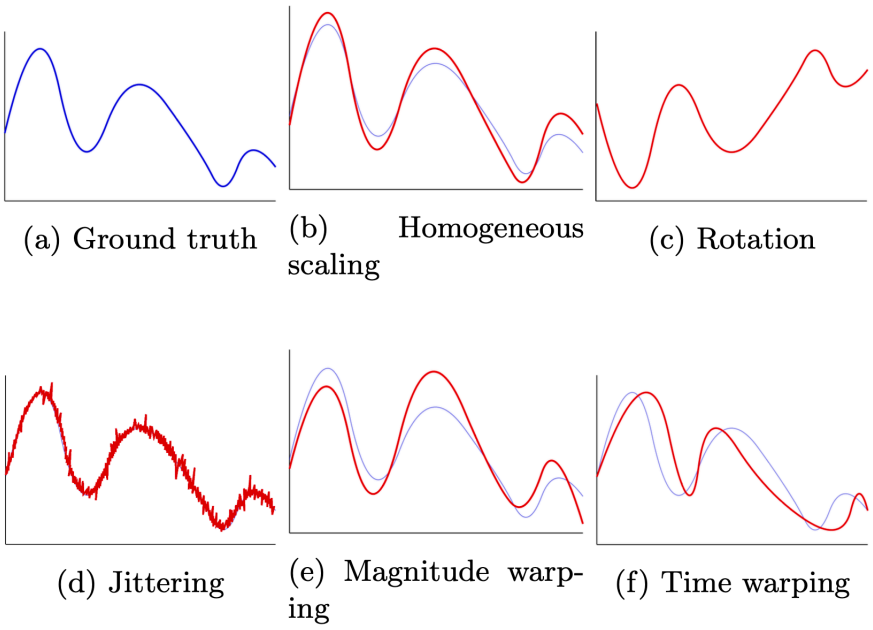


Figure 4.2: Data Augmentation Techniques [21].

All these techniques [21] have as their base the deformation, shortening, enlargement, or modification of the data samples of the dataset. In this thesis, we chose a method of data augmentation based on the superposition of random noise to address the RUL estimation problem, and also a time-slicing window during training.
The main reason is based on the fact that the historical capacity vector and the current capacity have a temporal relationship in lithium batteries. The accuracy of RUL prediction is based on the assumption that this relationship is discovered by the model, hence the sequentiality of the data.
Also, we had to take care of the fact that the range of capacity discharge in which we had to work was fixed by definition (EOL at 0.88 mAh) hence we could not use *scaling* [21] for example.
Some other techniques such as *permutations* [43] or *rotations* [43] were also not employable since they would have made us lose important sequential patterns in data.

Gaussian noise is commonly used in various fields, including signal processing, statistics, and data analysis, to model and simulate random variations in data or measurements. It is often used when the underlying sources of variation are not precisely known or when multiple sources of random variation can be reasonably approximated by a Gaussian distribution.
In our particular case, adding Gaussian noise to a set of existing batteries and including them afterward in the training dataset increases the generalization capacity of the neural

network by extending the initial dataset. This is also called *jittering*, and it is useful also to replicate capacity regeneration, capacity dips, and sensor mistakes during lithium battery operation, which enhances the adaptability and robustness of the model.
The following question is, by how much should we increase the dataset?
And also, how much noise should we add?
We propose in this work an empirical approach, where we experimented with a vast set of possible values for the noise and the number of batteries to be augmented.
We could not afford an excessively noisy dataset, since it is already known that the noise in the data can lead to a decrease in the accuracy of the model, especially when working with the capacity discharge curve and just temperature and cycle information [35].
These observations lead to the following conclusions:

- The noise should be small enough to not alter the general trend of the capacity discharge curve, but big enough to help the model to better generalize. After some experimentation, we found that a noise of 0.005 is a good compromise.
- The number of batteries to be augmented should be big enough to increase the dataset, but not too big to not trasnform too much the original dataset. Also here, after having done many trials, we found an optimal number consisting of augmenting all the batteries that had more than 540 cycles of life.

4.6. Transfer Learning and Fine-Tuning

Transfer learning and fine-tuning are advanced strategies in deep learning that aim to capitalize on pre-trained models, harnessing their potential to expedite and enhance the training process for new tasks. Instead of starting the training process from scratch, these techniques allow us to utilize the knowledge acquired from previously trained models, making them particularly advantageous when dealing with limited data or aiming for efficiency [42].

4.6.1. Transfer Learning

Transfer learning involves using the architecture and weights of a pre-trained model, typically trained on a large benchmark dataset, as the starting point for a new task [42]. The rationale behind this is that the features learned by these models on vast datasets can act as general features, beneficial for a broad range of tasks. Indeed, using networks trained on datasets having the same structure as the target one improves the capability of the transferred network to use the previously learned features.

The pre-trained model can be used as-is in cases such as ours where the original model is trained on similar data with the same objective. This yields generally respectable performance. But, considering that in the target domain we can use some data to improve performance leveraging fine-tuning, we did not stop on the simple transfer learning.

4.6.2. Fine-Tuning

Fine-tuning is a subsequent step to transfer learning, where the pre-trained model is further trained (or "tuned") on the target dataset. This can involve training the entire model or just a subset of layers. The process adjusts the weights, enabling the model to map the features for the new target task.

As highlighted in the literature [17, 42], there are some points to consider during fine-tuning:

- **Learning rate**

  A lower learning rate is often preferred to ensure that the model does not deviate significantly from the pre-trained weights.

- **Layers to train**

  Depending on the similarity of the new task to the original task, one might decide to train just the top layers, a few middle layers, or the entire network. For vastly different tasks, more layers might require fine-tuning.

- **Regularization**

  As fine-tuning is essentially a continuation of training, regularization techniques like dropout or weight decay can be vital to prevent overfitting.

In our case the best performance, after a few combinations of the previous parameters have been tried, was achieved by lowering the learning rate and training the whole model.

In conclusion, the amalgamation of transfer learning and fine-tuning presents a formidable approach to attain impressive performance with a shorter training duration and fewer data resources with respect to a model trained from scratch. These techniques underscore the principle of building upon existing knowledge, especially pivotal in scenarios with data constraints.

4.7. Our results

Now, we apply these methodologies to the task of remaining useful life estimation in lithium-ion batteries. These strategies enable us to adapt pre-existing knowledge from a source domain to the target domain, facilitating the accurate prediction of RUL. In this section, we delve into the practical implementation of transfer learning and fine-tuning, shedding light on their role in mitigating domain adaptation challenges in the context of our experiments and results.

4.7.1. Architecture

The Convolutional Long Short-Term Memory (ConvLSTM) model [34] blends convolutional layers with LSTM units, making it adept at modeling sequential data and extracting relevant features. This section presents the model's architecture and its role in mitigating domain adaptation challenges.

Model Structure

- **Input Layer:** Accepts time series data with shape *input_shape*.
- **Convolutional Layers:** Initial layers include 1D convolutional layers (*Conv1D*) for feature extraction. An optional second convolutional layer extracts more complex features.
- **LSTM Layer:** Captures sequential dependencies, returning hidden states and cell states.
- **Batch Normalization:** Optionally applied for regularization.
- **Decoder and Attention:** The decoder reverses the process, and an attention mechanism focuses on relevant input data.
- **Dense Layers:** Optional fully connected layers for more complex modeling.
- **Output Layer:** Predicts Remaining Useful Life for each time step.

4.7.2. Transfer Learning First Approach and Results

The model is initially utilized for transfer learning, offering a robust solution to address domain adaptation challenges. During this phase, the model is trained on two out of three available data batches and tested on the third one. The key concept here is to leverage knowledge from these source batches, allowing the model to adapt to domain-specific

Metrics:	MAE	RMSE	MAE 400	MAE 100	MAE 50
test b2-b3	29.6	52.6	25.9	5.5	4.5
T-L test b1	175.2	183.7	142.7	22.2	14.3

(a) Performance on target domain (Batch 1) with a model trained on Batch 2-3.

Metrics:	RMSE	MAE	MAE 400	MAE 100	MAE 50
test b1	18.52	13.97	9.11	4.25	2.62
test b2	262.12	227.67	228.42	135.00	78.79
test b3	214.51	160.24	59.67	20.50	9.34

(b) Performance on target domains (Batch 1, Batch 2, and Batch 3) with a model trained on Batch 1.

Table 4.1: Transfer Learning Results.

patterns and nuances present in the data.

Tables: In Table 4.1a we can see the results for training with Batches 2-3, testing on their test set and also testing on Batch 1 with this transfer learning approach. For completeness, in Appendix A we can find the results for the other possible training and testing combinations.

We have also decided to present the results in Table 4.1b for training, just using one batch (Batch 1) and testing with the other 2.

The outcomes from our initial application of transfer learning demonstrate its effectiveness as a foundational approach. However, the relatively high prediction error points towards the necessity for further refinements. In the subsequent section on Fine-tuning, we will explore how these improvements can be implemented to enhance the model's accuracy and overall performance. This step is crucial in advancing the model from providing basic predictions to achieving a more precise and reliable level of predictive accuracy.

4.7.3. Fine-Tuning: Initial Phase

Following the transfer learning phase, after having seen the results, to seek further improvements fine-tuning is performed. In the initial phase, the model is fine-tuned on 8 batteries out of 42 from the left-out batch of data for a limited number of epochs (i.e., 10 epochs). Fine-tuning works by further adjusting the model's weights to better align with the third batch's characteristics. It helps the model become more domain-aware and enhances its ability to make accurate RUL predictions in the target domain.

In Table 4.2a we can see the results for fine-tuning on Batch 1. In Table 4.2b we have decided to present, as before, the results for training using just one batch (Batch 1) and

Metrics:	MAE	RMSE	MAE 400	MAE 100	MAE 50
Fine-Tuning test b1	108.1	104	78.5	10.3	5.4

(a) Performance on target domain (Batch 1) after training on Batches 2-3 and fine-tuning for 10 epochs on Batch 1.

Metric	RMSE	MAE	MAE 400	MAE 100	MAE 50
test b2 FT	56.02	35.95	32.38	10.10	4.49
test b3 FT	105.98	71.19	34.45	13.87	7.13

(b) Performance on target domains (Batch 2, and Batch 3) with a model trained on Batch 1.

