Technische Universiteit Delft
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

Master thesis - Applied Mathematics

Transfer Learning Framework for Battery Lifetime Prediction Using Early Cycle Data: Addressing the Challenge of Limited Training Data Diversity

Author
Femke Schürmann

Responsible Supervisor
Dr. P. Chen

Other comittee members
Prof. Dr. Ir. G. Jongbloed
Dr. D. Kurowicka

March 2023

TU Delft

Acknowledgements

I would like to express my gratitude in this thesis to Dr. Piao Chen, my thesis supervisor, for providing guidance throughout the research process. Additionally, I would like to extend my appreciation to Msc. Wenda Kang for the daily support provided during my thesis work. Finally, I would like to acknowledge the valuable feedback received from Prof. Dr. Ir. Geurt Jongbloed and Dr. Dorota Kurowicka, which greatly assisted me in completing the research.

Abstract

The integration of large-scale battery storage systems can aid the transition to renewable energy and stabilize energy systems for optimization. However, batteries can be cost-prohibitive and unprofitable, highlighting the need for a more comprehensive understanding and modelling of battery degradation. Battery degradation prediction models play a crucial role in battery manufacturing, especially when they can be created using early cycle data. However, a challenge in battery prediction is the lack of diversity for training data, leading to models that are not robust and hard to generalize. Transfer learning can address this issue, as it doesn't require the test and training data to come from similar distributions. This thesis introduces a framework that uses early cycle data to predict battery lifetimes. The framework employs a regularized method, the elastic net, for battery lifetime prediction and Transfer Component Analysis (TCA) for transfer. The major contribution of this thesis is the transfer learning framework that is proposed for battery lifetime prediction, which involves the analysis of various aspects, including when what, and how to transfer. To demonstrate the framework's performance, a dataset based on early cycle data from a real-world case study is used. The results show that the proposed methods outperform existing methods in both the simulation and case study results. Different methods are used for selecting and weighing features before the transfer, resulting in 39 out of 42 improvements in the case study results. In particular, utilizing elastic net coefficients to weigh the features before the transfer yields the optimal approach in 15 out of 21 cases and enhances the RMSE and MAPE compared to not using transfer in 38 out of 42 cases. Additionally, this study, as one of the first studies in this field, provided innovative approaches to quantitatively examine negative transfer. It conducts a comprehensive analysis mainly for univariate distributions, utilizing a robust 2-sample goodness-of-fit test to gain a deeper understanding of the relationship between transfer performance and distributional differences.

Keywords: Battery degradation, remaining useful lifetime, Transfer learning, Transfer Component Analysis, 2-sample goodness-of-fit tests, variable selection

Contents

1 | Introduction

1.1 Background

The Netherlands is striving to decrease CO2 emissions and boost the usage of clean energy sources to meet its climate objectives [1]. However, the electricity grid is facing growing inadequacy in many areas and does not match the fluctuation of electricity supply and demand [2]. Large-scale battery storages can help with the acceleration of the energy transition, as they can be combined with renewable energy resources and therefore stabilize energy systems and make them more beneficial [3]. Additionally, smaller batteries can also reduce CO2 emissions in the transportation sector.

Lithium-Ion Batteries, a common type of battery, face issues with being costly and uneconomical [4]. The limited resources for batteries need to be utilized efficiently. To address these challenges, it's essential to enhance energy density and extend the battery's lifespan [5]. Understanding and modelling battery degradation is a crucial step to overcoming these challenges.

1.2 Related work

The ageing of battery cells is nonlinear and depends on their physical properties [6, 7]. It is caused by time (calendar ageing) and use (cycle ageing) [8, 9]. For modelling the degradation of batteries, there are generally 3 different approaches: physics-based, semi-empirical and data-driven [10, 11, 12, 13].

Physics-based models [14, 15, 16] simulate the chemical and physical interactions within batteries, and can provide precise predictions with a comprehensive understanding of battery chemistry and physics. However, they are intricate and depend on numerous factors, making them challenging to apply on a large scale [17, 18]. This makes sense as the theoretical model is often not in line with relevant operational conditions [19, 20].

Additionally, semi-empirical models [21, 22] in battery ageing analysis are mathematical models that aim to predict the behaviour and performance of batteries over time by incorporating both physical and empirical components. These models are based on a combination of experimental data and a theoretical understanding of battery physics and chemistry. The empirical part of the model is derived from actual data obtained through experiments and the physical part is based on the underlying physics and chemistry of battery behaviour. Semi-empirical models are widely used in battery research [23]. If the data is of high quality and accurately reflects real-world conditions, the model is more likely to produce accurate results. On the other hand, if the data is of poor quality, the model will likely produce inaccurate results. The operating conditions of the data used to generate the model can also significantly impact the performance of the model, as different operating conditions may affect the relationships between the factors in question. Therefore, it's crucial to carefully consider the quality of the data and the operating conditions when using (semi-)empirical models to estimate battery lifetimes.

Finally, data-driven models [24, 25, 26, 27], such as machine learning models, are commonly used for battery degradation modelling due to their ability to achieve high accuracy without a deep understanding of the underlying physical and chemical processes. These models learn from data and can identify patterns and relationships that may not be immediately apparent. However, the accuracy and robustness of these models are highly dependent on having a sufficient amount of diverse data. If the data used to train the model is limited or does not represent a wide range of conditions, the model may not generalize well to new data and produce inaccurate results. Therefore, it's crucial to have access to a large and diverse dataset in order to improve the quality and reliability of data-driven models for battery degradation modelling.

The main difficulty in using machine learning for battery lifetime prediction is obtaining sufficient relevant training data [28, 29]. Large amounts of diverse experimental data [29] are required to train and validate these models, and the accuracy of the model depends on the data quality. This is due to various factors such as battery type, operating conditions and manufacturing variations that can impact battery performance, making accurate prediction difficult. However, conducting experiments to determine battery lifespan can take several months or years to complete, limiting the progress of researchers.

Furthermore, many studies where machine learning models are used do not focus on the early cycle data as they require a certain percentage of degradation which does generally not occur in early cycle data [30, 31, 32, 33, 34]. Accurate predictions from early cycle data can revolutionize the battery industry by providing manufacturers with early feedback on the quality of their products as only a few test cycles are needed [35, 36]. This information can then be used to optimize the production processes and improve the cells, leading to longer cycle life and better performance. This enables significant benefits in terms of cost, performance, and environmental impact for both manufacturers and consumers.

In a case study introduced in 2019 by Severson et. al. [37], a dataset of 124 commercial lithium-ion phosphate/graphite cells was generated and cycled under fast-charging conditions. Derivatives of the discharge voltage curves from the early cycles of a battery, prior to degradation patterns appearing, are particularly interesting as they are a rich source for forecasting cycle lives [38, 39, 40]. In their approach, the authors apply linear and non-linear transformations to the raw data in order to extract relevant features for the battery degradation prediction. These features are then used within a regularized framework, the elastic net [41], to make predictions about the battery's lifetime. By using a regularized linear model, the authors are able to retain the interpretability of their method while still allowing for flexible complexity in the features. This means that the model can be adjusted to include a wide range of features and transformations, providing more accurate and robust predictions about battery degradation. The use of the first 100 cycles in their model gives promising results, but the generalization of battery lifetime prediction models remains a well-known challenge as there is a lack of diverse training data. Transfer learning can overcome this as it does not require training and test data to come from similar distributions.

Transfer learning (TL) can be a solution for the lack of diverse data in battery lifetime prediction as it enables the transfer of knowledge from a source problem to a target problem with similar characteristics but different underlying distributions [42]. For battery ageing studies, the source problem could be a related battery lifetime prediction task with a large and diverse dataset, while the target problem is the battery lifetime prediction task with limited data. Transfer learning can

leverage the knowledge learned from the source data to improve the performance of the model on the target task, even with limited data. Thus, transfer learning can provide a solution for battery lifetime prediction by leveraging knowledge from related problems. Pan and Yang's survey [43], which is considered a pioneering work, classifies transfer learning and evaluates the research advancement before 2010.

1.3 Problem statement

In this thesis, the aim is to set up an interpretable framework for battery lifetime prediction using early cycle data and transfer learning. In transfer learning, it is common to not only focus on *how* the knowledge is transferred from source to target data but also *when* a dataset is suitable to apply transfer learning to and *what* part of the data should be transferred. Especially, the verification of the performance of a transfer learning model when there is a lack of labelled data in the target domain is challenging [44, 45]. This study will propose and assess the performance of a feature-based transfer learning framework through a simulation study and apply it to the case study dataset [37] to evaluate its effectiveness in the battery lifetime prediction task.

The aforementioned aim of this thesis is realized through the following structure. In Chapter 2, the transfer learning framework is described and relevant examples are given. This framework is then applied to a simulation study in Chapter 3 and a case study in Chapter 4. Finally, based on this performance assessment, a conclusion is drawn in Chapter 5. Furthermore, a discussion is provided in Chapter 6 and potential future work will be discussed.

The Jupyter Notebook was used to write the codes for this thesis, and they are available in a Github repository[1].

[1] github.com/FemkeSchurmann/batteryrulthesis

2 | Transfer learning framework and methods

This chapter outlines the transfer learning framework. Prior to delving into this framework, Section 2.1 presents some background information and mathematical definitions on transfer learning. The framework employed in this thesis revolves around three key questions: *when* to transfer, *what* to transfer, and *how* to transfer. A detailed account of these questions and the framework is provided in Section 2.2.

To address the issue of *when* transfer can be employed, this study introduces two tests to quantify distributional differences: the Kolmogorov-Smirnov test [46] and its improved version [47]. Furthermore, these tests can offer valuable insights into the dependence of the dissimilarity between the source and target data and the transfer framework's performance. These tests will be described in Section 2.3.

Section 2.4 introduces the elastic net model, describing the concept of variable selection and how it can assist in addressing the question of *what* to transfer. Additionally, it illustrates how the elastic net model is utilized to predict data labels.

Subsequently, in Section 2.5, *Transfer Component Analysis* is presented as the chosen transfer method. This section elaborates on the question of *how* to transfer.

Finally, two metrics for measuring performance are introduced in Section 2.6, specifically the Rooted Mean Squared Error (RMSE) and Mean Absolute Percentage Error (MAPE).

2.1 Background and definitions of transfer learning

Before diving into the final framework used for generating the results, some definitions are introduced in order to understand what kind of methods are needed. For these definitions, ideas are used from Zhuang et al. [42].

Definition 2.1.1 (Domain)**.** *A domain $\mathcal{D}$ consists out of a feature space $\mathcal{X}$ and a marginal distribution $P(X)$ where X is defined as an instance set containing n instances, i.e. $X = \{x|x_i \in \mathcal{X}, i = 1, ..., N\}$. Then $\mathcal{D} = \{\mathcal{X}, P(X)\}$.*

Battery data can provide a variety of features, which can be utilized as input variables for making predictions. Examples of such features include discharge capacity or average charge time. It is important to note that if the same feature set is used for both the source and target data, the feature space of the source and target data is the same, but the marginal distributions $P(X)$ may differ across these distributions.

Definition 2.1.2 (Task)**.** *A task $\mathcal{T}$ consists of a label space $\mathcal{Y}$ and an implicit decision function f, i.e. $\mathcal{T} = \{\mathcal{Y}, f\}$.*

In battery lifetime prediction, the label space $\mathcal{Y}$ represents the output space, and the goal is to predict the lifetime of each battery i. For labelled instances, the lifetime is already known. Alternatively, another task could be to determine the remaining useful lifetime (RUL) of a battery. It's important to note that labels can also represent classes such as 'high lifetime' or 'low lifetime,' transforming the task into a classification problem instead of a regression problem.

In the context of machine learning, the objective is typically to identify the optimal decision function (or model) f^* from a set of candidate models in the hypothesis space $f \in \mathcal{H}$. The optimality of a model is determined by its ability to minimize the specified loss function $L(x, y, f)$.

Definition 2.1.3 (Transfer learning)**.** *Define source domain and learning task $\{(\mathcal{D}_S), \mathcal{T}_S\}$ containing one or more observations. Similarly define a target domain and learning task $\{(\mathcal{D}_T), \mathcal{T}_S\}$. Transfer learning aims to improve the learning of the decision function f^T for the target domain, using knowledge in the target domain and learning task.*

A frequent scenario in transfer learning involves having an adequate number of labelled instances or a well-trained model in the source domain, but a limited quantity or no labelled instances in the target domain. In such cases, the objective of transfer learning is to enhance the decision function for the target domain. In general, it is assumed that the source and target data may not come from the same distributions, which necessitates the use of transfer learning.

The algorithm used in this thesis will be a feature-based algorithm (see appendix Section 7.1) namely Transfer Component Analysis (TCA) [48]. Battery lifetime prediction models need to be robust and able to generalize to new batteries and conditions. When using instance-based transfer learning for this, the varying instances can be vastly different, resulting in overfitting and biased results. In contrast, feature-based transfer learning can help to identify the relevant features that are useful for battery lifetime prediction, even when there is variability in the data. It's worth mentioning that TCA solely employs feature data for transfer and excludes label information, implying that labels aren't necessary for either the source or target domain for utilizing TCA. Yet, the elastic net model requires label information during training. Furthermore, feature-based methods are suitable for regression which is the task in this thesis. The reasons outlined above have driven the motivation for utilizing a feature-based transfer technique. Details on TCA are described in Section 2.5.

2.2 Designing a transfer learning algorithm

In Figure 2.1, the framework for the transfer used in this thesis is proposed. When designing a transfer learning algorithm, three main issues should be considered, namely when to transfer, what to transfer and how to transfer. Shortly, this can be summarized as [49]:

- *'When to transfer'* questions in which situations the transfer will lead to improved or worse prediction results. In some situations, the transfer can be unsuccessful and maybe even hurt

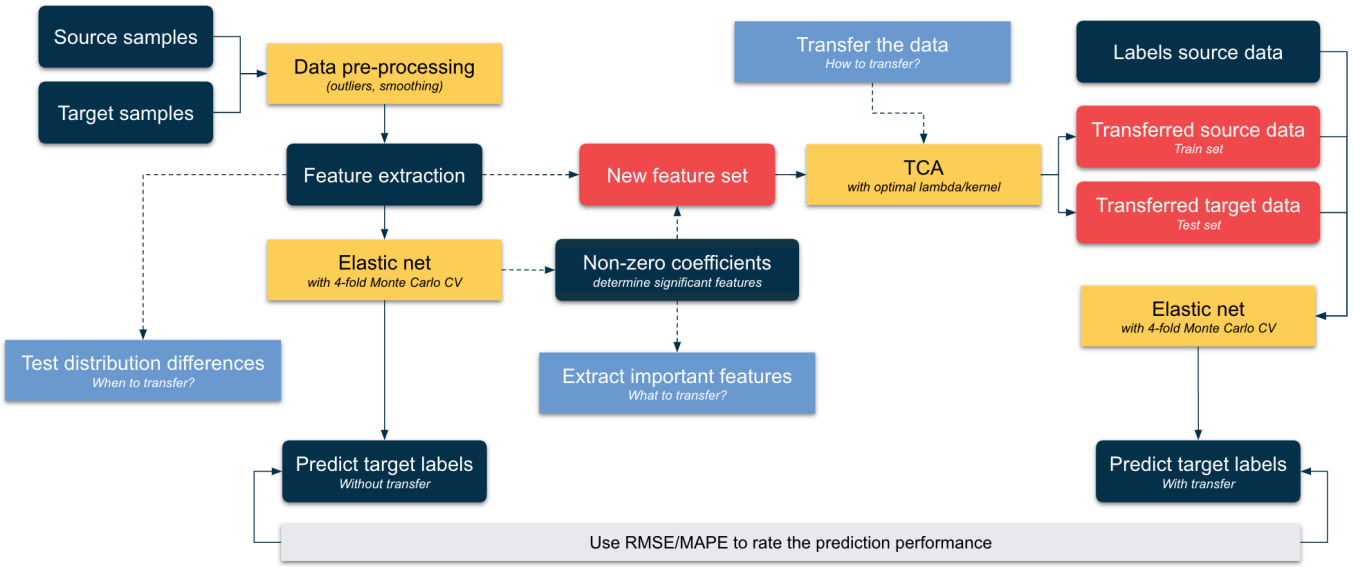


Figure 2.1: Framework for setting up the transfer and predicting the labels

the performance of the learning. This is often referred to as a *negative transfer*. Furthermore, when there is no labelled data available in the target domain, it is challenging to verify the performance of the transfer learning model [44]. The majority of transfer learning literature emphasizes what and how to transfer, while the evaluation of transfer performance remains an unresolved issue. Defining *negative transfer* and developing a reliable method for avoiding is furthermore a challenging task in itself [45].

- *What to transfer* refers to deciding which specific information or data should be included in the transfer process. It is important to think about what to transfer because not all knowledge learned in the source task may be useful or applicable to the target task. Therefore, it is important to think about what part of the data is useful for the transfer.

- *How to transfer* specifies what type of transfer learning method is suitable and which hyper-parameters should be chosen in the model.

The framework mentioned refers to a structure used to analyze the transfer learning procedure from the beginning to the end. The 3 questions stated above, which play a crucial role within this framework, are key components that need to be considered in order to fully understand the problem and come up with an appropriate solution. However, answering these questions is not straightforward and requires critical thinking. The rest of the chapter is dedicated to presenting various methods that can aid in providing answers to these crucial questions.

In Section 2.3, a distance metric called Maximum Mean Discrepancy (MMD), the Kolmogorov-Smirnov test and an improvement of this test are introduced. These metrics and tests will be used, analyzed and compared for determining *when to transfer*. Next, in Section 2.4, the elastic net model is proposed as a prediction method and it is explained how this model can be used for selecting the appropriate features before transfer. Therefore, this section contributes to *what to transfer*. Subsequently, in Section 2.5, Transfer Component Analysis is explained. This section addresses the

question of *how to transfer*. For the evaluation of the prediction results, the RMSE and MAPE will be used as prediction metrics and are explained in Section 2.6.

2.3 Distribution differences

This section is divided into two parts. In Subsection 2.3.1, the Maximum Mean Discrepancy (MMD) metric is explained, which is a metric for quantifying distribution differences. This metric plays a key role in many transfer learning methods, such as Transfer Component Analysis (Section 2.5). The second part of this section, Subsection 2.3.2, will focus on 2 tests, namely the Kolmogorov-Smirnov test and another, more powerful test, which will be called 'the Zk-test'. These tests give accurate insight into the similarity distributions, which is important to take into account when applying a transfer [48]. Later on, these tests can help identify the relationship between the similarity of the source and target distribution and the performance of the transfer. Therefore, the main goal of using these tests is to answer the question *when to transfer?*.

2.3.1 Maximum Mean Discrepancy

A widely used non-parametric metric in transfer learning for calculating the difference between 2 distributions is called Maximum Mean Discrepancy (MMD) [49]. Suppose $X^S \sim P$, $X^T \sim Q$, and define n^S and n^T as the sample sizes of X^S and X^T respectively. Finally, let Φ map each instance to the Hilbert space $\mathcal{H}$ associated with the kernel $k : \chi \times \chi \to \mathbb{R}$ where $k(\boldsymbol{x}_i^S, \boldsymbol{x}_j^T) = \Phi(\boldsymbol{x}_i^S)^\top \Phi(\boldsymbol{x}_j^T)$. Here, χ is the feature space of the source and target domain.

The MMD distance is calculated as follows:

$$\text{MMD}(X^S, X^T) = ||\frac{1}{n^S}\sum_{i=1}^{n^S}\Phi(\boldsymbol{x}_i^S) - \frac{1}{n^T}\sum_{j=1}^{n^T}\Phi(\boldsymbol{x}_j^T)||_{\mathcal{H}}^2. \tag{2.1}$$

The MMD can also be defined as the distance between two different projections of the means. A kernel trick can be used to calculate it. This leads to Equation 2.2 in order to calculate the MMD distance:

$$\text{MMD}(X^S, X^T) = ||\frac{1}{n^S}\sum_{i=1}^{n^S}\Phi(\boldsymbol{x}_i^S) - \frac{1}{n^T}\sum_{j=1}^{n^T}\Phi(\boldsymbol{x}_j^T)||_{\mathcal{H}}^2 = Tr(KL) \tag{2.2}$$

where:

$$K = \begin{bmatrix} K_{X^S,X^S} & K_{X^S,X^T} \\ K_{X^T,X^S} & K_{X^T,X^T} \end{bmatrix} \in \mathbb{R}^{(n^S+n^T)\times(n^S+n^T)} \tag{2.3}$$

so:

$$K_{ij} = \begin{cases} k(\boldsymbol{x}_i^S, \boldsymbol{x}_j^S) & i,j \leq n^S \\ k(\boldsymbol{x}_i^T, \boldsymbol{x}_j^T) & i,j > n^S \\ k(\boldsymbol{x}_i^S, \boldsymbol{x}_j^T) & \text{other.} \end{cases}$$

It is important to note that the MMD is a squared value, but it is often taken the square root to make it comparable to other distance measures. This is true since kernel matrices are symmetric

[50]. Note here that $\boldsymbol{x}_i^S \in \mathbb{R}^p$ is a sample vector from the source data. Similarly, $\boldsymbol{x}_i^T \in \mathbb{R}^p$ is a sample vector from the target data. Furthermore, $L \succeq 0$ with

$$L_{ij} = \begin{cases} \frac{1}{n^{S^2}} & x_i^S, x_j^S \in X^S \\ \frac{1}{n^{T^2}} & x_i^T, x_j^T \in X^T \\ -\frac{1}{n^S n^T} & \text{other.} \end{cases} \tag{2.4}$$

In this thesis, the following 4 kernels are used. Define $\boldsymbol{x} \in \mathbb{R}^p$ a sample vector of $X \sim P$ and $\boldsymbol{y} \in \mathbb{R}^p$ a sample vector of $Y \sim Q$. The different definitions of the kernels are with according parameters are defined as follows:

- The linear kernel: $k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^\top \boldsymbol{y}$
- The Polynomial kernel $k(\boldsymbol{x}, \boldsymbol{y}) = (\gamma \boldsymbol{x}^\top \boldsymbol{y} + c)^d$ where $d = 5$ (this is the same degree used for the polynomial fits on the case study data in Chapter 4), $\gamma = 1/n$ and $c = 1$.
- The (Radial Basis Function) RBF kernel: $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma ||\boldsymbol{x} - \boldsymbol{y}||^2)$ where $\gamma = 1/n$, also known as Gaussian kernel when $\gamma = \frac{1}{\sigma^2}$.
- The Laplacian Kernel is a variant on the RBF kernel: $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma ||\boldsymbol{x} - \boldsymbol{y}||_1)$ where $\gamma = 1/n$

For comparing different samples and their distributions, it is useful to scale the data before applying MMD. Scaling the data helps to ensure that the comparison between the distributions is not influenced by differences in the scale or magnitude of the features. This makes the MMD result more interpretable and meaningful. For this, a MinMaxScaler is used which is defined as

$$x_{\text{scaled}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}.$$

Here, $x_{\min}$ and $x_{\max}$ are the feature minimum and maximum over the entire sample. In Python, there is a function which can easily be applied for scaling data [51].

To compare the MMD with the other 2 tests, described in Subsection 2.3.2, a simulation example is provided. Suppose $X1$ and $X2$ are 2 samples with each 200 instances from a univariate normal distribution N(2,1) and $X3$ is a sample of 200 instances from a Gamma(1,1) distribution. Clearly, $X1$ and $X2$ are from the same distribution while $X3$ is from a different distribution. These samples are visualized in Figure 2.2 and will also be used for the tests introduced later on. Since 2 of these samples are from the same distribution, while the third sample is from a different distribution, it can be expected that the MMD between $X1$ and $X2$ is smaller than $X1$ and $X3$. However, with the kernels as defined above, only the polynomial kernel gives a higher MMD for $X1$ and $X3$ compared to the MMD of $X1$ and $X2$. All other kernels give the same MMD for both cases. This is a simple example with only a few samples and no fine-tuning of hyperparameters in the kernels. Still, it shows that the MMD is not extremely sensitive to distribution differences, but rather a metric that is able to identify differences between distributions computationally cheap and easy.

In the example given above, only the polynomial kernel was able to identify a smaller distribution discrepancy for the same distribution. This shows the importance of choosing the right kernel

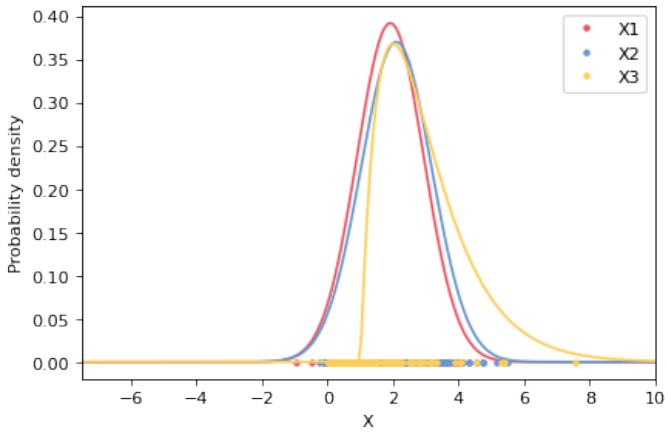


Figure 2.2: Three simulated data samples from the N(2,1) ($X1$ and $X2$) and Gamma(1,1) ($X3$) distributions

that is best at identifying underlying patterns in the data such that the distribution differences can be recognized. Choosing the most suitable kernel is a complex task. The original article of TCA [48] suggests that this kernel can be determined using methods such as cross-validation, but stresses the complexity of this task. In methods such as Maximum Mean Discrepancy Embedding (MMDE), described in appendix Equation 7.8, an implicit kernel function is used and found via linear programming. This can however be computationally expensive and therefore still not a suitable method for many applications. Finding the appropriate kernel with corresponding hyperparameters is a huge topic and therefore outside the scope of this research.

Other interesting distance metrics are Kullback-Leibler divergence [52], Jensen-Shannon Divergence [53], Bregman Divergence [54] and Hilbert-Schmidt independence Criterion [55]. The advantage of the MMD is that it is kernel-based, which means it can handle non-linear relationships and high dimensional data. Furthermore, it is computationally efficient, which is especially convenient for the development of feature-based transfer methods and suitable for measuring discrepancies between joint distributions. One limitation of MMD is that it only measures the difference in means between distributions and not other characteristics such as scale and shape. The example above shows that the MMD is not very sensitive to distribution differences. Still, while other metrics are better at detecting these differences, they can be difficult to compute and implement in transfer methods. Therefore, the MMD remains a useful metric for quantifying distribution discrepancies.

2.3.2 Tests for comparing marginal distributions

Since many transfer learning methods, specifically feature-based transfer methods [48], assume that the marginals of the source and target data are not necessarily from the same distribution, it is desired that this is analyzed before applying transfer learning. This could also help in the understanding of when a transfer can be applied. Since the MMD value is a metric that only uses feature means for estimation, it is hard to set up a test for this metric in order to test the distribution differences and it is also not a very powerful metric as it does not take into account other distribution characteristics such as shape and scale. Therefore, it is relevant to look at other tests that can tell whether 2

marginals come from the same distribution.

Before diving into possible tests, it is important to define what needs to be tested. Therefore, let $x_{i1}, x_{i2}, ..., x_{in_i}$ be a random sample corresponding to the continuous distribution function $F_i(x)$, $(i = 1, 2)$. The null hypothesis is that sample 1 and sample 2 come from the same marginal distribution. Therefore, we wish to test:

$$H_0 : \quad F_1(x) = F_2(x), \quad \forall x \in (-\infty, \infty) \tag{2.5}$$

against

$$H_1 : \quad F_1(x) \neq F_2(x), \quad \forall x \in (-\infty, \infty). \tag{2.6}$$

Since only sampled data is available, the exact corresponding distribution is hard to determine. Therefore, define

$$\hat{F}_i(t) = \frac{1}{n_i} \sum_{j=1}^{n_i} \mathbf{1}_{x_{ij} \leq t} \tag{2.7}$$

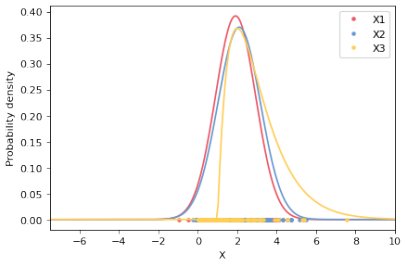
as the empirical distribution corresponding to sample i where $t \in (-\infty, \infty)$. Testing H_0 against H_1 is then equivalent to respectively $\tilde{H}_0 : \hat{F}_1(t) = \hat{F}_2(t)$ and $\tilde{H}_1 : \hat{F}_1(t) \neq \hat{F}_2(t)$ for all $t \in (-\infty, \infty)$. Note that this is true since $H_0 = \bigcap_{t \in (-\infty,\infty)} \{\tilde{H}_0\}_t$ and $H_1 = \bigcup_{t \in (-\infty,\infty)} \{\tilde{H}_1\}_t$. Indeed, when $\tilde{H}_0$ is true for all $t \in (-\infty, \infty)$, then H_0 is true (or the other way round). Similarly, when there exists a t for which $\tilde{H}_1$ is true, then H_1 is also true.

A well-known test is the *Kolmogorov-Smirnov test* introduced by [46] and [56]. The two-sample version of this test tests the null hypothesis which assumes that the 2 samples are from the same distribution. The test statistic is then defined by:

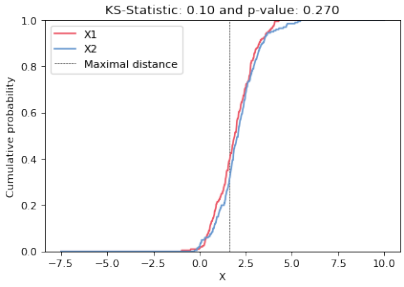
$$D_{n_1,n_2} = \sup_t |\hat{F}_{n_1}(t) - \hat{F}_{n_2}(t)| \tag{2.8}$$

where $\hat{F}_{n_1}$ and $\hat{F}_{n_2}$ are the empirical distribution functions as defined in Equation 2.7. Then, when test statistic D_{n_1,n_2} is large, the null hypothesis is rejected.

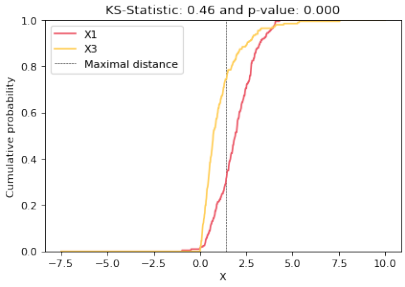
To visualize the Kolmogorov-Smirnov test statistic, a small example is given in Figure 2.3. Three samples are again generated with each containing 200 points. The first two samples $X1$ and $X2$ are from the same distribution, namely N(2,1), while the third sample $X3$ is generated from a Gamma(1,1) distribution. This means that the test applied to the first two samples should accept the null hypothesis, while the test between the first and third samples should reject the null hypothesis. In order to test this, a significance level of 0.05 is set. The p-value measures the probability of having the observed results, assuming the null hypothesis is true, so when the p-value is lower than 0.05, the null hypothesis is rejected. In Figure 2.3b, the empirical cumulative distribution functions are plotted for both samples and the KS-statistic and p-value are given. The dashed lines show at which value for X, the distance between the two functions is maximal (in other words, the KS statistic). In Figure 2.3b, the empirical cumulative distribution functions are shown for the first two samples. Since these two samples are from the same distribution, the test should accept the null hypothesis. As the p-value is 0.27, which is larger than 0.05, the null hypothesis is correctly accepted. Similarly,



(a) Three simulated data samples from the N(2,1) (twice) and Gamma(1,1) distributions



(b) The empirical cumulative distribution functions for $X1$ and $X2$, with the dashed line indicating the maximal distance between the two functions



(c) The empirical cumulative distribution functions for $X1$ and $X3$, with the dashed line indicating the maximal distance between the two functions

Figure 2.3: Example of the Kolmogorov-Smirnov statistic in practice

the test for samples 1 and 3 in Figure 2.3c should reject the null hypothesis, which is also in line with the p-value.

Tests that have been developed more recently are able to detect differences between distributions more effectively, including variations in location, scale, and shape. In contrast, traditional two-sample tests are only able to detect differences in location [47].

In this thesis, the focus will be on a test originally introduced by [57] as an improvement of the KS-test. In this thesis, this test is referred to as the 'Zk-test'. Define $x_1, ..., x_n$ as the pooled sample where $n = n_1 + n_2$ and let $X_{(1)}, ..., X_{(n)}$ be the ordered sample. The Zk-statistic is then defined as:

$$Zk = \max_{1 \leq k \leq n} [\sup_{x_{(k)} \leq t \leq x_{(k+1)}} G_t^2].$$

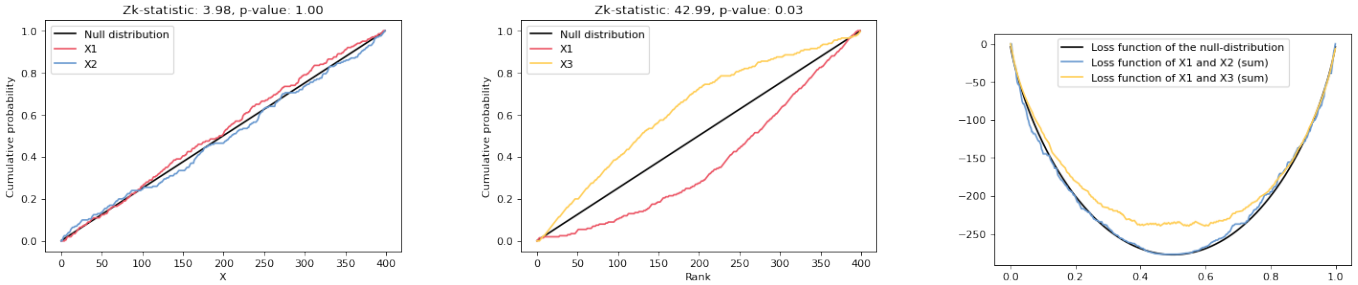
This is equivalent to

$$Zk = \max_{1 \leq k \leq n} [\sum_{i=1}^{2} n_i (F_{ik} \log \frac{F_{ik}}{F_k} + (1 - F_{ik}) \log \frac{1 - F_{ik}}{1 - F_k})] \tag{2.9}$$

Where modifications on $\hat{F}(t)$ are made at its discontinuous points $X_{(k)}$. In Equation 2.9, $F_k = \hat{X}_{(k)}$ is the empirical distribution of the ordered pooled sample and $F_{ik} = \hat{F}_i(X_{(k)})$ is the empirical distribution corresponding to sample i. These are defined as follows.

Define R_{ij} to be the rank of the j-th ordered statistic in the i-th sample. This means for each sample, a set of ranked data can be generated which contains the ranks in the pooled sample of the data points in that sample. Note that in $X_{(k)}$, k corresponds to the rank in the pooled sample. Define $F_k = \frac{k-.5}{n}$ so that F_k ranks $X_{(k)}$ relatively as $k \in (1, ..., n)$ and whenever $k = R_{ij}$ for some j, so the $X_{(k)}$ is in the i-th sample, $F_{ij} = \frac{j-.5}{n_i}$. Here, F_{ij} ranks $X_{(k)} = X_{(j)}$ relatively in sample i as $j \in (1, ..., n_i)$. However, if $R_{ij} < k < R_{ij+1}$, where $R_{i0} := 1$ and $R_{in_i+1} := n + 1$, then let $F_{ik} = \frac{j}{n_i}$.