Table 4.2: Fine-Tuning results.

Metrics:	MAE	RMSE	MAE 400	MAE 100	MAE 50
Fine-Tuning (+20 Epochs)	108.9	104	77.6	7.5	4.6
Fine-Tuning (only 10 epochs)	108.1	104	78.5	10.3	5.4

Table 4.3: Fine-Tuning Extended Results, target domain (Batch 1) after +20 Epochs of fine-tuning.

testing with the other 2 after having Fine-Tuned the model on the chosen subset of the two batches.

For completeness, in A we can find the results for the other possible training and testing combinations, which yield similar outcomes.

4.7.4. Fine-Tuning: Extended Phase

Subsequently, an extended phase of fine-tuning is undertaken with an increased number of training epochs (i.e., 20 epochs) making it a total of 30 epochs. However, during this extended fine-tuning, no significant improvement in the model's performance was observed. The results suggest that the model's adaptation has already reached a point of diminishing returns between 10 and 30 epochs, where further fine-tuning does not yield substantial benefits even hindering performance.

This outcome may be attributed to the model already achieving a high level of domain adaptation during the initial fine-tuning phase, leaving little room for further improvement in the extended phase. The data characteristics of the third batch may have been sufficiently captured by the model.

Displayed in Table 4.3 the results on the same combination of Batches 2-3, the others can be found in Appendix A.

4.7.5. Final Transfer Learning and Fine-Tuning Observations

Transfer learning proved effective in our study due to its ability to leverage pre-trained models on the source domain, exploiting shared underlying relationships with the target domain. However, fine-tuning, especially with limited labeled data, excelled in adapting to the unique characteristics of the target domain, resulting in enhanced predictive accuracy, the performance uplift is around 50% when compared to simple transfer learning. This adaptability to domain-specific insights made fine-tuning a valuable choice, particularly when the target domain significantly deviated from the source domain, as with Batch 2 shown in Appendix A.

5 | Experiments

In this chapter, we present the experimental procedures and methodologies we employed in the current thesis research.

5.1. Structure of the Dataset

The *MIT-Toyota* dataset was collected for a joint study between the Massachusetts Institute of Technology and Toyota Motor Corporation. Originally, it has been used in [35]. The dataset is composed of 124 commercial lithium-ion batteries that were cycled to failure through a fast charge policy to obtain the relevant data.
The cells have a nominal capacity of 1.1 Ah and a nominal voltage of 3.3 V.

5.1.1. Charge-Discharge Policy

The cells in this dataset are charged with a one-step or two-step fast-charging policy. This policy has the format "C1(Q1)-C2", in which C1 and C2 are the first and second constant-current steps, respectively, and Q1 is the state-of-charge (SOC, %) at which the currents switch. This means that if we had for example 6C(40%)-3C for a battery that has a rated capacity of 1000 milliampere-hours (mAh), then "1C" for this battery would be 1000 mA, and 6C would be 6000 mA. Thoroughly explained in 4.4
This current would be kept constant until we reached 40% of the battery's state of charge, and then the current would be switched to 3C, which would be kept constant until the battery was fully charged.
In this specific dataset, the initial charging current is 3960mA ($C1 = 3C$) once the battery reaches 80% of its state of charge ($Q1 = 80\%SOC$), the current is switched to 1100mA ($C2 = 1C$) and kept constant until the battery is fully charged (CC-CV).
The max and min voltage applied while charging/discharging the batteries are 2.0 V and 3.6 V. The temperature of the environment in which the charge cycles took place was kept constant at 30 °C.
The discharging phase is the same for all batteries, constant and with a fixed value at 4C. Since the objective was to have enough diverse data for the study, different charging

policies have been applied. The conjunction of this factor with the strong nonlinearity of battery degradation led to a very heterogeneous dataset, where the life duration of batteries ranges from 150 to 2200 cycles.

5.1.2. Data Collection

Each battery cycle is organized into a hierarchical structure with two levels: firstly, a summary of measurements for the entire cycle, and secondly, detailed measurement data for each specific time step within that cycle.

The *Cycle* structure contains:

- **t**: vector of time stamps (sec) which indicates the time instant at which sensors read the measurements.
- **Q charge**: cumulative moved charge in the charge phase (Ah).
- **I**: current (A).
- **V**: voltage (V).
- **T**: temperature (°C).
- **Q discharge**: cumulative moved charge in the discharge phase (Ah).
- **Discharge dQdV**: derivative of Q(V) in the discharge phase.

The *Summary* structure contains:

- **Q charge**: charge moved in the charge phase of each cycle.
- **Q discharge**: charge moved in the discharge phase of each cycle.
- **T_max**: maximum temperature reached in the cycle.
- **T_min**: minimum temperature reached in the cycle.
- **T_avg**: average cycle temperature.
- **Charge time**: time needed for a full charge.
- **Cycle**: number of the cycle.
- **IR**: measured internal resistance.

We are going to work with the features selected from the Summary structure.

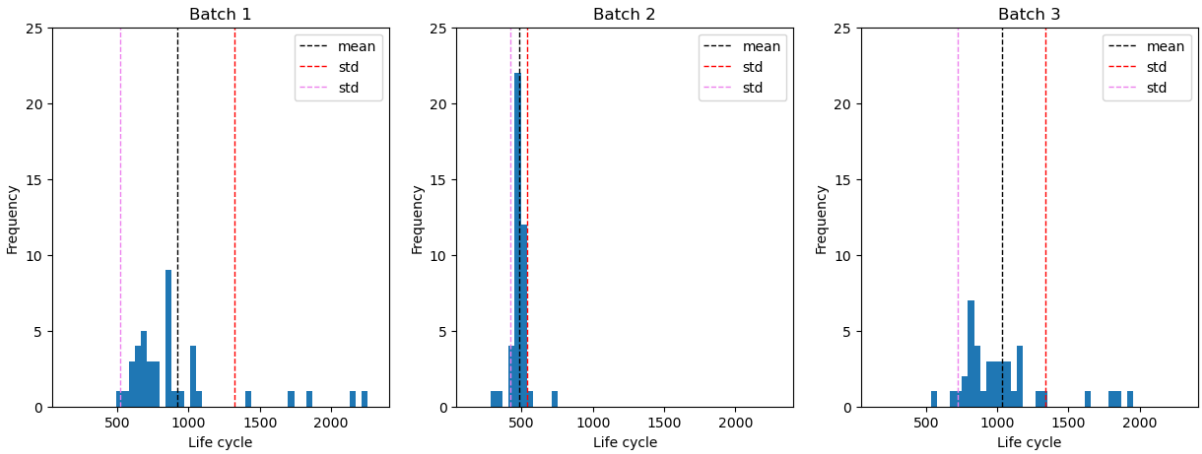


Figure 5.1: Distribution of the number of cycles to failure for each batch.

5.1.3. Batches Division and Analysis

The dataset used in this thesis is organized into three separate parts, referred to as "batches". Each of these batches represents data from about 48 individual cells. The distinguishing factor for each batch is the "batch date," which indicates the date when testing began for that particular group of cells. The reason for dividing the dataset into three batches is practical: it is due to the use of the same testing machines for conducting all the tests. By splitting the dataset, it becomes easier to manage and analyze the data from the different batches, each of which was tested at a different time.
We simply kept the division in batches and used categorical numbering to identify them. Each batch ends up having a relevant difference in the feature's distribution, making the MIT- Toyota dataset more similar to 3 diverse datasets jointly collected. This poses the challenge and the opportunity of training a model on one or two batches and applying the domain adaptation techniques, previously introduced 2.6. The diversity previously mentioned is reflected in the different distributions of the following adopted features:

- **Cycles to failure**

  Number of cycles before the battery fails.

  As we can see in Figure 5.1, the distribution of the number of cycles to failure for each batch looks more similar for Batches 1 and 3, while Batch 2 has a completely different distribution, this made us explicit how the specific charging and discharging conditions previously mentioned are affecting the battery's degradation. It also gave us a hint on how the different batches could be used to train and test the model and how the performance might relate to these differences, as we will see.

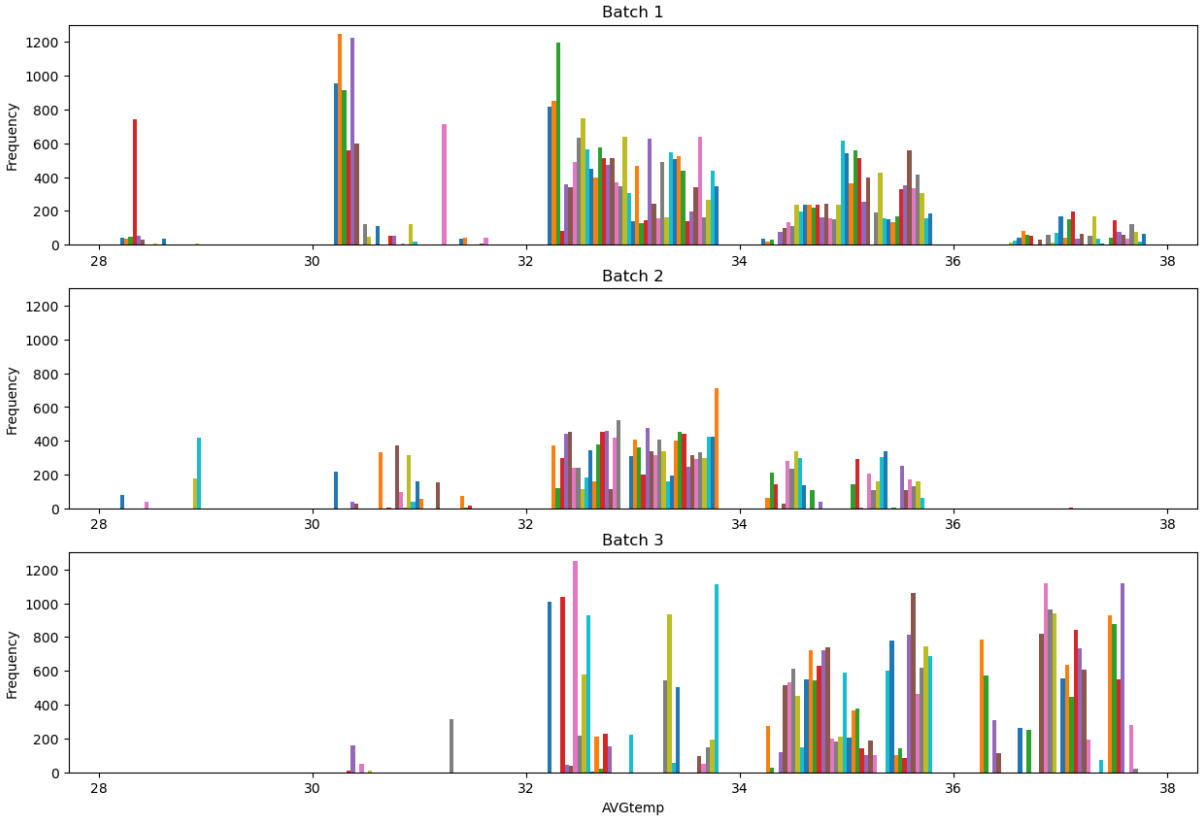


Figure 5.2: Distribution of the average temperature for each batch.

- **Average temperature**

  The average temperature of the battery during the cycle.

  As we can see in Figure 5.2, the average temperature frequency distribution is significantly different for all 3 batches.

- **Discharge capacity**

  Cumulative moved charge in the discharge phase (Ah) for each cycle until EOL is reached.

  Also, relevant differences emerge in the curves of the discharge capacity for each batch, as we can see in Figure 5.3. We have observed a significant decrease in the rate of decline of QD (Discharge Capacity) values during the final 150 cycles of a battery's life. To clarify, when a battery has a lifespan of, for example, 400 cycles, its QD value may drop rapidly from 0.98 to 0.88 within a short span of cycles, as opposed to a battery lasting 1000 cycles where the decline occurs more gradually, going for example from 0.91 to 0.88 in the last 150 cycles. This phenomenon is a clear indicator of the nonlinearity of the problem faced and highlights the current

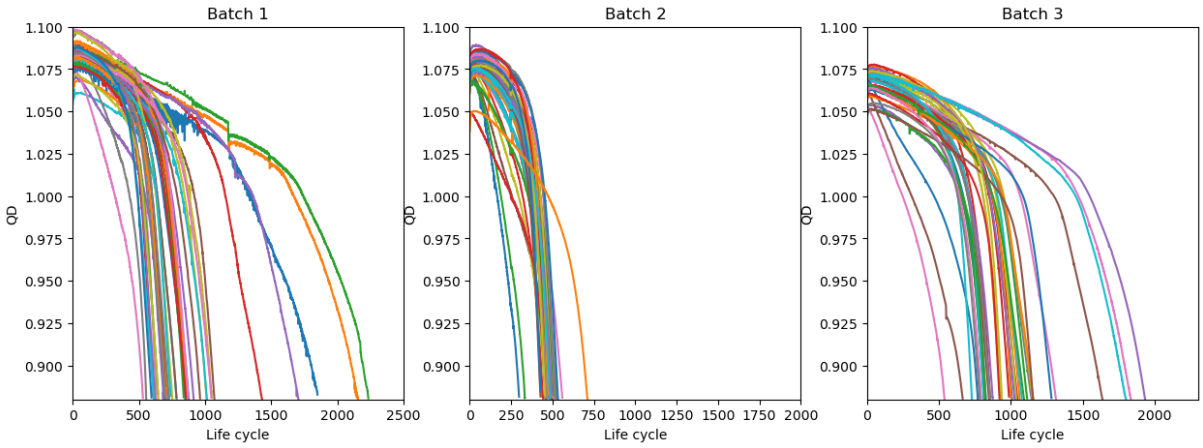


Figure 5.3: Discharge Capacity for each batch.

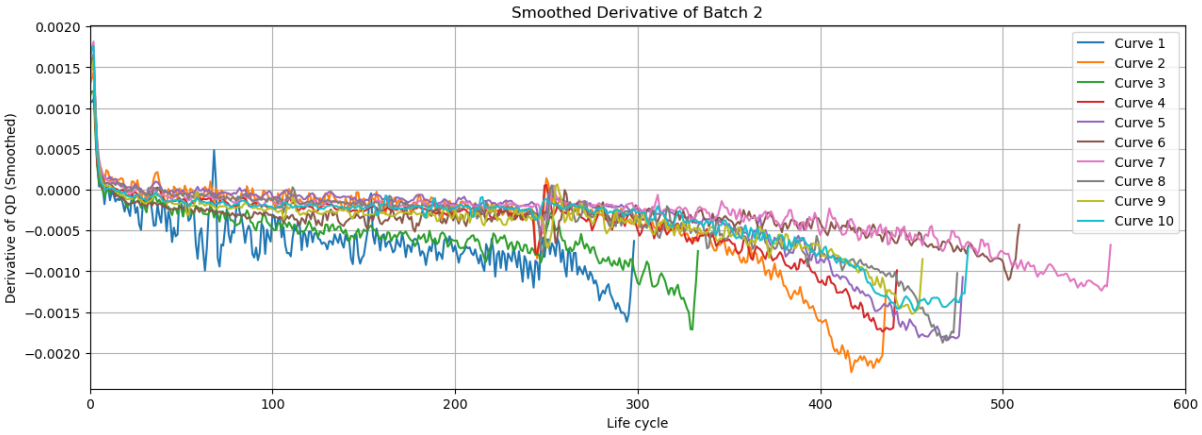


Figure 5.4: Derivative of the discharge capacity of the first 10 batteries in Batch 2.

challenge. It is worth noting that we have not yet accounted for the noise in the data, as QD values may oscillate at certain points.

To gain a deeper understanding of the steepness of the last cycles of each cell, we have chosen to examine the derivative of discharge capacity within the second batch, which serves as our case study. The visual representation of this analysis is provided in Figure 5.4, where we have intentionally selected only 10 batteries at random from this batch for the sake of clarity.

What becomes evident is that, as each battery in this subset nears the end of its lifespan, typically around 150 cycles before, there is a sharp decline in the derivative of the discharge capacity. This observation highlights a distinct and consistent pattern in the dataset.

5.2. Data Preprocessing Overview

The initial phase of data preprocessing was handled by the MIT Toyota Research Group, the custodians of the dataset. They conducted an initial culling by excluding a selected group of battery cells that did not reach at least 80% of their capacity, keeping those above. These excluded cells were deemed irrelevant as they had not reached the end of their operational life (EOL), making them unsuitable for training a model that directly predicts RUL as previously explained. Following this, we took over the data preprocessing.

5.2.1. Data Cleaning

Our efforts began with the identification and treatment of outliers. This involved replacing anomalous data points, including null values and sensor measurement errors, with a calculated average derived from adjacent data points in the battery discharge curve. This step was pivotal in ensuring data quality.

Subsequently, we segmented the battery capacity fade curve at the 80% mark of the battery's nominal capacity, equivalent to 0.88 Ampere-hours (Ah). This demarcation point represented the EOL threshold, helping us determine the actual number of useful life cycles each battery had undergone.

Recognizing the inherent noise in the charge-discharge (QD) curves, we opted to apply a smoothing algorithm. This straightforward technique involved substituting data points at a given cycle with the average of the preceding and succeeding cycles if a drop of 0.005 (Ah) or more was observed. This adjustment led to more consistent and reliable data, making the model less susceptible to erroneous readings.

5.2.2. Data Selection and Maximum Prediction Horizon

Finally, we choose to consider only the most recent 800 aging cycles. This approach prioritizes the latter part of the battery's life, as providing RUL estimates for early-life cycles is less crucial. We can say that focusing on the cycles towards the EOL aligns with the idea of a model that pays maximum attention to the *last 800 predictions*, which are deemed the most relevant for PHM purposes in our case.

5.3. Hyperparameter and Model Optimization

Building on the foundational research presented in a previous thesis [29], our objective was to adapt an LSTM model with attention mechanisms for our specific datasets. Our

datasets are distinct, featuring a smaller number of batteries (48 in total) in comparison to those in prior studies, such as the Sandia Dataset which consists of 90 batteries (referenced in Section 2.5). This smaller dataset size necessitated a meticulous fine-tuning of both the model and its hyperparameters to ensure optimal performance.

5.3.1. Initial Manual Adjustments

Before resorting to automated tools, we manually adjusted certain aspects of the model and its hyperparameters. Some primary adjustments included altering the number of layers, adding, removing, or changing regularization layers and techniques (Dropout, Batch Normalization, L1_L2 regularization ...). This manual optimization phase served as a benchmark, setting the stage for more systematic and rigorous optimization strategies.

5.3.2. Automated Hyperparameter Tuning with Keras-Tuner

To systematically explore the vast hyperparameter space and enhance the model's performance, we used Keras-Tuner, a powerful tool for hyperparameter optimization.

Optimization Techniques Employed:

1. **Random search**

   Random search serves as a robust and straightforward approach. It samples random combinations of hyperparameters and determines the best combination. Surprisingly, random search got some nice improvements with respect to manual optimization.

2. **Bayesian optimization**

   A probabilistic model-based approach that seeks the input values to a function that results in an output that is better than the current best-known value. Using Bayesian optimization, we observed fast and reliable searches. We were convinced that this could be the be-all and end-all of algorithms with a run time of under an hour.