The idea behind this test statistic is that when two samples come from the same distribution, combining them and assigning ranks to each element in the pooled sample should result in a uniform



(a) Ranks compared between $X1$ and $X2$

(b) Ranks compared between $X1$ and $X3$

(c) Log-loss function for the null distribution, and the sum of the log-los function for 2 samples ($X1$ and $X2$, $X1$ and $X3$)

Figure 2.4: Example of the Zk-statistic in practice

distribution of ranks between 1 and n. This is considered the *null* distribution. However, if one sample has a larger proportion of small values compared to the other sample, the lower ranks in the pooled sample will mostly come from the sample with more small values, while the higher ranks will come from the sample with fewer small values. In order to quantify this difference, the log-loss function is used:

$$L = R_i \log(P(R_i)) + (n - R_i) \log(1 - P(R_i)).$$

Here, R_i is the rank of each observation and $P(R_i)$ is the cumulative probability of being smaller than that rank. Compared to the uniform case, when a sample contains many small values, $\log(1 - P(R_i))$ will be more dominant while for a sample with many small values, $\log(P(R_i))$ will be more dominant.

In Figure 2.4, the ranks of the samples are compared and the loss function is visualized. The Zk statistic equals the maximum difference between the log-loss function of the null distribution and the sum of the log-loss of the samples. The p-value is calculated by simulating many 'uniform' distributions between 1 and n, and calculating their test statistics. From these statistics, a distribution can be built and the p-value can be calculated. In Figure 2.4b, there is much difference in ranks which indicates a bigger difference between the distributions. This also follows from the corresponding p-value, which correctly indicates the rejection of the null hypothesis. For this example, only 100 simulations are done in order to calculate the distribution of Zk. Adding more simulations will lead to a more accurate p-value, however, if the Zk-test is only based on 100 samples, the sampling for the p-value should be in line with this with corresponding accuracy. It is notable to mention the Zk-statistic has a maximum which is reached whenever the 2 samples are non-overlapping. This maximum value is related to the number of samples, of which an example where the number of source and target samples are equal, is visualized in appendix Figure 7.6.

2.3.3 Comparison between the power of the Kolmogorov-Smirnov test and Zk-test

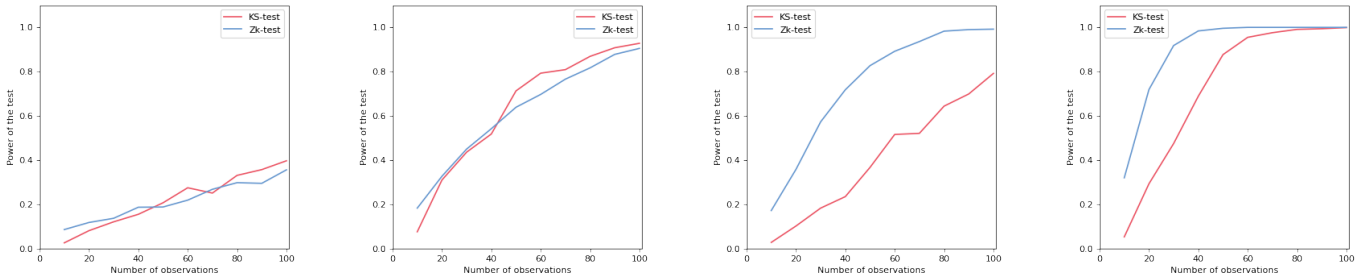
Since the Zk-test uses a loss function based on the ranks of both distributions, it is much more sensitive to distribution differences compared to the Kolmogorov-Smirnov test, which only looks at

the maximum difference between both Empirical Cumulative Distribution functions. The original article which introduces the Zk-test [47] provides simulation examples which show the power of their tests versus many tests. In this thesis, a few of these examples are shown in order to give more insight into this.

First of all, the power of a statistical test is the probability that the test correctly rejects the null hypothesis when the alternative hypothesis is actually true. In other words, it is the ability of a statistical test to detect a true effect or difference when one actually exists. Generally, the power of a test is influenced by several factors, including the sample size, the level of significance chosen for the test, the effect size (i.e., the magnitude of the difference or relationship between the variables of interest), and the variability of the data. Increasing the sample size or the effect size generally increases the power of the test, whereas increasing the level of significance or the variability of the data decreases the power of the test.

In this experiment, which is similar to the experiment done in the original paper, the significance value is set to $\alpha = 0.05$. This is also the significance level that is used throughout this thesis. The observations will be varied from 10 to 100 with stepsize 10, for both the source and target data (so in total 20-200), where the source and target data will always have the same sample size. Furthermore, the source distribution is always sampled from a standard univariate normal distribution and the target distribution will be drawn from a $N(\mu, \sigma^2)$ distribution, where μ and σ^2 will be varied. Note that therefore, the only changes will be in location and scale and there is no difference in shape or skewness. The target distribution will always be different from the source distribution, so it is desired that the null hypothesis is rejected by the test. A power of 0.6 means that 600/1000 simulations correctly rejected the null hypothesis.

In Figure 2.5, 4 different scenarios are shown. In the first 2 scenarios, the target distribution has a different mean whereas in the first case, the difference is smaller compared to the second case. Likewise, in the other 2 scenarios, the target distribution has a different variance, where also the magnitude of difference in the first case is smaller compared to the second case.



(a) N(0,1) vs. N(0.3,1) (b) N(0,1) vs. N(0.6,1) (c) N(0,1) vs. N(0,2) (d) N(0,1) vs. N(0,3)

Figure 2.5: Power of a test plotted against the number of observations for 4 different scenarios

From these figures, it can be seen that the tests are comparable for differences in mean, but for differences in variance, the Zk-test is much stronger. Note, in particular for distributions which are slightly different, that the power around 40 observations is not very high. In the case study data, this number of observations is used for predictions. Therefore, it should be taken into account that even though the Zk-test is more powerful compared to the KS-test, it can still lead to the wrong conclusion.

2.4 Elastic Net and feature selection

The method used by Severson et al. [37] is a regularization model which combines the lasso [58] and ridge [59] regression methods, called Elastic net [41]. Lasso regression penalizes features in order to prevent over-fitting. It is based on the linear model:

$$\hat{\mathbf{y}}_i = \mathbf{x_i}\hat{\beta}$$

where $\mathbf{x_i}$ is a p-dimensional sample vector and $\hat{\beta}_i$ is a p-dimensional coefficient vector. In the classic linear model, the least-squares optimization technique is used to find the coefficients for $\hat{\beta}_i$. For regularization techniques, a penalty $\alpha P(\beta)$ is added:

$$\hat{\beta}_i = \text{argmin}_\beta ||\mathbf{y} - \mathbf{X}\beta||_2^2 + \alpha P(\beta)$$

In the elastic net, this penalty $P(\beta)$ is defined as:

$$P(\beta) = \frac{1-\lambda_1}{2}||\beta||_2^2 + \lambda_1|\beta| \tag{2.10}$$

The λ_1 term is called the 'l1-ratio', also known as the mixing parameter. It determines the balance between the L1 and L2 regularization terms. A value of 0 corresponds to L2 regularization (Ridge), and a value of 1 corresponds to L1 regularization (Lasso).

The regularization strength (α) controls the overall strength of the regularization term. A larger value of α corresponds to stronger regularization, and a smaller value corresponds to weaker regularization.

Overall, the Elastic Net model is a good choice when there are multiple correlated features in the dataset and you want to select a subset of them while also shrinking the remaining coefficients.

The title of the original paper where the Elastic Net model was introduced [41] is 'Regularization and Variable Selection via Elastic Net', which implies that the elastic net is also very suitable for variable selection. The L1 regularization term encourages many of the coefficients to be zero, which can be used for variable selection. The L2 regularization term helps to prevent overfitting. This combination of L1 and L2 regularization allows Elastic Net to balance the trade-off between fitting the data well and having a small number of variables. It also makes it less prone to overfitting than Lasso regression which uses only L1 regularization.

The magnitude of the (absolute value of the) coefficient can not directly be seen as the variable importance, but generally, larger coefficients could be seen as more important features.

Different variable selection methods will be used in the framework and compared based on their prediction results. In order to refer to different ways of feature selection before the transfer, the following abbreviations are used:

- *NoTL:* No transfer learning used.
- *AllTL:* No feature selection, all features are included in the transfer.

- *SigTL:* From the elastic net model, the features with non-zero coefficients are used for the transfer (in a univariate scenario, this is equivalent to *AllTL*).
- *CoefTL:* The elastic net is applied to the data and the absolute value of the coefficients are used as 'weights' for each feature before transfer.

The difference between these methods is especially important for data with many features, so less important features can be taken out before transferring. This might help the transfer to 'focus' and provide better results.

2.5 Transfer Component Analysis

This thesis will focus on a feature-based method called Transfer Component Analysis (TCA) because the proposed method in the case study is feature-based and the method is furthermore, suitable for regression problems. The history of methods which motivated the development of TCA is described in the appendix: Section 7.2. The key assumption for this method is that the marginals of the source and target data are not identically distributed, however, the conditionals of the embedded dataset is. In other words: $P \neq Q$, but $P(Y^S|\phi(X^S)) = Q(Y^T|\phi(X^T))$ under a mapping ϕ on the input.

Transfer component analysis was introduced by Pan et al. [48] in 2011. First of all, the distance between the marginal distributions is minimized. Similar methods such as MMDE [60] do this by setting up a Semi-Definite Programming (SDP) problem, which can be expensive. Therefore, a framework is introduced based on kernel feature extraction. In this way, the expensive SDP is avoided and the kernel generalizes unseen patterns in the data.

Minimizing the distance between source and target data Define K as in Equation 2.3 and decompose $K = KK^{-1}K = (KK^{-1/2})(K^{-1/2}K)$, often referred to as the empirical kernel map. Let $\tilde{W} \in \mathbb{R}^{(n^S+n^T)\times m}$ be a matrix which transforms the kernel map features to an m dimensional space. This leads to a resultant kernel matrix which is equal to

$$\tilde{K} = (KK^{-1/2}\tilde{W})(\tilde{W}K^{-1/2}K) = KWW^\top K \tag{2.11}$$

where $W = K^{-1/2}\tilde{W}$. Using this definition for $\tilde{K}$, the distance between the mapped feature spaces can be calculated using the MMD distance:

$$D(\phi(X^S), \phi(X^T)) = Tr((KWW^\top K)L) = Tr(W^\top KLKW). \tag{2.12}$$

Note that L is the same coefficient matrix as defined in Equation 2.4. For minimizing this distance, often a regularization parameter λ and term $Tr(W^\top W)$ is often needed to control the complexity of W.

Preserving the properties of the source and target data TCA also proposes that ϕ should also preserve the important properties of the source and target data such as the data variance. Since

$W^\top K$ is the embedding of the data in the latent space, the covariance matrix of the projected samples can be calculated by:

$$W^\top KHKW \tag{2.13}$$

where $H = I_{n^S+n^T} - (\frac{1}{n^S+n^T})\mathbf{1}_{n^S+n^T}$ is the centering matrix. Here, $I \in \mathbb{R}^{(n^S+n^T)\times(n^S+n^T)}$ is the identity matrix and $\mathbf{1} \in \mathbb{R}^{(n^S+n^T)\times(n^S+n^T)}$ is the matrix where all elements contain 1s. It is used to remove column means from a matrix so that the covariance matrix can be calculated this way. This can be showed by noticing that for a vector $\boldsymbol{x} \in \mathbb{R}^n$ and $H = I_n - \frac{1}{n}\mathbf{1}_n$:

$$H\boldsymbol{x} = \boldsymbol{x} - \overline{\boldsymbol{x}}, \tag{2.14}$$

where $\overline{\boldsymbol{x}}$ is an n-dimensional vector with indices $\frac{1}{n}\sum_{i=1}^n x_i$; the mean of the vector. From this, it is easy to see that

$$\boldsymbol{x}^\top H\boldsymbol{x} = \sum_{i=1}^n (x_i - \overline{x})^2. \tag{2.15}$$

Therefore, when $X = [\boldsymbol{x}_1, ..., \boldsymbol{x}_n]^\top$ is an $n \times p$ matrix with row vectors which represent instances and contain p features, then

$$HX = [\boldsymbol{x} - \overline{\boldsymbol{x}}, ..., \boldsymbol{x}_n - \overline{\boldsymbol{x}}]^\top.$$

Here, $\overline{\boldsymbol{x}}$ is the p-dimensional sample mean of $\boldsymbol{x}_1, ..., \boldsymbol{x}_n \in \mathbb{R}^p$ and indeed H subtracts the column mean from each column in H. It follows that, by using Equation 2.15,

$$X^\top HX = \sum_{i=1}^n (\boldsymbol{x}_i - \overline{x})(\boldsymbol{x}_i - \overline{x})^\top \tag{2.16}$$

is (proportional to) the sample covariance matrix of X. Using Equation 2.16, it follows that $W^\top KHKW$ equals the covariance matrix of the projected data KW.

Since TCA proposes that ϕ should preserve the important properties of the source and target data, it is an obvious choice to maximally preserve the data variance. This can be done by setting the constraint that $W^\top KHKW = I_{n^S+n^T}$. The sample covariance matrix of a dataset describes the variance and correlation between the features in the dataset. By setting this matrix to the identity matrix, it means that the features in the dataset have been transformed so that they have no correlation and the same variance. This results in eigenvectors or independent components that are more interpretable as they represent patterns in the data that are independent of each other. Additionally, this independence in the data also helps to preserve the variance explained by the eigenvectors which are the directions in the feature space that explain the most variance in the data.

Transfer Component Analysis Combining the two constraints where the distance between the source and target data is minimized and the properties of the source and target data are preserved, leads to the kernel learning problem as in Equation 2.17:

$$\boxed{\begin{array}{ll} \min\limits_{W} & Tr(W^\top KLKW) + \lambda Tr(W^\top W) \\ \text{s.t.} & W^\top KHKW = I_{n^S+n^T}. \end{array}} \tag{2.17}$$

Which can be rewritten to a trace maximization problem using the Lagrangian [48]. Indeed, let Φ be a diagonal matrix containing Lagrange multipliers i.e. $\Phi = \text{Diag}([\phi_1, ..., \phi_{n^S+n^T}])$. The Lagrangian of the optimization problem in Equation 2.17 can be written as:

$$\min_W \ \text{Tr}(W^\top (KLK + \lambda I_{n^S+n^T})W) - \text{Tr}((W^\top KHKW - I_{n^S+n^T})\Phi). \tag{2.18}$$

Then the derivative w.r.t W and setting this to 0 leads to:

$$(KLK + \lambda I_{n^S+n^T})W = (KHK)W\Phi.$$

This is known as a generalized eigenvalue problem and can be solved using generalized eigendecomposition. Therefore, calculate the eigenvectors and eigenvalues of

$$(KLK + \lambda I_{n^S+n^T})^{-1}(KHK).$$

Since it is a minimization problem, the eigenvalues and corresponding eigenvectors should be sorted from small to large values. The resulting matrix W contains the first m eigenvectors corresponding to the m smallest eigenvalues.

Note that when W is the identity matrix, it is not the same as *'no transfer preferred'* since it minimizes the discrepancies between the distributions in the projected space and not in the original space. Choosing the best kernel and value for λ is challenging and could be separate research. In the thesis, the optimal kernel and λ will be chosen based on the RMSE on the test data.

TCA algorithm and key assumptions

- Let $X^S = \{\boldsymbol{x_i^S}\}_{i=1}^d, \boldsymbol{x_i^S} \in \mathbb{R}^{n^S}$ and $X^T = \{\boldsymbol{x_i^T}\}_{i=1}^d, \boldsymbol{x_i^T} \in \mathbb{R}^{n^T}$ be the source and target data respectively with both containing d features.
- Choose a kernel and construct matrices K and L from their definitions:

$$K = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S} & K_{T,T} \end{bmatrix} \in \mathbb{R}^{(n^S+n^T)\times(n^S+n^T)}$$

- Calculate $L \succeq 0$ with

$$L_{ij} = \begin{cases} \frac{1}{n^{S^2}} & \boldsymbol{x}_i, \boldsymbol{x}_j \in X^S \\ \frac{1}{n^{T^2}} & \boldsymbol{x}_i, \boldsymbol{x}_j \in X^T \\ -\frac{1}{n^S n^T} & \text{other.} \end{cases}$$

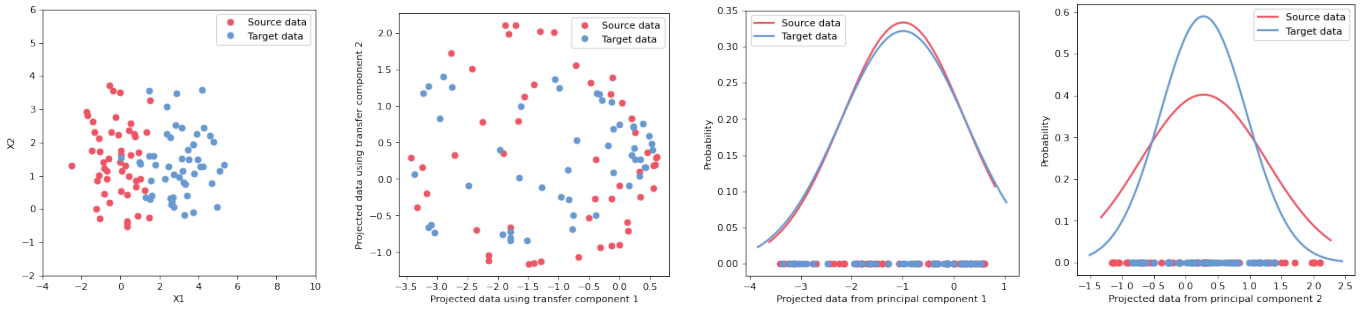
  Choose $\lambda > 0$ and $m \in 1, ..., d$ and calculate:

$$(KLK + \lambda I_{n^S+n^T})$$

- Build $H = I_{n^S+n^T} - \frac{1}{n^S+n^T}\mathbf{1}_{n^S+n^T}$ and calculate KHK
- Construct W by finding the $m < n^S + n^T - 1$ smallest *absolute* eigenvalues and corresponding eigenvectors of the generalized eigenvalue problem. To determine the number of components m, the cumulative explained eigenvalue ratio can be used which is explained in appendix Subsection 7.2.1.

- Calculate the transferred data points by calculating KW, which is the embedding of the data in the latent space.

Applying this method to a simulation set using an RBF kernel leads to the results in Figure 2.6. Details on this simulation and how it helps understand the difference between other methods are provided in the appendix (Section 7.2). From this figure, it becomes clear that indeed, the 2-



(a) Original source and target data $X \in \mathbb{R}^{50\times 2}$ from the simulation

(b) TCA with an RBF kernel, represented in 2 dimensions

(c) Projection onto 1 dimension based on the first transfer component using TCA (RBF kernel), with normal fit

(d) Projection onto 1 dimension based on the second transfer components using TCA (RBF kernel), with normal fit

Figure 2.6: TCA applied to the simulation data with fitted normal distributions with an RBF kernel

dimensional source and target dataset are much more overlapping after transfer looking at the first 2 transfer components.

2.6 Prediction metrics

In order to compare methods, metrics are also used in the original case study in order to determine the performance of various models. These metrics are the Root Mean Squared Error (RMSE) and the Mean Absolute Percentage Error (MAPE).

Suppose $\hat{y}_i$ is the predicted cycle life for battery i and y_i is the actual cycle life for batteries $i = 1...n$. Then:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.19}$$

Similarly, the average percentage error can be calculated by:

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{y_i} \times 100 \tag{2.20}$$

In both cases, a lower error means a more accurate model. However, there is often a tradeoff [61] between minimizing the MAPE and minimizing the RMSE. The MAPE is a measure of the accuracy

of a model that is easy to understand and interpret, as it is expressed as a percentage. The RMSE, on the other hand, is a measure of the difference between the predicted and actual values, with larger differences being penalized more heavily.

In general, the RMSE is more sensitive to outliers than the MAPE, so a model that minimizes the RMSE may not necessarily have the lowest MAPE. As a result, it may be necessary to trade-off between the two measures to achieve the best overall model performance. The bigger challenge in the case study [37] is to predict outliers accurately. Consequently, RMSE will be the main focus of our thesis while MAPE will still be acknowledged.

3 | Simulation Study

Simulated data will be used to generate results before applying the methods to the case study data. This allows for control over conditions, making the results more explainable. By manipulating the source and target data, the effect of dataset differences can be controlled and monitored. The simulated data will be generated according to the case study to ensure consistency with the case study results. The source and target data will be varied in different sections. First of all, a univariate scenario is analyzed in Section 3.1, which resembles the variance model in the case study, introduced later on in Chapter 4. In Section 3.2, the multivariate scenario is evaluated which resembles the full model in the case study.

As introduced in Section 2.4, different variable selection methods will be used in order to 'focus' the transfer. For these methods, the following abbreviations are used:

- *NoTL:* No transfer learning used.
- *AllTL:* No feature selection, all features are included in the transfer.
- *SigTL:* From the elastic net model, the features with non-zero coefficients are used for transfer (in a univariate scenario, this is equivalent to *AllTL*).
- *CoefTL:* The elastic net is applied to the data and the absolute value of the coefficients are used as 'weights' for each feature before transfer.

Note that for the univariate case, where the feature space consists out of 1 feature, the *AllTL* and *SigTL* methods are equivalent.

3.1 Univariate scenarios

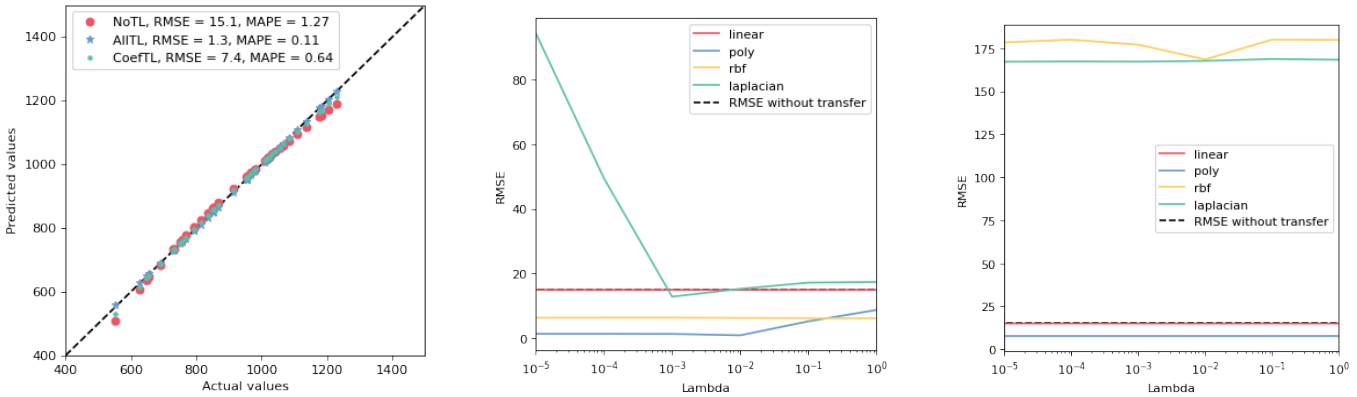
First, univariate samples will be generated and evaluated. In Subsection 3.1.1, the source and target data are sampled from the same distribution. In Subsection 3.1.2, the source and target data are sampled from different distributions. The goal of these first 2 sections is to analyze different metrics before and after the transfer in order to get an idea of the relationship between those metrics and the performance of the transfer. Finally, in Subsection 3.1.3, the target distributions will be shifted/scaled distributions of the source data in order to analyze the relationship between the magnitude of distribution difference and the performance of the transfer. Here, the relationship is analyzed between the distribution difference and the transfer performance. For this, the source distribution is sampled 100 times and shifts/scales will be applied to this source distribution to generate varying target distributions. On these distributions, different methods are applied and the results are analyzed. This is computationally heavy and therefore, the 'DelftBlue' supercomputer of the TU Delft is used [62].

3.1.1 Source and target data from 1 univariate normal distribution

In the initial phase of this simulation study, the framework will be evaluated using a dataset that has one feature and the source and target data are drawn from the same distribution. To ensure the simulation closely resembles the case study, the samples will be generated from a univariate normal distribution with a mean and variance similar to the variance feature (over all samples) from the variance model. Since in the case study, most scenarios have circa 40 samples in the train (or source) data and 40 samples in the test (or target) data, 80 samples will be generated in total. The labels for these samples will be generated by setting:

$$y_i = -x_i^5. \tag{3.1}$$

Before transferring the data, the Zk-test is used to quantify the distributional difference between source and target data. Since the p-value for this test is above 0.05, the null hypothesis should be accepted which correctly indicates that both samples are from the same distribution.



(a) Predicted values for the *NoTL*, *AllTL* and *CoefTL* method in the univariate case for optimal kernel and λ.

(b) *AllTL*, RMSE values for different kernels and lambda

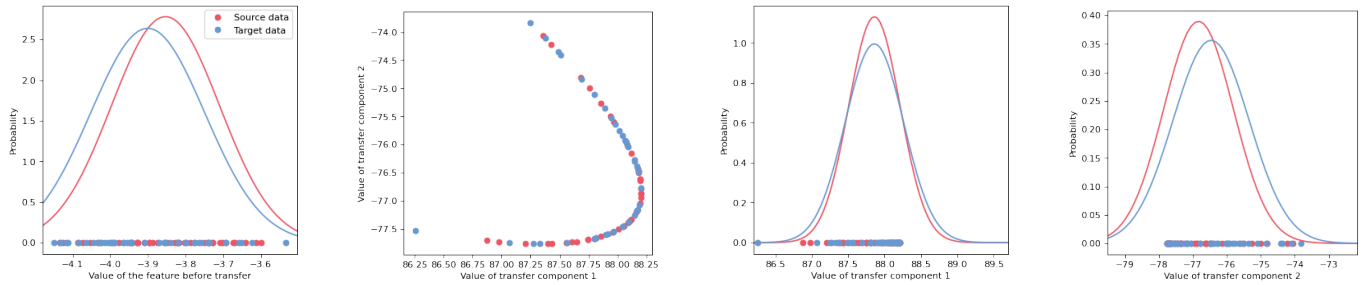
(c) *CoefTL*, RMSE values for different kernels and lambda

Figure 3.1: RMSE values of the predictions for different kernels and lambda

In Figure 3.1a, the results of the prediction are shown for the 3 models, *NoTL*, *AllTL* and *CoefTL*. Since for the univariate case, the *AllTL* and *SigTL* methods are the same, the *SigTL* method is left out. The transfer improved the results as seen from the prediction metrics. Particularly, the predictions were more accurate for values in the tails after transfer. Note that the results are very close to each other, which is probably caused by the fact that no noise is added to the features/label data.

The other plots in Figure 3.1 show the RMSE for different kernels and values for lambda for both transfer models. In both cases, the polynomial kernel leads to the lowest RMSE, which is notable since the labels were also generated using the polynomial relation. The elastic net after transfer selects in the *AllTL* method the first 2 transfer components, which indirectly correspond to the first 2 eigenvectors with the lowest eigenvalue in the projection matrix. The *CoefTL* method has only 1 significant transfer component which also passes the Zk-test.

To understand what happens to the data after the transfer, the marginals of source and target data before transfer and the significant transfer components after transfer for the *AllTL* method are plotted in Figure 3.2. It appears that for the *CoefTL* method, the significant transfer components look similar as Figure 3.2a.



(a) Source and target data before transfer with a normal fit

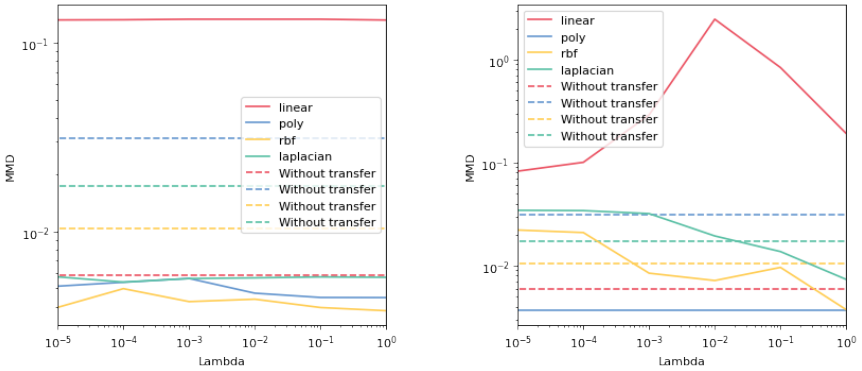
(b) Source and target data after transfer, only the significant components

(c) Source and target data after transfer with a normal fit, transfer component 1

(d) Source and target data after transfer with a normal fit, transfer component 2

Figure 3.2: Feature and transfer components before and after transfer with normal fits, *NoTL* and *AllTL* methods

From these plots, it can be seen that before the transfer, the source and target data slightly differ. After transfer for the *AllTL* method, there is a clear relationship between the first and second transfer component. The distributions of the first transfer components have a more similar location but different variances. The distributions of the second component have more similar variances but a slightly different mean.



(a) *AllTL* (b) *CoefTL*

Figure 3.3: MMD for different values of λ and different kernels, with and without transfer

In Figure 3.3, the MMD values are given before and after transferring using the corresponding kernel for calculating the MMD and applying the transfer. In order to compare the MMD between different kernel types, the data will be scaled as described in Section 2.3.

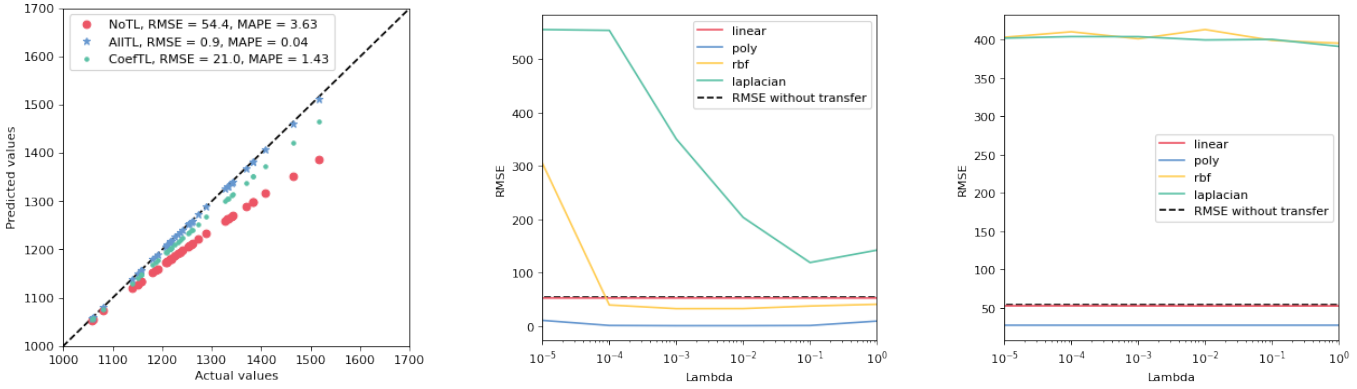
For the *AllTL* method, all MMDs become smaller after transfer except for the linear kernel. This could imply that the transfer was successful for all methods except the linear kernel. In Figure 3.1, it was seen that the linear kernel indeed does not give any improvement. This shows that even with

an unsuccessful transfer, the elastic net was still able to give moderate predictions using the data from the projected space. The Laplacian kernel only improved upon the results without transfer for 1 value of lambda, namely $1e-3$. This is not in line with the results from Figure 3.3a, showing that successfully minimizing the distribution distances can still lead to worse predictions. The results from the polynomial and RBF kernel improved for all values of lambda, which is in line with the indication from the MMD before and after transfer.

For the *CoefTL*, only the polynomial kernel has a lower MMD compared to the MMD without transfer for this kernel for all values of lambda. This is also the only kernel that improves upon the prediction results for all values of lambda. For the RBF and Laplacian kernel, it appears that the MMD is minimized for some values of lambda while the predictions do not improve. This shows again that the elastic net might not be able to improve predictions using data from the project space. For the linear kernel, no improvements were seen in the predictions which are in line with MMD values. Looking at these results, the MMD might not be a good metric for indicating a successful transfer. It could also be that since the underlying assumption of transfer learning is not satisfied, the model acts differently and therefore it is not a useful example.

In summary, the transfer improves the results in the case where distributions are very similar and it is not necessarily expected that the transfer will perform better than the results without transfer, as this is not in line with the assumption of TCA. The transfer's ability to predict values accurately in the tail could be due to its recognition of patterns not captured by traditional machine learning methods, thus explaining the improvement in predictions. Furthermore, the MMD in this scenario appears not to be an accurate indication of a successful transfer.

3.1.2 Source and target data from 2 univariate normal distributions



(a) Predicted for the *NoTL*, *AllTL* and *CoefTL* method in the univariate case

(b) *AllTL*, RMSE values for different kernels and lambda

(c) *CoefTL*, RMSE values for different kernels and lambda

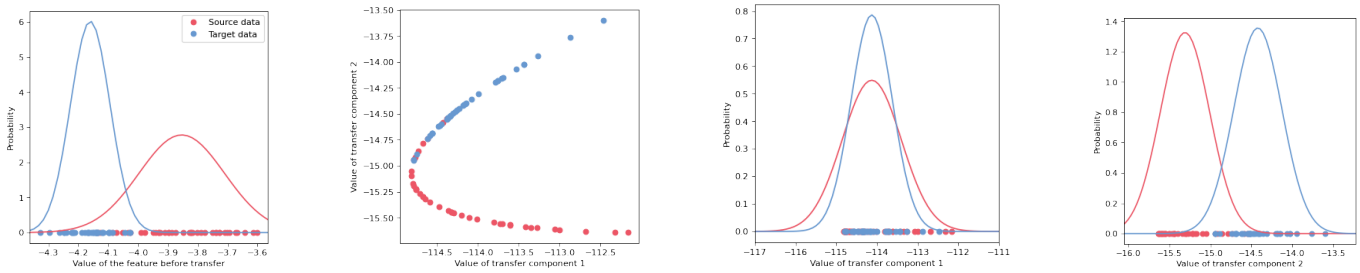
Figure 3.4: Predictions and RMSE values of the predictions for different kernels and lambda

In this section, the target data is drawn from a univariate distribution that has a similar mean and variance as the target data from the case study's variance feature. This indicates that the source

and target data originate from distinct distributions. This is verified by the Zk-test, which produces a p-value of 0.

The predictions for all methods, except the *SigTL* method, are given in Figure 3.4a. Indeed, the transfer methods improve upon the results without transfer, in particular, higher values are estimated more accurately. From Figure 3.4b and Figure 3.4c, it follows that both variable selection methods prefer the polynomial kernel, where the *AllTL* method is slightly more sensitive for the value of lambda. The *AllTL* method gives better predictions compared to the *CoefTL* method.

It appears that the *AllTL* method selects the first 2 transfer components as significant, however, the second transfer component does not pass the Zk-test. Therefore, Figure 3.5 shows plots and normal fits of these two significant components for the *AllTL* method.



(a) Source and target data before transfer with a normal fit

(b) Source and target data after transfer, only the significant components

(c) Source and target data after transfer with a normal fit, transfer component 1

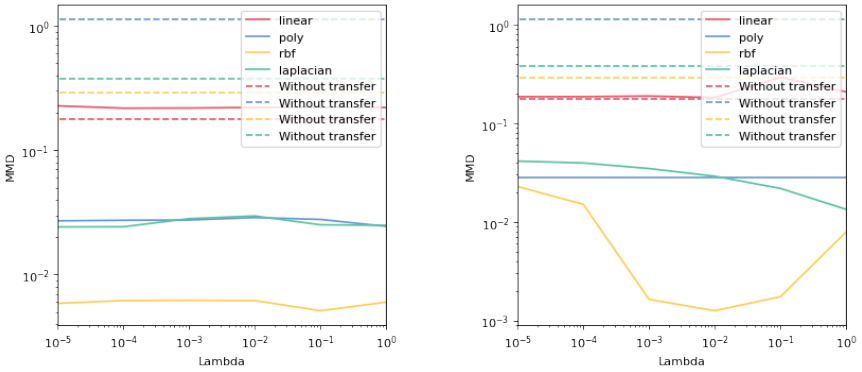
(d) Source and target data after transfer with a normal fit, transfer component 2

Figure 3.5: Feature and transfer components before and after transfer with normal fits, *NoTL* and *AllTL* methods

Again, there is an interesting relationship between the first 2 components. Where the first transfer component of the source and target data have a similar location but different variance, the second component is different in location but has a similar variance. It is important to keep in mind that the Zk-test only compares marginals, so even when the second component does not pass the test, there may still be a joint relationship with the first component. Additionally, the component after transfer might have not the same distribution for source and target, however, they can still be similar.