3. **Hyperband**

   At last, we gave a shot to Hyperband, an adaptive resource allocation and early-stopping strategy. It is designed to converge to the best model by allocating more resources to promising configurations. Initial trials with Hyperband showed even better performance. But there was a cost to this, the search time (on GPU) was

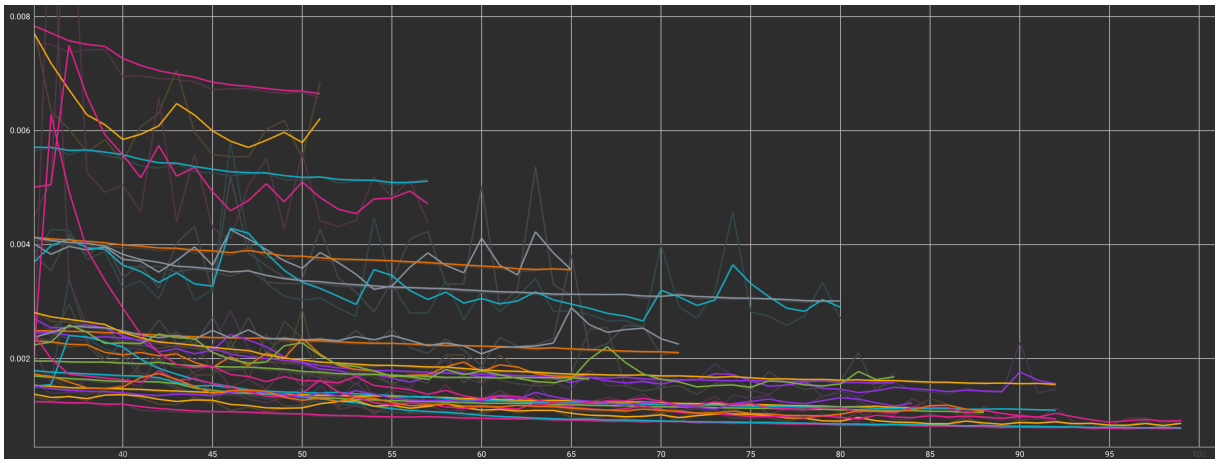


Figure 5.5: Hyperband Regression Error (Loss) per epoch for each optimization attempt, accuracy on the x-axes and epochs on the y-axes.

Hyperparameters	MAE	RMSE	RMSE 400	RMSE 100	RMSE 50
Original	29.8	43.7	33	10.3	9.2
Hyperband	14.7	39.5	28.0	4.0	2.4

Table 5.1: Performance before and after Hyperparameter optimization, on Batches 2-3.

longer than 10 hours. But the results are astonishing, the best and most reliable ones. In Figure 5.5, the algorithm's performance is showcased, highlighting that only the most promising hyperparameter sets (i.e., those yielding the highest accuracy on the validation dataset) are carried forward to the target epoch number.

5.3.3. Final Model Configuration

Post-optimization, the optimal LSTM with attention model for our dataset, in tandem with the optimized hyperparameters, showed an almost 50% average improvement in prediction accuracy compared to the base model from the previous thesis on our datasets, in Table 5.1.

Through a combination of manual adjustments and systematic hyperparameter tuning using various optimization techniques, we achieved a significant enhancement in our LSTM with Attention model's performance. This optimization journey underscores the importance of hyperparameter tuning, especially when transitioning between datasets with distinct characteristics.

5.4. Domain Adversarial Neural Networks

The presented methodology refines the domain adversarial neural network for the unique task of estimating the remaining useful life of lithium-ion batteries. Unlike the original DANN's implementation [14] on image classification, this adaptation introduces several key modifications tailored for RUL prediction.

Feature Extraction: Instead of its image-based roots, the DANN has been reconfigured to facilitate the extraction of information from battery data sequences. This involves the application of 1D convolutional and LSTM layers, tailored to effectively capture meaningful features inherent in sequential battery data.

RUL Prediction: The nature of the task has shifted from image classification to RUL estimation, a regression problem. Consequently, the model now integrates a RUL predictor designed to predict continuous RUL values. This module is equipped with multiple dense layers, optimized for the intricacies of regression tasks.

- **Domain Adaptation**:

  A central concern in this method is the challenge of domain adaptation, which seeks to map the relevant feature distributions, in a domain-invariant manner between two distinct data domains: the source domain, often representative of the training dataset, and the target domain, which encapsulates data sourced from batteries for which RUL estimations are required.

- **Loss Functions**:

  The method uses a pair of loss functions. The first one is a customized loss function tailored to the nuances of RUL estimation [29] for the primary task, the regression one. The secondary task, meaning the domain classification, implements a loss called *Binary Cross-Entropy*. These functions cater to the distinctive requirements of the tasks, encompassing both the primary RUL prediction objective and the secondary domain adaptation component, thus guiding the model's training process effectively.

- **Performance Monitoring**:

  To assess the model's performance, a range of metrics is employed, the main ones, reported in the experiments section, are Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

5.4.1. Structure and Configuration

A DANN consists of three primary components: a feature extractor, a regressor, and a domain classifier. Its strength lies in ensuring the feature extractor produces features beneficial for label prediction, while also being domain-agnostic. We can see the structure of our model in Figure 5.6.

- **Feature extractor**

  It is composed of a pair of one-dimensional convolutional and pooling layers, before an LSTM layer using ReLU activation functions.

- **Regressor**

  All LSTM outputs are used to build the attention and decoder part of the regressor. These, flattened, go straight into a stack of 4 dense layers. It is implemented a softmax output layer for RUL prediction.

- **Domain classifier**

  It adopts a sigmoid activation function for determining the domain of the input. After multiple attempts, we got the best results by feeding as input of the domain classifier the output of the first LSTM layer, we tried to put it right after the convolutional layers or after the second LSTM layer (decoder) but we got worse performance.

5.4.2. Training Strategy and Domain Adversarial Process

Within the domain adversarial neural network framework, a key component is the domain predictor, equipped with a gradient reversal layer (GRL). This layer plays an essential role in orchestrating an "adversarial game" between the domain classifier and the primary RUL estimation task. The central player in this dynamic is the λ parameter, which governs the degree of domain adaptation throughout the training phases.

During backpropagation, the GRL inverts the sign of the gradient, a process modulated by λ. By altering the gradient weights, λ effectively converts the standard gradient descent process into a competitive algorithm. This competition is between the primary task of RUL estimation and the secondary task of mapping domain-invariant features. Such a mechanism can lead to complex behaviors in the loss function, including the emergence of periodic orbits and behaviors reminiscent of chaotic systems [18], thus presenting a challenging landscape for training the network.

Batch	RMSE	MAE	MAE 400	MAE 100	MAE 50
source 2-3	90.6	64.8	66.7	16.2	14.6
target 1	166.1	124.4	104.4	11.4	10.0

Table 5.2: Here displayed are the results on combination 2-3, the others can be found in Appendix A

After some meticulous research [13, 15], we went for an empirical approach, since in the literature there was no evidence of a better strategy for the λ parameter tuning, and so we tried three main strategies:

- **Fixed λ**

  We started with a fixed λ value, and we tried to find the best value for it. We found that this approach was not very effective, since the model was not able to converge to a good solution.

- **Dynamic λ**

  We tried to change the λ value during the training process, starting from 0.1 and then increasing it to different sets of values, algorithmically chosen between 0.1 and 1. We found out that this approach was not very effective, since the model was not able to converge to a good solution.

- **Dynamic λ reversed**

  We tried to change the λ value during the training process, starting from a higher set of values, giving hence more relevance to the domain adaptation task in the first epochs and then decreasing it with multiple possible steepness, until a different set of lower or higher values was reached, depending on the specific chosen implementation. We found that this approach, scheduler like, was the best one since the model was able to converge to a good solution.

One of the last big design decisions we made was to decide where the domain classifier head should attach. This was quite challenging as a lot of tries were made to find the sweet spot. In the end, we decided to give the domain classifier the "knowledge" coming from the encoder, excluding the attention and decoder layers, because of the limited amount of data Figure 5.7c. The choice of using the encoder as the last layer of the feature extractor Figure 5.7a came from the literature [15] and the knowledge we gained through experience, that the output of a layer geared to time series prediction was better suited for the job.

In our most recent round of DANN model optimization, we are satisfied with the performance levels achieved. A simplified representation of the entire model can be found in Figure 5.6 as detailed in Table 5.2, it is expected that the performance within the source domain outshines that in the target domain. Notably, the discrepancy in error between source and target predictions is minimal. Key to this success is the model's training approach, which is tailored to extract domain-agnostic features. By doing so, the DANN is equipped to perform well not only in the domain it was originally trained on but also in new, different domains.

5.4.3. Lessons and Takeaways

The intrinsic design of the DANN, with its three major components: feature extractor, regressor, and domain classifier, ensures the extraction of domain-agnostic features, a pivotal requirement for our application. Our exploration into its structure revealed the effectiveness of leveraging LSTM layers within the feature extractor and the role of attention in the regressor, with the domain classifier leveraging the outputs of the initial LSTM layer.

Through our exploration and experimentation, it became clear that understanding and correctly calibrating the interplay between the feature extraction and domain classification processes is pivotal to harnessing the true potential of the DANNs for domain adaptation. Moreover, the selection of the encoder as the last layer of the feature extractor reaffirms the synergy between time series prediction and domain adaptation, illuminating a pathway for future applications and optimizations in this domain.

5.5. Comparison

Upon training, the DANN's performance was juxtaposed against two other strategies, namely transfer learning with and without fine-tuning introduced in section 4.6. The reason for the different outcomes performance-wise is to be found in the training process. When training the model for TL, we train the model on source domain data without any domain adaptation techniques. Differently, when training the DANN, domain adaptation is intrinsic to the model.

The results presented in Table 5.3 reveal that the performance is quite similar to that of transfer learning combined with fine-tuning. When examining predictions from the last 100 and 50 cycles, the fine-tuning model emerges as the leader in performance, closely followed by the DANN and then basic transfer learning. However, when we shift our

Target	RMSE	MAE	MAE 400	MAE 100	MAE 50
TL	183.7	175.2	142.7	22.2	14.3
TF with F-T	104	105.0	74.4	**12.4**	**5.8**
DANN	**103.0**	**76.2**	**43.0**	14.0	8.8

Table 5.3: Performance on target domain (Batch 1), comparison between Transfer learning with and without fine-tuning and DANN.

focus to longer-term predictions, the DANN outperforms the others, exhibiting distinctly superior results. We assume that this is attributed to the DANN's enhanced capability for generalization. As a more intricate model, the DANN excels at extracting features that are not strictly tied to a specific domain. This proficiency is especially evident in the early cycles, where the domain's uniqueness is not as influential, and in contrast, the generalization capabilities are more relevant since in the early cycles the battery behavior is similar. This allows the DANN to deliver a stronger performance.

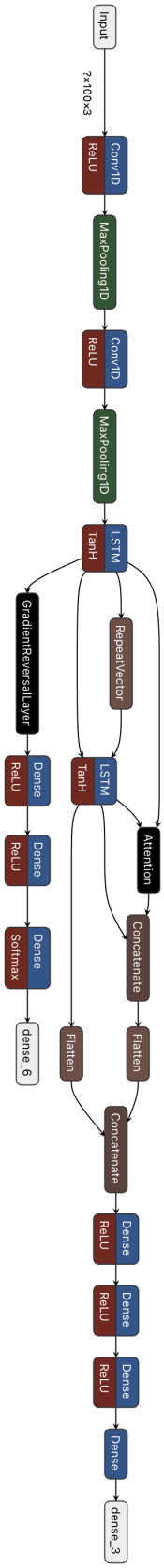
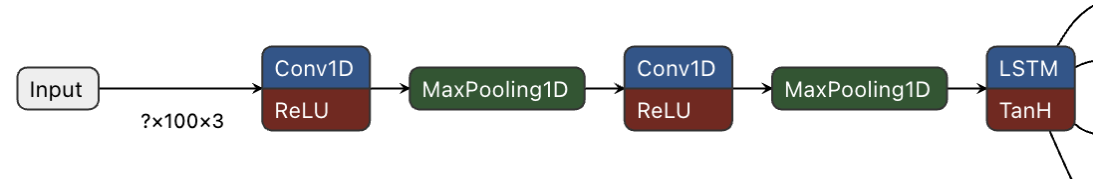
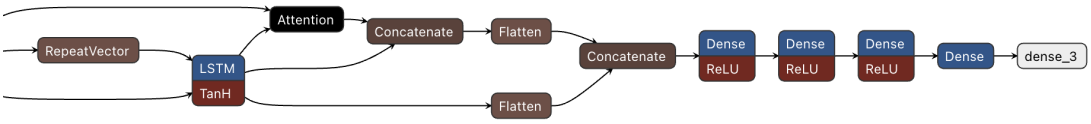


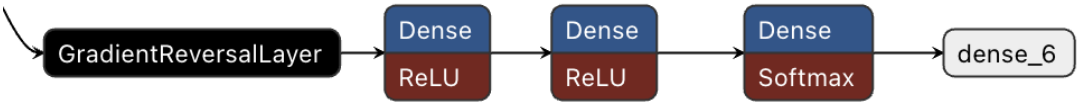
Figure 5.6: Final DANN model.



(a) Feature extractor



(b) Regressor head



(c) Domain predictor

Figure 5.7: Final DANN model decomposed to show better the single components of each part of the network. Figure 5.7b and Figure 5.7c both use the outputs of the last layer of Figure 5.7a.

6 | Conclusions and Future Developments

In this thesis, we delved into the domain adaptation challenges in Remaining Useful Life (RUL) estimation of Li-ion batteries. To tackle these issues, we introduced a novel methodology that was previously employed in entirely different contexts.

This consisted in the implementation of a Domain Adversarial Neural Network (DANN) to adjust the neural network's representations of specific features to make them similar across different domains, effectively aligning the selected feature distributions.

We showed that utilizing a DANN for our dataset is a promising approach worthy of significant attention. This stems from the model's unsupervised learning paradigm, distinguishing it from conventional transfer learning methods. Its adaptability and intuitive foundation further enhance its appeal.

DANNs enable addressing domain adaptation without needing explicit labels from the source or target domains, which is invaluable in situations where labeled data is either unavailable or costly. Moreover, the flexibility of the model allows its easy integration into various neural architectures, making it an enticing choice for domain adaptation tasks.

However, in cases with limited data, like our study, involving *only eight battery* cells' data from the target domain, we observed that the effectiveness of fine-tuning, given the presence of labeled data, surpasses domain adversarial learning. This is probably because fine-tuning capitalizes on target-specific insights, enhancing predictive accuracy by aligning closely with the target domain's unique attributes. Conversely, domain adversarial learning, since it aims for domain invariance, might miss some critical domain-specific details of the target.

Furthermore, our work revealed a moderate performance improvement when we incorporated traditional data augmentation techniques, such as jittering, into the previous approaches. This enhancement was most pronounced when manually adjusted to align with the dataset's specific characteristics.

It is also worth mentioning that, in this thesis, we also introduced a systematic method for optimizing hyperparameters in the convolutional-LSTM model. Our approach demonstrated a remarkable increase in average prediction accuracy while reducing the time investment compared to earlier manual methods.

In conclusion, the major contributions arising from our research shed light on the capabilities of deep domain adaptation in the realm of RUL estimation, demonstrating significant and robust performance gains. While our methods showed promise, it is crucial to exercise caution during transfer learning. We have observed that a brute-force transfer approach can potentially lead to a degradation in our model's performance.

A central focus of our thesis was the exploration of homogeneous domain adaptation, which implies that there is a relationship between the data from the source domain and the target domain, and we presume that deep neural networks can uncover some shared representation from these two domains.

Future research could focus on discovering innovative data augmentation techniques designed to mitigate the scarcity of data while preserving the inherent characteristics of sensor data, or using a different model, maybe based on the transformer architecture.

Indeed, one promising avenue for augmentation techniques exploration involves leveraging Generative Adversarial Networks (GANs) [30] to generate synthetic data. This approach enables more effective training of the network with a diverse yet consistent dataset. By incorporating this strategy, the model becomes exposed to various patterns during training, which can later prove beneficial during online deployment or testing, thereby enhancing overall performance.

The other potential avenue for future research could be driven by the recent advancements in the capabilities of the transformer architecture [26]. Notably, this architecture has recently surpassed the recurrent neural network family in various tasks, including computer vision and natural language processing. However, its application in prognostic health management tasks, such as the one discussed, remains an area yet to be explored.

Bibliography

[1] Q. Ajao, L. Sadeeq, O. Oludamilare, and M. Qasim. An overview of lithium-ion battery dynamics for autonomous electric vehicles: A matlab simulation model. *OALib*, 10:1–16, 01 2023. doi: 10.4236/oalib.1110272.

[2] K. S. S. Alamin, Y. Chen, E. Macii, M. Poncino, and S. Vinco. A machine learning-based digital twin for electric vehicle battery modeling. In *2022 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*, pages 1–6. IEEE, 2022.

[3] Z. E. Amogne, F.-K. Wang, and J.-H. Chou. Transfer learning based on transferability measures for state of health prediction of lithium-ion batteries. *Batteries*, 9, 2023. doi: 10.3390/batteries9050280. URL `https://doi.org/10.3390/batteries9050280`.

[4] D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014. URL `https://api.semanticscholar.org/CorpusID:11212020`.

[5] M. H. A. Banna, T. Ghosh, M. J. A. Nahian, K. A. Taher, M. S. Kaiser, M. Mahmud, M. S. Hossain, and K. Andersson. Attention-based bi-directional long-short term memory network for earthquake prediction. *IEEE Access*, 9:56589–56603, 2021. doi: 10.1109/ACCESS.2021.3071400.

[6] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira. Analysis of representations for domain adaptation. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Proceedings of advances in Neural Information Processing Systems*, volume 19, pages 137–144. MIT Press, 2006. URL `https://proceedings.neurips.cc/paper_files/paper/2006/file/b1b0432ceafb0ce714426e9114852ac7-Paper.pdf`.

[7] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13:281—-305, feb 2012. ISSN 1532-4435.

[8] M. Chen, Z. Xu, K. Q. Weinberger, and F. Sha. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Coference on International Conference on Machine Learning*, ICML'12, pages 1627—-1634, Madison, WI, USA, 2012. Omnipress. ISBN 9781450312851.

[9] B. cheng Wen, M. qing Xiao, X. qi Wang, X. Zhao, J. feng Li, and X. Chen. Data-driven remaining useful life prediction based on domain adaptation. *PeerJ Computer Science*, 7:e690, 2021.

[10] J.-H. Chou, F.-K. Wang, and S.-C. Lo. A novel fine-tuning model based on transfer learning for future capacity prediction of lithium-ion batteries. *Batteries*, 9:325, 2023. doi: 10.3390/batteries9060325. URL `https://doi.org/10.3390/batteries9060325`.

[11] P. R. de Oliveira da Costa, A. Akçay, Y. Zhang, and U. Kaymak. Remaining useful lifetime prediction via deep domain adaptation. *Reliability Engineering & System Safety*, 195:106–682, 2020. ISSN 0951-8320. doi: https://doi.org/10.1016/j.ress.2019.106682. URL `https://www.sciencedirect.com/science/article/pii/S0951832019304946`.

[12] A. Farahani, S. Voghoei, K. Rasheed, and H. R. Arabnia. A brief review of domain adaptation. In *Proceedings from ICDATA 2020 and IKE 2020*, pages 877–894. Springer, 2021.

[13] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *Procedings of the international conference on machine learning*, pages 1180–1189. PMLR, 2015.

[14] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17(1):2096–2030, jan 2016. ISSN 1532-4435.

[15] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

[16] H. Guan and M. Liu. Domain adaptation for medical image analysis: a survey. *IEEE Transactions on Biomedical Engineering*, 69(3):1173–1185, 2021.

[17] Y. Guo, H. Shi, A. Kumar, K. Grauman, T. Rosing, and R. Feris. Spottune: transfer learning through adaptive fine-tuning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4805–4814, 2019.

[18] T. Hazra and K. Anjaria. Applications of game theory in deep learning: a survey. *Multimedia Tools and Applications*, 81(6):8963–8994, 2022.

[19] H. He, O. Queen, T. Koker, C. Cuevas, T. Tsiligkaridis, and M. Zitnik. Domain adap-

tation for time series under feature and label shifts. *arXiv preprint arXiv:2302.03133*, 2023.

[20] X. G. i Nieto (UPC). Insight@dcu insight deep learning workshop. *GitHub*, 2017. URL `https://github.com/telecombcn-dl/dlmm-2017-dcu`.

[21] G. Iglesias, E. Talavera, Á. González-Prieto, A. Mozo, and S. Gómez-Canaval. Data augmentation techniques in time series domain: a survey and taxonomy. *Neural Computing and Applications*, 35(14):10123–10145, May 2023. ISSN 1433-3058. doi: 10.1007/s00521-023-08459-3. URL `https://doi.org/10.1007/s00521-023-08459-3`.

[22] S. Joshi, J. A. Owens, S. Shah, and T. Munasinghe. Analysis of preprocessing techniques, keras tuner, and transfer learning on cloud street image data. In *In Proceedings of the 2021 IEEE International Conference on Big Data (Big Data)*, pages 4165–4168, 2021. doi: 10.1109/BigData52589.2021.9671878.