The *CoefTL* method selects 1 transfer component which does not pass the Zk-test. This could be interpreted as an unsuccessful transfer, which is actually not in line with the results.

This example shows that it is challenging to determine a precise metric for a successful transfer. Especially since the Zk-test is designed for marginal distributions and therefore does not take into account the joint relationship between features. Similarly, the MMD, which is suitable for multivariate distributions, appears not to be a reliable metric for indicating a successful transfer. As can be observed in Figure 3.6, the MMD shows that the distance between the source and target distributions was successfully minimized for many kernels. The only kernel, however, that did show improvement for all lambda and both methods are the polynomial kernel. Using a linear kernel did not lead to an improvement in the results which is also in line with the MMD. However, for the RBF and Laplacian kernel, the MMD remains unreliable for determining a successful transfer.



(a) *AllTL* (b) *CoefTL*

Figure 3.6: MMD for different values of λ and different kernels, with and without transfer

In conclusion, the transfer is able to improve the results when the distribution of the source and target data is not the same. Furthermore, the *AllTL* method gives better predictions compared to the *CoefTL* method. This is not surprising, as selecting and weighing variables before the transfer is expected to become more relevant for larger feature spaces. It is also worth mentioning that TCA was especially effective in accurately predicting tail values. Identifying a reliable metric for determining a successful transfer remains challenging. The MMD showed that for most kernels, the distance between the source and target distributions was minimized, but this did not always align with the results. The Zk-test provides some insight, but its focus on marginal distributions makes it difficult to interpret the results, making it challenging to assess the success of the transfer.

3.1.3 Relationship between the distribution difference and transfer performance

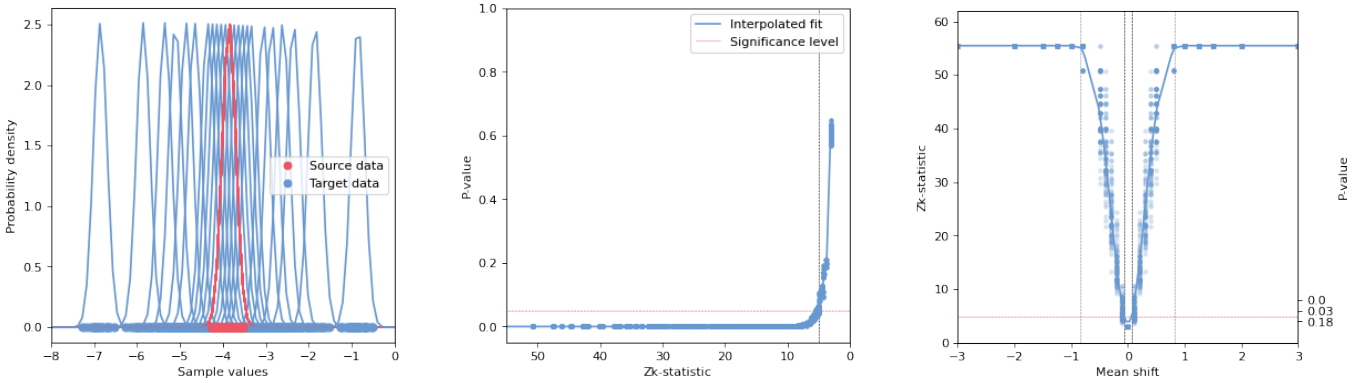
In order to gain a deeper understanding of the relationship between different distributions and the effectiveness of transfer learning, various simulations will be conducted. First, a univariate normal distribution is simulated with similar mean and variance as in the univariate model (variance model) of the case study. By using a standard uniform distribution to generate 40 random probabilities and applying the normal distribution's quantile function with the corresponding mean and variance, the samples for all examples will be identical, making it simpler to compare results.

In the first part, the means will be shifted while the variance will be kept the same. In the second part, both the mean and variance are varied. Since the normal distribution is symmetric, it is assumed that the results will be symmetric as well. Therefore, the variance is only scaled in one way, assuming that for the corresponding 'inverse scale', the results will be mirrored. This will be repeated 100 times so that the results are based on 100 source samples with shifted/scaled target samples and the results are more robust.

The second analysis is for negatively skewed data, which aligns with the source data of the case study. Multiple distributions are fit to the source data and an uncommon distribution is selected for sampling to generate target data and labels. As the data is not symmetric, the variance will be adjusted in both directions while the mean is shifted. This approach demonstrates the functioning of the framework under varying scenarios and sheds light on the results of the case study.

Since in the previous section it was observed that there was no reliable metric after transfer for indicating a successful transfer, this will not be analyzed in the following simulation results.

Varying means and fixed variance for symmetric source and target distributions The mean of the target distribution will be shifted with values between -3 and 3, including the value 0 which means that the source and target data are exactly the same. The labels will be generated similarly as described before and the kernel and value for λ will be fixed with a polynomial kernel and $\lambda = 1e - 2$.



(a) Source distribution with different target distributions for 1 source sample

(b) Zk-statistic distribution with corresponding p-values and interpolated fit

(c) Zk-statistic for different target distributions due to shifts in mean

Figure 3.7: Different target distributions and Zk-statistic distribution with corresponding p-values and interpolated fit

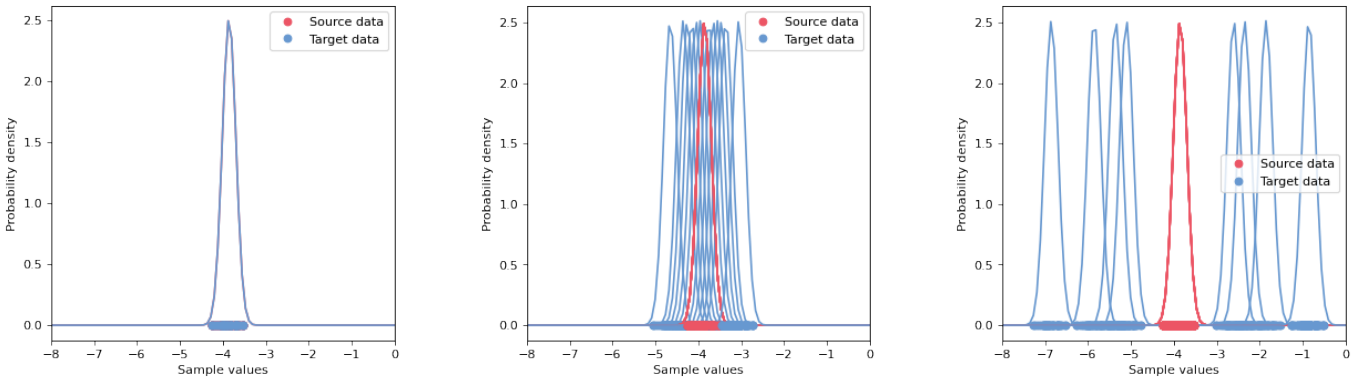
In Figure 3.7, one source sample with different target distributions is visualized with corresponding Zk-statistics and p-values. For each source sample, 23 different shifts are applied resulting in 23 target distributions for each source samples with different corresponding Zk-statistics and p-values. Based on these values, a distribution can be plotted and by interpolating[1] a line, the statistic can be derived for which the p-value equals 0.05. It can be seen in Figure 3.7c that the Zk-statistics still show some uncertainty. Only when distributions are very similar or very different, the statistic, with the corresponding p-value, is more certain. This uncertainty will reduce whenever the sample size increases, as was shown in Section 2.3. However, as this simulation study should resemble the case study, the sample size is kept similar. Since the experiment is repeated 100 times, the mean values still lead to reliable conclusions.

The different dashed lines indicate 3 ranges:

- **Similar distributions** (Figure 3.8a) The range delineated by the black dashed lines corresponds to distributions that are similar, with a p-value greater than 0.05. These values are also obtained through interpolation.
- **Significantly different, but overlapping distributions** (Figure 3.8b) The dissimilarity of the distributions is indicated by the span between the grey and black dashed lines. Although the p-value obtained from the Zk-test for these distributions is less than 0.05, there is still some degree of overlap between them. As a result, the Zk-statistic will not reach its highest value.

[1]For this, the Python function *'scipy.stats.interp1d'* is used

- **Very different, non-overlapping distributions** (Figure 3.8c) The grey lines that mark the outer range indicate a significant dissimilarity between the distributions. When the Zk-statistic reaches its maximum value, it can be inferred that the two distributions no longer overlap. This is dependent on the number of observations, as can be found in appendix Figure 7.6. The values for the mean shift that corresponds to this are obtained through interpolation.



(a) Similar distributions with p-value > 0.05

(b) Significantly different, but overlapping distributions with p-value ≤ 0.05 and the Zk-statistic below its maximum

(c) Very different, non-overlapping distributions for which the Zk-statistic is maximal

Figure 3.8: Different classes for classifying the difference between source and target distributions

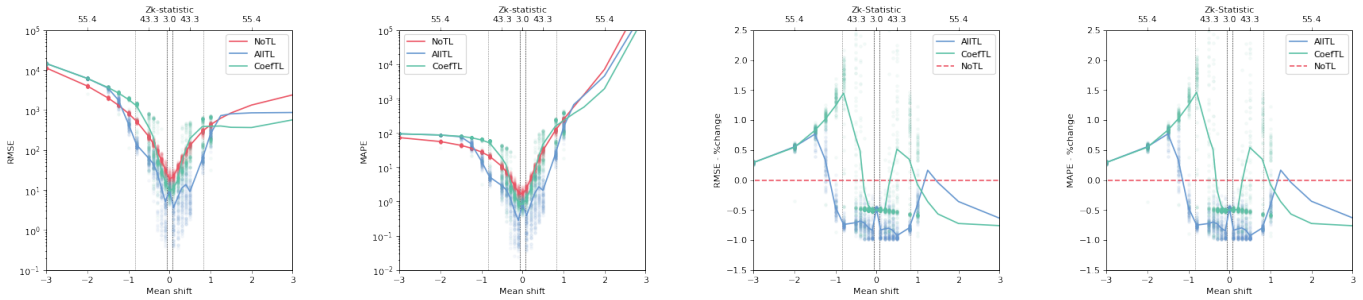
In Figure 3.9, the absolute and relative prediction results are shown for the different shifts in mean for all three methods, averaged over 100 repetitions. The dashed lines correspond to the ranges which are described above. The following conclusions can be drawn from these results. These are mainly based on the results where the relative change in RMSE and MAPE is visualized.

	Similar distributions	**Significantly different, but overlapping distributions**	**Very different, non-overlapping distributions**
Symmetric source samples with shifted target distributions	Both transfers improve upon the results without transfer	At least 1 transfer method improves upon the results, the other becomes worse than NoTL	At some stage, a negative shift will result in poorer transfer outcomes, while a positive shift will enhance transfer performance in at least one instance.

Table 3.1: Summary of the results for the symmetric case with shifted target distributions

It's worth noting that in the scenario where the source and target data are identical, both transfers produce superior results. This outcome may seem unexpected, given that no transfer might be expected as optimal in this case. However, TCA consistently employs a kernel to project the data into a new space, and hence the elastic net is always applied to data that has undergone projection. This explains the difference in results. It is worth noting that the outcomes exhibit an asymmetrical pattern despite the symmetry of the shifts. This asymmetry is evident even in cases where no transfer of learning occurs, suggesting that several factors contribute to this phenomenon. One such factor is the 5-degree polynomial relationship between the label and feature data. Consequently, target distributions that are closer to 0 exhibit smaller differences in their label data, while those further from 0 have more significant differences between their labels. This observation explains why the

model's RMSE is higher for a negative shift, as it is less sensitive to these larger differences. On the other hand, this heightened sensitivity can result in a bias that clarifies why the MAPE increases substantially for a positive shift with smaller differences in the label data. This is referred to as the variance/bias trade-off [63].



(a) RMSE vs. mean shift

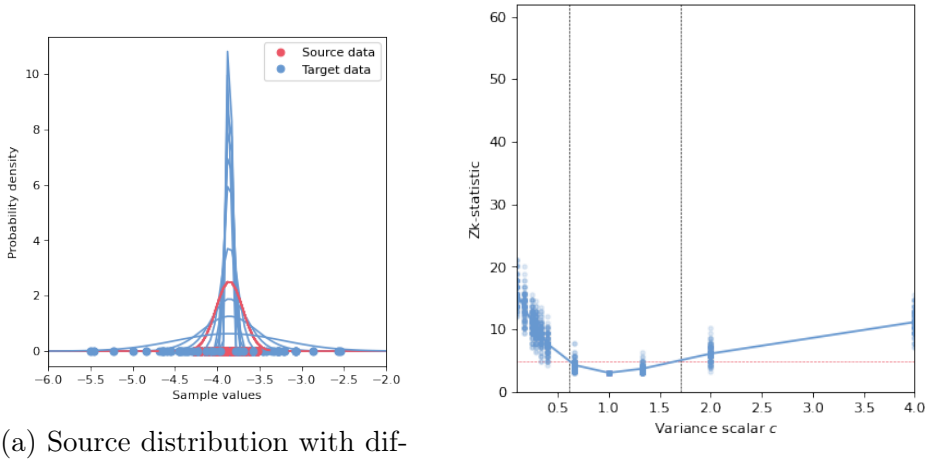
(b) MAPE vs. mean shift

(c) Relative change in RMSE vs. mean shift (zoomed)

(d) Relative change in MAPE vs. mean shift (zoomed)

Figure 3.9: Average prediction results for 100 symmetric source samples with shifted target samples for *NoTL*, *AllTL* and *CoefTL*

Fixed means and varying variance for symmetric source and target distributions The variance of the target distribution will be scaled by c which varies between 0.1 and 4, including the value 1 which means that the source and target data are exactly the same. The scaled variance then becomes $\sigma^2 \cdot c$. The labels will be generated similarly as described before and the kernel and value for λ will be fixed with a polynomial kernel and $\lambda = 1e - 2$.



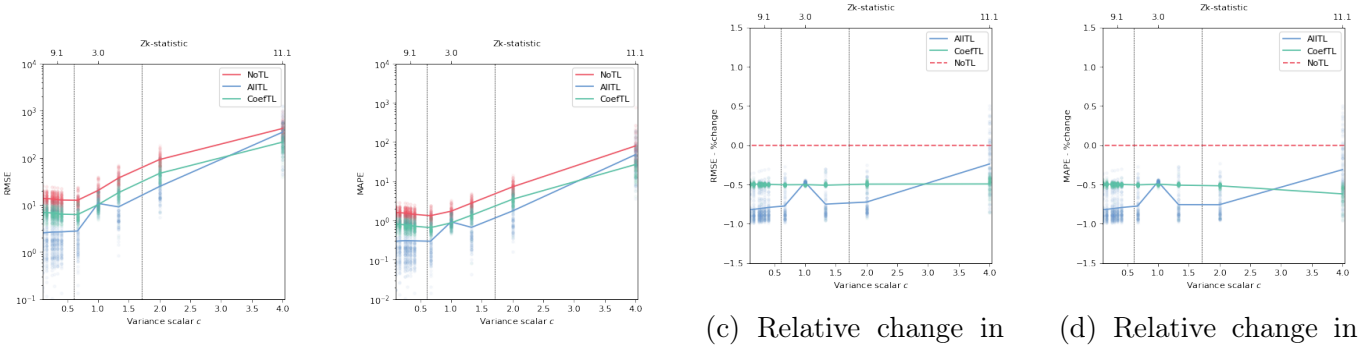
(a) Source distribution with different target distributions for 1 source sample

(b) Zk-statistic for different target distributions due to different variances

Figure 3.10: Different target distributions and corresponding Zk-statistic and p-value

In Figure 3.10, the different target distributions for 1 one source sample are visualized and the Zk-statistics and p-values are plotted against the scalar of the variance. The vertical lines indicate for which target distributions the p-value is below 0.05, thus significantly similar to the source sample.

Note that the source and target distribution always overlap which explains why the maximum Zk-statistic is never reached. The scenario where the variance is very small or very large is not very realistic. Therefore, this example is included in order to see the impact of the variance on the prediction results rather than a realistic example.



(a) RMSE vs. scaled variance

(b) MAPE vs. scaled variance

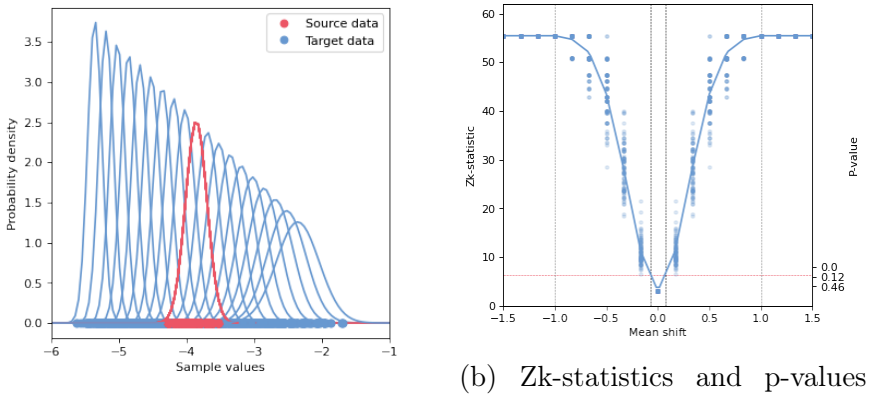
(c) Relative change in RMSE vs. scaled variance

(d) Relative change in MAPE vs. scaled variance

Figure 3.11: Average prediction results for 100 symmetric source samples with different target samples for *NoTL*, *AllTL* and *CoefTL*

From the results in Figure 3.11, it can be seen that the transfer always improves upon the results. This indicates that the results remain in line with the previous example.

Varying means and variance for symmetric source and target distributions In a similar way, the mean can be shifted and the variance can be scaled so that both vary for different target distributions. The mean is shifted between $[-1.5, 1.5]$, while the variance is scaled by $\sigma^2 \cdot c$ where c is linearly divided (in 19 steps) over the range $[0.67, 2]$. This leads to the target samples and corresponding Zk-statistics in Figure 3.12.



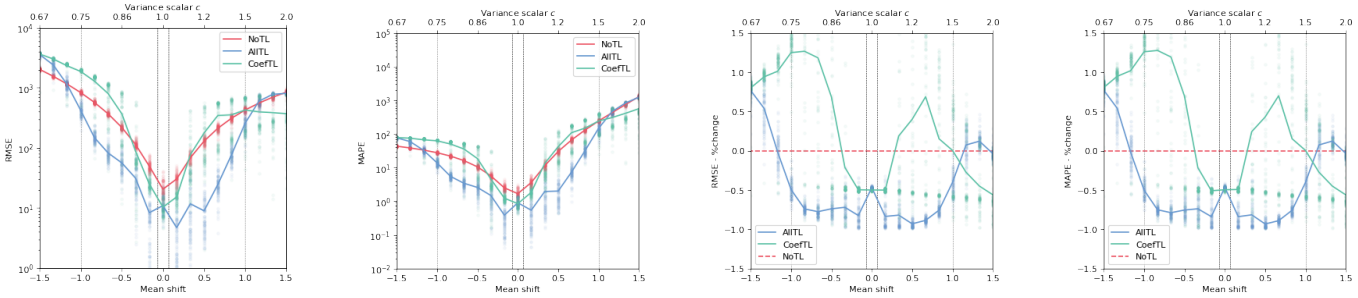
(a) Source distribution with different target distributions for 1 source sample

(b) Zk-statistics and p-values for different shifted/scaled target distributions with interpolated fit

Figure 3.12: Different target distributions and corresponding Zk-statistic and p-value

It is notable that the Zk-statistical distribution is symmetric while there is an asymmetry in the way

the target distributions are shifted/scaled. This shows that the Zk-test is much more sensitive to mean shifts compared to the variance difference. The prediction results can be found in Figure 3.13. The observed patterns in these figures are very similar to what was observed in the first case, where only the mean shifted. Therefore, the results can be concluded similar as in Table 3.1.



(a) RMSE vs. mean shift

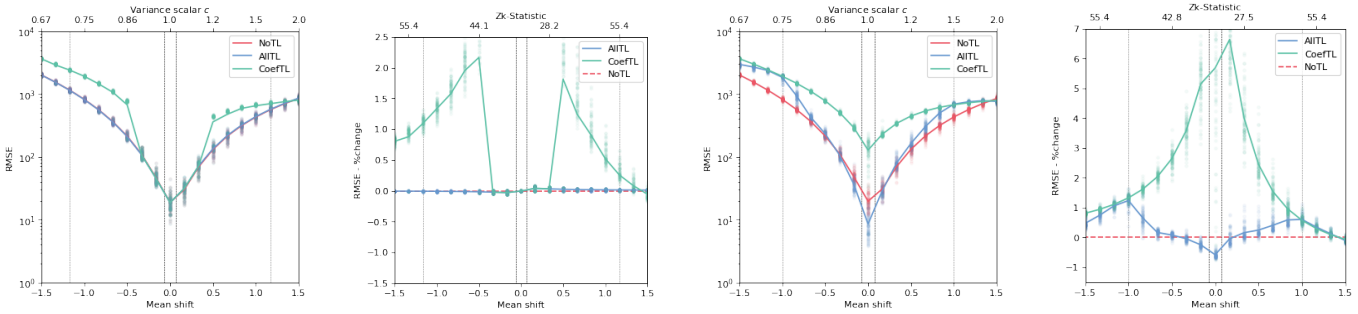
(b) MAPE vs. mean shift

(c) Relative change in RMSE vs. mean shift

(d) Relative change in MAPE vs. mean shift

Figure 3.13: Average prediction results and boxplots for 100 symmetric source samples with different target samples for *NoTL*, *AllTL* and *CoefTL*, varying means and variances

It should be remembered that the performance of the methods depends on the choice of the kernel and other parameters. In Figure 3.14, as the MAPE behaves similarly to the RMSE, only the RMSE results are shown for the linear kernel and RBF kernel (with $\lambda = 1e-2$). From this, it is clear that the overall relationship between the difference in distribution and the performance is more or less similar to the polynomial kernel, but does not always improve upon the results without transfer anymore. Choosing the correct kernel and corresponding value for lambda is, however, outside the scope of this research.



(a) RMSE vs. mean shift, linear kernel

(b) Relative change in RMSE vs. mean shift, linear kernel

(c) RMSE vs. mean shift, RBF kernel

(d) Relative change in RMSE vs. mean shift, RBF kernel

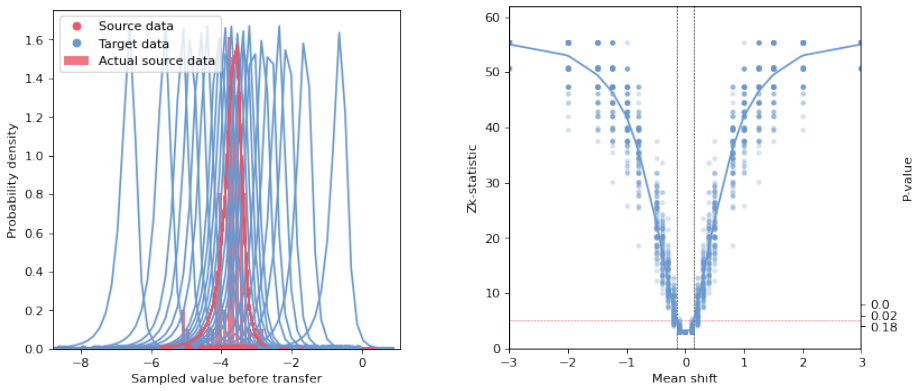
Figure 3.14: Average prediction results for 100 symmetric source samples with different target samples for *NoTL*, *AllTL* and *CoefTL* and the Linear and RBF kernel

Conclusion for symmetric source and target distribution with varying means and variance For a symmetric source and target distribution with different scales and shifts, the results can be summarized as in Table 3.1. The summary of the results aims to provide more general observations rather than specific details on different variable selection methods. This means that the focus

is on overarching patterns and trends, rather than specific details. Some additional observations should be taken into account as well.

- **Kernel choice sensitivity:** The results of the study or analysis are highly dependent on the choice of kernel. This means that the outcomes can vary significantly based on the type of kernel used in the analysis.
- **RMSE and MAPE tradeoff:** A trade-off can be observed between the RMSE and the MAPE. In some cases, an improvement in RMSE may result in a worsening of MAPE, and vice versa. This trade-off highlights the importance of considering both measures when evaluating the performance of a model.

Varying means with skewed source and target distributions By fitting many distributions to the source data of the case study (batches 1 and 2) a sample can be generated closer to the case study data. From this distribution, different means will be added similar to before, and corresponding Zk-test results will be provided. Furthermore, labels are generated similarly as before and predictions will be done and analyzed without and with the transfer. This will be done for multiple kernels, but 1 fixed value for lambda, namely $\lambda = 1e-2$. By using the '*Fitter*' package [64] in Python, 106



(a) Source distribution with different target distributions for 1 source sample

(b) Zk-statistics and p-values for different shifted/scaled target distributions with interpolated fit

Figure 3.15: Sampled skewed source data with different shifted target distributions and corresponding Zk-statistic and p-value

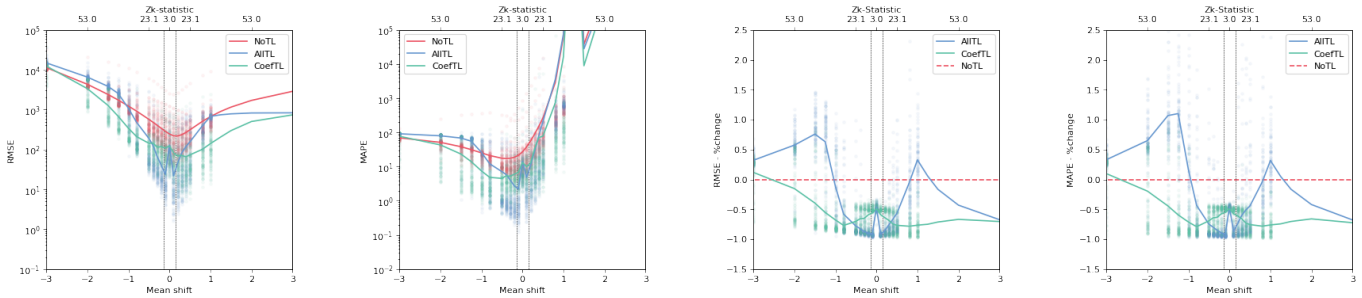
common and uncommon distributions. This package provides a combination of metrics such as the Kolmogorov-Smirnov test (Section 2.3), the sum of squared errors (SSE), AIC [65] and BIC [66], so that the user can determine which distribution fit is most accurate according to their wishes. From this, *Johnson's SU* [67] is selected as it outperforms all distributions on all of these metrics. This distribution has probability density:

$$f(x,a,b) = \frac{b}{\sqrt{x^2+1}} \cdot \phi(a + b\log\left(x + \sqrt{x^2+1}\right)) \tag{3.2}$$

where a, b ($b > 0$) are shape parameters and ϕ is the pdf of the normal distribution. The density in Equation 3.2 is defined in the standardized form so the non-standardized form $f(x,a,b,\mu,\sigma^2)$ is

equivalent to $f(y, a, b)$ with $y = \frac{X-\mu}{\sigma^2}$ where μ is the mean and σ^2 is the variance. The *Fitter* function automatically fits the optimal shape parameters for a, b. Again, probabilities are sampled from a standard uniform distribution and by using the quantile function of Johnson's SU distribution, the sample for the source data, with size 40, is generated. This leads to a negatively skewed distribution, meaning that the distribution is asymmetric and the mass of the distribution is concentrated on the right of the distribution. The labels are again similar as in Equation 3.1. This leads to the samples and Zk-test results in Figure 3.15.

The dashed lines correspond to the values of the mean shift for which the null-hypothesis should be rejected, implying that source and target data are from different distributions. These dashed lines correspond to the dashed lines in the prediction results in Figure 3.16. It is notable that the Zk-statistics appear to be symmetrical even though the data is skewed. The prediction results for the



(a) RMSE vs. mean shift

(b) MAPE vs. mean shift

(c) Relative change in RMSE vs. mean shift

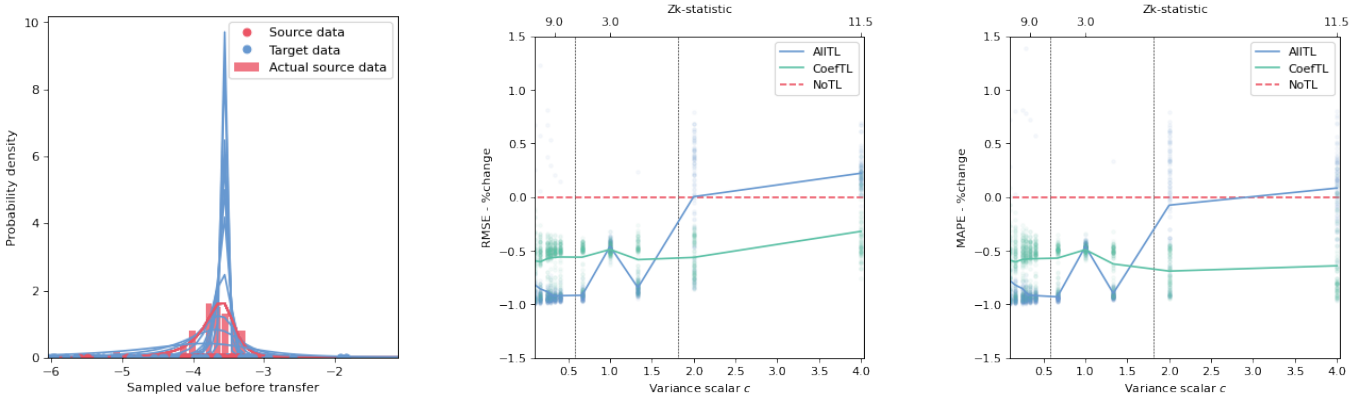
(d) Relative change in MAPE vs. mean shift

Figure 3.16: Average prediction results and boxplots for 100 symmetric source samples with different target samples for *NoTL*, *AllTL* and *CoefTL*, varying means

polynomial kernel are given in Figure 3.16. From these plots, similar conclusions can be drawn as in the symmetric case. The only difference is that at some point, for a negative shift, the results after transfer become slightly worse compared the results without transfer. Therefore, this contradicts with the conclusion that for dissimilar, but overlapping distributions, at least one transfer improves the results.

It is notable that the *CoefTL* method for the skewed case outperforms the *AllTL* method, whereas for the symmetric case, this was generally the other way around. This shows that the variable selection methods have different effects on the prediction results which makes in some cases one more suitable compared to the other.

Fixed means and varying variance for negatively skewed source and target distributions
In this part, the mean is again fixed while the variance is scaled. The variances are scaled in a similar way as the symmetric case where the mean was fixed and the variance varied. For the polynomial kernel, this leads to the results in Figure 3.17. The dashed line here indicates the variance without scaling.



(a) Source distribution with different target distributions for 1 source sample

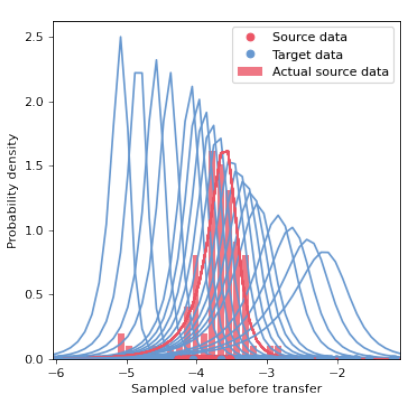
(b) Relative change in RMSE vs. scaled variance

(c) Relative change in MAPE vs. scaled variance

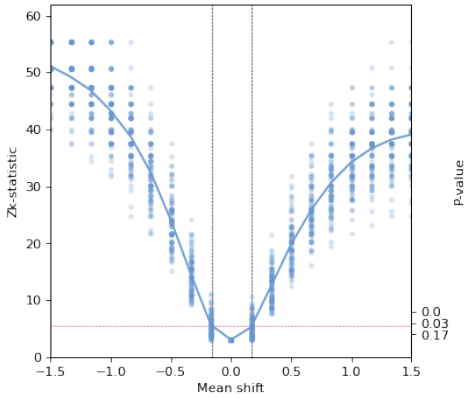
Figure 3.17: Average prediction results for 100 skewed source samples with different target samples for *NoTL*, *AllTL* and *CoefTL* with the polynomial kernel, fixed means and varying variances

The outcomes closely match what was seen in the symmetric case when variances were altered but the mean was constant. It is only that in this case, one transfer leads to worse results when the variance becomes larger at some point. Nonetheless, on the whole, at least one transfer consistently produces better outcomes than no transfer. For smaller variances, the transfer improves upon the results.

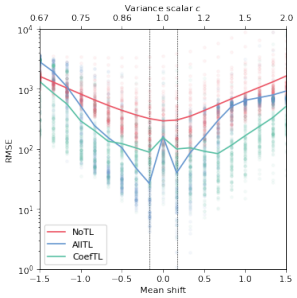
Varying means and variance for negatively skewed source and target distributions Finally, both the mean and variance are shifted and scaled. This is done for a mean shift between $[-1.5, 1.5]$ and corresponding scaled variances where the scaled variance becomes $\sigma^2 \cdot c$ with $c \in [1.5, 1.5]$. Again, 19 different target distributions are generated and the experiment is repeated 100 times. For the polynomial kernel, this leads to the results in Figure 3.18.



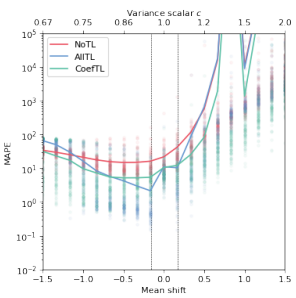
(a) Source distribution with different target distributions for 1 source sample



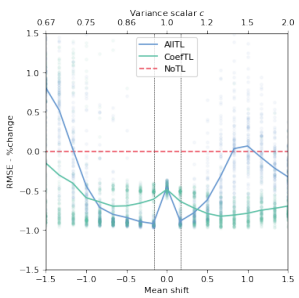
(b) Zk-statistics for different shifted/scaled target distributions



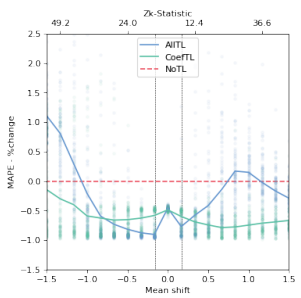
(c) RMSE vs. mean shift



(d) MAPE vs. mean shift



(e) Relative change in RMSE vs. mean shift



(f) Relative change in MAPE vs. mean shift

Figure 3.18: Average prediction results for 100 skewed source samples with different target samples for *NoTL*, *AllTL* and *CoefTL* with polynomial kernel, varying means and variances

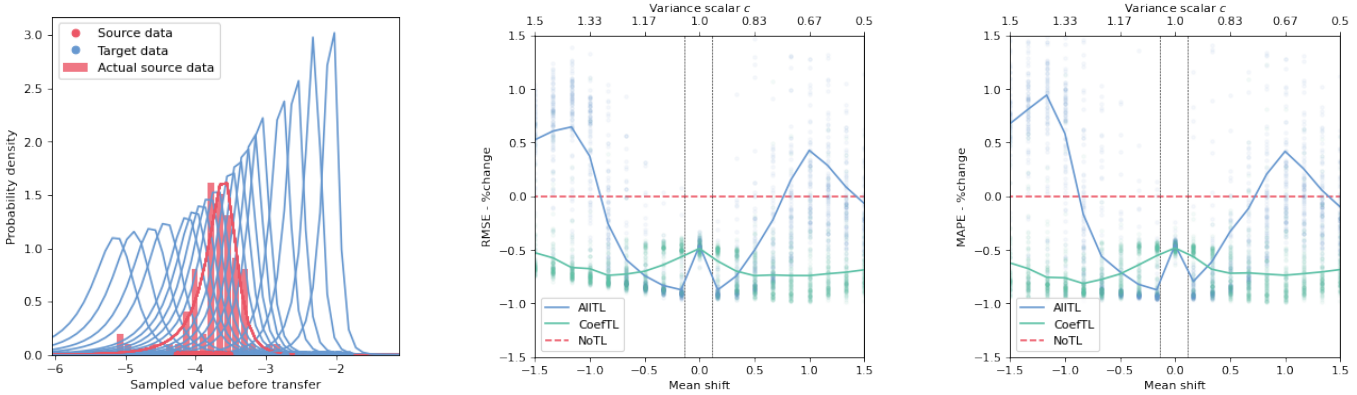
It is first of all notable that the Zk-statistic stays below the maximum loss, indicating that there is always an overlap between the source and target distributions. As seen in the previous sections, more similar results lead to improved transfer results. This is indeed reflected in the RMSE and MAPE, where especially for the RMSE, the transfer results almost always improve the results, except at some point for a negative shift with a smaller variance. The conclusions from the results are very similar to the symmetric case where the target distributions were both shifted and scaled, except that the case where the distributions are non-overlapping does not occur. This summary is given in Table 3.2.

Instead of scaling the variance by $\sigma^2 \cdot c$, it will be scaled by σ^2/c where $c \in [1.5, 0.5]$. This leads to smaller variances for more positively shifted target distributions and the other way around. The prediction results for this simulation using a polynomial kernel are shown in Figure 3.19.

These results are also similar to the previous case and the symmetric case. It is notable, that for

	Similar distributions	**Dissimilar but overlapping distributions**	**Very different, non-overlapping distributions**
Symmetric source samples with shifted target distributions	Both transfers improve upon the results without transfer	At least 1 transfer method improves upon the results, the other becomes worse than NoTL at some point	-

Table 3.2: Summary of the results for the skewed case with shifted/scaled target distributions



(a) Source distribution with different target distributions for 1 source sample

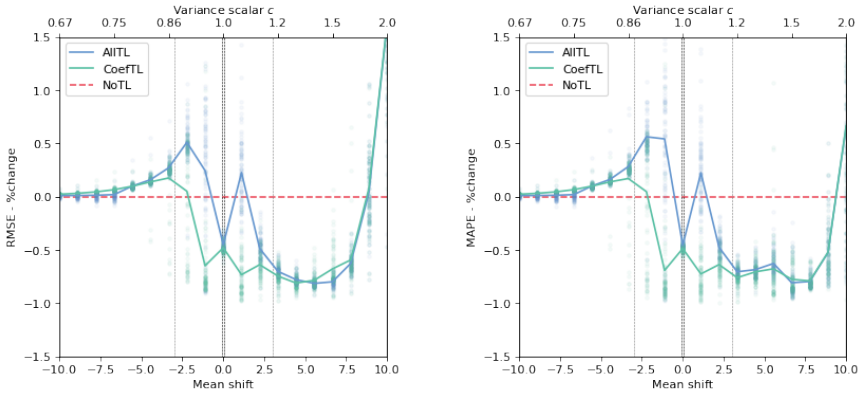
(b) Relative improvement in RMSE vs. mean shift

(c) Relative improvement in MAPE vs. mean shift

Figure 3.19: Average prediction results for 100 skewed source samples with different target samples for *NoTL*, *AllTL* and *CoefTL* with polynomial kernel, varying means and variances

the skewed distributions, the *CoefTL* is generally better than the *AllTL* method, whereas in the symmetric case, this was the other way around. This again shows that the *CoefTL* appears to be more powerful for skewed data, whereas the *AllTL* method is more powerful for more symmetric data.

Varying means and variance for extremely different skewed distributions Finally, as in the previous cases, it is not clear what happens for very different distributions. Therefore, the variance will be scaled similarly as in the previous case, but the mean will be shifted linearly in the range $[-10, 10]$. These results are given in Figure 3.20.



(a) Relative change in RMSE vs. mean shift and variance, polynomial kernel

(b) Relative change in MAPE vs. mean shift and variance, polynomial kernel

Figure 3.20: Average prediction results for 100 skewed source samples with very different target samples for *NoTL*, *AllTL* and *CoefTL*

It can be seen that for this case, the results behave asymmetrically. For a negative shift, the transfer, first of all, becomes worse, as was seen before for non-overlapping distributions, but becomes comparable to the *NoTL* method at some point. For a positive shift, however, the results of at least one transfer method outperform the results without transfer and only at some point do the transfer methods become worse.

Conclusion for negatively skewed source and target distribution with varying means and variance The results for the skewed case are very similar to those for the symmetric case. A difference is that for the skewed case, the *CoefTL* method gave better results whereas, for the symmetric case, the *AlLTL* method was optimal. It should generally be noted that it appears that the transfer generally improves upon the results whenever there is more overlap between the source and target samples. Since the skewed source and target distributions have more overlap compared to their symmetric version with similar mean and variance for the target data, the transfer results are overall relatively better for the skewed case compared to the prediction results of the *NoTL* method. When distributions don't overlap anymore, the results are inconsistent and therefore hard to interpret.

3.1.4 Conclusion of the simulations for the univariate case

In the first part of this section, an elaborate analysis was executed for 2 cases where samples were generated from either 1 or 2 univariate normal distributions. From this, the transfer could be analyzed both before and after the transfer. These results show that transfer learning can improve results, even when source and target data are very similar and it was not expected that the transfer would give improvements. Not passing the test after transfer does not necessarily imply an unsuccessful transfer as it is important to consider the overall distribution and not just the marginals when interpreting Zk-test results. Indicating a successful transfer using some kind of metric appears overall to be challenging and could be stated as a limitation of TCA.

In the second part of this section, an extensive analysis was done in order to see the impact of changes in mean and variance on the performance of the transfer methods. This was done both for symmetric and skewed source and target distributions and repeated 100 times in order to secure robust results. These results are similar as in Table 3.1 and again summarized in Table 3.3. The summary of the results focuses on giving broader insights rather than delving into specific variable selection methods. This means that the emphasis is on identifying general patterns and trends, not specific details.

	Similar distributions	**Dissimilar but overlapping distributions**	**Very different, non-overlapping distributions**
Symmetric/skewed source samples with shifted target distributions	Both transfers improve upon the results without transfer	At least 1 transfer method improves upon the results, the other becomes worse than NoTL	At some stage, a negative shift will result in poorer transfer outcomes, while a positive shift will enhance transfer performance in at least one instance.

Table 3.3: Summary of the results for all univariate cases with shifted/scaled target distributions

Only for the skewed case in Figure 3.16, where the means were shifted and the variance was fixed, it was seen that when there is a small overlap between the source and target distribution for a negative shift, the transfer led to slightly worse results. Some additional observations are also relevant to mention, namely:

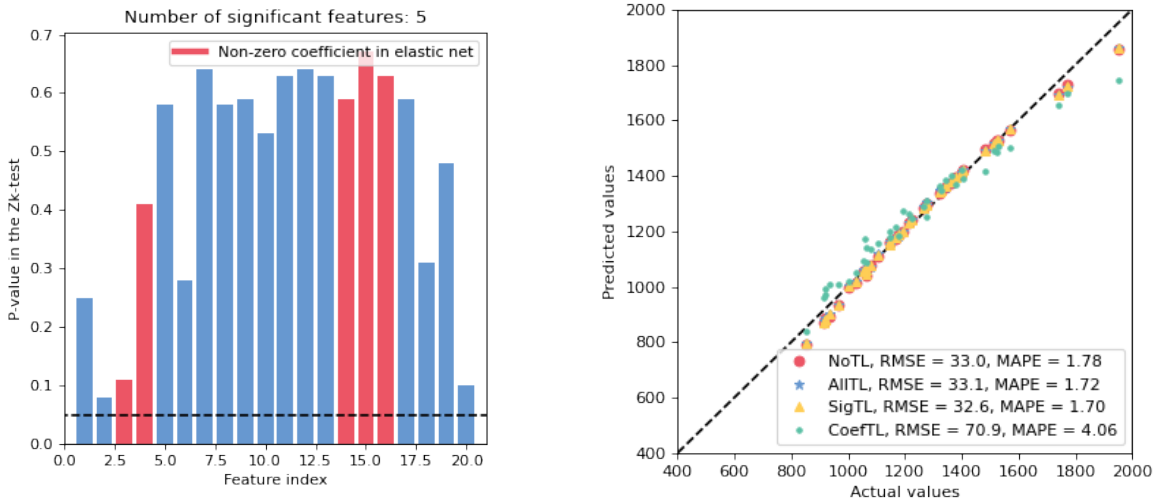
- **Kernel choice sensitivity:** The results of the study or analysis are highly dependent on the choice of kernel. This means that the outcomes can vary significantly based on the type of kernel used in the analysis.
- **Zk-test:** The Zk-test is more powerful for detecting mean differences between the source and target distributions compared to differences in variances. Whenever there are more observations, the Zk-test becomes more deterministic and the performance of the transfer can be more aligned with the conclusions stated in Table 3.3.
- **RMSE and MAPE trade-off:** A trade-off can be observed between the RMSE and the MAPE. In some cases, an improvement in RMSE may result in a worsening of MAPE, and vice versa. This trade-off highlights the importance of considering both measures when evaluating the performance of a model.
- **CoefTL vs. AllTL:** In the univariate case, it appears that for the symmetric data, the *AllTL* method performs generally best while for skewed data, the *CoefTL* performs best.

- **Asymmetry in the label data** It is noteworthy that despite the shifts being symmetrical, the outcomes exhibit an asymmetric pattern. This pattern is observable even in situations where no transfer of learning occurs, indicating that multiple factors contribute to this phenomenon. One such factor is the 5-degree polynomial relationship between the label and feature data. As a result, target distributions that are closer to 0 have smaller differences in their label data, while those that are farther from 0 have more substantial differences between their labels. This observation explains why the model's RMSE is higher for a negative shift, as it is less sensitive to these larger differences. However, this heightened sensitivity can also lead to a bias, which accounts for the MAPE becoming significantly larger for a positive shift with less noticeable differences in the label data. This is also in line with the observed RMSE and MAPE trade-off mentioned above.

3.2 Multivariate scenarios

In this section, similar simulations will be applied and evaluated for the multivariate case. Again, the goal is to simulate data which is related to the data from the case study. First, the scenario where source and target data are from the same distribution will be simulated and evaluated in Subsection 3.2.1. Then in Subsection 3.2.2, the target marginal distribution of 1 significant feature will be drawn from a different distribution so that the marginal distributions for the source and target data of that feature are different. Finally, in Subsection 3.2.3, the source and target data will be sampled from 2 different multivariate normal distributions and results will be generated and evaluated. Note that for this multivariate case, the simulation study becomes more complex and harder to interpret. It is difficult to determine whether the assumptions for transfer learning are satisfied and to assess the impact of the transfer on the results. Therefore, the analysis will be less extensive for this multivariate scenario compared to the univariate study.

3.2.1 Source and target data from 1 multivariate normal distribution



(a) P-values of the Zk-test for all features without transfer

(b) Predicted values for all variable selection methods

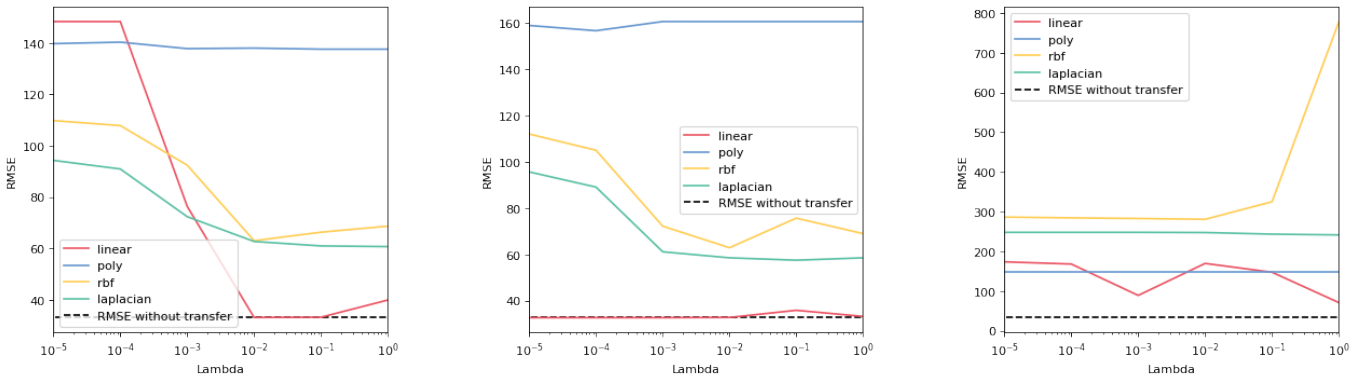
Figure 3.21: Zk-test and predictions for similar source and target data

In this section, the source and target data will be from the same source and target distribution. A source and target sample will be generated with 20 features and 40 instances each. These features have similar means and covariance as in the case study dataset and will be sampled from a multivariate normal distribution. By setting:

$$y_i = \frac{1}{4000}\overline{\boldsymbol{x}_{\boldsymbol{i}}^{\mathbf{5}}}, \tag{3.3}$$

where $\boldsymbol{x}_i \in \mathrm{R}^{20}$ and $\overline{\boldsymbol{x}}$ is the mean of the vector, the labels will be more or less similar to the case study. Note that by doing this, there is a polynomial relationship between the feature set and the labels.

In Figure 3.21a, the Zk-test results for the features before transfer are shown. As expected, all features pass the test so it can be assumed that the source and target data features are from the same distributions. Figure 3.21b implies that the results with transfer lead to higher RMSE and MAPE for *CoefTL*. From the predicted values by this method, it is also seen that lower values are overestimated by this method while higher values are underestimated. The results from the *AllTL* are more or less similar to the results of *NoTL*. The *SigTL* method does give slightly better prediction results, although this is not directly visible from the predictions in Figure 3.21b. It is notable that these results are different from the univariate case, where it was seen that the transfer could still improve the results even when source and target distributions are similar. This was particularly because the transfer was able to give more accurate predictions for values in the tail. In this multivariate case, similar results are however not observed.

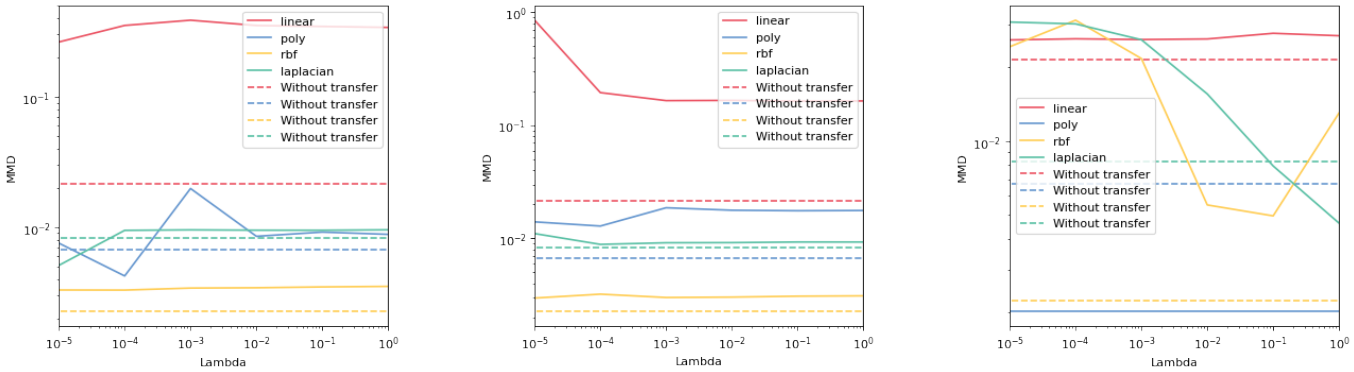


(a) *AllTL* (b) *SigTL* (c) *CoefTL*

Figure 3.22: RMSE values of the predictions for different kernels and lambda

In Figure 3.22, the RMSE values for different kernels and lambdas are given for the different variable selection methods and in Figure 3.23, MMD values are given for all kernels and models before and after transfer.

For all three models, the linear kernel is in this multivariate case optimal. This is not in line with the MMD, which actually implies that the MMD increased after transfer. All other kernels also do not decrease after transfer for most values of lambda, which is actually in line with the prediction results. Consequently, it can be stated that the MMD remains an unreliable metric for determining a successful transfer. In summary, when the source and target data have the same distribution, the Zk-test confirms this for the marginal distributions. In this case, it is unlikely that transferring the data will improve the results, as seen in the results after the transfer. While the *SigTL* method slightly improves results, the improvements are minimal. On the other hand, the *CoefTL* method leads to the worst prediction results, indicating that transferring data when the assumptions of TCA do not hold can result in a negative transfer. Additionally, it is worth noting that the transfer methods are computationally more expensive. As a result of this simulation, it is recommended to use the *NoTL* method when the source and target distributions are very similar.

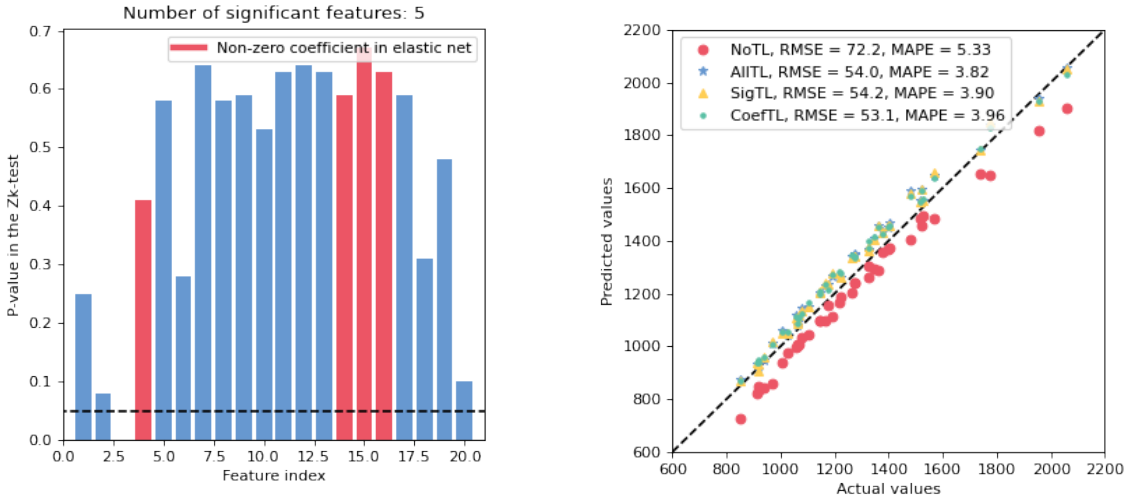


(a) *AllTL* (b) *SigTL* (c) *CoefTL*

Figure 3.23: MMD for different values of λ and different kernels, with and without transfer

3.2.2 Changing 1 marginal of a significant feature in the target data

Changing the first feature that is significant (feature 3) in the target data to a Gamma distribution with a shape equal to the absolute value of the mean of that feature and a scale equal to 1 lead to the results in Figure 3.24. Here, the label data for the target data is again calculated using Equation 3.3 using the new target data with the adapted marginal.



(a) P-values of the Zk-test for all features without transfer

(b) Predicted values for all variable selection methods

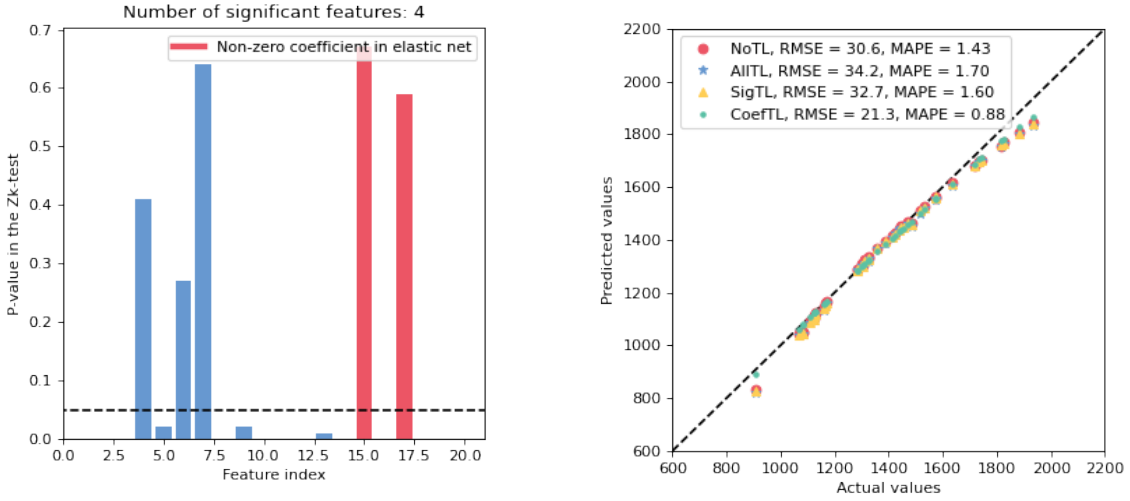
Figure 3.24: Zk-test and predictions for source and target data with 1 differing marginal

Indeed, the third feature does not pass the Zk-test anymore. Since this is 1 out of the 4 features, it seems to give too much noise which influences the predictions negatively. The transfer methods improve these predictions. Where the *NoTL* method mainly underestimates the actual values, the transfer methods slightly overestimate the results but are slightly more accurate. This is also reflected by the MAPE. For the tail values, the transfer methods give much more accurate results, which is reflected by the RMSE.

From the Zk-test, it follows that all methods generate 2 significant transfer components which both pass the test. Also, the relationship of the RMSE and the choice of kernel and λ is for all three methods with transfer more or less similar and all methods prefer the polynomial kernel, with various values of λ. Therefore, it is not surprising that all three methods give almost similar prediction results. The *CoefTL* is slightly more accurate, especially for higher values, which is reflected by the RMSE, whereas the *AllTL* method gives on average more accurate results, reflected by its MAPE.

In conclusion, even changing only 1 marginal can influence the results a lot. The transfer is able to overcome this and still gives good results.

3.2.3 Source and target data from 2 multivariate normal distributions



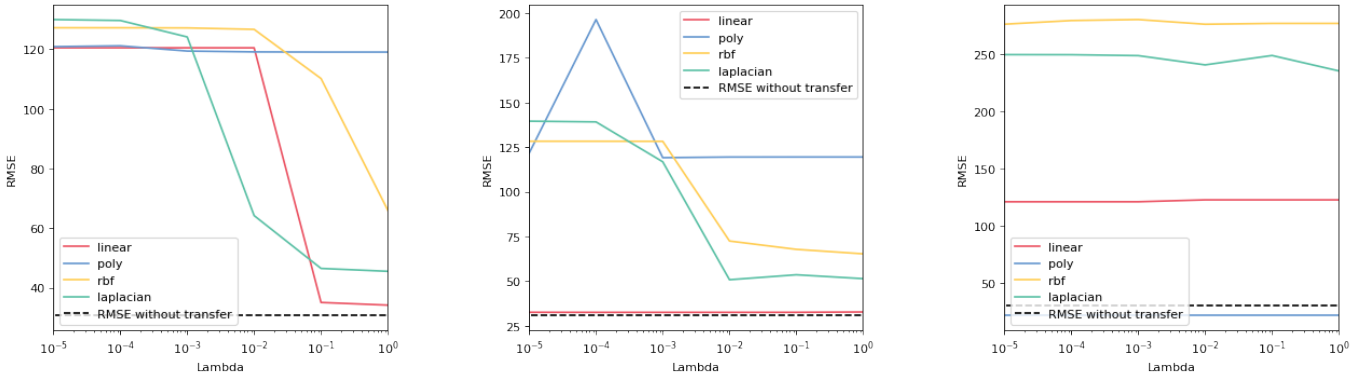
(a) P-values of the Zk-test for all features without transfer

(b) Predicted values for all variable selection methods

Figure 3.25: Zk-test and predictions for different source and target data

Instead of sampling the source and target data from 1 distribution, 2 distributions will be used. Here, 40 source samples are generated from a multivariate normal distribution with similar means and covariance as in the feature set from batches 1 and 2. Also, 40 target samples are generated from a multivariate normal distribution with similar means and covariance as the data from batch 3. Labels are generated by again fitting a linear regression model for the source and target data separately, and using the coefficients in order to calculate the labels. This means again that there is a polynomial relationship between the feature set and the labels.

In Figure 3.25a, the Zk-test results for the features before transfer are shown. In this case, 4 features are selected as significant by the elastic net model, where only 2 of them pass the Zk-test. This means that 50% of the included features come from different marginal distributions according to the tests. From Figure 3.25b, it becomes clear that only the *CoefTL* method improves upon the results without transfer. It is notable that the *AllTL* and *SigTL* methods give similar results slightly worse than the *NoTL* method.



(a) *AllTL* (b) *SigTL* (c) *CoefTL*

Figure 3.26: RMSE values of the predictions for different kernels and lambda

In Figure 3.26, the RMSE values for different kernels and lambda for the different variable selection methods. The *AllTL* and *SigTL* both give optimal results using a linear kernel with λ near 1 for the *AllTL* method and all values of λ for the *SigTL* method. In the *AllTL* method, this leads to 5 significant transfer components of which 2 do not pass the Zk-test. For the *SigTL* method, the 4 transfer components corresponding to the smallest eigenvalues are significant of which 2 do not pass the test. Since in both cases 1 transfer component does not pass the test, a successful transfer is not necessarily implied. The *CoefTL* method prefers the polynomial kernel with various values for λ. Here, 3 transfer components appear to be significant in the elastic net model. In this case, only 1 component passes the Zk-test.

In conclusion, only the *CoefTL* method was able to improve upon the prediction results. Perhaps, 'focusing' the transfer by first selecting and weighing the important features can help recognise complex underlying patterns which leads to better results. This is especially relevant when the feature set contains many features.

3.2.4 Conclusion of the multivariate scenarios

In general, it is hard to determine when the transfer can improve the results for a multivariate dataset. Since the Zk-test is defined for marginals and not joint distributions, it is not a sufficient metric in order to determine whether the assumptions for transfer learning are satisfied. Furthermore, it is challenging to determine when the transfer was actually successful and led to improved results. When there are more extreme values, the transfer seems to be able to give more accurate predictions, especially for these values. To fully analyze the effectiveness of transfer learning in multivariate cases, it may be necessary to use a more comprehensive goodness-of-fit test that is specifically suited for multivariate data.

3.3 Conclusion of the simulation results

The simulation results show a few important conclusions. For univariate cases, the transfer can improve results for both similar and dissimilar distributions but can lead to worse results only when the distributions are very different. This suggests that transfer methods can *relax* the strict assumption of traditional machine learning that the source and target samples must come from the same distribution.

However, transfer learning has limitations. It is challenging to find a suitable metric for determining a successful transfer. MMD before and after the transfer has no clear relationship with transfer performance, while Zk-test is effective for detecting distribution differences but is limited to marginal distributions and has reduced sensitivity to differences in variance.

It was furthermore observed that the transfer methods were particularly effective for predicting tail values. This is also reflected by the RMSE, which punishes bad predictions for outliers more compared to the MAPE. A trade-off can be observed between the RMSE and the MAPE. In some cases, an improvement in RMSE may result in a worsening of MAPE, and vice versa. This trade-off highlights the importance of considering both measures when evaluating the performance of a model.

Finally, the results of the study or analysis are highly dependent on the choice of kernel. This means that the outcomes can vary significantly based on the type of kernel used in the analysis.

When it comes to multivariate cases, determining the success of transfer learning becomes even more difficult as the metrics used in univariate cases, such as MMD, are not reliable. The Zk-test, which was effective in detecting distribution differences in univariate cases, is limited in multivariate cases as it is only defined for marginal distributions.

To fully evaluate transfer learning in multivariate cases, a more comprehensive goodness-of-fit test that specifically handles multivariate data may be required. This would help to better understand the success of transfer in such cases and provide a clearer picture of the performance.

Despite these challenges, the results show that transfer learning can still be effective in predicting tail values in multivariate scenarios. This suggests that transfer methods may be useful in improving predictions, even in complex and multi-dimensional situations. However, a more thorough evaluation of transfer learning in multivariate cases is necessary to fully understand its effectiveness.

4 | Case Study

In this chapter, the results are presented according to the framework explained in Chapter 2. In order to have multiple results, 7 different scenarios are used. These scenarios are explained in Section 4.2. From then, the results will be given and analyzed for all scenarios. Each section includes all parts of the framework, namely the Zk-test results on the source and target marginal distributions before transfer and the significance of the features in the elastic net model with the corresponding prediction results.

4.1 Data description

In this section, an overview is given of the data which is used in this thesis for the case study. Furthermore, it is shown that early cycle data is used while degradation patterns are not present yet. By extracting and transforming this data and calculating summary statistics such as mean and variance, it is shown that this leads to a suitable feature set which is highly correlated with the battery lifetimes. In Subsection 4.1.1, a description is given of what can be found in the dataset from the case study. This is followed by a description of the feature-based approach in Subsection 4.1.2. In this final section, the pre-processing of the data is done such as smoothing specific features and taking out outliers. The original source of the data can be found here[1].

4.1.1 Data overview

The provided dataset from the case study [37] contains 3 batches with in total 124 (41/43/40 resp. in batch 1/2/3) lithium-ion phosphate/graphite (manufactured by A123 Systems model APR18650M1A, 1.1 Ah nominal capacity) batteries cycled with varying fast-charging conditions in a temperature controlled room. The discharge conditions however are identical for all batteries (4C to 2.0V, where 1C is 1.1A). In each batch, approximately 48 battery cells can be found. An important note is that batches 1 and 2 were generated from the same experiment, while batch 3 was generated independently. For each battery, the following data is attached:

- **Cycle Life:** The number of cycles until the battery's capacity has decreased below 80% (and for batch 2, 75%) of its nominal capacity is calculated by multiplying the discharge current (measured in Amperes) by the discharge time (measured in hours).
- **Charge policy:** All batteries are charged according to a specific charging policy. These policies are defined for each battery using the format 'C1(Q1)-C2', where C1 represents the first constant current-constant voltage (CC-CV) step until a certain state-of-charge (SoC) percentage, Q1, is reached. From this point, the charging continues at a current of C2 until the SoC

[1] https://data.matr.io/1/

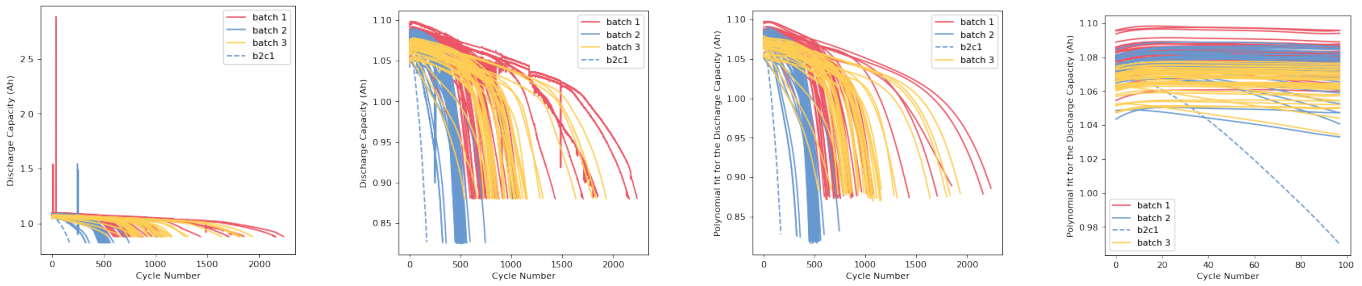
reaches 80% (0.88 Ah). After this, the batteries are allowed to rest for a brief period (ranging from a few seconds to slightly over 1 minute, depending on the batch) before being charged with 1C CC-CV. The manufacturer recommends a fast-charging policy protocol of 3.6C.

- **Summary Data:** The summary data contains information for each cycle, including the cycle number, discharge capacity, charge capacity, internal resistance, maximum temperature, average temperature, minimum temperature, and charging time.
- **Cycle Data:** The information contained within a cycle includes the time (t), charge capacity (Qc), current (I), voltage (V), temperature (T), and discharge capacity (Qd). Additionally, calculated variables such as the discharge rate dQ/dV, the discharge capacity interpolated linearly (Qdlin), and temperature interpolated linearly (Tdlin) are also included.

4.1.2 Formatting the data features

For the feature-based approach, several features are extracted and transformed so that they are highly correlated with the labels and therefore suitable for predicting the cycle life. In this section, the extraction and transformation of these features are described. Since the same feature set will be used as in [37], the background is provided to show why this set is chosen and suitable for prediction.

Since the lifetime of a battery is defined by its discharge capacity, it makes sense to zoom in on this feature. For each cycle, the discharge capacity for that cycle is given. From the plot in Figure 4.1a, it can be seen that some curves have major outliers. From the type of battery, it is known that the capacity can not be higher than 1.1 Ah, which gives an upper bound for these curves. Taking out these outliers leads to the curves in Figure 4.1b. Finally, fitting a polynomial on this data gives the results in Figure 4.1c, which will be used for the final feature extraction and model features.



(a) Discharge capacity curves without pre-processing

(b) Discharge capacity curves with left out points that included a value above the upper bound of 1.1 Ah

(c) Discharge capacity curves with polynomial fits degree 5

(d) Discharge capacity curves with polynomial fits degree 5, over the first 100 cycles

Figure 4.1: Discharge capacity curves for all cycles and all batches, without and with pre-processing

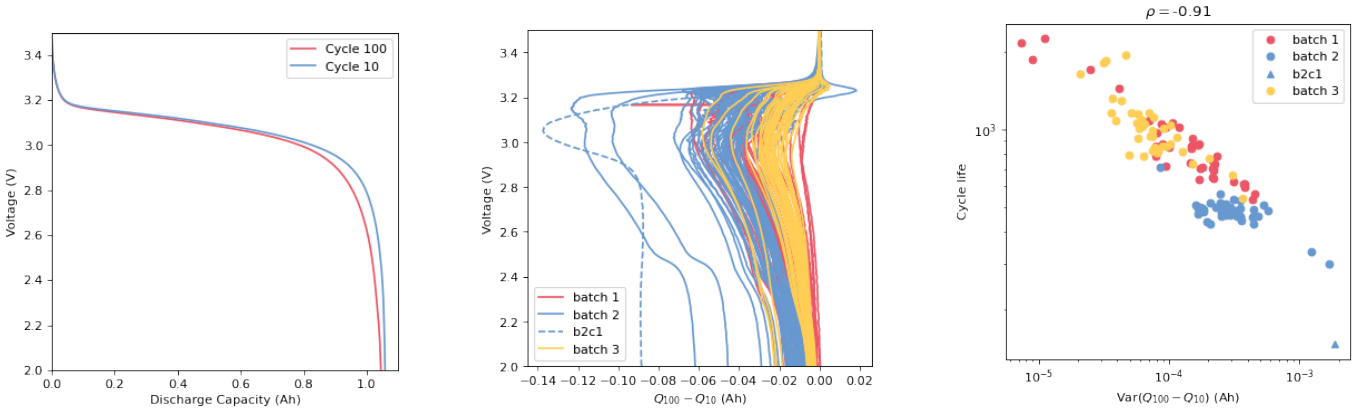
Zooming in on the first 100 cycles gives the discharge curves as a function of the cycles in Figure 4.1d. In this figure, it can be seen that the capacity fade is negligible in the first 100 cycles. This is supported by the weak correlations between the cycle life and several derivatives of the discharge

curves within the first 100 cycles. Therefore, additional features are needed which have a strong correlation with the end-of-life cycle life.