[23] A. Karajgi. How neural networks solve the xor problem. *Towards Data Science*, 2020. URL `https://towardsdatascience.com/how-neural-networks-solve-the-xor-problem-59763136bdd7`.

[24] M. T. Lawder, B. Suthar, P. W. Northrop, S. De, C. M. Hoff, O. Leitermann, M. L. Crow, S. Santhanagopalan, and V. R. Subramanian. Battery energy storage system (bess) and battery management system (bms) for grid-scale applications. *Proceedings of the IEEE*, 102(6):1014–1030, 2014.

[25] M. H. Lipu, M. Hannan, A. Hussain, M. Hoque, P. J. Ker, M. H. M. Saad, and A. Ayob. A review of state of health and remaining useful life estimation methods for lithium-ion battery in electric vehicles: Challenges and recommendations. *Journal of cleaner production*, 205:115–133, 2018.

[26] X. Liu, C. Yang, Y. Meng, J. Zhu, and Y. Duan. Capacity estimation of Li-ion battery based on transformer-adversarial discriminative domain adaptation. *AIP Advances*, 13(7):75–113, 07 2023. ISSN 2158-3226. doi: 10.1063/5.0152038. URL `https://doi.org/10.1063/5.0152038`.

[27] M. Long, Y. Cao, J. Wang, and M. Jordan. Learning transferable features with deep adaptation networks. *arXiv preprint arXiv:1502.02791*, 2015.

[28] G. Loye. Attention mechanism. *Floydhub*, 2019. URL `https://blog.floydhub.com/attention-mechanism/`.

[29] L. Martiri. A method for estimating the remaining useful life of lithium-ion batteries when nearing end of life. Master's thesis, Scuola di Ingegneria Industriale e

dell'Informazione, Politecnico di Milano, May 2023. URL https://hdl.handle.net/10589/211681. Anno accademico 2022/2023.

[30] F. Naaz, A. Herle, J. Channegowda, A. Raj, and M. Lakshminarayanan. A generative adversarial network-based synthetic data augmentation technique for battery condition evaluation. *International Journal of Energy Research*, 45(13):19120–19135, 2021.

[31] K. Nguyen, T. Le, V. Nguyen, T. Nguyen, and D. Phung. Multiple kernel learning with data augmentation. In R. J. Durrant and K.-E. Kim, editors, *Proceedings of The 8th Asian Conference on Machine Learning*, volume 63 of *Proceedings of Machine Learning Research*, pages 49–64, The University of Waikato, Hamilton, New Zealand, 16–18 Nov 2016. PMLR. URL https://proceedings.mlr.press/v63/nguyen19.html.

[32] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. doi: 10.1109/TKDE.2009.191.

[33] H. Qin, X. Fan, Y. Fan, R. Wang, Q. Shang, and D. Zhang. A transferable prediction approach for the remaining useful life of lithium-ion batteries based on small samples. *Applied Sciences*, 13:84–98, 2023. doi: 10.3390/app13148498. URL https://doi.org/10.3390/app13148498.

[34] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak. Convolutional, long short-term memory, fully connected deep neural networks. In *Proceedings of 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4580–4584. Ieee, 2015.

[35] K. A. Severson, P. M. Attia, N. Jin, N. Perkins, B. Jiang, Z. Yang, M. H. Chen, M. Aykol, P. K. Herring, D. Fraggedakis, et al. Data-driven prediction of battery cycle life before capacity degradation. *Nature Energy*, 4(5):383–391, 2019.

[36] S. Shao, S. McAleer, R. Yan, and P. Baldi. Highly accurate machine fault diagnosis using deep transfer learning. *IEEE Transactions on Industrial Informatics*, 15(4):2446–2455, 2019. doi: 10.1109/TII.2018.2864759.

[37] A. Sicilia, X. Zhao, and S. J. Hwang. Domain adversarial neural networks for domain generalization: when it works and how to improve. *Machine Learning*, 112(7):2685–2721, 2023. doi: 10.1007/s10994-023-06324-x. URL https://doi.org/10.1007/s10994-023-06324-x.

[38] X. Tang, K. Liu, K. Li, W. D. Widanage, E. Kendrick, and F. Gao. Recovering large-scale battery aging dataset with machine learning. *Patterns*, 2(8):100–302, 2021. ISSN 2666-3899. doi: https://doi.org/10.1016/j.patter.2021.100302. URL `https://www.sciencedirect.com/science/article/pii/S2666389921001458`.

[39] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

[40] B. University. Bu-409: Charging lithium-ion. *Battery University*, 2021. URL `https://batteryuniversity.com/article/bu-409-charging-lithium-ion`.

[41] F.-K. Wang, Z. E. Amogne, J.-H. Chou, and C. Tseng. Online remaining useful life prediction of lithium-ion batteries using bidirectional long short-term memory with attention mechanism. *Energy*, 254:124344, 2022.

[42] K. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.

[43] Q. Wen, L. Sun, F. Yang, X. Song, J. Gao, X. Wang, and H. Xu. Time series data augmentation for deep learning: A survey. *arXiv preprint arXiv:2002.12478*, 2020.

[44] L. Wu, X. Fu, and Y. Guan. Review of the remaining useful life prognostics of vehicle lithium-ion batteries using data-driven methodologies. *Applied Sciences*, 6 (6):166, 2016.

[45] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille. Adversarial examples for semantic segmentation and object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1369–1378, 2017.

[46] S. Yang, C. Zhang, J. Jiang, W. Zhang, L. Zhang, and Y. Wang. Review on state-of-health of lithium-ion batteries: Characterizations, estimations and applications. *Journal of Cleaner Production*, 314:0959–6526, 2021.

[47] Z. Yang, W. Cohen, and R. Salakhudinov. Revisiting semi-supervised learning with graph embeddings. In *In Proceedings of the International conference on machine learning*, pages 40–48. PMLR, 2016.

[48] Z. Ye and J. Yu. State-of-health estimation for lithium-ion batteries using domain adversarial transfer learning. *IEEE Transactions on Power Electronics*, 37(3):3528–3543, 2021.

[49] K. Zhang, B. Schölkopf, K. Muandet, and Z. Wang. Domain adaptation under target

and conditional shift. In *In Proceedings of the International Conference on Machine Learning*, pages 819—827, 2013.

[50] F. Zhao, W. Liu, and C. Wen. A new method of image classification based on domain adaptation. *Sensors*, 22(4):1315, 2022.

A | Extended Results

This appendix comprehensively presents the complete set of results derived from our experimental investigations. Due to space constraints and the extensive nature of our data, it was not feasible to include these detailed results within the main body of the thesis. Here, readers will find a thorough compilation of tables that provide an in-depth view of the outcomes and findings from our research. This extended presentation of results is aimed at offering a more complete understanding of our study and facilitating a deeper analysis of the data and conclusions drawn in the main text.

A.1. Transfer Learning and Fine-Tuning

A.1.1. Transfer Learning

Tables: In Table A.1a we can see the results for training with batches 1-3, and testing on batch 2 with this transfer learning approach. While in Table A.1b there are the results for training with batches 1-2 and testing with batch 3.

Batch:	MAE	RMSE	MAE 400	MAE 100	MAE 50
Transfer learning b2	204	241.5	199.3	148.1	56.2

(a) Performance on target domain (batch 2) with a model trained on batch 1-3.

Batch:	MAE	RMSE	MAE 400	MAE 100	MAE 50
Transfer learning b3	142.0	176.3	107.8	29.5	16.8

(b) Performance on target domain (batch 3) with a model trained on batch 1-2.

Table A.1: Transfer Learning Results.

A.1.2. Fine-Tuning

Tables: In Table A.2a we can see the results for training with batches 1-3, and testing on batch 2 after applying fine-tuning for 10 epochs on 8 batteries. In Table A.2b there are the results for training with batches 1-2 and testing with batch 3 after applying fine-tuning for 10 epochs on 8 batteries.

Batch:	MAE	RMSE	MAE 400	MAE 100	MAE 50
Fine-Tuning b2	45.1	79.1	40.3	13.2	6.1

(a) Performance on target domain (batch 2) after fine-tuning for 10 epochs.

Batch:	MAE	RMSE	MAE 400	MAE 100	MAE 50
Fine-Tuning b3	89.3	130.8	49.7	16.8	10.7

(b) Performance on target domain (batch 3) after fine-tuning for 10 epochs.

Table A.2: Fine-tuning results.

A.2. Domain Adversarial Neural Network

A.2.1. Two-Batch DANN Performance

Tables containing data when training DANN

Batch	RMSE	MAE	MAE 400	MAE 100	MAE 50
source b1-b2	73.2	47.2	29.5	7.2	6.1
target b3	120.6	89.1	60.5	24.3	12.7

Table A.3: DANN trained on batches 1-2.

Batch	RMSE	MAE	MAE 400	MAE 100	MAE 50
source b1-b3	56.6	39.0	20.0	11.8	12.1
target b2	123.6	101.3	117.6	85.4	52.5

Table A.4: DANN trained on batches 1-3.

A.2.2. Single Batch DANN Performance

Tables containing data when training with a single batch DANN

Batch	RMSE	MAE	MAE 400	MAE 100	MAE 50
train b1	56.4	40.6	17.6	9.4	4.4
test b2	148.7	138.8	171.0	124.2	107.7
test b3	107.3	82.5	67.6	26.3	15.8

Table A.5: DANN trained on batch 1.

Batch	RMSE	MAE	MAE 400	MAE 100	MAE 50
train 2	19.4	16.7	16.7	9.2	4.2
test 1	129.5	92.7	67.4	12.4	4.4
test 3	189.7	146.2	105.5	40.3	24.6

Table A.6: DANN trained on batch 2.

Batch	RMSE	MAE	MAE 400	MAE 100	MAE 50
train b3	64.1	48.3	32.6	17.7	11.8
test b2	197.3	164.7	105.5	81.6	71.8
test b1	146.6	109.5	180.6	28.0	28.3

Table A.7: DANN trained on batch 3.

B | Machine Learning Background

In this appendix, we want to give a brief description of what machine learning deals with, what a neural network is in general, what is a recurrent neural network, and why we chose an LSTM architecture with the attention mechanism.