Many features proposed by the case study are based on domain knowledge of lithium-ion batteries such as *initial discharge capacity*, *charge time* and *cell can temperature*. Furthermore, several features are derived from the so-called discharge voltage curve: within each cycle, the discharge capacity values are given during the entire cycle (charging and discharging) and can be evaluated as a function of voltage $Q(V)$. In order to be able to use this data, linear interpolation is fitted and evaluated on a range of 1000 equally spaced points from 3.5 V to 2.0 V. These curves and their derivatives are a great source for degradation diagnosis. Define $\Delta Q_{100-10}(V)$ as the difference between the discharge voltage curves of cycles 100 and 10. Since Q_{100-10} should be below 0, all values which are interpolated above this value are eliminated. Figure 4.2a shows these two curves for a representative cell (b2c10) and gives clarity on the difference between the two curves. Plotting these differences as a function of voltage gives the curves in Figure 4.2b.

Note that in some of these curves, the difference between the discharge capacity for cycle 10 and cycle 100 is above 0, while this is not realistic, since some degradation should have occurred after 90 cycles. This could be a consequence of not accurate interpolation or noise in the measured data. However, deleting these values leads to a lower correlation with the lifetimes. The case study plotted a similar figure, where these values were deleted while keeping a high correlation with the lifetimes. Therefore, it is not clear if they did or did not include these positive values in their variance feature. In this thesis, it is decided to leave the values in the dataset in order to have a similar correlation as in the case study.

Finally, summary statistics can be calculated from these curves such as *minimum*, *mean*, *variance*, *kurtosis* and *skewness*. The log variance of $\Delta Q_{100-10}(V)$ is plotted against the log cycle lives in Figure 4.2c. With a correlation of -0.91, the variance feature can contribute as a useful feature for model prediction.



(a) Discharge voltage curves for the 100th and 10th cell for a representative cell (b2c10)

(b) Discharge voltage curves for the 100th and 10th cell for all cells

(c) Variance of $\Delta Q_{100-10}(V)$ against battery cycle life for all cells

Figure 4.2: Discharge voltage curves and the variance feature

Since the features based on $\Delta Q_{100-10}(V)$ have high predictive power, three different models will be

investigated:

- *Variance model*: Only uses $\log(\text{Var}(|\Delta Q_{100-10}(V)|))$ to predict log cycle lives.
- *Discharge model*: Next to (the logarithm absolute value of) summary statistics of $\Delta Q_{100-10}(V)$ such as minimum, mean, variance, skewness, kurtosis and $\Delta Q(V = 2)$, additional candidate features obtained during discharge are used. These are the slope and intercept of the linear fits for the capacity fade curve cycle 2 to 100 and cycle 91 to 100. Furthermore, the discharge capacity at cycle 2 ($Q(n = 2)$) and 100 ($Q(n = 100)$) are included and the difference between the maximal discharge capacity for a cell and cycle 2 ($\max_n(Q(n)) - Q(n = 2)$).
- *Full model*: For this model, additional features are used from different data streams such as temperature and internal resistance. From the temperature curves cycle 2 to 100, the maximum and minimum are used and the integral over time for these cycles is used. This integral of temperature over time is not clearly defined. For this thesis, it is interpreted as the sum of the difference between the interpolated temperature data in the 100th and 2nd cycle divided by 1000 (so the average difference):

$$\text{Temperature integral, cycle 2 and 100} = \frac{1}{1000}\sum_{i=1}^{1000}(T(c = 100) - T(c = 2))_i.$$

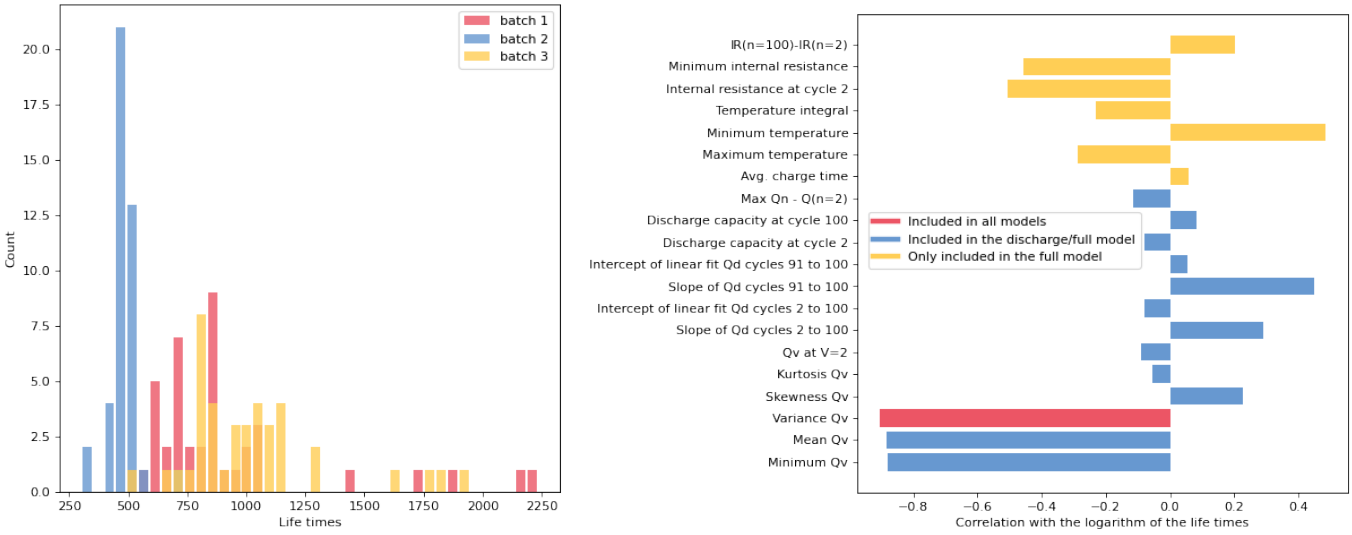
  Furthermore, the full model includes the average charge time of the first 5 cycles, the internal resistance at cycle 2, the minimum internal resistance for cycles 2 to 100 and the difference between cycle 100 and cycle 2.

For the full model, derivatives of the temperature and internal resistance are used for prediction. The temperature is given as measured and interpolated over 1000 equal-spaced points. In the summary statistics, the minimum, maximum and average values of the temperature are provided. The temperature of the chamber is forced to 30 degrees Celsius. We expect the measured temperate to be around this number as well. Not including the temperature of 'cycle 1' and setting a lower bound of 25 degrees Celsius gives a suitable dataset for prediction (especially for the first 100 cycles).

The internal resistance is only given in the summary statistics for each cycle and each battery. These measurements were obtained during charging at 80% SOC by averaging 10 pulses of ±3.6C with a pulse width of 30 ms (batch 1 and 2) or 33 ms (batch 3). Here are some outliers towards 0, which gives reason to again state a lower bound of 0.005.

Deletion of battery 'b2c1' In the figures, this battery has been highlighted several times. As this battery rapidly reaches an 80% state of health (after 148 cycles), it could be taken out of the dataset as it gives too much noise to the results. This is also suggested in the case study.

In Figure 4.3a, a histogram of the label data can be found where the colours indicate different batches. Cycle lives vary from around 300-2200. The cycle life of the batteries in batch 1 and batch 3 are for most batteries higher compared to the batteries in batch 2. Finally, insight is given into the relationship between the labels (log cycle life) and the features. Therefore, the correlations between the labels and features are plotted in Figure 4.3b. From the figure, it becomes clear that the minimum, mean and variance of $\Delta Q_{100-10}(V)$ are highly correlated with the log lifetimes. Other features have correlations up to ±0.5.



(a) Histogram of the label data (Cycle life), batches are indicated by colour

(b) Correlations of the features with the log lifetimes

Figure 4.3: Label data histograms and correlation analysis between the labels and features for all batches

4.2 Scenarios for source and target data

In this section, 7 scenarios for source and target data are used and compared. These scenarios are:

- **Scenario 1:** Source/train set: Batch 1, Target/test set: Batch 2
- **Scenario 2:** Source/train set: Batch 1, Target/test set: Batch 3
- **Scenario 3:** Source/train set: Batch 2, Target/test set: Batch 3
- **Scenario 4:** Source/train set: Batch 1 & 2, Target/test set: Batch 3
- **Scenario 5:** Source/train set: Original train set (from the article), Target/test set: Original primary test set (from the article)
- **Scenario 6:** Source/train set: Original train set (from the article), Target/test set: Original secondary test set/Batch 3 (from the article)
- **Scenario 7:** Source/train set: Original primary test set (from the article), Target/test set: Original secondary test set/Batch 3 (from the article)

Note that the source data is always used as the train sort and should be labelled data, whereas the target set is also the test set.

A visualization of the different scenarios is shown in Figure 4.4. Note that this is more an intuitive figure than an actual description of the scenarios as the number of cells in each batch is not corresponding to the actual number of cells.

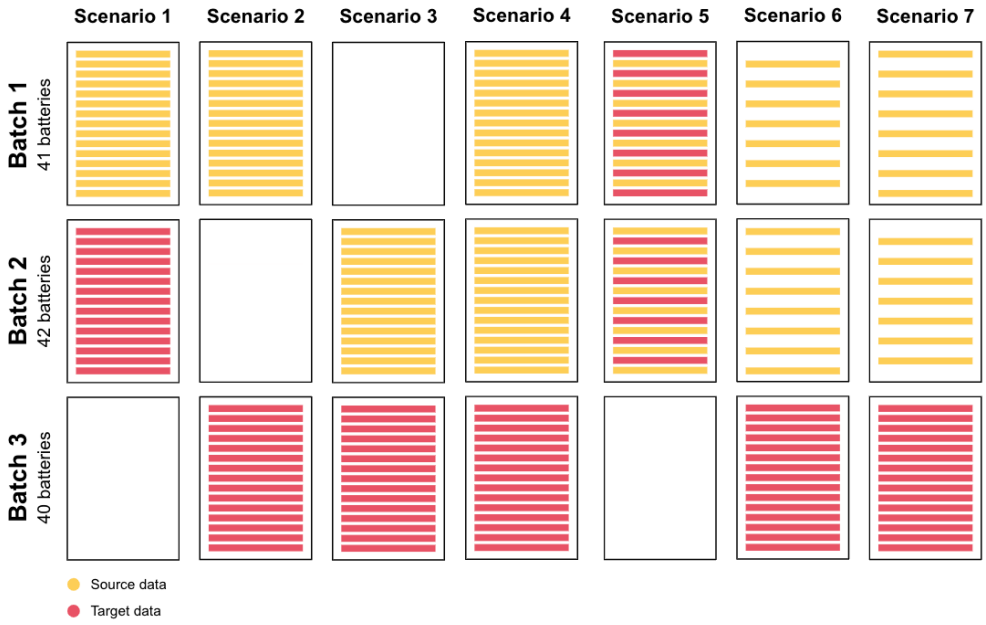


Figure 4.4: Different scenarios for source and target data

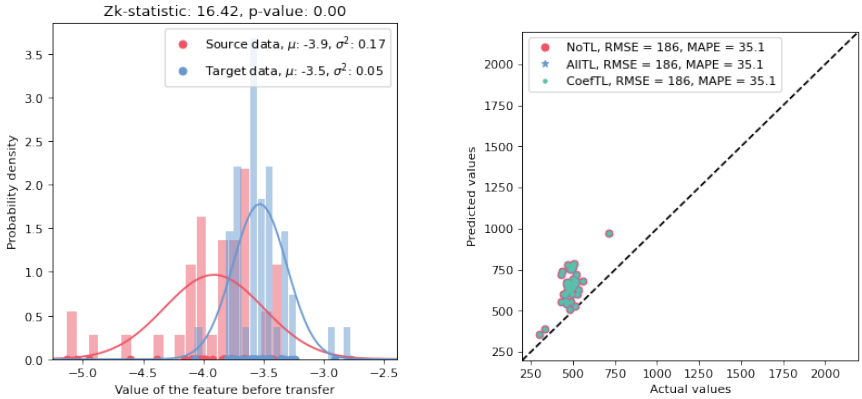
4.3 Case study results

This section applies the framework to the case study data. Since the variance model is a univariate model, its results are easier to understand and align better with the simulated examples. Consequently, a thorough explanation will be provided in this case. The results of the discharge and full model will be presented without in-depth analysis, as the multivariate nature of these models makes their results more difficult to interpret.

The variance model consists of 1 feature which is highly correlated with the labels. Note that for 1 feature, there is no difference between the *AllTL* and *SigTL* method, thus the *SigTL* method is left out for the variance model. For the discharge and full model, all variable selection methods are used and it can be expected that the difference between the methods, in this case, becomes clearer.

It's crucial to note that selecting the right kernel and lambda can be a difficult task, which is not covered in this research. Instead, in most cases, the kernel and lambda will be picked based on the lowest RMSE. However, as the simulations show, there may be a trade-off between RMSE and MAPE, implying that this approach to hyperparameter selection can negatively impact the MAPE. Therefore, when the lowest RMSE leads to a negative impact on the MAPE, in some cases also other values will be checked which can lead to a slightly higher RMSE but also a better MAPE. This will also be stated in the description of that model. The main reason to do this is to show that a transfer is able to improve the results.

4.3.1 Scenario 1



(a) Distribution differences between source and target data with a normal fit, variance model

(b) Prediction results for the variance model

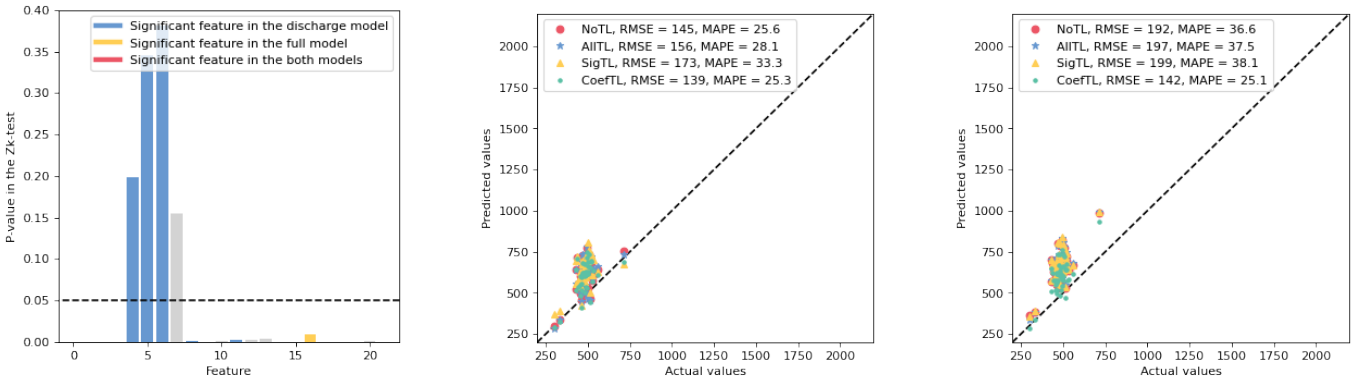
Figure 4.5: Distribution differences and prediction results for the variance model, scenario 1

Scenario 1 consists out of batch 1 as the source set and batch 2 as the target set. From the prediction results without transfer, it was seen that the labels are concentrated around similar values, which are generally overestimated by the elastic net model. In Figure 4.5, the distributions with normal fits are shown for the source and target data and the prediction results for the different transfer methods are given.

When the Zk-test is applied to the source and target data of scenario 1, the p-value implies that the null hypothesis that the source and target data come from the same distribution should be rejected. This shows that in batches 1 and 2, which are from the same experiment, a difference in distributions can be observed. However, this is not surprising as the charging conditions for these 2 batches varied as well. The *NoTL*, *AllTL* and *CoefTL* provide equivalent predictions. In the simulation study, it was seen that when distributions are different, but overlapping, at least 1 transfer can still improve the results. However, given that a linear kernel was selected in this case as the optimal kernel, the difference in results compared to the simulation, which predominantly relied on a polynomial kernel, is understandable. In fact, it was noted that with a linear kernel, the results from the three methods can be quite comparable, which aligns with the findings of this case study.

The Zk-test and prediction results for the discharge and full model are visualized in Figure 4.6. From the Zk-test, it can be seen that not all features pass the Zk-test, indicating distribution differences. Furthermore, the variance feature is the only feature that is significant in both the discharge and full model. The elastic net in the discharge model selects 8 features as significant, where 3 pass the Zk-test. The full model selects 4 features of which 0 pass the Zk-test. This could be seen as an indication that in the discharge model, the source and target distribution are slightly more similar compared to the full model. This could explain why the transfer improves the results of the full model relatively more compared to the results of the discharge model.

For the discharge and full model, the *CoefTL* is the only variable selection method that improves upon the results without transfer. This highlights the importance of selecting and weighing features



(a) Zk-test results, the discharge and full model have 8 and 4 significant features resp.

(b) Prediction results for the discharge model

(c) Prediction results for the full model

Figure 4.6: Zk-test results before transfer and the predictions for scenario 1, multivariate models

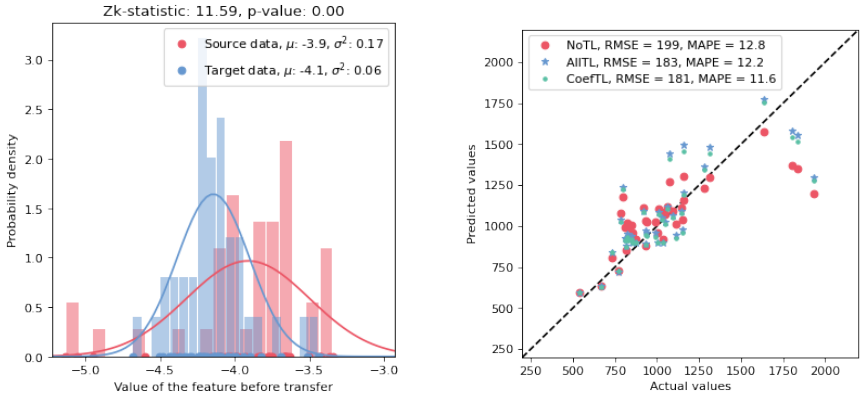
before transferring them in the multivariate scenario. Furthermore, the label with a higher value is accurately predicted by all methods in the discharge model but overestimated in the full model. The results for all models are summarized in Table 4.1.

	RMSE NoTL	RMSE AllTL	% Change AllTL	RMSE SigTL	% Change SigTL	RMSE CoefTL	% Change CoefTL	MAPE NoTL	MAPE AllTL	% Change AllTL	MAPE SigTL	% Change SigTL	MAPE CoefTL	% Change CoefTL
Variance model	186	186	0.00%	-	-%	186	0.00%	35.1	35.1	0.00%	-	-%	35.1	0.00%
Discharge model	145	156	**7.59%**	173	**19.31%**	139	-4.14%	25.6	28.1	**9.77%**	33.3	**30.08%**	25.3	-1.17%
Full model	192	197	**2.60%**	199	**3.65%**	142	-26.04%	36.6	37.5	**2.46%**	38.1	**4.10%**	25.1	-31.42%

Table 4.1: Summary of the prediction results for the variance, discharge and full model for scenario 1

4.3.2 Scenario 2

A similar analysis as the previous section will be applied to the second scenario. This scenario uses batch 1 as the source set and batch 3 as the target set. In Figure 4.7, the source and target data are shown with normal plots and the prediction results are given for the variance model. The Zk-test's

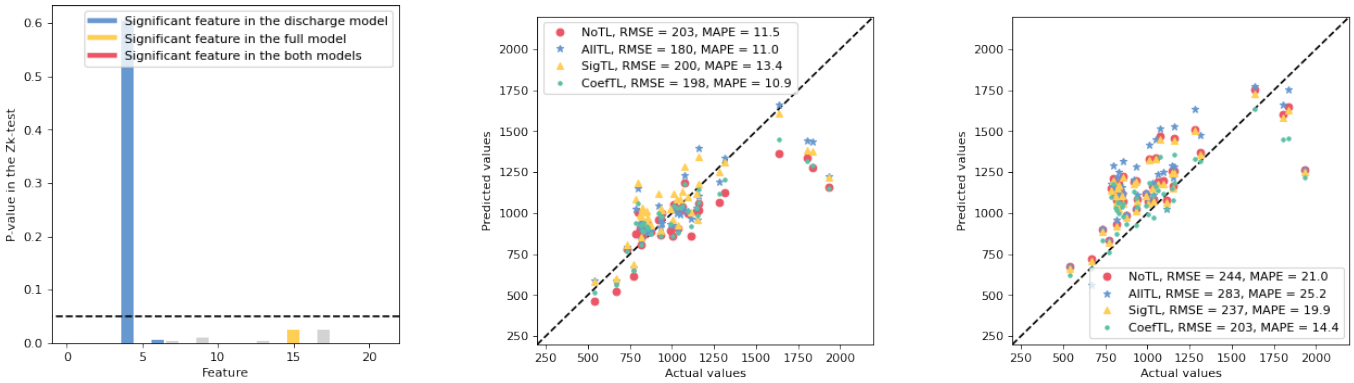


(a) Distribution differences between source and target data with a normal fit, variance model

(b) Prediction results for the variance model

Figure 4.7: Distribution differences and prediction results for the variance model, scenario 2

p-value indicates that the source and target data come from different distributions. This can be explained by the fact that batches 3 and 1 are from different experiments. In the simulation study, it was seen that for similar distributions, both transfer methods outperform the results without transfer. This is indeed in line with the results observed in Figure 4.7 for both the RMSE and MAPE.



(a) Zk-test results, the discharge and full model have 8 and 4 significant features resp.

(b) Prediction results for the discharge model

(c) Prediction results for the full model

Figure 4.8: Zk-test results before transfer and the predictions for scenario 2, multivariate models

The results for the discharge and full model are given in Figure 4.8. The Zk-test reveals that there are fewer features passing the test in the multivariate cases of scenario 2 compared to scenario 1, indicating greater differences in the source and target distributions. This aligns with expectations,

given that batch 3 is from a distinct experiment from batch 1 and batch 2. Transfer learning improves results in almost all cases, with the full model showing more improvement than the discharge model. This could be due to the fact that the source and target distributions of the full model are more dissimilar than those of the discharge model, as suggested by the Zk-test results for the significant features. Additionally, the full model has a larger number of features, but only 4 are considered significant, suggesting that the *CoefTL* method is effective in targeting the transfer, which may have accounted for the superior results

For the discharge model, the *NoTL* method appears to underestimate many values, while the transfer methods provide more accurate predictions. Notably, the *AllTL* method is optimal for this model, contrasting with the full model.

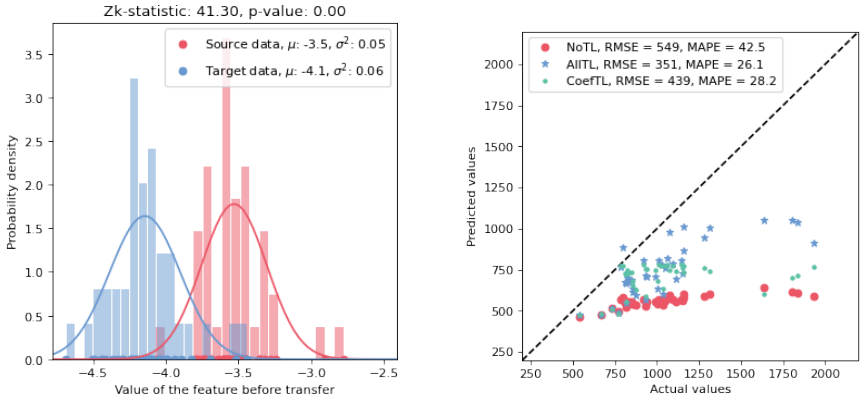
In the full model, without transfer, many average values are overestimated and higher values are underestimated. The *AllTL* method leads to even higher predictions for average values, resulting in improved results for the discharge model but worse results for the full model. The *SigTL* method gives similar predictions as the *NoTL* method, while the *CoefTL* method is particularly effective for average values.

In Table 4.2, the results are summarized for the variance, discharge and full model for scenario 2.

	RMSE NoTL	RMSE AllTL	% Change AllTL	RMSE SigTL	% Change SigTL	RMSE CoefTL	% Change CoefTL	MAPE NoTL	MAPE AllTL	% Change AllTL	MAPE SigTL	% Change SigTL	MAPE CoefTL	% Change CoefTL
Variance model	199	183	-8.04%	-	-%	181	-9.05%	12.8	12.2	-4.69%	-	-%	11.6	-9.38%
Discharge model	203	180	-11.33%	200	-1.48%	198	-2.46%	11.5	11.0	-4.35%	13.4	**16.52%**	10.9	-5.22%
Full model	244	283	**15.98%**	237	-2.87%	203	-16.80%	21.0	25.2	**20.00%**	19.9	-5.24%	14.4	-31.43%

Table 4.2: Summary of the prediction results for the variance, discharge and full model for scenario 2

4.3.3 Scenario 3

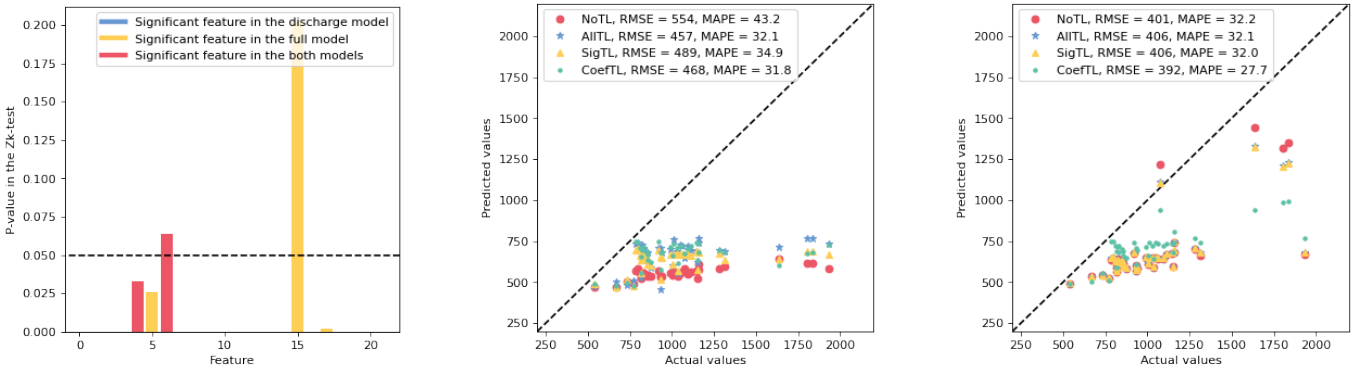


(a) Distribution differences between source and target data with a normal fit, variance model

(b) Prediction results for the variance model

Figure 4.9: Distribution differences and prediction results for the variance model, scenario 3

In scenario 3, batch 2 is used as source data and batch 3 is the target data. In Figure 4.9, the source and target distributions are shown for the variance model with the corresponding predictions. In this scenario, it is noteworthy that the variance of the source and target distributions are quite similar, but the mean of the target distribution has shifted negatively by 0.6. This shift still resulted in the fact that the source and target data are dissimilar, according to the Zk-test results. Both source and target distributions have positive skewness, with values of 1.02 and 0.71 respectively. This scenario resembles the negatively skewed source and target data scenario in the simulation study but is then mirrored. In the simulations, it was seen that for dissimilar, overlapping distributions, at least 1 transfer improves upon the results. In this case, both transfer methods improved upon the results. It is notable that the *AllTL* method is optimal in this case whereas, in the simulations, the *CoefTL* method gave better results for skewed data. The results after transfer do improve but remain for most values underestimated. For the discharge and full model, the Zk-test and prediction results



(a) Zk-test results, the discharge and full model have 5 and 10 significant features respectively

(b) Prediction results for the discharge model

(c) Prediction results for the full model

Figure 4.10: Zk-test results before transfer and the predictions for scenario 3, multivariate models

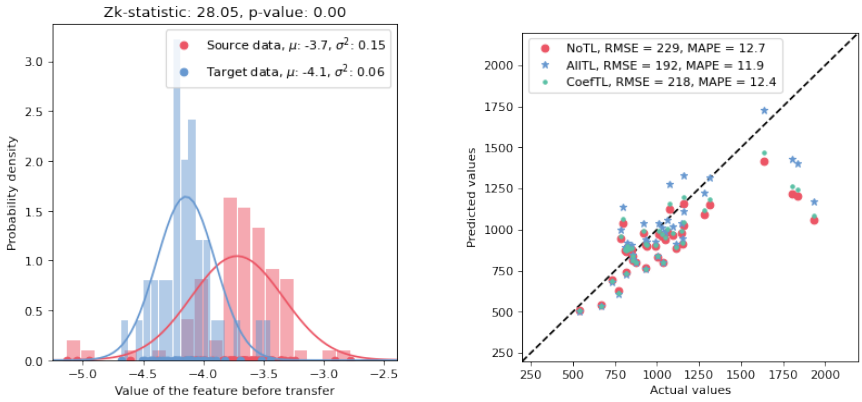
are given in Figure 4.10. It can be seen from the Zk-test that some of the features do not pass the Zk-test, indicating that the corresponding source and target samples are from different distributions. This is for the discharge model 1/5 significant components that do not pass the test, and for the full model 2/10. The results improve in both models, where for the discharge model, all transfer methods improve upon the results without transfer and the *AllTL* method is optimal. In the full model, the *NoTL* performs much better compared to the predictions in the other two models for the *NoTL* method. In this case, the transfer methods do improve for the MAPE, but for the RMSE, only the *CoefTL* is able to give better results. In this case, especially average values are predicted more accurately while tail values are underestimated. Still, the method gives overall better results according to the prediction metrics. A summary of the results is given in Table 4.3.

	RMSE NoTL	RMSE AllTL	% Change AllTL	RMSE SigTL	% Change SigTL	RMSE CoefTL	% Change CoefTL	MAPE NoTL	MAPE AllTL	% Change AllTL	MAPE SigTL	% Change SigTL	MAPE CoefTL	% Change CoefTL
Variance model	549	351	-36.07%	-	-%	439	-20.04%	42.5	26.1	-38.59%	-	-%	28.2	-33.65%
Discharge model	554	457	-17.51%	489	-11.73%	468	-15.52%	43.2	32.2	-25.46%	34.9	-19.21%	31.8	-26.39%
Full model	401	406	**1.25%**	406	**1.25%**	392	-2.24%	32.2	32.1	-0.31%	32.0	-0.62%	27.7	-13.98%

Table 4.3: Summary of the prediction results for the variance, discharge and full model for scenario 3

4.3.4 Scenario 4

In scenario 4, batches 1 and 2 are used as source data and batch 3 is the target data. In Figure 4.11, the source and target distributions are shown for the variance model with the corresponding predictions. The target data is negatively shifted and has a smaller variance. Since the skewed sample was



(a) Distribution differences between source and target data with a normal fit, variance model

(b) Prediction results for the variance model

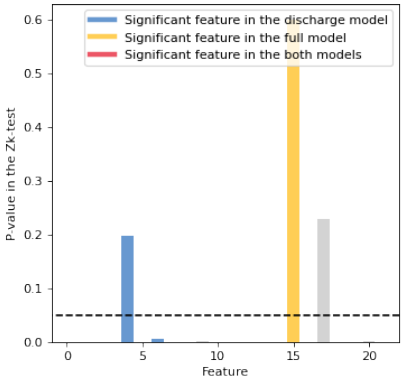
Figure 4.11: Distribution differences and prediction results for the variance model, scenario 4

generated by fitting a distribution on batches 1 and 2, this simulation is most in line with scenario 4. In this simulation, it was seen that for a target distribution that was negatively shifted and has a smaller variance, the results generally improve after transfer when distributions are different but still overlap. In the prediction results, it can be seen that the *NoTL* particularly gives bad predictions for the higher tail values. The transfer methods improve on these predictions but overestimate in some cases the average values. This is reflected by the prediction metrics, where mainly the RMSE improves and the MAPE stays behind. The *AllTL* is optimal and both methods are still able to improve upon both prediction metrics.

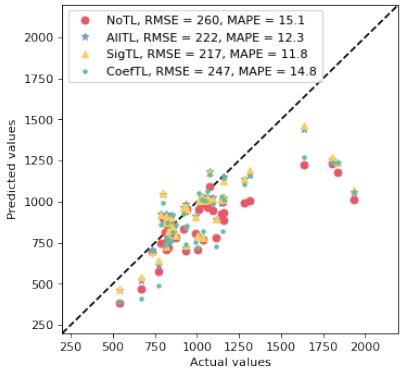
For the discharge and full model, the Zk-test and prediction results are given in Figure 4.12. From the Zk-test, it can be seen that 1/8 of the significant features pass for the discharge model. This could be seen as an indication that the source and target distribution are very different. In the univariate case, too different distributions can lead to worse results after transfer but in this case, it can be seen that the transfer methods improve upon the results without transfer. Most values are underestimated by the *NoTL* method, whereas the transfer methods are more accurate. In particular, the *SigTL* method is optimal with a significant improvement for both the RMSE and MAPE.

	RMSE NoTL	RMSE AllTL	% Change AllTL	RMSE SigTL	% Change SigTL	RMSE CoefTL	% Change CoefTL	MAPE NoTL	MAPE AllTL	% Change AllTL	MAPE SigTL	% Change SigTL	MAPE CoefTL	% Change CoefTL
Variance model	229	192	-16.16%	-	-%	214	-6.55%	12.7	11.9	-6.30%	-	-%	12.4	-2.36%
Discharge model	260	222	-14.62%	217	-16.54%	247	-5.00%	15.1	12.3	-18.54%	11.8	-21.85%	14.8	-1.99%
Full model	289	220	-23.88%	213	-26.30%	274	-5.19%	16.1	12.5	-22.36%	12.0	-25.47%	16.2	**0.62%**

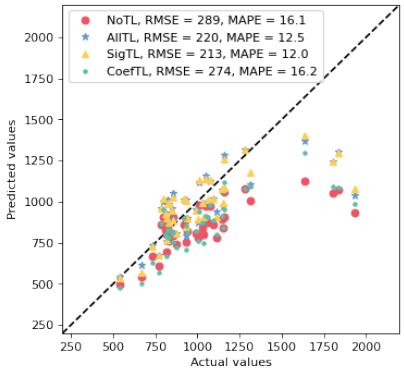
Table 4.4: Summary of the prediction results for the variance, discharge and full model for scenario 4



(a) Zk-test results, the discharge and full model have 8 and 4 significant features respectively



(b) Prediction results for the discharge model

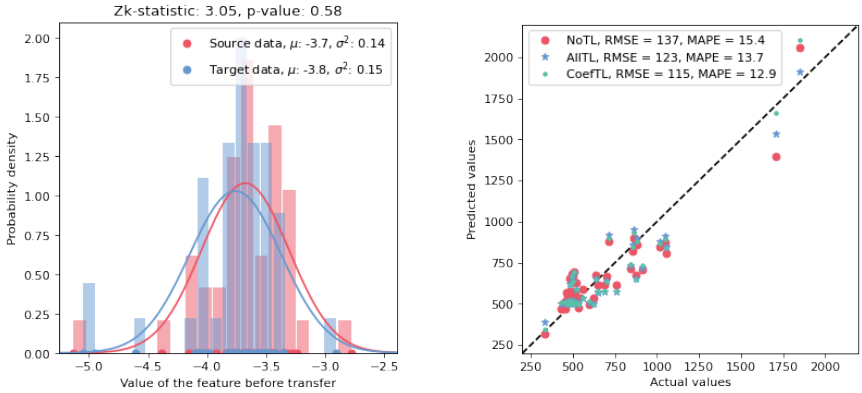


(c) Prediction results for the full model

Figure 4.12: Zk-test results before transfer and the predictions for scenario 4, multivariate models

For the full model, 1 out of 4 significant features passes the Zk-test also indicating quite some difference. The *NoTL* method still underestimates most values, leading to even worse predictions compared to the discharge model. The transfer methods improve upon these results, especially as average values are not underestimated anymore and high tail values are predicted more accurately. For lower tail values, the results are however underestimated by the *CoefTL* and the improvement for high values is not significant. The optimal model here is also the *SigTL* method, which improves on both the RMSE and MAPE. A summary of the results is given in Table 4.4. An interesting observation is that the Zk-test suggests a greater dissimilarity between the significant source and target features for the discharge model, compared to the full model. Despite this, the full model demonstrates greater relative improvement in results. This may be attributed to the fact that in the full model, only 4 features are selected as significant, indicating that a targeted transfer approach can lead to better performance.

4.3.5 Scenario 5

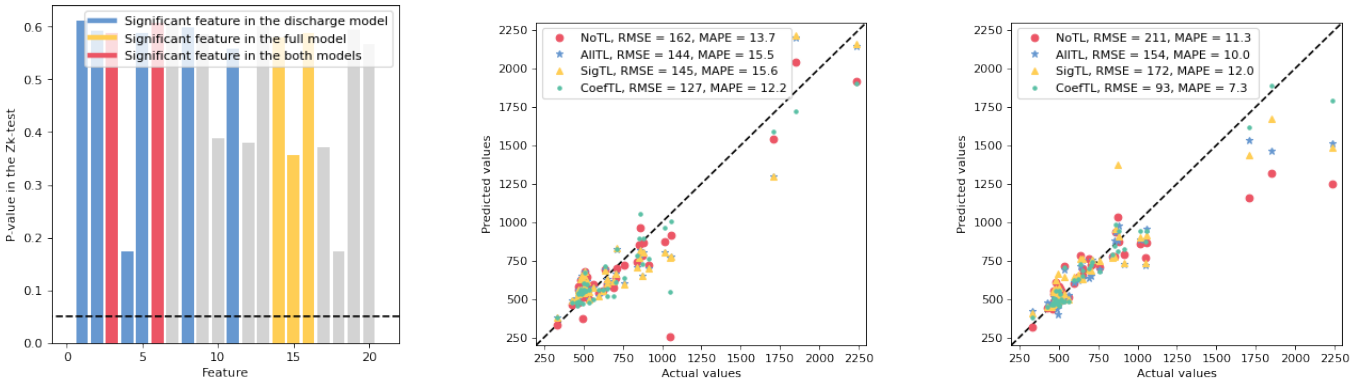


(a) Distribution differences between source and target data with a normal fit, variance model

(b) Prediction results for the variance model

Figure 4.13: Distribution differences and prediction results for the variance model, scenario 5

In scenario 5, the train and test set from the original case study are used, which is a combination of batches 1 and 2. In Figure 4.13, the source and target distributions are shown for the variance model with the corresponding predictions. From the Zk-test results, it can be seen that the source and target distribution are similar. In the simulation study, it was seen that for these cases the transfer can still improve the results significantly. Indeed from the prediction results, it can be seen that both transfer methods improve both the RMSE and MAPE, where the *CoefTL* method is optimal.



(a) Zk-test results, the discharge and full model have 8 and 5 significant features respectively

(b) Prediction results for the discharge model

(c) Prediction results for the full model

Figure 4.14: Zk-test results before transfer and the predictions for scenario 5, multivariate models

For the discharge and full model, the Zk-test and prediction results are given in Figure 4.14. From the Zk-test, a similar scenario where all marginals pass the Zk-test was also simulated in 3.2.1. However, the label data is much different, resulting in different outcomes in the case study. The *NoTL* method accurately predicts average values, but heavily underestimates high tail values in both the discharge and full models. In the simulation, *CoefTL* performed poorly, but here it is the optimal method

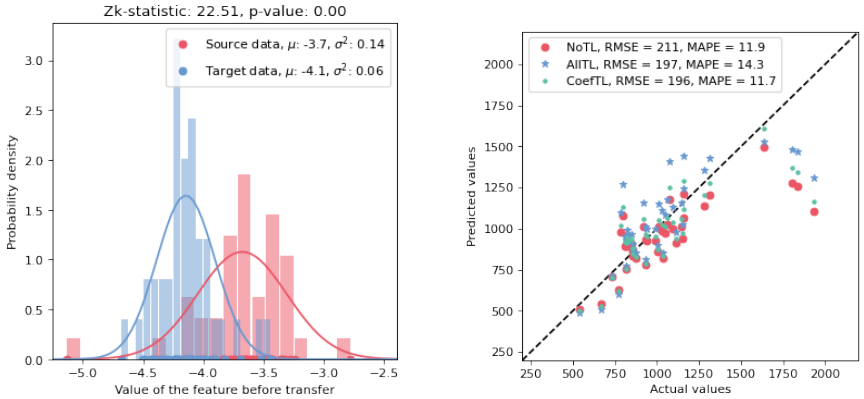
in both models. The discharge model shows a significant improvement in predictions for both the RMSE and MAPE. In the full model, both metrics improve, indicating more accurate predictions for the label data. A summary of the results for scenario 5 is given in Table 4.5.

	RMSE NoTL	RMSE AllTL	% Change AllTL	RMSE SigTL	% Change SigTL	RMSE CoefTL	% Change CoefTL	MAPE NoTL	MAPE AllTL	% Change AllTL	MAPE SigTL	% Change SigTL	MAPE CoefTL	% Change CoefTL
Variance model	137	123	-10.22%	-	-%	115	-16.06%	15.4	13.7	-11.04%	-	-%	12.9	-16.23%
Discharge model	162	144	-11.11%	145	-10.49%	127	-21.60%	13.7	15.5	**13.14%**	15.6	**13.87%**	12.2	-10.95%
Full model	211	154	-27.01%	172	-18.48%	93	-55.92%	11.3	10.0	-11.50%	12.0	**6.19%**	7.3	-35.40%

Table 4.5: Summary of the prediction results for the variance, discharge and full model for scenario 5

4.3.6 Scenario 6

In scenario 6, the train and secondary test set from the original case study are used, which is a combination of batches 1 and 2 for the train set and batch 3 as the test set. In Figure 4.15, the source and target distributions are shown for the variance model with the corresponding predictions. This scenario is actually relatively similar to scenario 4, where the target data is shifted negatively

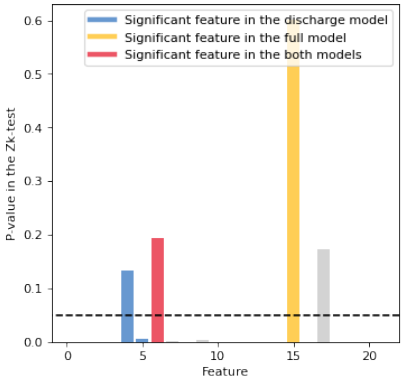


(a) Distribution differences between source and target data with a normal fit, variance model

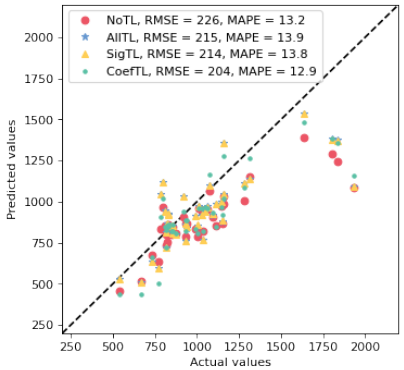
(b) Prediction results for the variance model

Figure 4.15: Distribution differences and prediction results for the variance model, scenario 6

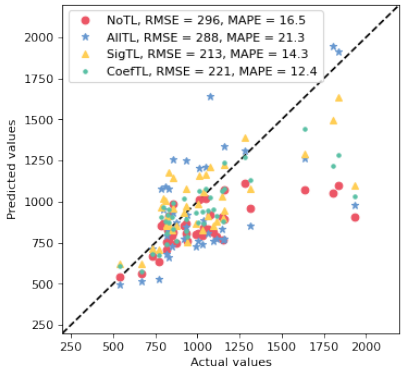
and has a smaller variance, leading to a Zk-test indicating a difference in source and target data. The distributions however are slightly more similar in this case as the variance difference is smaller. In scenario 4, it was seen that the transfer methods were able to improve upon the results mainly for the RMSE. This is similar to the results for scenario 6, where the RMSE for both transfer methods improves, but the MAPE has not necessarily improved. This is probably caused by the fact that the *NoTL* method underestimates most higher tail values, where the transfer methods are more accurate leading to an improved RMSE. However, the transfer methods also overestimate many of the higher average values, leading to a worse MAPE. The *CoefTL* is in this case optimal, whereas, in scenario 4, the *AllTL* method was optimal. The *CoefTL* method is furthermore able to improve both the RMSE and MAPE. For the discharge and full model, the Zk-test and prediction results are given in Figure 4.16. From the Zk-test, it is observed that for the discharge model, only 2/8



(a) Zk-test results, the discharge and full model have 8 and 5 significant features respectively



(b) Prediction results for the discharge model



(c) Prediction results for the full model

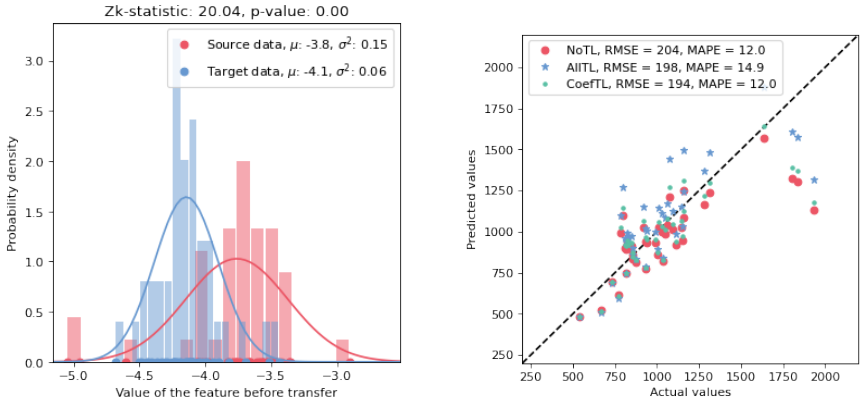
Figure 4.16: Zk-test results before transfer and the predictions for scenario 6, multivariate models

significant features pass the Zk-test whereas, for the full model, 2/5 significant features pass. This could be seen as an indication that the data used for the discharge model, is more different in source and target data compared to the full model. For the discharge model, the *NoTL* method generally underestimates the values, especially for higher tail values. The RMSE for all transfer methods improves, as the tail values are mainly predicted more accurately. Only for the *CoefTL* method, also the MAPE is optimal which makes this variable selection method the most optimal method. For the full model, the *SigTL* method is optimal, improving on both the RMSE and MAPE, also here, the *CoefTL* method improves upon both the RMSE and MAPE and gives a better performance for the MAPE compared to the *SigTL* method. A summary of the results is given in Table 4.6.

	RMSE NoTL	RMSE AllTL	% Change AllTL	RMSE SigTL	% Change SigTL	RMSE CoefTL	% Change CoefTL	MAPE NoTL	MAPE AllTL	% Change AllTL	MAPE SigTL	% Change SigTL	MAPE CoefTL	% Change CoefTL
Variance model	211	197	-6.64%	-	-%	196	-7.11%	11.9	14.3	**20.17%**	-	-%	11.8	-0.84%
Discharge model	226	215	-4.87%	214	-5.31%	204	-9.73%	13.2	13.9	**5.30%**	13.8	**4.55%**	12.9	-2.27%
Full model	296	288	-2.70%	213	-28.04%	221	-25.34%	16.5	21.3	**29.09%**	14.3	-13.33%	12.4	-24.85%

Table 4.6: Summary of the prediction results for the variance, discharge and full model for scenario 6

4.3.7 Scenario 7

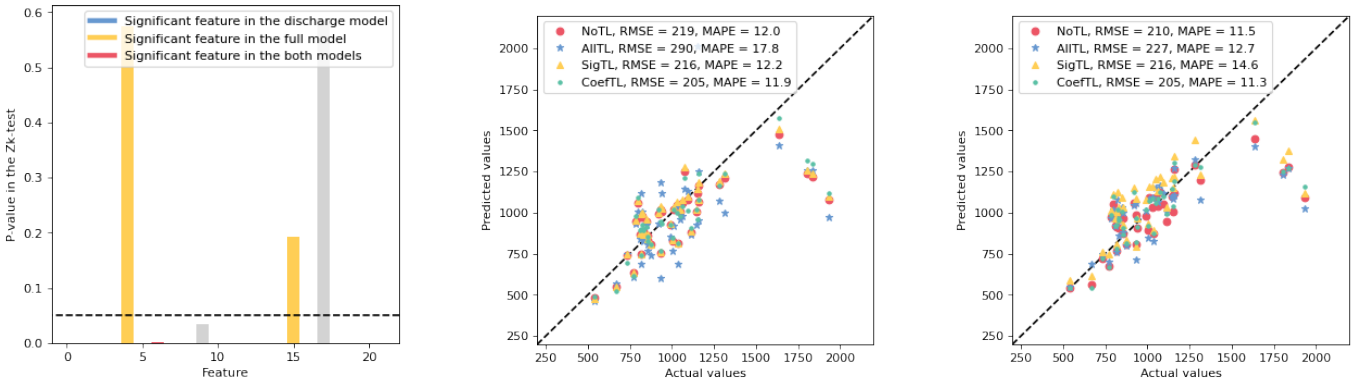


(a) Distribution differences between source and target data with a normal fit, variance model

(b) Prediction results for the variance model

Figure 4.17: Distribution differences and prediction results for the variance model, scenario 7

In scenario 7, the primary and secondary test sets from the original case study are used, which is a combination of batches 1 and 2 for the primary test set (here the source/train set) and batch 3 as the secondary test set. In Figure 4.17, the source and target distributions are shown for the variance model with the corresponding predictions. This scenario is also very similar to scenarios 4 and 6, only here the mean shift is 0.1 less leading to slightly more similar source and target data. From the transfer results, it can be seen that the high tail values, which are underestimated by the *NoTL* method, are predicted more accurately after transfer. However, in particular, for the *AllTL* method, many average values are overestimated, leading to a worse MAPE. For the *CoefTL*, which is the optimal transfer method, the RMSE also improves whereas the MAPE stays more or less the same.



(a) Zk-test results, the discharge and full model have 2 and 6 significant features respectively

(b) Prediction results for the discharge model

(c) Prediction results for the full model

Figure 4.18: Zk-test results before transfer and the predictions for scenario 7, multivariate models

For the discharge and full model, the Zk-test and prediction results are given in Figure 4.18. Here, 0/2 and 2/6 significant features in the discharge model and full model respectively do not pass the

Zk-test. It can be seen in both models, the *CoefTL* method is optimal with an improved RMSE and MAPE compared to the results without transfer.

	RMSE NoTL	RMSE AllTL	% Change AllTL	RMSE SigTL	% Change SigTL	RMSE CoefTL	% Change CoefTL	MAPE NoTL	MAPE AllTL	% Change AllTL	MAPE SigTL	% Change SigTL	MAPE CoefTL	% Change CoefTL
Variance model	204	198	-2.94%	-	-%	194	-4.90%	12	12.1	**0.83%**	-	-%	12	0.00%
Discharge model	219	290	**32.42%**	216	-1.37%	205	-6.39%	12	17.8	**48.33%**	12.2	**1.67%**	11.9	-0.83%
Full model	210	227	**8.10%**	216	**2.86%**	205	-2.38%	11.5	12.7	**10.43%**	14.6	**26.96%**	11.3	-1.74%

Table 4.7: Summary of the prediction results for the variance, discharge and full model for scenario 7

A summary of the results is given in Table 4.7. In all 3 models, the *CoefTL* method improves upon the results without transfer for the RMSE. The MAPE after the transfer is more or less the same. It is not exactly clear why this is the case as the scenario is very similar to scenarios 4 and 6 which show more improvement.

4.3.8 Conclusion on the case study results

In this section, a summary is provided with the conclusions for all 3 models. For each model and scenario, a small description is given on the source data, with corresponding Zk-test results. This is followed by the RMSE and MAPE from the *NoTL* method. Finally, the optimal transfer method is pointed out with the corresponding kernel choice and the prediction results. The value for the regularization parameter lambda is left out as the results are less sensitive to this value. Furthermore, the kernel and lambda choice are outside the scope of this research so no conclusion will be drawn from this choice. The reason to provide the kernel is to give the full background on the results, which could be relevant for future studies. Some overall conclusions will be given on these summary statistics.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7
Target distribution with respect to the source distribution	Positive shift with smaller variance	Negative shift with smaller variance	Negative shift with similar variance	Negative shift with smaller variance	Similar mean and variance	Negative shift with smaller variance	Negative shift with smaller variance
Zk-test Statistic/P-value	16.4 / 0.0	11.6 / 0.0	41.3 / 0.0	28.1 / 0.0	3.1 / 0.58	22.5 / 0.0	20.0 / 0.0
NoTL RMSE/MAPE	186 / 35.1	199 / 12.8	549 / 42.5	229 / 12.7	137 / 15.4	211 / 11.9	204 / 12.0
Optimal variable selection method	AllTL / CoefTL	CoefTL	AllTL	AllTL	CoefTL	CoefTL	CoefTL
Kernel	Linear	Laplacian	RBF	RBF	Laplacian	RBF	RBF
RMSE Abs. / %	186 / 0.0%	181 / -9.1%	351 / -36.1%	192 / -16.2%	115 / -16.1%	196 / -7.1%	194 / -4.9%
MAPE Abs. / %	35.1 / 0.0%	11.6 / -9.4%	26.1 / -38.6%	11.9 / -6.3%	12.9 / -16.2%	11.8 / -0.8%	12.0 / 0%

Table 4.8: Summary of the results for the variance model

The summary statistics for the variance model are given in Table 4.8. In all scenarios, the results after transfer improve or stay similar. It can be seen that for scenario 3, the improvement after the transfer is the most. This was also the scenario with the most different distributions according to the Zk-statistic. The target data was negatively shifted but had a similar variance. This is followed by scenario 5, which also improves a lot for both the RMSE and MAPE. In this scenario, the source and target data were very similar, which is also indicated by the Zk-test results. Similar results were also seen in the simulation results.

Scenarios 4, 2, 6, and 7 also showed improvement, with scenario 4 showing the largest improvement in RMSE and scenario 7 showing the least. For scenarios 4, 6 and 7, the Zk-test was not passed,

meaning that the source and target distributions were different. It was seen that the distributions still overlapped. Thus, the results were in line with the simulation study, where for dissimilar distributions which still overlapped, at least 1 transfer improved the results.

For scenario 2, the p-value from the Zk-test indicates that the source and target data are from the same distribution and show more similarity compared to scenario 1. Also, these results improve significantly after transfer whereas the results for scenario 1 after transfer are exactly the same.

From the univariate case study results, a few conclusions can be drawn. First of all, the results appear to improve a lot for either very similar or very different source and target distributions. For distributions which were slightly different, the results only improved marginally.

	Scenario 1	**Scenario 2**	**Scenario 3**	**Scenario 4**	**Scenario 5**	**Scenario 6**	**Scenario 7**
Number of significant components that pass the test	3/8 (37.5%)	1/8 (12.5%)	1/5 (20%)	1/8 (12.5%)	8/8 (100%)	2/8 (25%)	0/2 (0%)
NoTL RMSE/MAPE	145 / 25.6	203 / 11.5	554 / 43.2	260 / 15.1	162 / 13.7	226 / 13.2	219 / 12.0
Optimal variable selection method	CoefTL	CoefTL	AllTL	SigTL	CoefTL	CoefTL	CoefTL
Kernel	Linear	RBF	Laplacian	Linear	Laplacian	Laplacian	Poly
RMSE Absolute / %	139 / -4.1%	198 / -2.5%	457 / -17.5%	217 / -16.5%	127 / -39.8%	204 / -9.7%	205 / -6.4%
MAPE Absolute / %	25.3 / -1.2%	10.9 / -5.2%	32.2 / -25.46%	11.8 / -21.9%	12.2 / -11.0%	12.9 / -2.3%	11.9 / -0.8%

Table 4.9: Summary of the results for the discharge model

For the discharge model, a summary of the results is shown in Table 4.9. Scenario 5 has the greatest improvement in RMSE after the transfer, likely due to the close similarity between the source and target distributions. Meanwhile, scenario 3 also showed significant improvement. In the variance model, this scenario had the biggest difference in source and target data. The Zk-test in this case can only be applied to marginals in a multivariate scenario, making it difficult to determine differences.

After scenario 3, scenario 4 showed the largest improvement, followed by scenarios 6 and 7, this order aligns with the results from the variance model, where scenario 4 was the most different and scenario 7 the most similar. The number of significant features that pass the Zk-test also supports this observation. However, interpreting these results in a multivariate scenario remains challenging.

Compared to the variance model, scenario 2 showed only marginal improvement and scenario 1 showed improvement, whereas, in the variance model, all methods performed similarly.

Overall, the results are more or less in line with the variance model. It is only notable that in this case, the transfer methods performed better for all scenarios and both the RMSE and MAPE. It is finally notable that only scenario 3 has the *AllTL* as the most optimal method while for all the other methods, the *CoefTL* (in 5 cases) and *SigTL* (in 1 case) are optimal. This shows the relevance of determining *What to transfer* before applying for the transfer.

In Table 4.10, a summary of the results is given for the full model. It can be seen that the optimal transfer method always leads to an improvement for both the RMSE and MAPE. Furthermore, only variable selection methods (*CoefTL* and *SigTL*) are optimal, again pointing out the relevance of focusing the transfer. Again, scenario 5 has the largest improvement while source and target data are in this case very similar. This is followed by scenarios 4 and 6, which improve relatively similarly after transfer.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7
Number of significant components that pass the test	0/4 (0%)	0/4 (0%)	2/10 (20%)	1/4 (25%)	5/5 (100%)	2/5 (40%)	2/6 (33.3%)
NoTL RMSE/MAPE	192 / 36.6	244 / 21.0	401 / 32.3	289 / 16.1	211 / 11.3	296 / 16.5	210 / 11.5
Optimal variable selection method	CoefTL	CoefTL	CoefTL	SigTL	CoefTL	SigTL	CoefTL
Kernel	Linear	RBF	Laplacian	Linear	Laplacian	Laplacian	Linear
RMSE Absolute / %	142 / -26.0%	203 / -16.8%	392 / -2.2%	213 / -26.3%	93 / -55.9%	213 / -28.0%	205 / -2.4%
MAPE Absolute / %	25.1 / -31.4%	14.4 / -31.4%	27.7 / -14.0%	12.0 / -22.4%	7.3 / -35.4%	14.3 / -13.3%	11.3 / -1.7%

Table 4.10: Summary of the results for the full model

Scenario 3 in this case has much less relative improvement, which is caused by the fact that the full model becomes much better without transfer leading to a relatively low improvement in RMSE. The MAPE still improves significantly for this scenario.

In scenario 1, both the RMSE and MAPE showed significant improvement, likely due to overfitting causing the *NoTL* to perform worse. The use of only significant features for prediction focuses the model and leads to improvement, as evidenced by the improved results of the *NoTL* from both the variance and discharge models when the transfer was not used. The transfer further enhances the results in the full model, leading to an improved RMSE and MAPE. Finally, scenario 7 improves marginally. It is not clear why this is exactly the case.

Overall, where the results of the variance and discharge model were in line with each other, it is much harder to explain the results of the full model. It is particularly notable that the *CoefTL* method is optimal for 15 out of 21 cases and does not lead to worse RMSE for all cases and to a (slightly) worse MAPE in only 1 out of 21 cases. These results are summarized in Table 4.11.

	Scenario 1		Scenario 2		Scenario 3		Scenario 4		Scenario 5		Scenario 6		Scenario 7	
	RMSE	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**	**RMSE**	**MAPE**
Variance model	0.00%	0.00%	-9.05%	-9.38%	-20.04%	-33.65%	-6.55%	-2.36%	-16.06%	-16.23%	-7.11%	-0.84%	-4.90%	0.00%
Discharge model	-4.14%	-1.17%	-2.46%	-5.22%	-15.52%	-26.39%	-5.00%	-1.99%	-21.60%	-10.95%	-9.73%	-2.27%	-6.39%	-0.83%
Full model	-26.04%	-31.42%	-16.80%	-31.43%	-2.24%	-13.98%	-5.19%	**0.62%**	-55.92%	-35.40%	-25.34%	-24.85%	-2.38%	-1.74%

Table 4.11: Summary of the relative changes after applying the *CoefTL* methods for all scenarios, both RMSE and MAPE

5 | Conclusion

Battery optimization is crucial in the energy transition as large battery storage systems can support the integration of renewable energy sources and stabilize energy systems. Despite the importance of battery efficiency and longevity, it can be challenging to profitably produce batteries. To better understand battery lifetimes and improve them, models such as physics-based, semi-empirical, and data-driven models can be used. However, these models require large amounts of relevant data for validation and conducting experiments to determine a battery's lifespan can take months or even years. A study by Severson et al [37] proposed a feature-based method that can predict battery lifetimes using early cycle data before degradation occurs, reducing the experimental time, but generalizing these findings is difficult due to limited battery data. Transfer learning offers a solution by transferring knowledge from one problem with similar characteristics to another with different distributions.

The goal of this thesis was to develop an interpretable transfer learning framework for battery lifetime prediction using early cycle data. The framework takes into account *when* to transfer, *what* to transfer, and *how* to transfer knowledge from a source to a target problem with similar characteristics but different distributions. The framework introduced a 2-sample goodness-of-fit test referred to as the 'Zk-test' [47] to analyze the relationship between the source and target domain.

The framework uses an elastic net model [41] to pre-train the model and derive coefficients for the features to reduce the number and magnitude of features before transfer. For this, different variable selection methods were used and tested. Specifically, the *AllTL* method transfers all the original features in the source and target domain, whereas the *SigTL* would only transfer the features with a non-zero coefficient in the elastic net. Finally, the *CoefTL* method uses the pre-trained coefficients as weights so that the weighted features are transferred.

The transfer was done using Transfer Component Analysis [48], which projects the data into a different feature space using different kernels. The final model was trained using the elastic net on the new source and target domains and evaluated using the RMSE and MAPE prediction metrics.

Both the simulation and case study results presented several key conclusions. For the univariate case, such as the variance model, it was generally seen that the transfer improves the results particularly when source and target distributions are very similar. These results become worse and less unstable when the source and target distribution become more different. For example, in the simulation study, for very different distributions the transfer in some cases leads to worse results whereas in other cases it leads to improved results. In the case study, however, the scenario with very different source and target distributions showed a large improvement in the prediction performance after transfer. A threshold for defining a 'very' different distribution was established by using the highest value of the Zk-statistic, which is attained when there is no intersection between the source and target distributions. It was observed that if there is any degree of overlap between the source and target data, at least one transfer can lead to better results than without a transfer. However, in Figure 3.16 in the simulation study, where the means were shifted and the variance was fixed for skewed source and target distributions, there was a slight overlap between the source and target data, yet the transfers led to slightly inferior outcomes compared to those without transfer.

For the multivariate cases, it was seen that the transfer was especially effective for predicting the tail values of labels. This resulted in improved results after the transfer, even for similar source and target distributions. For both the discharge and full models, it was observed that all results improved after transfer. The full model especially showed the importance of variable selection methods as the *SigTL* and *CoefTL* gave the best results.

It remains challenging to find a suitable metric for determining a successful transfer without using the prediction results. The MMD before and after transfer showed no clear relationship with the results. The Zk-test shows good performance for detecting differences in distributions, but is limited to marginal distributions and also less vulnerable to differences in variance. Especially for multivariate cases, determining the success of transfer learning becomes even more challenging because MMD remains unreliable and the Zk-test is limited in multivariate cases as it only applies to marginal distributions. To fully assess transfer learning in multivariate cases, a more comprehensive goodness-of-fit test that deals specifically with multivariate data may be necessary. This would help to gain a better understanding of the success of transfer in these cases and provide a clearer understanding of the performance.

In general, the *CoefTL* method turned out to be the most successful. In 15 out of 21 cases, the *CoefTL* method is the optimal method. Furthermore, it does not lead to worse RMSE for all cases and to a (slightly) worse MAPE in only 1 out of 21 cases. it was the best model and only in a few instances did it result in worse predictions. Transfer learning often improved the RMSE, indicating better predictions for high tail values. This highlights that transfer can lead to a more accurate model capable of detecting complex patterns in the data.

In conclusion, TCA can generally lead to improved results where selecting the variables before the transfer, using the *CoefTL* method is optimal. It remains challenging to find a suitable metric for indicating a successful transfer as the MMD is unreliable and the Zk-test is not suitable for multivariate scenarios. From the univariate scenarios, it was seen that there is a relationship between the Zk-test and the results, but further research is needed to make actual conclusions on this relationship. For example, the choice of kernel should be taken into account and an elaborate analysis should be done for each variable selection method separately.

Transfer learning has proven to be effective for both comparable and dissimilar source and target distributions. As a result, it is particularly valuable in the context of battery lifetime prediction since it can handle scenarios where there is a shortage of varied training data.

6 | Discussion

This chapter outlines several points for discussion that have arisen from the examination of the framework presented in this thesis. Some of these points can be considered topics for further discussion.

6.1 Discussion

First of all, many discussion points on the dataset used for the case study and the model applied to it are presented by the original paper [37]. Hence, it is advisable to review the accompanying notes provided by this article. It is acknowledged that the outcomes of the case study may not align with the results obtained for similar scenarios in this thesis, which could be due to misinterpretation of certain features, particularly the temperature integral, and variations in the pre-processing of the data. Additionally, differences in the elastic net model and cross-validation could have resulted in a difference in predictions. Because of this, the results in this thesis before and after transfer should be compared within this thesis, and not with the results from the original case study.

It's important to note that the Transfer Component Analysis method [48] assumes that the marginal distribution of the source and target data are different, but the conditionals of the label data given the projected transfer component data are from the same distribution. However, the results of simulations in the thesis suggest that the transfer performance almost always improves when the source and target samples are exactly the same or similar. This implies that the assumption made by TCA is not a stringent one and could be relaxed in future work.

Another discussion point is that the results of this thesis are heavily influenced by the choice of kernel and regularization parameter (lambda). The main objective of the thesis was to develop and evaluate a transfer learning framework, examining the effect of different variable selection methods on performance and the relationship between performance and distribution differences. In many cases, the kernel was selected based on its performance, rather than being incorporated into the framework itself. When there is no label data available in the target domain, this becomes an even more important issue. Incorporating a systematic approach to selecting kernel and lambda can help reduce variability in results and make the conclusions more robust, particularly in such cases. Furthermore, such an approach can help determine a successful transfer in general, which is still a challenge. In addition to outlining a transfer learning framework, an upper limit was also established for the dissimilarity between the source and target distributions that allows for successful transfer. To enhance the precision of this threshold, a starting point for future exploration could involve expanding this analysis to various types of data.

The choice of variable selection method is also a significant factor in the performance of the transfer learning framework. These methods are particularly important for multivariate datasets, which are more commonly encountered in practice. The results obtained using the *CoefTL* method were encouraging, but further research is needed to fully understand and justify its use in the transfer

method. It would be beneficial to provide more examples and theoretical explanations to support the use of the *CoefTL* method and to establish clear assumptions for when it is likely to produce successful results. In this way, the framework could be made more robust and widely applicable, especially for multivariate datasets.

6.2 A Personal Note

Throughout the period of writing this thesis, there has been a significant focus worldwide on recent advancements in artificial intelligence (AI). For example, the recent launch of ChatGPT[1] is a major development in the field, and Microsoft's acquisition [68] of the technology suggests that we may soon see AI integrated into their applications. While there has been a great deal of positive attention given to these developments, there has also been criticism expressed by those who have concerns. Even though the findings and conclusions presented in this thesis may not be directly related to current advancements in this field, they can still contribute as a small piece of a larger puzzle, which is why it is necessary for me to include a brief note in this thesis.

Reflecting on the potential benefits and risks of artificial intelligence, I am reminded of my experience in a machine learning course where we accurately classified tumours as malignant or benign using PET and CT scans. This example illustrates how AI can be used for good in healthcare, while my thesis on predicting battery life has the potential to advance the development of batteries and, therefore, play a crucial role in the energy transition.

Nevertheless, I acknowledge the concerns raised about AI in other contexts, such as the use of recommender systems by social media platforms like Instagram and TikTok. As highlighted in the documentary 'The Social Dilemma' [2], these algorithms can create a feedback loop that reinforces specific beliefs and behaviours, potentially leading to a loss of autonomy and privacy for users. Furthermore, AI's potential to replace jobs, including those in creative industries, is another significant concern. Some artists fear that AI-generated art may eventually replace human-created art, leading to a loss of jobs and creativity [69]. Additionally, there are concerns that AI could erode critical thinking skills in students [70], as automated solutions may lead to a reliance on technology and hinder the development of independent problem-solving skills. These issues have long-term implications for education and the workforce.

It is important to ensure that users maintain some level of control when interacting with AI algorithms. In my view, allowing users to give AI a task and then critically evaluate its output is a good way to work with AI. Education should adapt to this shift and prepare students to work with AI in a critical and informed way. However, on social media platforms like Instagram and TikTok, there is often limited input from the user possible, which should be viewed as a concern. Addressing these concerns remains essential in order to prevent further loss of control by the users.

In conclusion, it is important to consider the ethical implications when developing AI systems. These technologies must be designed and deployed in a way so that the autonomy of the user remains

[1] https://openai.com/blog/chatgpt

[2] Orlowski, J. (Director). (2020). The Social Dilemma [Motion Picture]. United States: Exposure Labs.

maintained through interaction with the application is critical. This approach can enable us to harness the positive potential of AI while mitigating its potential negative impacts. After these advancements are made, individuals must embrace them and incorporate them into their careers, while maintaining a critical mindset and being conscious of their potential dangers. While people should not be fearful that AI will necessarily replace their jobs, it is essential to recognize that those who utilize AI may have a competitive advantage.

While other concerns exist, such as potential impacts on human rights [71], this note focused on the autonomy of the user. However, it is essential to carefully consider the implications of AI and algorithms on human decision-making and to ensure that these technologies are developed and used in a responsible and ethical manner while balancing the potential benefits with potential risks.

7 | Appendix

7.1 Additional background for transfer learning

Transfer learning methods are generally categorized in the following four algorithms [42]:

- *Instance-based algorithms* Instance-based transfer learning approaches aim to adjust the weights of samples in the source domain to account for differences in marginal distributions, with the goal of leveraging these reweighted instances during training in the target domain. By incorporating the reweighted source samples, the target learner can focus on extracting only the relevant knowledge from the source domain.

- *Feature-based algorithms* In feature-based transfer learning, knowledge is transferred by projecting the data onto learning a new feature representation for both the source and target domains. This way, the source-labelled data can be reused for training. The projected features can be used as a bridge to transfer knowledge.

- *Model-based algorithms* Knowledge is transferred by the assumption that the source and target domains share various (hyper)parameters of the learning models. It is motivated by the idea that a well-trained source model captures a lot of good structure, which is general and can be adapted to a more specified target model. For deep learning methods, the model-based approach has recently been used widely as a pre-training technique since it can train a deep learning model on sufficient source data and only a few target data points are needed to fine-tune some parameters of the pre-trained model.

- *Relation-based transfer learning algorithms* Knowledge is transferred by setting rules which specify relations between instances across domains and tasks. Once these common instances are extracted, they can be used as knowledge for transfer learning. In contrast to the previous 3 methods, relation-based transfer learning approaches do not assume that the source and target domain are independent and identically distributed.

7.2 Transfer Component Analysis - Related work

In the sections below, related methods will be explained that motivated the development of TCA. This starts with 2 dimensionality reduction methods, namely Principal Component Analysis (PCA) Subsection 7.2.1 and Kernel-PCA Subsection 7.2.2 followed by Maximum Variance Unfolding (MVU) Subsection 7.2.3 which is generalized by Maximum Mean Discrepancy Embedding (MMDE) Subsection 7.2.4. These methods are very similar to Transfer Component analysis but aim to implicitly find the kernel function using a linear programming setup.