B.1. Fundamentals of Machine Learning

B.1.1. Supervised vs. Unsupervised Learning

Machine learning encompasses a rich landscape of models, each designed to address specific problem domains. To navigate this diversity, we categorize problems based on the availability of labeled data. This categorization forms the foundation of supervised and unsupervised learning.

In supervised learning, we work with data where the target variable, representing the outcome we want to predict, is known and available for our independent set of features. The essence of this approach lies in teaching the model by showing it examples, allowing it to learn the relationships between the features and the target variable. The model adjusts its predictions based on the feedback provided by the known labels, striving to make increasingly accurate predictions over time.

On the other hand, unsupervised learning is the realm of problems where we lack labeled data, leaving the model to explore and identify patterns in the data on its own. It autonomously groups or organizes data points based on inherent similarities or structures, often revealing hidden insights and associations within the data.

B.1.2. Training, Validation, and Testing Sets

In the realm of machine learning, working directly with all available data for model training is often impractical. This is because exposing the model to the entire dataset may lead to overlooking crucial aspects of its performance. Two fundamental considerations arise:

1. **Generalization:**

   A fundamental objective of machine learning is to create models capable of generalizing their knowledge to new, unseen data. However, if a model is exclusively trained on all available data, it may become too specialized, fitting the training data perfectly (overfitting) while struggling to interpret data deviating even slightly from the training dataset's distribution.

2. **Performance assessment:**

   To determine how well a model performs, we require a mechanism to evaluate its performance throughout the training process. Without validation and testing sets, we would be in the dark about how the model is likely to behave with new, unseen data.

To address these challenges, we employ a practice known as data set partitioning. We divide our dataset into three distinct subsets, commonly referred to as the training set, validation set, and test set.

- **Training set**

  This set serves as the foundation for instructing the model. The training data is used to adapt and teach the model by providing it with examples.

- **Validation set**

  The validation set plays a pivotal role in assessing the performance of different models during the training process. It helps gauge how well each model generalizes to new data and aids in the selection of the best model.

- **Test set**

  The test set is kept separate from the beginning and is not used during the training or validation phases. It serves as a final, unbiased assessment of the model's performance. Evaluating the model with data it has never encountered provides a realistic measure of its effectiveness.

This division into three sets empowers us to strike a balance between training, assessing, and ultimately ensuring that our machine learning model can not only learn from data but also apply its knowledge effectively to new, real-world situations.

B.1.3. Loss Functions and Optimizers

A loss function is essentially a mathematical function that quantifies the disparity between the predicted outcomes of a model and the true labels or target variables for a given sample. It serves as a vital tool for assessing how effectively the model is performing its task, providing a measure of the deviations between predictions and actual outcomes. In our work, we have explored several loss functions, including Root Mean Squared Error (RMS) and Mean Absolute Error (MAE).

An optimizer, on the other hand, is a specific algorithm employed to minimize the loss function and iteratively adjust the model's parameters. Optimizers play a crucial role in training machine learning models by fine-tuning the model's parameters to reduce the error between predictions and actual values. Some of the most relevant optimizers include Gradient Descent (GD), Stochastic Gradient Descent (SGD), and the Adam optimizer.

These concepts of loss functions and optimizers are fundamental components of machine learning and deep learning that play a critical role in training models and enhancing their predictive capabilities.

B.2. Basics on neural networks

B.2.1. Basic Concept of Artificial Neurons

Neural network theory is based on the concept of neurons. This is the atomic element of every neural network. This component aims to mimic the inner workings of a single neuron in the brain. Indeed, it takes one or, usually, multiple inputs, performs a weighted sum, and adds a bias adjustment. The activation function is then applied to introduce non-linearity, as shown in B.1, allowing the neuron to make decisions by activating or not activating in response to the processed input. The output is then sent to the next layer of neurons in the network.

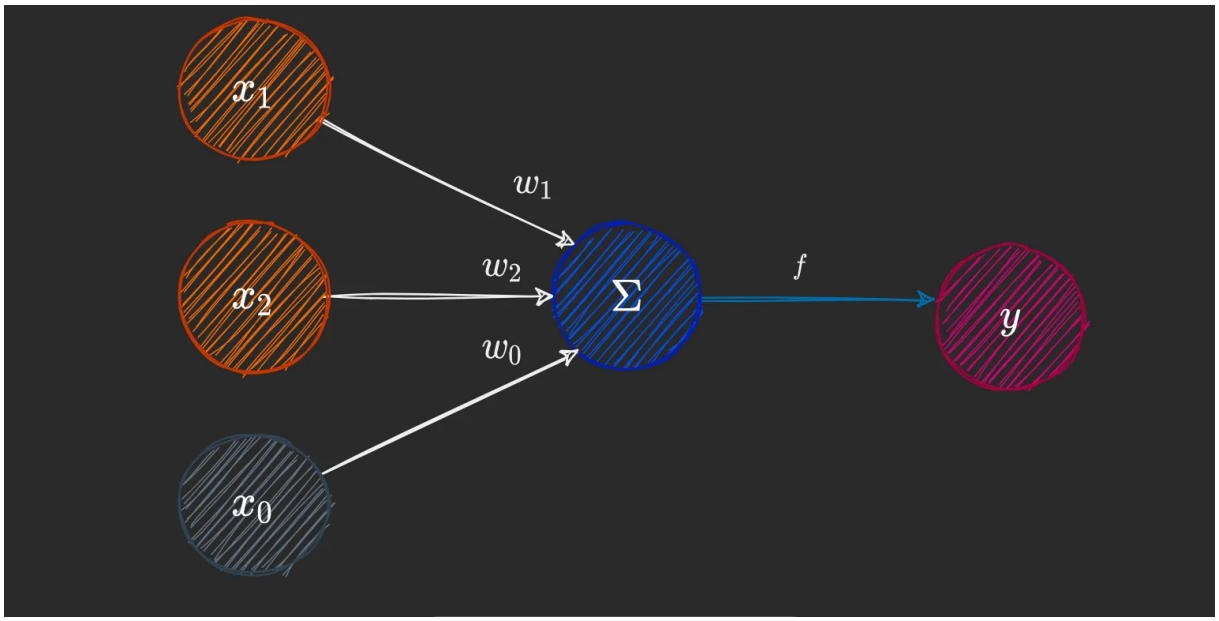


Figure B.1: A neuron, how the output is computed [23].

B.2.2. Overview of Neural Networks

Envision a neural network as a team of neurons working together. Each neuron's output contributes to a layer, and these layers stack together to form a complex network capable of performing intricate tasks. Think of a neural network like a layered orchestra, where each section plays its part to contribute to the overall symphony. Neural networks learn through training, where they iteratively adjust the weights of connections (weights and biases) to predict outcomes more accurately, similar to learning from experience. Connections are updated when the output is wrong, the more the error the bigger the updates.

B.2.3. Types of Neural Networks

Neural networks come in different flavors, each with a unique structure suited for specific tasks. The Feed Forward Neural Network (FNN) is the simplest form, where information travels in one direction only—forward. On the other hand, recurrent neural networks boast a feedback system that gives them a form of memory, making them ideal for tasks like language modeling where past information is crucial. Convolutional neural networks excel in identifying hierarchical patterns, which is why they are the go-to choice for image recognition. Each network's architecture is tailored to the complexities of the task at hand, whether it is recognizing speech, translating languages, or driving autonomous cars.

B.3. Recurrent Neural Networks

B.3.1. The Concept of Sequence Data and Why Use RNNs for Sequence Data

Sequences are a fundamental data type with significant relevance in numerous real-world applications. They represent a series of data points that follow a specific order, such as time series data, natural language text, or sensor measurement sequences such as in our case. The sequential nature of these data types presents unique challenges and opportunities.

Recurrent neural networks are a class of neural networks well-suited for handling sequence data. Unlike traditional feedforward neural networks, RNNs possess a memory mechanism that allows them to capture and model temporal and sequential dependencies within the data. This capability is particularly advantageous in scenarios where the order and context of data points are crucial for making accurate predictions or understanding the underlying patterns.

The advantages of RNNs in modeling sequence data include their ability to maintain internal state information, adapt to variable-length sequences, and dynamically process data points sequentially.

B.3.2. Challenges with Basic RNNs: Vanishing and Exploding Gradients

Training traditional RNNs has been associated with specific challenges that affect their performance. Two of the most prominent challenges are the issues of vanishing and exploding gradients.

In the context of training RNNs, the vanishing gradient problem occurs when the gradients of the loss function with respect to the model's parameters become minuscule during the backpropagation process. This is especially problematic when dealing with long sequences. When gradients vanish, it means that the model has difficulty learning and updating its parameters effectively, which can hinder its ability to capture long-term dependencies in the data.

Conversely, the exploding gradient problem arises when the gradients become exceptionally large during training. This can result in unstable and divergent training processes, making it difficult to optimize the model's parameters effectively.

Addressing these challenges is crucial for improving the effectiveness of RNNs in modeling sequence data. Several advanced RNN variants and techniques, such as LSTM and Gated Recurrent Unit (GRU), have been developed to mitigate the vanishing and exploding gradient problems.

B.4. Long Short-Term Memory Networks

B.4.1. Understanding the LSTM Unit

Presents the specialized architecture of LSTM units that enables learning long-term dependencies.

- **Forget gate**

  Explains how LSTMs decide which information to discard from the cell state.

- **Input gate**

  Describes how LSTMs determine new information to be added to the cell state.

- **Cell state**

  Introduces the component of LSTMs that carries relevant information throughout the sequence processing.

- **Output Gate**

  Details how LSTMs decide what is next to output based on the cell state and the input.

B.4.2. The LSTM Cell State: The Key to Long-term Dependencies

Elucidates the core feature of LSTMs that maintains long-term relationships in data.