7.2.1 Principal component analysis (1901)

Principal Component Analysis (PCA) was introduced in 1901 by Karl Pearson. PCA is a method providing the best m-dimensional subspace to project data onto from a d-dimensional space, where $m < d$. This subspace will then be the projection of the original vectors onto m directions, i.e. the *principal components*, which span the subspace. Intuitively, it would be the original space, viewed from different axes/dimensions. For deriving these components, the simplest way is to maximize the variance. The first principal component is in the direction where projections have the largest variance. The second principal component is in the direction of the projections with the largest variance orthogonal to the first component. The i-th component is in the direction of the projections with the largest variance orthogonal to the previous $i-1$ components. Maximizing the variance is done by finding the eigendecomposition of the covariance matrix and sorting the eigenvectors based on the size of the corresponding eigenvalues. Next, depending on the number of principal components chosen, the new features can be calculated from the eigenvectors. By applying this change of basis, it is reasonable that the new features are the features with the largest variance and thus, most important. Features with small variances can, on the other hand, be discarded. After this more intuitive explanation, the mathematical derivation of PCA is given below.

Let $X \in \mathbb{R}^{n \times d}$ a dataset containing the column vectors $\{\boldsymbol{x}_i\}_{i=1}^{d} \in \mathbb{R}^n$ contains n datapoints and d is the number of features. PCA assumes that data is centred, so define a matrix Z with $\mu_k = \frac{1}{n}\sum_k \boldsymbol{x}_k$ the mean over column vector (or feature) k and σ_k its standard deviation, one can centre the data using for example via $\boldsymbol{x}_{ik} \to \frac{\boldsymbol{z}_{ik}-\mu_k}{\sigma_k}$ $\forall i \in [1, ..., n]$ and $k \in [1, ..., d]$. In short, Z contains column-wise zero empirical mean or the sample mean of each column has been shifted to zero). The covariance matrix is then proportional to:

$$\Sigma_Z \propto Z^\top Z. \tag{7.1}$$

Note that $\Sigma_Z \in \mathbb{R}^{d \times d}$, and since the centred dataset has sample means for each feature equal to zero, i.e. for each $i \in [1, ..., d]$, $E(\boldsymbol{z}_i) = \frac{1}{n}\sum_{k=1}^{n} z_{ki} = 0$ and the covariance matrix indices can indeed be expressed as:

$$\begin{aligned}
[\Sigma_Z]_{ij} = \mathrm{Cov}(\boldsymbol{z_i}, \boldsymbol{z_j}) &= E((\boldsymbol{z_i} - E(\boldsymbol{z_i}))(\boldsymbol{z_j} - E(\boldsymbol{z_j}))^\top) \\
&= E(\boldsymbol{z_i}\boldsymbol{z_j}^\top) \propto \sum_{k=1}^{n} z_{ki} z_{kj} \\
&= \sum_{k=1}^{n} [Z]_{ki}[Z]_{kj} \\
&= \sum_{k=1}^{n} [Z^\top]_{ik}[Z]_{kj} = [Z^\top Z]_{ij}
\end{aligned}$$

It can be concluded that Equation 7.1 is true. It follows that Σ_Z is positive semi-definite and hence has eigenvectors which form an orthonormal basis for $\mathbb{R}^d$. In order to show that the (sub)space spanned by subspace eigenvectors with the largest eigenvalues is the optimal projection of the original data, we will prove the following, more generalized, theorem. The proof is based on ideas from [72] and [73].

Theorem 7.2.1. *Let $W = Z^\top Z$, it follows that W is positive semi-definite and has eigenvectors $\boldsymbol{u}_1, ..., \boldsymbol{u}_d$ with corresponding eigenvalues $\lambda_1, ..., \lambda_d$ which form an orthonormal basis for $\mathbb{R}^d$. Also assume, without loss of generality, that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_d$. Finally, let $P_V^\top = P_V : \mathbb{R}^{d\times d} \to \mathbb{R}^{d\times d}$ be the projection onto V matrix and define the projection error by $||\boldsymbol{z}_i - P_V \boldsymbol{z}_i||_2^2$. For any integer $0 < m < d$ $U_m := span\{u_1, ..., u_m\}$ is the subspace which minimizes the sum of the projection errors:*

$$\sum_{i=1}^{n} ||\boldsymbol{z}_i - P_{U_m}\boldsymbol{z}_i||_2^2 = \min_{V\subset\mathbb{R}^d} \sum_{i=1}^{n} ||\boldsymbol{x}_i - P_V \boldsymbol{z}_i||_2^2. \tag{7.2}$$

Proof. Choose any $m < d$ and let $V \in \mathbb{R}^m \subset \mathbb{R}^d$. Note that the Frobenius norm for a real matrix $A \in \mathbb{R}^{m\times n}$ is defined as $||A||_F^2 = \sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2 = \mathrm{Tr}(A^\top A)$. Then:

$$\begin{aligned}
\sum_{i=1}^{n} ||\boldsymbol{z}_i - P_V \boldsymbol{z}_i||_2^2 &= ||Z - P_V Z||_F^2 \\
&= \mathrm{Tr}((Z - P_V Z)^\top (Z - P_V Z)) \\
&= \mathrm{Tr}(Z^\top Z - Z^\top P_V Z - P_V^\top Z^\top Z + P_V^\top Z^\top Z P_V) \\
&= \mathrm{Tr}(W - W P_V - P_V W + P_V W P_V) \\
&= \mathrm{Tr}(W) - \mathrm{Tr}(P_V W)
\end{aligned}$$

It is used here that $\mathrm{Tr}(AB) = \mathrm{Tr}(BA)$ and $P_V^2 = P_V$. Since $\mathrm{Tr}(W)$ is a constant, minimizing Equation 7.2 is the same as maximizing $\mathrm{Tr}(P_V W)$.

Let $\{\boldsymbol{v}_1, ..., \boldsymbol{v}_m\}$ be an orthonormal basis for V. It follows that $P_V = \sum_{i=1}^m \boldsymbol{v}_i^\top \boldsymbol{v}_i$ from which we have:

$$\mathrm{Tr}(P_V W) = \mathrm{Tr}(\sum_{i=1}^{m} \boldsymbol{v}_i^\top \boldsymbol{v}_i W) = \sum_{i=1}^{m} \mathrm{Tr}(\boldsymbol{v}_i^\top W \boldsymbol{v}_i)$$

Note that $\boldsymbol{v}_i$ has unit length, i.e. $\boldsymbol{v}_i \boldsymbol{v}_i \top = 1$ and we want $\sum_{i=1}^m \mathrm{Tr}(\boldsymbol{v}_i^\top W \boldsymbol{v}_i)$ to be maximal. Using the Lagrange multiplier λ with objective function $\boldsymbol{v}_i^\top W \boldsymbol{v}_i$ and constraint $\boldsymbol{v}_i \boldsymbol{v}_i \top - 1$, this leads to an unconstrained optimization problem by maximizing:

$$\mathcal{L}(\boldsymbol{v}, \lambda) := \boldsymbol{v}_i^\top W \boldsymbol{v}_i - \lambda(\boldsymbol{v}_i^\top \boldsymbol{v}_i - 1).$$

This can be optimized using the derivative w.r.t λ and $\boldsymbol{v}_i$:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \lambda} &= \boldsymbol{v}_i^\top \boldsymbol{v}_i - 1 = 0 \\
\frac{\partial \mathcal{L}}{\partial \boldsymbol{v}_i} &= 2W\boldsymbol{v}_i - 2\lambda \boldsymbol{v}_i = 0
\end{aligned}$$

given:

$$\begin{aligned}
\boldsymbol{v}_i^\top \boldsymbol{v}_i &= 1 \\
W\boldsymbol{v}_i &= \lambda \boldsymbol{v}_i
\end{aligned}$$

We can conclude that when v_i is an eigenvector of W (or the covariance matrix) with the largest eigenvalue λ_i and

$$\mathrm{Tr}(P_V W) = \sum_{i=1}^{m} \mathrm{Tr}(\boldsymbol{v}_i^\top W \boldsymbol{v}_i)$$

is maximized by setting $\lambda_1 \geq ... \geq \lambda_d$ (in a maximization problem, the eigenvalues are sorted from largest to the smallest value) [**?**] and letting v_i correspond to its eigenvalue λ_i. □

By setting $W = \Sigma_Z$, we have that the projection onto the subspace spanned by the first m eigenvectors with the largest eigenvalue minimizes the projection error while maximizing the variance of the projection onto the first m principal components, since their total variance equals $\sum_{i=1}^{m} \lambda_i$. The projected dataset can finally be calculated by $P_{U_m} Z$ where Z is the standardized matrix of the original dataset and P_{U_m} a symmetric $m \times m$ containing the first m eigenvectors of $Z^\top Z$ with the largest corresponding eigenvalues.

Choosing the number of principal components m can be done by using the explained variance ratio. This can be calculated by dividing the variance of a principal component and the total variance, which is the sum of the variance of all the principal components. By calculating the cumulative explained variance ratios for the first m components, and setting this to be over a certain threshold between 95-99%, one can determine how many principal components are needed to make the dimensionality reduction sufficient.

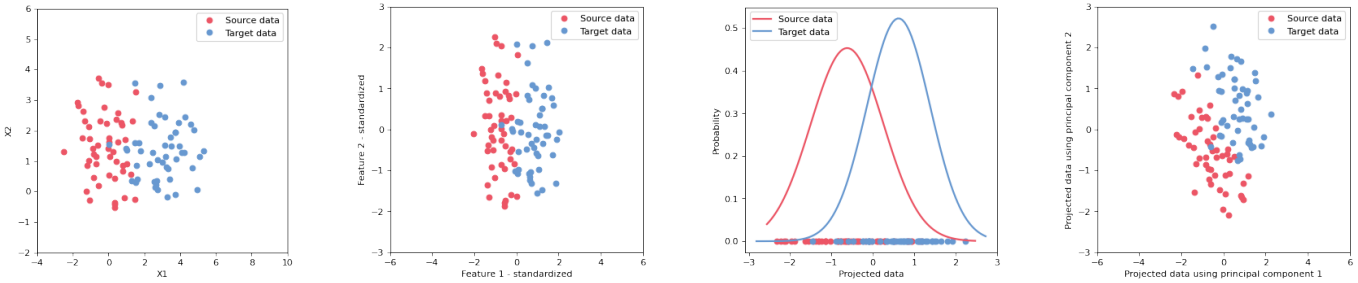
PCA algorithm and key assumptions

- Select a dataset $X \in \mathbb{R}^{n \times d}$ with n datapoints and d features.
- Create a matrix Z which is the centered version of X, i.e. each columns should have mean 0. This can be done for example by applying: $\boldsymbol{z_i} = \boldsymbol{x_i} - \frac{1}{n} \sum_{k=1}^{n} x_{ki}$ to each column vector $\boldsymbol{z_i}_{i=1}^{d}$.
- Calculate the covariance matrix of Z via $Z^\top Z$ and note that this is a positive semi-definite matrix which means that there exist eigenvectors and eigenvalues which form an orthonormal basis for $\mathbb{R}^d$.
- Find the eigenvectors and eigenvalues of $Z^\top Z$ and sort them based on their eigenvalues.
- Select the first $m < d$ eigenvalues and corresponding eigenvectors and build P_{U_m}. Choose m in this case by using the explained variance ratio, as explained above.
- Find the projected datapoints by $Y = P_{U_m} Z$ where $Y \in \mathbb{R}^{n \times m}$.

Applying PCA to the simulated dataset mentioned in Section 2.5 leads to the results in Figure 7.1.

7.2.2 Kernel Principal component analysis (1996)

Kernel Principal Component Analysis (KPCA) was introduced by Schölkopf et al. in 1996 [74] as a non-linear generalization of PCA. Since PCA always finds linear principal components to represent data in lower dimensions, KPCA is able to find non-linear principal components as well. Intuitively, this could be seen as applying PCA in a new, transformed space.



(a) Original source and target data $X \in \mathbb{R}^{50\times 2}$ from the simulation

(b) Standardized dataset Z

(c) Projection onto 1 dimension based on the first principal component ($\lambda_1 = 106.58$), with normal fit

(d) Projection onto 2 dimensions based on both principal components ($\lambda_2 = 93.42$)

Figure 7.1: PCA applied to the simulation data with different kernels and projected to different dimensions

In order to do this, a mapping function ϕ to a non-linear space is introduced, however never explicitly calculated. This leads to the possibility that very high dimensional ϕ's can be introduced without actually evaluating data in that so-called feature space. Instead, a trick is used called 'the kernel trick'. To elaborate on this, the following definition is introduced:

Definition 7.2.1. *Let $\phi : \mathcal{X} \to \mathcal{V}$ be a map between a vector inner product space $\mathcal{X}$ to a feature inner product space $\mathcal{F}$. Let $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ be 2 vectors in $\mathcal{X}$. Then $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}), \phi(\boldsymbol{x'}) \rangle_F$, where $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ denotes the inner product in space $\mathcal{F}$. Here, k is the kernel function.*

Note that these spaces can be any type of space, as long as the key restriction of having a proper inner product is satisfied. To give a bit more intuition, calculating $\langle \phi(\boldsymbol{x}), \phi(\boldsymbol{y}) \rangle_F$ usually requires calculating $\phi(\boldsymbol{x})$ and $\phi(\boldsymbol{y})$ first, followed by calculating the dot product which leads to a scalar as a result. The kernel trick is a clever way to avoid expensive computations in high dimensions.

Let $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j) \rangle_F = \phi(\boldsymbol{x}_i)^\top \phi(\boldsymbol{x}_j)$, then the input can be represented in a lower dimension using PCA. To do this, the features need to be centred, so that the covariance matrix can be calculated. This can be done by centering the features via:

$$\tilde{\phi}(\boldsymbol{x_k}) = \phi(\boldsymbol{x}_k) - \frac{1}{n}\sum_{i=1}^{n} \phi(\boldsymbol{x}_i)$$

and finding the corresponding centered kernel:

$$\begin{aligned}
\tilde{K}_{ij} &= \tilde{\phi}(\boldsymbol{x_i})^\top \tilde{\phi}(\boldsymbol{x_j}) \\
&= (\phi(\boldsymbol{x}_i) - \frac{1}{n}\sum_{i=1}^{n} \phi(\boldsymbol{x}_i))^\top (\phi(\boldsymbol{x}_j) - \frac{1}{n}\sum_{j=1}^{n} \phi(\boldsymbol{x}_j)) \\
&= K_{ij} - \frac{1}{n}\sum_{k=1}^{n} K_{ik} - \frac{1}{n}\sum_{k=1}^{n} K_{jk} + \frac{1}{n^2}\sum_{l,k=1}^{n} K_{lk} \\
&= K_{ij} - \frac{2}{n}\sum_{k=1}^{n} K_{ik} + \frac{1}{n^2}\sum_{l,k=1}^{n} K_{lk}
\end{aligned}$$

Therefore, let $\mathbf{1}_{1/n}$ be a matrix with all elements equal to $\frac{1}{n}$, then:

$$\tilde{K} = K - 2\mathbf{1}_{1/n}K + \mathbf{1}_{1/n}K\mathbf{1}_{1/n} \tag{7.3}$$

With this centred kernel, the process can be continued in a similar way as PCA. The transferred datapoints $Y \in \mathbb{R}^{n\times m}$, with $m < d$ the desired dimension, can be calculated by

$$Y = P_V\tilde{K}.$$

In short, the assumptions and algorithm of KPCA are stated below.

KPCA algorithm and key assumptions

- Choose a kernel function k and build the kernel matrix by $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$.
- Center the kernel by calculating

$$\tilde{K} = K - 2\mathbf{1}_{1/n}K + \mathbf{1}_{1/n}K\mathbf{1}_{1/n}$$

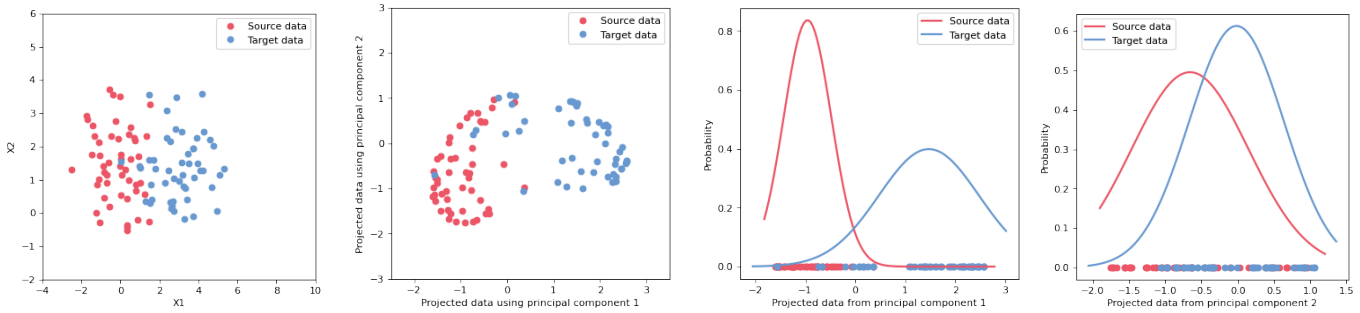
  .
- Apply the PCA algorithm to the centred matrix $\tilde{K}$. Note that the matrix is already centred so the algorithm starts with calculating the covariance matrix and then finding its eigendecomposition.
- Calculate the projected values by $Y = P_{U_m}\tilde{K}$.

Doing this using the RBF kernel gives the results in Figure 7.2.

The main freedom in KPCA is choosing the kernel function or specifying the kernel matrix. These kernels are similar to the kernels used in Support Vector Machines (SVM) [75]. Actually, much work revolves around the question of which kernel to choose. MVU focuses on learning this kernel matrix by finding an implicit mapping into the feature space which will 'unfold' the manifold from which the data was sampled.

7.2.3 Maximum Variance Unfolding (2004)

Maximum Variance Unfolding (MVU) is a type of KPCA and was introduced by [76] in 2004. As mentioned in the previous paragraph, the choice of the kernel plays an important role when applying KPCA. Different types of kernels can reveal different structures in lower dimensions. In this paragraph, the problem of learning a kernel is resolved by setting up the problem as an instance



(a) Original source and target data $X \in \mathbb{R}^{50\times 2}$ from the simulation

(b) KPCA with an RBF kernel, represented in 2 dimensions

(c) Projection onto 1 dimension based on the first principal component using KPCA (RBF kernel), with normal fit

(d) Projection onto 1 dimension based on the second principal components using KPCA (RBF kernel), with normal fit

Figure 7.2: KPCA applied to the simulation data with fitted normal distributions with an RBF kernel

of semi-definite programming (SDP). A semi-definite program is an optimization problem and in the standard form usually looks like [77]:

$$\begin{aligned} \max_{X} \quad & C \circ X \\ \text{s.t.} \quad & A_i \circ X = b_i, \forall i = 1, ..., m \\ & X \succeq 0 \end{aligned} \tag{7.4}$$

where X is a matrix containing variables and C contains various constants. In Equation 7.4, $C \circ X$ is called the objective function and A_i's are the constraint matrices and b_i are scalars. Note that all constraints must be linear.

The advantage of MVU is that, whereas PCA only has a good solution (one that preserves distances well) when $\boldsymbol{x}_i$ lie near a $m-$dimensional subspace of $\mathbb{R}^d$, MVU finds a non-linear embedding of the points. This is similar to what KPCA does, but instead of choosing a kernel which transforms the space of the data, MVU maximizes the variance without specifying a kernel. Instead, MVU maximizes the matrix K with the constraint that the local isometry of the (non-linear) manifold should be preserved. Mathematically this can be described as: given $\{\boldsymbol{x_i}\}_{i=1}^d \in \mathbb{R}^n$, find $\{\boldsymbol{y_i}\}_{i=1}^d \in \mathbb{R}^n$ where $m \leq d$ such that $||\boldsymbol{y_i} - \boldsymbol{y_j}||^2 \approx ||\boldsymbol{x_i} - \boldsymbol{x_j}||^2$ for nearest neighbors $(i, j) \in E$, where E is an edge set. To clarify on this, let's give an example.

Suppose we have the so called 'Swiss roll' (a commonly used figure to sample from in manifold learning) from Figure 7.3. Imagine that each point has k-nearest neighbours and these points in the roll are pulled as far apart as possible, where $\phi(\cdot)$ records the final positions. All points will have the same nearest neighbours, but will also flatten while the variance increases captured by the dominant principal components. This intuition can be formalized by setting up an optimization problem as described in Equation 7.4.

Instead of empirically choosing a kernel function and calculating matrix K, the objective of MVU is to find a matrix K via SPD using constraints. From there, PCA is applied to the resulting matrix

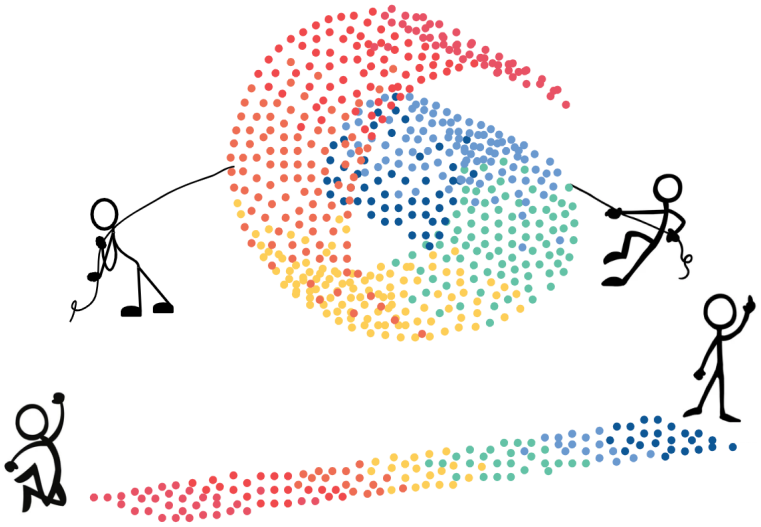


Figure 7.3: Maximum variance unfolding of Swiss roll [78]

K in order to find the principal components in the linear projected subspace. Therefore, K must be positive semi-definite and centered, i.e. $K \succeq 0$ and $\sum_{ij} K_{ij} = 0$. Note that these are both linear constraints which are required by the SPD problem definition. Furthermore, the local isometry of neighbours is preserved. Therefore, for all nearest neighbors i, j:

$$\begin{aligned} ||\phi(\boldsymbol{x_i}) - \phi(\boldsymbol{x_j})||^2 = ||\boldsymbol{x_i} - \boldsymbol{x_j}||^2 &= \phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_i) + \phi(\boldsymbol{x}_j)\phi(\boldsymbol{x}_j) - \phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_j) - \phi(\boldsymbol{x}_j)\phi(\boldsymbol{x}_i) \\ &= K_{ii} + K_{jj} - K_{ij} - K_{ji} = G_{ii} + G_{jj} - G_{ij} - Gji \\ &= \boldsymbol{x}_i \cdot \boldsymbol{x}_i + \boldsymbol{x}_j \cdot \boldsymbol{x}_j - \boldsymbol{x}_i \cdot \boldsymbol{x}_j - \boldsymbol{x}_j \cdot \boldsymbol{x}_i. \end{aligned} \tag{7.5}$$

Here, $G_{ij} = \boldsymbol{x}_i \cdot \boldsymbol{x}_j$ is the Gram matrix of the inputs. Note that this is also a linear constraint. Since only the distances of the neighbours are preserved, we have that the objective of MVU is to maximize the pairwise distance of each pair of points, i.e. for $i \neq j$ maximize $\sum_{ij} ||\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)||^2$. Since $K_{ij} = \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$ and by the centering constraint that $\sum_{ij} K_{ij} = 0$ we have that:

$$\begin{aligned} &\sum_{ij} ||\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}_j)||^2 \\ &= \sum_{ij} (\phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_i) + \phi(\boldsymbol{x}_j)\phi(\boldsymbol{x}_j) - 2\phi(\boldsymbol{x}_i)\phi(\boldsymbol{x}_j)) \\ &= \sum_{ij} (K_{ii} + K_{jj} - 2K_{ij}) = \sum_{ij} (K_{ii} + K_{jj}) \propto \mathrm{Tr}(K) \end{aligned} \tag{7.6}$$

Thus, the objective function for the optimization is maximizing the trace of the kernel. This leads to the following semi-definite programming problem:

$$\begin{aligned} \max_{K} \quad & Tr(K) \\ \text{s.t.} \quad & K \succeq 0, \\ & \sum_{ij} K_{ij} = 0, \\ & K_{ij} + K_{jj} - K_{ij} - K_{ji} = G_{ij} + G_{jj} - G_{ij} - G_{ji}, \\ & \forall i, j \quad s.t. \quad \eta_{ij} = 1. \end{aligned} \tag{7.7}$$

Since this is a convex optimization problem since all the constraints are linear, there are no local maxima. Also, K is positive semi-definite and centred, so eigendecomposition can be applied in order to find the principal components in the linear space. This procedure is similar to we saw in PCA and KPCA.

For solving the problem, we use the 'SCS' solver from the 'cvxpy' package in Python. The algorithm we use is inspired on the 'MVU' algorithm[1] from Github. The algorithm is applied on the same simulated dataset which is used in the previous examples. The results can be found in Figure 7.4. MVU is used as a motivation for the transfer learning method called Maximum Mean Discrepancy Embedding.

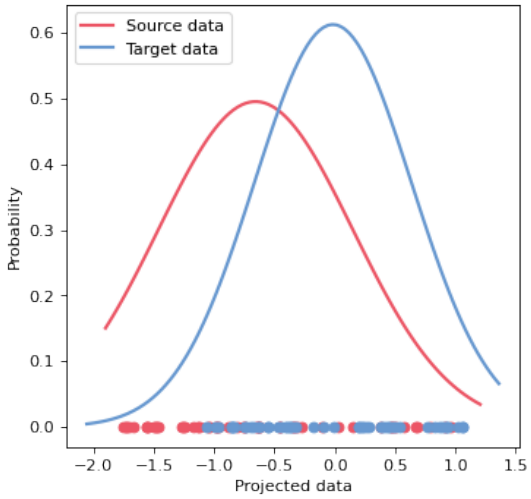


Figure 7.4: 1D projection by MVU with fitted normal distribution

7.2.4 Maximum Mean Discrepancy Embedding (2008)

This method was introduced by Pan et. al. [60] in 2008 and is motivated by MVU [76]

The goal of Maximum Mean Discrepancy Embedding (MMDE) is to project the source and target data in such a way, that the distributions of the projections are close to each other. In other words, the distances between the distributions of the projected data are minimized. This is different from the above-mentioned methods, as they mainly focus on maximizing the data variance rather than minimizing the distances between the distributions. Let X^S and X^T describe the distributions of respectively the source and target data. Then, the MMD metric is minimized, i.e.:

$$\text{MMD}(X^S, X^T) = ||\frac{1}{n^S}\sum_{i=1}^{n^S}\phi(\boldsymbol{x}_i^S) - \frac{1}{n^T}\sum_{i=1}^{n^T}\phi(\boldsymbol{x}_i^T)||_{\mathcal{H}}^2, \tag{7.8}$$

for some $\phi \in \mathcal{H}$. Here, n^S and n^T are the sample sizes of respectively the source and target domain. Let again $k(\boldsymbol{x_i}, \boldsymbol{x_j}) = \phi(\boldsymbol{x}_i)^\top\phi(\boldsymbol{x}_j)$ be the associated universal kernel. Then, by applying the kernel

[1] https://github.com/buquicchiol/MaximumVarianceUnfolding

trick, Equation 7.8 can be written as:

$$\text{MMD}(X^S, X^T) = Tr(KL)$$

where:

$$K = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S} & K_{T,T} \end{bmatrix} \in \mathbf{R}^{(n^S+n^T)\times(n^S+n^T)} \tag{7.9}$$

in which K_S and K_T are defined by k on the data in respectively the source and target domain, and $L \succeq 0$ with

$$L_{ij} = \begin{cases} \frac{1}{(n^S)^2} & x_i, x_j \in X^S \\ \frac{1}{(n^T)^2} & x_i, x_j \in X^T \\ -\frac{1}{n^S n^T} & \text{other.} \end{cases}$$

The goal is to learn the kernel matrix K instead of the universal kernel k. For this, the learned kernel matrix must correspond however to a universal kernel since we want K to be positive definite in order to apply PCA. For this, the following property can be used:

Theorem 7.2.2. *A kernel is universal if, for any arbitrary set of distinct points, it induces strictly positive definite kernel matrices.*

There is a specific group of strictly positive definite kernel matrices which also induce universal kernels, namely:

Theorem 7.2.3. *If a kernel matrix K can be written as*

$$K = \tilde{K} + \epsilon I$$

where $\epsilon > 0, \tilde{K} \succeq 0$ and I is the identity matrix, then the kernel function corresponding to K is universal.

Therefore, the learned kernel matrix should be of this form.

Besides minimizing the trace of KL, the constraints are set as in the MVU problem, i.e.:

- Positive semi-definite Kernel matrix $K \succeq 0$,
- The embedded data is centered, i.e. $\sum_{ij} K_{ij} = 0$,
- Local isometry, i.e. $K_{ii} + K_{jj} - 2K_{ij} = ||x_i - x_j||^2$ where $(i, j) \in \eta$ (x_i, x_j are k-nearest neighbors),
- The trace of K is maximized.

This problem can be formulated as the following optimization problem:

$$\begin{aligned} \min_{K=\tilde{K}+\epsilon I} \quad & Tr(KL) - \lambda Tr(K) \\ \text{s.t.} \quad & K \succeq 0, \\ & \sum_{ij} K_{ij} = 0, \\ & K_{ij} + K_{jj} - 2K_{ij} = ||x_i - x_j||^2, \\ & \forall i, j \quad s.t. \quad \eta_{ij} = 1. \end{aligned} \tag{7.10}$$

Here, $\epsilon > 0$ is a small positive constant and the relative weightings are controlled by $\lambda \geq 0$, which is chosen empirically or via cross-validation. Then, Equation 7.10 is equivalent to:

$$\begin{aligned} \min_{\tilde{K} \succeq 0} \quad & Tr(\tilde{K}L) - \lambda Tr(\tilde{K}) \\ \text{s.t.} \quad & K \succeq 0, \\ & \sum_{ij} \tilde{K}_{ij} = -\epsilon, \\ & \tilde{K}_{ij} + \tilde{K}_{jj} - 2\tilde{K}_{ij} + 2\epsilon = ||x_i - x_j||^2, \\ & \forall i, j \quad s.t. \quad \eta_{ij} = 1. \end{aligned} \tag{7.11}$$

Again, for solving this problem, the same solver is used as in the MVU problem. When $\tilde{K}$ is found, the principal components can be found similar as in MVU. This leads to the results in Figure 7.5. Clearly, the space learned by MMDE consists of two datasets close to each other. This is why MMDE

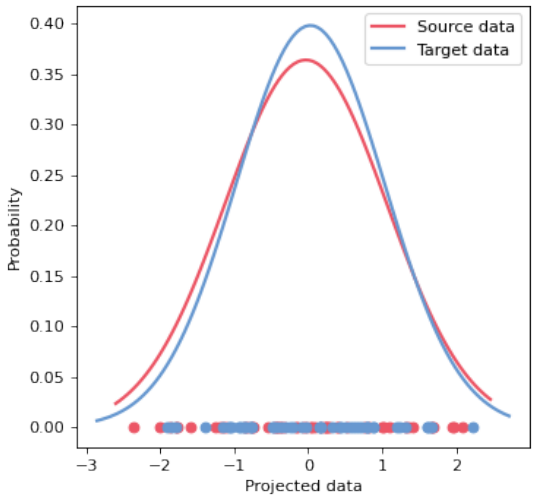


Figure 7.5: 1D projection by MMDE with fitted normal distribution

can help transfer learning effectively. MMDE has a few limitations. The most important limitation is that since the problem is formulated as an SPD problem, it is computationally expensive and therefore less attractive to use. Being SDP also required K to be semi-positive definite and needs post-processing by PCA, which may discard useful information in K. This is a motivation for using Transfer Component Analysis (TCA) [48].

7.3 Additional figures for the results

7.3.1 Maximum of the Zk-statistic

In Figure 7.6, the maximum Zk-statistic is shown based on the number of observations. It can be seen that there is a linear relationship between the total number of observations and the maximum value of the Zk-statistic.

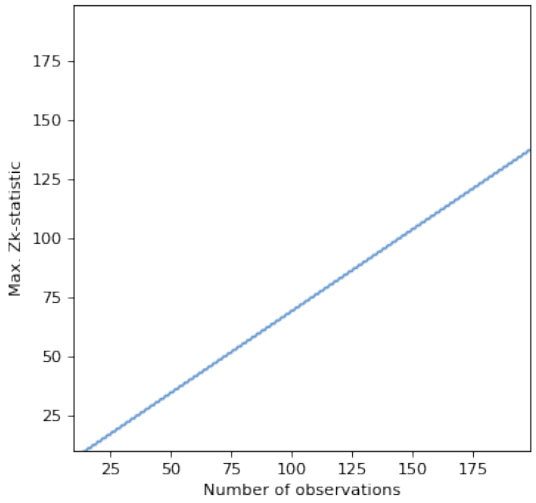
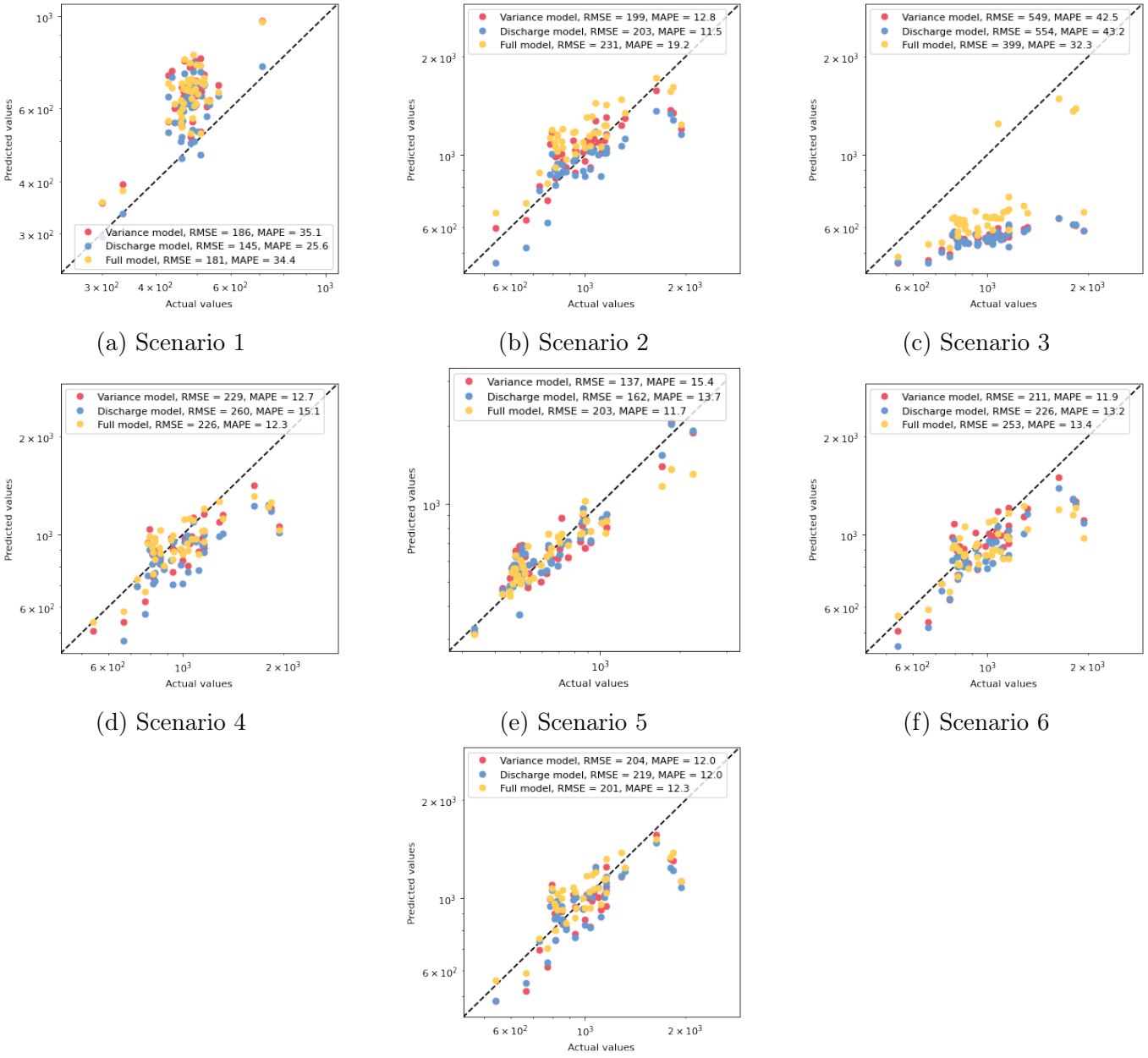


Figure 7.6: Maximum Zk-statistic based on the total number of observations of the source and target data

7.3.2 Prediction results without transfer

In this section, the results without transfer are plotted and some results are explained. In Figure 7.7, the prediction results for all 3 models are given for all scenarios. It can be seen that the variance model for most scenarios outperforms the discharge model and sometimes also the full model. This could be due to the fact that the inclusion of more features leads to overfitting and therefore worse results.