B.5. Training LSTMs

Training recurrent neural networks is a little trickier than simple FNNs. In fact, since the prediction is sequence-based, as in it depends on the previous n values to predict the next one, the training must be shaped that way too. Simple backpropagation techniques are extended to support the time factor too, in fact, they are called Backpropagation Through Time (BPTT). This training strategy may result unstable if the exploding gradient is

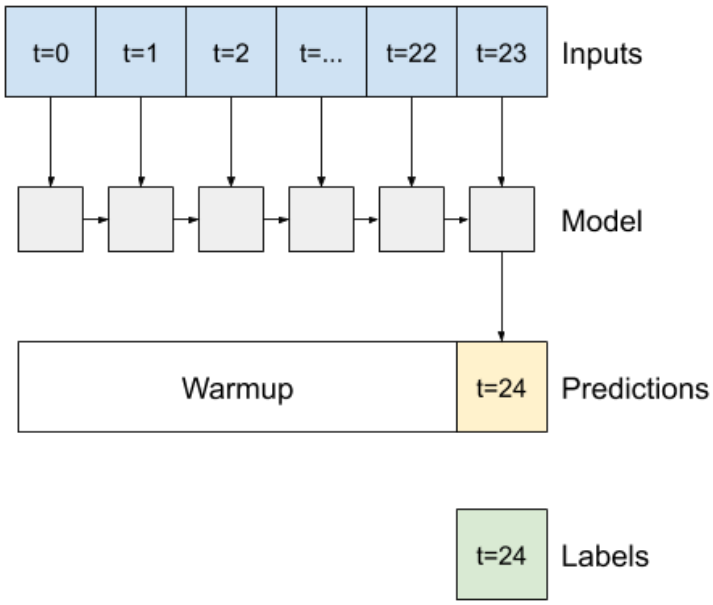


Figure B.2: Simple forecast example that can be done with LSTM.

considered, but the right regularization techniques are enough to tackle this potential issue.

One key aspect of LSTM is choosing the right hyperparameters, here hyperparameter tuning is key to get the best performance for each particular application.

B.5.1. Time Series Prediction

LSTMs have become a mainstay for forecasting in time series data due to their ability to capture long-term dependencies. Their architecture is well-suited for analyzing sequential data for stock markets, weather patterns, and *predictive maintenance*, helping predict future events based on historical data. The gated mechanism of LSTMs allows them to retain relevant past information and discard non-useful data, leading to more accurate predictions over time.

B.5.2. Natural Language Processing

In natural language processing, LSTMs handle complex language structures, making them effective for tasks such as machine translation, sentiment analysis, and text summarization. Their sequential data processing capability enables them to understand context and

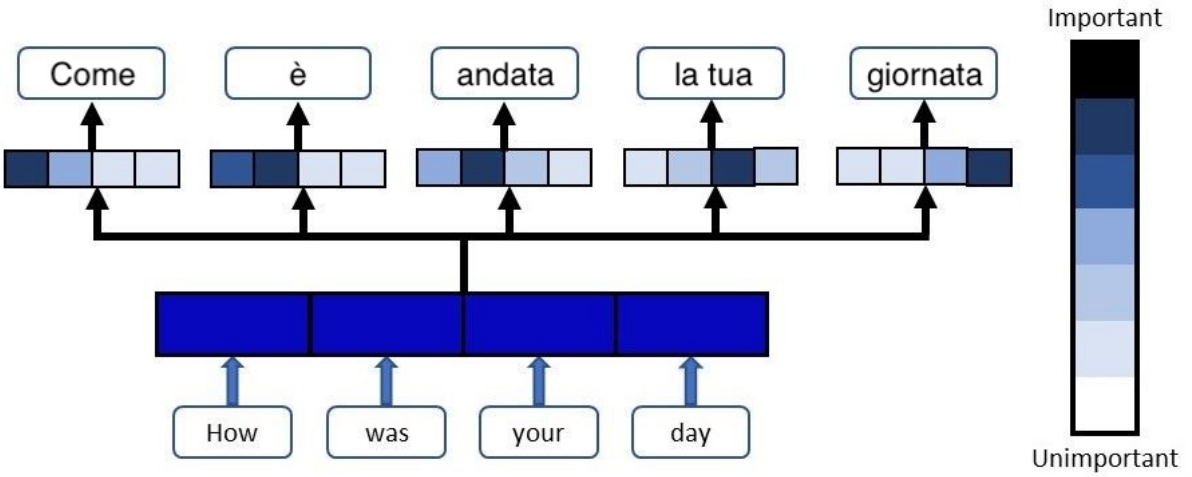


Figure B.3: Translated image from Loye G.'s article [28], shows how attention mechanism helps dynamically focus on different parts of the input sequence in language translation.

generate coherent language patterns, a crucial factor in capturing the intricacies of human language.

B.5.3. Speech Recognition

Speech recognition benefits from LSTM's ability to process audio data over time. These networks excel at modeling the temporal sequences in speech, facilitating the development of advanced voice-activated technologies. LSTMs can differentiate between phonetic nuances by learning from continuous speech flow, significantly improving speech-to-text accuracy and enabling their use in real-time transcription and interactive voice response systems.

LSTM's capacity to learn from sequences makes it a versatile tool for any application requiring the understanding and prediction of temporal data.

B.6. Attention Mechanism

The attention mechanism is a transformative concept in machine learning that addresses the limitations of traditional sequence-to-sequence models such as LSTM networks. It mimics cognitive attention by allowing models to dynamically focus on different parts of the input sequence for each step of the output B.3, thereby significantly improving the handling and interpretation of sequential data.

B.6.1. General Mechanism

Attention provides a way to weight the input data by relevance, enabling the model to access any part of the input sequence directly. This selective focus mechanism is akin to how human attention works, allowing for a more efficient and context-aware processing of information. By doing so, models can overcome the bottleneck of having to encode all information into a fixed-length vector, as was common in earlier seq2seq models.

B.6.2. Attention within LSTMs

Integrating attention with LSTM networks has yielded substantial improvements, particularly in tasks where the input and output sequences are of variable lengths. In standard LSTMs, the capacity to remember information across long sequences diminishes with increasing length due to the constraints of the hidden state's size. With attention, LSTMs can alleviate this issue by referencing the entire input sequence directly, which enhances their ability to maintain and utilize long-range dependencies.

This enhancement is crucial for complex sequence modeling tasks such as language modeling and text generation, where the context can significantly influence the meaning and subsequent prediction. Attention allows LSTM to perform "soft" searches for relevant parts of the input sequence that are most informative for predicting each word or character in the output sequence, leading to more accurate and coherent results.

List of Tables

List of Figures

Acknowledgements

Un ringraziamento speciale va ai professori Francesco Amigoni e Loredana Cristaldi, e ai dottorandi Davide Azzalini e Luca Martiri che ci hanno guidato e supportato per la stesura di questa tesi, insegnandoci tanto.

Luho

Non potevamo prescindere da un paio di parole di ringraziamento per tutte quelle persone che negli ultimi due anni sono state fondamentali per il raggiungimento di questo traguardo. Sento che le prime e più importanti vanno come sempre alla mia famiglia, Mamma, Babbo e Lara, con tutti i nostri difetti e le nostre crisi riusciamo comunque ad esserci sempre l'uno per l'altro. Grazie, siete le mie radici.
Chiara, senza la tua pazienza e il tuo amore sarebbe stato tutto molto più difficile, grazie davvero per condividere questa strada mano nella mano con me.
Peppe sei stato un amico prima che un compagno di tesi, e quindi grazie di cuore, per il supporto, il push e la capacità di farmi spingere oltre le mie aspettative.
Franz, Fora, Pietro, Lore, Chiara, Emma, Milo, Max, Bea, Francesca, Scafibus, Sara, Alessia e tutti i compagni di corso e di Agorà, ognuno di voi è stato importante a modo suo ed ha contribuito umanamente prima che accademicamente a questo punto di arrivo. Grazie. Ringrazio anche i miei coinquilini, Federico e Matilde, per avermi fatto sentire a casa mia anche fuori da casa mia. Via san gregorio avrà sempre un posto importante nel mio cuore grazie a voi. In fine ma non perchè siete gli ultimi, ma anzi in quanto siete la mia seconda famiglia, grazie Sanso, Cris, Giulio e Claudio. Siete riusciti ad essere presenti anche a Milano in qualche modo. Gli amici di Firenze vorrei ringraziarli tutti, Cosimo, Losi, Carca, Mario e tutti quelli che non trovano il nome qua ma sanno che abbiamo comunque condiviso qualcosa.
Danke.

Melo

Come avrete già visto dal testo della tesi scrivere non è il mio forte. In ogni caso questi ringraziamenti sono per tutte quelle persone che mi sono state accanto e mi hanno supportato in questo percorso.

Mi sembra giusto partire dalla mia famiglia. Ringrazio mia mamma Valeria e mio papà Ignazio che, nonostante tutti i casini che hanno combinato, sono riusciti comunque a starmi vicino e a supportarmi. Mia sorella Maria Giorgia perché è grazie alla sua forza che siamo riusciti, insieme, a superare tante sfide in questi anni e che è sempre pronta a supportarmi anche quando non ci sentiamo per tanto tempo.

Ringrazio mia nonna Lina e mio nonno Lello, mio cugino Dario, i miei zii Aurelio, Gabriella, Bianca, Sergio, Salvino e Lia per tutti i consigli e l'incoraggiamento, da sempre.

A pari merito (quasi) con mia mia famiglia la mia seconda famiglia, i miei amici Paolo, Andrea, Chiara, Marta, Roto, Gianluca, Diego, Bianca, Alessio, Simone, Fulvio, Fabrizio, Fiamma, Sabrina, Gabriele, Giuseppe, Edoardo, per cui mi sento infinitamente fortunato e che ringrazio di cuore di essermi stati vicino e avermi supportato e sopportato in questi anni. Non so se sarei qua ora se non fosse stato per loro.

Ringrazio tutti i miei compagni di università a Palermo, in particolare Riccardo, Federico, Giorgio e Giuseppe.

Devo ringraziare anche Milano che mi ha permesso di conoscere tantissime persone stupende che mi hanno supportato, aiutato, guidato e insegnato tanto, Francesca, Clara, Beatrice, Francesco, Massimiliano, Lorenzo, Chiara, Matteo, Emilio e tutti i compagni dell'Agorà.

Non ho citato Luca perché è anche grazie a lui che tutto questo è stato possibile, grazie per tutti i momenti di studio pazzi, per esserci stato sempre e per tutte le scorreggiate di cervello.

Ma, sopratutto, sono grato alla città che tanto mi ha fatto soffrire per avermi fatto conoscere Sara la persona più bella che io abbia mai conosciuto, che ringrazio infinitamente per tutto quello che ha fatto per me e per tutto l'amore che mi ha dato in questo bellissimo anno.

Grazie.