(a) Scenario 1 (b) Scenario 2 (c) Scenario 3

(d) Scenario 4 (e) Scenario 5 (f) Scenario 6

(g) Scenario 7

Figure 7.7: Prediction results before transfer for the variance, discharge and full model.

In scenario 1, it is observed that all the models tend to overestimate most values and the labels are densely packed around similar values with only one outlier value at the upper end and a few at the

lower end.

For scenario 2, it appears that the discharge and full models predict the average values with less accuracy compared to the variance model. The variance predicts the low values accurately but also underestimates them with the discharge model. For higher values, both models underestimate the labels, whereas the full model performs better in predicting these irregular values, possibly due to the inclusion of more features, which helps identify underlying patterns leading to more accurate predictions.

This pattern of overestimating average values and underestimating tail values, especially by the variance and discharge model, is also observed in scenarios 4, 6, and 7.

For scenario 3, many labels are highly underestimated. Only for a few tail values on the higher end, the full model is able to improve the predictions, leading to much lower RMSE and MAPE values.

Finally, in scenario 5, it is found that many values are accurately predicted by all models. However, the full model tends to underestimate the labels for higher tail values, which is reflected in the RMSE score that places a higher penalty on mispredicted high values. The full model outperforms the others in predicting average values, as evidenced by the MAPE score. In contrast, the variance and discharge models are better at predicting the higher tail values, but their performance for predicting average values is weaker.

7.4 Classification methods

In this section, the classification setting for transfer learning in our setting is explained and later on applied. The feature set of this classification model is similar to in [37]. Therefore, the main focus is on explaining the methods and results. It is encouraged to look into the original article for more details. The most important note here is that in the classification setting, only data from the first 5 cycles is used.

7.4.1 Joint Distribution Adaptation (2013)

Another domain adaptation method, which can also be seen as an extension of TCA, was introduced by Long et. al [79]. For this method, the scenario is assumed that both the marginal and conditional distributions are different. The idea of this method is also to reduce the distance between the source and target marginal distributions $P_S(X^S)$ and $P_T(X^T)$ using the MMD criterion this is done in a similar way as TCA. Basically, a map is introduced which minimizes the MMD by using the kernel trick:

$$D(\phi(X^S), \phi(X^T)) = Tr((KWW^\top K)L) = Tr(W^\top KLKW). \tag{7.12}$$

For clarity further on, define $L_0 := L$. Now, JDA also focuses on adapting the conditional source and target distributions $P_S(Y^S|X^S)$ and $P_T(Y^T|X^T)$ in such a way, that they become similar as well. It is nontrivial, however, to match these conditional distributions since we assume that there are no target

labels available. JDA proposes the exploration of pseudo-labels for the target domain in order to match the conditional distributions. These pseudo-labels are calculated using known methods such as standard learners or transfer learners. It can be sufficient to estimate $P_S(Y^S|X^S)$ and $P_T(Y^T|X^T)$ using the class-conditional distributions $P_S(X^S|Y^S)$ and $P_T(X^T|Y^T)$ for large datasets [79]. The class-conditional distributions can be matched by using the true source labels and pseudo target labels: $P_S(X^S|Y^S = c)$ and $P_T(X^T|\hat{Y}_T = c)$ w.r.t. each class $c \in \{1, ..., C\}$ in the label set. Note that this method is developed for classification and therefore, if applied to a regression problem, an extension should be introduced. Again, the MMD is used to measure the distance between the class-conditional distributions.

$$||\frac{1}{n^S}^{(c)} \sum_{x_i \in \mathcal{D}_S^{(c)}} W^\top x_i - \frac{1}{n^T}^{(c)} \sum_{x_i \in \mathcal{D}_T^{(c)}} W^\top x_i||^2 = Tr((KWW^\top K)L_c) = Tr(W^\top K L_c K W) \tag{7.13}$$

where $\mathcal{D}_S^{(c)}$ contains the source data, with all available true labels and $n^{S(c)} = |\mathcal{D}_s^{(c)}|$. This means that when all source data is labeled, $\mathcal{D}_S^{(c)} = \mathcal{D}_S$ and $n^{S(c)} = n^S$. The class-target domain $\mathcal{D}_T^{(c)}$ contains all the data from the original target domain with pseudo (predicted) labels corresponding to class c, i.e. $\mathcal{D}_T^{(c)} = \{\boldsymbol{x}_j : \boldsymbol{x}_j \in \mathcal{D}_t \text{ and } \hat{y}(\boldsymbol{x}_j) = c\}$. Furthermore, $n^{T(c)} = |\mathcal{D}_t^{(c)}|$ and each MMD matrix L_c for class c is calculated by:

$$[L_c]_{ij} = \begin{cases} \frac{1}{n^{S(c)} n^{S(c)}} & x_i, x_j \in \mathcal{D}_S^{(c)} \\ \frac{1}{n^{T(c)} n^{T(c)}} & x_i, x_j \in \mathcal{D}_T^{(c)} \\ -\frac{1}{n^{S(c)} n^{T(c)}} & x_i \in \mathcal{D}_S^{(c)},\ x_j \in \mathcal{D}_T^{(c)}. \\ -\frac{1}{n^{S(c)} n^{T(c)}} & x_i \in \mathcal{D}_T^{(c)},\ x_j \in \mathcal{D}_S^{(c)}. \\ 0 & \text{other.} \end{cases}$$

As JDA aims to adapt both the marginal and conditional distribution in such a way, that they become similar, we could formally represent

$$\begin{aligned} D(\mathcal{D}_S, \mathcal{D}_T) &\approx D(\phi(X^S), \phi(X^T)) + D(P(Y^S|X^S), P(Y^T|X^T)) \\ &= Tr(W^\top K L_0 K W) + \sum_{c=1}^{C} Tr(W^\top K L_c K W). \end{aligned} \tag{7.14}$$

This leads to the resulting minimization problem:

$$\boxed{\begin{aligned} \min_W \quad & Tr(W^\top K L K W) + \sum_{c=1}^{C} Tr(W^\top K L_c K W) + \mu Tr(W^\top W) \\ \text{s.t.} \quad & W^\top K H K W = I. \end{aligned}} \tag{7.15}$$

Again, a regularization parameter $\mu > 0$ is added to control the complexity of matrix W. Important to note is that TCA is a special case of JDA with $C = 0$. Now, Equation 7.15 can be rewritten to a trace maximization Equation 7.16 using the Lagrangian [48].

$$\boxed{\max_W \quad Tr((W^\top (K(L_0 + \sum_{c=1}^{C} L_c)K + \mu I_m)W)^{-1}(W^\top K H K W)).} \tag{7.16}$$

Again, the resulting matrix W contains the m leading eigenvectors of $(K(L_0+\sum_{c=1}^{C} L_c)K+\mu I_m)^{-1}KHK$. The procedure is to predict L_0 similar to TCA. Then, use a model to predict pseudo labels for the target data. From here, construct the MMD matrices and solve the eigendecomposition problem from Equation 7.16. Finally, train a model on transformed data to predict the labels of the target data.

7.4.2 Balanced Distribution Adaptation (2017)

This model is an extension of JDA. It suggests that just matching both marginal and conditional distributions is not enough as they are not necessarily equally important. Therefore, BDA adds a balance factor λ to leverage the importance of the distributions:

$$\begin{aligned} D(\mathcal{D}_S, \mathcal{D}_T) &\approx (1-\lambda)D(\phi(X^S), \phi(X^T)) + \lambda D(P(Y^S|X^S), P(Y^T|X^T) \\ &= (1-\lambda)Tr(W^\top K L_0 K W) + \lambda \sum_{c=1}^{C} Tr(W^\top K L_c K W) \end{aligned} \tag{7.17}$$

where $\lambda \in [0, 1]$. Note the following:

- When $\lambda \to 0$, the datasets are less similar, therefore the marginal distributions are more important to adapt.
- When $\lambda \to 1$, the datasets are more similar, therefore the conditional distributions are more important to adapt.
- When $\lambda = 0$, the method is actually the same as TCA.
- When $\lambda = 1$, the method is actually the same as JDA.

The rest of the method is similar to JDA, therefore, only the final trace optimization problem is formulated in Equation 7.18.

$$\boxed{\max_{W} \quad Tr((W^\top (K((1-\lambda)L_0 + \lambda \sum_{c=1}^{C} L_c)K + \mu I_m)W)^{-1}(W^\top KHKW)).} \tag{7.18}$$

7.4.3 Logistic Elastic Net

In [37], logistic regression is used for classification with an $l1$-regularization penalty. In order to keep things in line with the method used in the regression model, we will use Logistic Elastic Net regression, which is suitable for both the variable selection before transfer and predictions after transfer. It also appears that in the Python package[2], this method is possible in combination with the 'Saga' solver which is much faster than the only other solver which is suitable for the $l1$-penalty.

[2] sklearn.linear_model.LogisticRegressionCV

Logistic Elastic Net [80] is a type of regularized regression that combines the L1 and L2 penalties of Ridge and Lasso regression. It is used for binary classification problems and aims to find the optimal balance between the two types of regularization to improve the prediction accuracy and interpretability of the model. The L1 regularization term encourages sparse solutions and helps to select a subset of important predictors, while the L2 regularization term prevents overfitting by shrinking the coefficients of less important predictors. This is similar to what we have seen in the regression version of this method.

The mathematical definition of logistic Elastic Net is a variation of the logistic regression model, where the objective function is the negative log-likelihood of the data, subject to a combination of L1 and L2 regularization. The logistic elastic net loss function can be defined as

$$L = -\frac{1}{n}\sum_{i=1}^{n}(y_i)\log(p_i) + (1-y_i)\log(1-p_i)) + \alpha((1-\lambda_i)\frac{1}{2}\sum_{j=1}^{n} w_j^2 + \lambda_i \sum_{j=1}^{n}|w_j|). \tag{7.19}$$

Here, n is the number of observations, y_i is the label corresponding to sample i, p_i is the predicted probability of y_i, w_j is the $j-th$ coefficient and α, λ_1 are the regularization parameters, similar as in Section 2.4.

7.4.4 Prediction metric

Similar to [37], accuracy will be used as a measure for the performance of the prediction. In classification, accuracy is a commonly used metric to evaluate the performance of a model. It is defined as the ratio of the number of correctly classified instances to the total number of instances. The accuracy of a model is computed as the proportion of correctly classified instances to the total number of instances in the dataset. It can be represented mathematically as:

$$\text{Accuracy} = \frac{\text{number of correctly classified instances}}{\text{total number of instances}}$$

It can be represented as a percentage, where a higher percentage indicates better performance of the model. The accuracy metric does not provide information about the specific errors made by the model. Furthermore, if the train set contains mainly 1 type of class, the model highly tends to predict this class more likely for all labels in the test set. Therefore, the model will give good predictions in for example scenario 3 as 90% of the labels are out of 1 class and only a few labels are in the other class. Therefore, the accuracy metric can be misleading when the dataset is imbalanced, meaning that it contains a disproportionate ratio of observations in each class.

7.5 Classification Results

For the classification results, similar scenarios and variable selection methods are used as described in Chapter 4. In [37], the classification is only applied to the variance and full model, therefore, only results will be given for these 2 models.

For the transfer, BDA will be used as it is a combination of TCA and BDA. Depending on μ, the most suitable model will be chosen, or a combination of both. When $\mu = 0$, TCA as used while when $\mu = 1$, JDA is preferred. The rest of the method is similar to the regression setting such as choosing the number of transfer components and the variable selection methods.

It is noteworthy that before presenting the results, pseudo-labels are created for the target data in order to compute the necessary conditional probabilities. It is crucial that these pseudo-labels for the target data include a representation of all unique labels present in the source dataset. This is also described in Subsection 7.5.1.

Since in the dataset, batches 1 and 3 have much higher labels compared to batch 2. This leads to the fact that setting a threshold of 550 cycles, leads to almost only 1 type of label in batch 2, while batches 1 and 3 contain almost only the other label. When training the model in scenarios 1, 2 and 3, the labels of the source and target data appear to be too similar or dissimilar leading to assigning all labels to 1 class and none to the other. This is also the case for scenario 4 in the full model. Therefore, the pseudo-labels that are generated are not suitable for transfer. This could be seen as a limitation of these methods as source and target data by definition do not have similar distributions and therefore, it is not uncommon that label data can vary a lot between these sets. Because of this limitation, the results for the variance and full model will only be shown for these scenarios.

Finally, while in the regression setting 'b2c1' was deleted, it will not be deleted in the classification setting. Furthermore, the temperature integral is not clearly defined, which has led to a different interpretation of this feature. The original article [37] stresses the importance of this feature, which could have an effect on our results.

7.5.1 Labels of the data

In the classification setting, all instances with lifetimes below 550 cycles get assigned a 'Low' and all instances with lifetimes (strictly) higher than 550 get assigned a 'High'. The labels are divided as follows:

- Batch 1 contains 40 batteries with a 'high' lifetime and 1 battery with a 'low' lifetime.
- Batch 2 contains 2 batteries with a 'high' lifetime and 41 batteries with a 'low' lifetime.
- Batch 3 contains 39 batteries with a 'high' lifetime and 1 battery with a 'low' lifetime.

This confirms the disbalance in the labels, as described earlier. This stresses the reason for only using

scenarios 4,5,6 and 7, which are combinations of the batches and therefore include also instances from batch 2.

7.5.2 Variance classifier

Since the *AllTL* method and *SigTL* method are the same for the variance model, only results will be shown for the *AllTL* method. In Table 7.1, the results are shown for the variance classifier.

Scenario	No TL	AllTL	Relative change NoTL / AllTL	Mu	CoefTL	Relative change NoTL / CoefTL	Mu
4	95.0%	97.5%	-2.6%	1.00	97.5%	-2.6%	1.00
5	79.1%	79.1%	0.0%	1.00	79.1%	0.0%	0.20
6	95.0%	97.5%	-2.6%	1.00	97.5%	-2.6%	1.00
7	95.0%	97.5%	-2.6%	1.00	97.5%	-2.6%	1.00

Table 7.1: Accuracy's of the predictions for the classification setting (Variance Classifier)

The results for the *NoTL* are for scenario 4,6 and 7 the same. Only 2 batteries are misclassified. The transfer for both variable selection methods improves upon these results and only misclassifies 1 instance. Note that the balance parameter μ shows that JDA is preferred for most models.

7.5.3 Full classifier

Scenario	No TL	AllTL	Relative change NoTL / AllTL	Mu	SigTL	Relative change NoTL / SigTL	Mu	CoefTL	Relative change NoTL / CoefTL	Mu
Scenario 5	88.4%	93.0%	-5.3%	0.7	90.7%	-2.6%	0.3	88.4%	0.0%	1.00
Scenario 6	45.0%	52.5%	-16.7%	0.7	97.5%	-116.7%	0.9	73.2%	-62.6%	0.60
Scenario 7	87.5%	97.6%	-11.5%	0.9	97.5%	-11.4%	0.9	97.5%	-11.5%	0.90

Table 7.2: Accuracy's of the predictions for the classification setting (Full Classifier)

In Table 7.2, the results are given for the full classifier. In all scenarios, at least one of the transfer methods improves upon the results without transfer.

References

[1] Rijksoverheid. Ontwerp Beleids-programma Klimaat, 2022. URL: https://www.rijksoverheid.nl/documenten/publicaties/2022/06/02/ontwerp-beleidsprogramma-klimaat.

[2] Chris Jongsma. Omslagpunt grootschalige batterijopslag, 2021. URL: https://ce.nl/publicaties/omslagpunt-grootschalige-batterijopslag/.

[3] Bruce Dunn, Haresh Kamath, and Jean Marie Tarascon. Electrical energy storage for the grid: A battery of choices. *Science*, 334(6058):928–935, 2011. URL: https://www.science.org/doi/10.1126/science.1212741, doi:10.1126/SCIENCE.1212741/SUPPL{_}FILE/DUNN-SOM.PDF.

[4] Richard Schmuch, Ralf Wagner, Gerhard Hörpel, Tobias Placke, and Martin Winter. Performance and cost of materials for lithium-based rechargeable automotive batteries. *Nature Energy 2018 3:4*, 3(4):267–278, 4 2018. URL: https://www.nature.com/articles/s41560-018-0107-2, doi:10.1038/s41560-018-0107-2.

[5] Wiljan Vermeer, Gautham Ram Chandra Mouli, and Pavol Bauer. A Comprehensive Review on the Characteristics and Modelling of Lithium-ion Battery Ageing. *IEEE Transactions on Transportation Electrification*, 2021. doi:10.1109/TTE.2021.3138357.

[6] S. Tamilselvi, S. Gunasundari, N. Karuppiah, Abdul Razak Rk, S. Madhusudan, Vikas Madhav Nagarajan, T. Sathish, Mohammed Zubair M. Shamim, C. Ahamed Saleel, and Asif Afzal. A Review on Battery Modelling Techniques. *Sustainability 2021, Vol. 13, Page 10042*, 13(18):10042, 9 2021. URL: https://www.mdpi.com/2071-1050/13/18/10042/htmhttps://www.mdpi.com/2071-1050/13/18/10042, doi:10.3390/SU131810042.

[7] Md Sazzad Hosen, Joris Jaguemont, Joeri Van Mierlo, and Maitane Berecibar. Battery lifetime prediction and performance assessment of different modeling approaches. *iScience*, 24(2):102060, 2 2021. doi:10.1016/J.ISCI.2021.102060.

[8] Shuangqi Li, Hongwen He, Chang Su, and Pengfei Zhao. Data driven battery modeling and management method with aging phenomenon considered. *Applied Energy*, 275:115340, 10 2020. doi:10.1016/J.APENERGY.2020.115340.

[9] Xuebing Han, Languang Lu, Yuejiu Zheng, Xuning Feng, Zhe Li, Jianqiu Li, and Minggao Ouyang. A review on the key issues of the lithium ion battery degradation among the whole life cycle. *eTransportation*, 1, 8 2019. doi:10.1016/J.ETRAN.2019.100005.

[10] Yunhe Yu, Nishant Narayan, Victor Vega-Garita, Jelena Popovic-Gerber, Zian Qin, Marnix Wagemaker, Pavol Bauer, and Miro Zeman. Constructing Accurate Equivalent Electrical Circuit Models of Lithium Iron Phosphate and Lead–Acid Battery Cells for Solar Home System Applications. *Energies 2018*, 11(9), 9 2018. URL: https://www.mdpi.com/1996-1073/11/9/2305/htmhttps://www.mdpi.com/1996-1073/11/9/2305, doi:10.3390/EN11092305.

[11] Aden Seaman, Thanh Son Dao, and John McPhee. A survey of mathematics-based equivalent-circuit and electrochemical battery models for hybrid and electric vehicle simulation. *Journal of Power Sources*, 256:410–423, 6 2014. doi:10.1016/J.JPOWSOUR.2014.01.057.

[12] Anthony Barré, Benjamin Deguilhem, Sébastien Grolleau, Mathias Gérard, Frédéric Suard, and Delphine Riu. A review on lithium-ion battery ageing mechanisms and estimations for automotive applications. *Journal of Power Sources*, 241:680–689, 11 2013. doi:10.1016/J.JPOWSOUR.2013.05.040.

[13] Yi Li, Kailong Liu, Aoife M. Foley, A. Zülke, Maitane Berecibar, E. Nanini-Maury, J. Van Mierlo, and Harry E. Hoster. Data-driven health estimation and lifetime prediction of lithium-ion batteries: A review. *Renewable and Sustainable Energy Reviews*, 113:109254, 10 2019. doi:10.1016/J.RSER.2019.109254.

[14] Venkatasailanathan Ramadesigan, Paul W. C. Northrop, Sumitava De, Shriram Santhanagopalan, Richard D. Braatz, and Venkat R. Subramanian. Modeling and Simulation of Lithium-Ion Batteries from a Systems Engineering Perspective. *Journal of The Electrochemical Society*, 159(3):R31, 1 2012. URL: https://iopscience.iop.org/article/10.1149/2.018203jeshttps://iopscience.iop.org/article/10.1149/2.018203jes/meta, doi:10.1149/2.018203JES.

[15] Malin Andersson, Moritz Streb, Jing Ying Ko, Verena Löfqvist Klass, Matilda Klett, Henrik Ekström, Mikael Johansson, and Göran Lindbergh. Parametrization of physics-based battery models from input–output data: A review of methodology and current research. *Journal of Power Sources*, 521:230859, 2 2022. doi:10.1016/j.jpowsour.2021.230859.

[16] Xiaosong Hu, Dongpu Cao, and Bo Egardt. Condition Monitoring in Advanced Battery Management Systems: Moving Horizon Estimation Using a Reduced Electrochemical Model. *IEEE/ASME Transactions on Mechatronics*, 23(1):167–178, 2 2018. doi:10.1109/TMECH.2017.2675920.

[17] Wendi Guo, Zhongchao Sun, Søren Byg Vilsen, Jinhao Meng, and Daniel Ioan Stroe. Review of "grey box" lifetime modeling for lithium-ion battery: Combining physics and data-driven methods. *Journal of Energy Storage*, 56:105992, 12 2022. doi:10.1016/J.EST.2022.105992.

[18] M. Varini, Pietro Elia Campana, and Göran Lindbergh. A semi-empirical, electrochemistry-based model for Li-ion battery performance prediction over lifetime. *Journal of Energy Storage*, 25:100819, 10 2019. doi:10.1016/J.EST.2019.100819.

[19] M. A. Hannan, M. S.H. Lipu, A. Hussain, and A. Mohamed. A review of lithium-ion battery state of charge estimation and management system in electric vehicle applications: Challenges and recommendations. *Renewable and Sustainable Energy Reviews*, 78:834–854, 10 2017. doi:10.1016/J.RSER.2017.05.001.

[20] Antti Aitio, Scott G. Marquis, Pedro Ascencio, and David Howey. Bayesian Parameter Estimation Applied to the Li-ion Battery Single Particle Model with Electrolyte Dynamics. *IFAC-PapersOnLine*, 53(2):12497–12504, 2020. doi:10.1016/j.ifacol.2020.12.1770.

[21] I. Bloom, B. W. Cole, J. J. Sohn, S. A. Jones, E. G. Polzin, V. S. Battaglia, G. L. Henriksen, C. Motloch, R. Richardson, T. Unkelhaeuser, D. Ingersoll, and H. L. Case. An accelerated calendar and cycle life study of Li-ion cells. *Journal of Power Sources*, 101(2):238–247, 10 2001. doi:10.1016/S0378-7753(01)00783-2.

[22] M. Broussely, S. Herreyre, P. Biensan, P. Kasztejna, K. Nechev, and R. J. Staniewicz. Aging mechanism in Li ion cells and calendar life predictions. *Journal of Power Sources*, 97-98:13–21, 7 2001. doi:10.1016/S0378-7753(01)00722-4.

[23] Kailong Liu, T. R. Ashwin, Xiaosong Hu, Mattin Lucu, and W. Dhammika Widanage. An evaluation study of different modelling techniques for calendar ageing prediction of lithium-ion batteries. *Renewable and Sustainable Energy Reviews*, 131:110017, 10 2020. doi:10.1016/J.RSER.2020.110017.

[24] Lifeng Wu, Xiaohui Fu, and Yong Guan. Review of the Remaining Useful Life Prognostics of Vehicle Lithium-Ion Batteries Using Data-Driven Methodologies. *Applied Sciences 2016, Vol. 6*, 6(6), 5 2016. URL: https://www.mdpi.com/2076-3417/6/6/166/htmhttps://www.mdpi.com/2076-3417/6/6/166, doi:10.3390/APP6060166.

[25] Shaheer Ansari, Afida Ayob, Molla Shahadat Hossain Lipu, Aini Hussain, and Mohamad Hanif Md Saad. Data-Driven Remaining Useful Life Prediction for Lithium-Ion Batteries Using Multi-Charging Profile Framework: A Recurrent Neural Network Approach. *Sustainability 2021*, 13(23):13333, 12 2021. URL: https://www.mdpi.com/2071-1050/13/23/13333/htmhttps://www.mdpi.com/2071-1050/13/23/13333, doi:10.3390/SU132313333.

[26] Daoquan Chen, Weicong Hong, and Xiuze Zhou. Transformer Network for Remaining Useful Life Prediction of Lithium-Ion Batteries. *IEEE Access*, 10:19621–19628, 2022. doi:10.1109/ACCESS.2022.3151975.

[27] Weihan Li, Haotian Zhang, Bruis van Vlijmen, Philipp Dechent, and Dirk Uwe Sauer. Forecasting battery capacity and power degradation with multi-task learning. *Energy Storage Materials*, 11 2022. URL: https://arxiv.org/abs/2111.14937v2, doi:10.48550/arxiv.2111.14937.

[28] Donal P. Finegan, Juner Zhu, Xuning Feng, Matt Keyser, Marcus Ulmefors, Wei Li, Martin Z. Bazant, and Samuel J. Cooper. The Application of Data-Driven Methods and Physics-Based Learning for Improving Battery Safety. *Joule*, 5(2):316–329, 2 2021. doi:10.1016/J.JOULE.2020.11.018.

[29] Donal P. Finegan and Samuel J. Cooper. Battery Safety: Data-Driven Prediction of Failure. *Joule*, 3(11):2599–2601, 11 2019. doi:10.1016/J.JOULE.2019.10.013.

[30] Abhinav Saxena, Brian Bole, Matthew Daigle, and Kai Goebel. Prognostics for Batteries Aging Experiments and Modeling NASA Battery Workshop 2012. 2012. URL: http://www.prognostics.nasa.gov].

[31] Stephen J. Harris, David J. Harris, and Chen Li. Failure statistics for commercial lithium ion batteries: A study of 24 pouch cells. *Journal of Power Sources*, 342:589–597, 2 2017. doi:10.1016/J.JPOWSOUR.2016.12.083.

[32] Xiaosong Hu, Jiuchun Jiang, Dongpu Cao, and Bo Egardt. Battery Health Prognosis for Electric Vehicles Using Sample Entropy and Sparse Bayesian Predictive Modeling. *IEEE Transactions on Industrial Electronics*, pages 1–1, 2015. doi:10.1109/TIE.2015.2461523.

[33] Yongzhi Zhang, Rui Xiong, Hongwen He, and Michael G. Pecht. Lithium-Ion Battery Remaining Useful Life Prediction With Box–Cox Transformation and Monte Carlo Simulation. *IEEE Transactions on Industrial Electronics*, 66(2):1585–1597, 2 2019. doi:10.1109/TIE.2018.2808918.

[34] Yongzhi Zhang, Rui Xiong, Hongwen He, and Michael G. Pecht. Long Short-Term Memory Recurrent Neural Network for Remaining Useful Life Prediction of Lithium-Ion Batteries. *IEEE Transactions on Vehicular Technology*, 67(7):5695–5705, 7 2018. doi:10.1109/TVT.2018.2805189.

[35] Zicheng Fei, Fangfang Yang, Kwok Leung Tsui, Lishuai Li, and Zijun Zhang. Early prediction of battery lifetime via a machine learning based framework. *Energy*, 225:120205, 6 2021. doi:10.1016/J.ENERGY.2021.120205.

[36] Calum Strange and Gonçalo dos Reis. Prediction of future capacity and internal resistance of Li-ion cells from one cycle of input data. *Energy and AI*, 5:100097, 9 2021. doi:10.1016/J.EGYAI.2021.100097.

[37] Kristen A. Severson, Peter M. Attia, Norman Jin, Nicholas Perkins, Benben Jiang, Zi Yang, Michael H. Chen, Muratahan Aykol, Patrick K. Herring, Dimitrios Fraggedakis, Martin Z. Bazant, Stephen J. Harris, William C. Chueh, and Richard D. Braatz. Data-driven prediction of battery cycle life before capacity degradation. *Nature Energy 2019 4:5*, 4(5):383–391, 3 2019. URL: https://www.nature.com/articles/s41560-019-0356-8, doi:10.1038/s41560-019-0356-8.

[38] D. Anseán, M. Dubarry, A. Devie, B.Y. Liaw, V.M. García, J.C. Viera, and M. González. Fast charging technique for high power LiFePO4 batteries: A mechanistic analysis of aging. *Journal of Power Sources*, 321:201–209, 7 2016. doi:10.1016/j.jpowsour.2016.04.140.

[39] D. Anseán, M. Dubarry, A. Devie, B.Y. Liaw, V.M. García, J.C. Viera, and M. González. Operando lithium plating quantification and early detection of a commercial LiFePO4 cell cycled under dynamic driving schedule. *Journal of Power Sources*, 356:36–46, 7 2017. doi:10.1016/j.jpowsour.2017.04.072.

[40] Ira Bloom, Andrew N. Jansen, Daniel P. Abraham, Jamie Knuth, Scott A. Jones, Vincent S. Battaglia, and Gary L. Henriksen. Differential voltage analyses of high-power, lithium-ion cells. *Journal of Power Sources*, 139(1-2):295–303, 1 2005. doi:10.1016/j.jpowsour.2004.07.021.

[41] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 4 2005. URL: https://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/full/10.1111/j.1467-9868.2005.00503.xhttps://onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/abs/10.1111/j.1467-9868.2005.00503.xhttps://rss-onlinelibrary-wiley-com.tudelft.idm.oclc.org/doi/10.1111/j.1467-9868.2005.00503.x, doi:10.1111/J.1467-9868.2005.00503.X.

[42] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Senior Member, Hui Xiong, and Qing He. A Comprehensive Survey on Transfer Learning. 2020.

[43] Sinno Jialin Pan and Qiang Yang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 10 2010. doi:10.1109/TKDE.2009.191.

[44] Kailong Liu, Qiao Peng, Yunhong Che, Yusheng Zheng, Kang Li, Remus Teodorescu, Dhammika Widanage, and Anup Barai. Advances in Applied Energy Transfer learning for battery smarter state estimation and ageing prognostics : Recent progress , challenges , and prospects. *Advances*

in Applied Energy, 9:100117, 2023. URL: https://doi.org/10.1016/j.adapen.2022.100117, doi:10.1016/J.ADAPEN.2022.100117.

[45] Zirui Wang, Zihang Dai, Barnabás Póczos, and Jaime Carbonell. Characterizing and Avoiding Negative Transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2018.

[46] A. Kolmogorov. Sulla determinazione empirica di una legge di distribuzione. *G. Ist. Ital. Attuari*, 4:83–91, 1933. URL: https://cir.nii.ac.jp/crid/1571135650766370304.

[47] Jin Zhang. Powerful two-sample tests based on the likelihood ratio. *Technometrics*, 48(1):95–103, 2 2006. doi:10.1198/004017005000000328.

[48] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain Adaptation via Transfer Component Analysis. *IEEE transactions on neural networks*, 2010.

[49] Qiang Yang, Yu Zhang, Wenyuan Dai, and Sinno Jialin Pan. Transfer Learning. *Transfer Learning*, 1 2020. URL: https://www-cambridge-org.tudelft.idm.oclc.org/core/books/transfer-learning/CCFFAFE3CDBC245047F1DEC71D9EF3C7, doi:10.1017/9781139061773.

[50] Michael I. Jordan. Inner products and positive semi-definite matrices, 2017. URL: https://people.eecs.berkeley.edu/~jordan/kernels/.

[51] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, Grisel O., Blondel M., Prettenhofer P., R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html.

[52] S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 3 1951. doi:10.1214/aoms/1177729694.

[53] D.M. Endres and J.E. Schindelin. A new metric for probability distributions. *IEEE Transactions on Information Theory*, 49(7):1858–1860, 7 2003. doi:10.1109/TIT.2003.813506.

[54] L.M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1 1967. doi:10.1016/0041-5553(67)90040-7.

[55] Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring Statistical Dependence with Hilbert-Schmidt Norms. In *Algorithmic Learning Theory: 16th International Conference, ALT 2005*, pages 63–77, Singapore, 10 2005. Springer Berlin Heidelberg. doi:10.1007/11564089{_}7.

[56] N. Smirnov. Table for Estimating the Goodness of Fit of Empirical Distributions. 19(2):279–281, 6 1948. URL: https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-19/issue-2/Table-for-Estimating-the-Goodness-of-Fit-of-Empirical-Distributions/10.1214/aoms/1177730256.fullhttps://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-19/issue-2/Table-for-Estimating-the-Goodness-of-Fit-of-Empirical-Distributions/10.1214/aoms/1177730256.short.

[57] John H.J. Einmahl and Ian W. Mckeague. Empirical likelihood based hypothesis testing. 9(2):267–290, 4 2003. URL: https://projecteuclid.org/journals/bernoulli/volume-9/issue-2/Empirical-likelihood-based-hypothesis-testing/10.3150/bj/1068128978.fullhttps://projecteuclid.org/journals/bernoulli/volume-9/issue-2/Empirical-likelihood-based-hypothesis-testing/10.3150/bj/1068128978.short.

[58] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Source: Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[59] Arthur E. Hoerl and Robert W. Kennard. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55, 2 1970. doi:10.2307/1267351.

[60] Sinno Jialin Pan, James T Kwok, and Qiang Yang. Transfer Learning via Dimensionality Reduction. In *AAAI Conference on Artificial Intelligence*, 2018. URL: www.aaai.org.

[61] Cort J. Willmott and Kenji Matsuura. Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance. *Climate Research*, 30(1):79–82, 12 2005. doi:10.3354/CR030079.

[62] Delft High Performance Computing Centre (DHPC). DelftBlue Supercomputer (Phase 1), 2022. URL: https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1.

[63] Brady Neal. On the Bias-Variance Tradeoff: Textbooks Need an Update. *arXiv:1912.08286*, 12 2019.

[64] Thomas Cokelaer. Python package: Cokelaer/fitter: Fit data to many distributions, 2019. URL: https://fitter.readthedocs.io/en/latest/index.html.

[65] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 12 1974. doi:10.1109/TAC.1974.1100705.

[66] Gideon Schwarz. Estimating the Dimension of a Model. *The Annals of Statistics*, 6(2), 3 1978. doi:10.1214/aos/1176344136.

[67] N. L. Johnson. Systems of Frequency Curves Generated by Methods of Translation. *Biometrika*, 36(1/2):149, 6 1949. doi:10.2307/2332539.

[68] D. Milmo. Microsoft confirms multibillion dollar investment in firm behind ChatGPT, 2023. URL: https://www.theguardian.com/technology/2023/jan/23/microsoft-confirms-multibillion-dollar-investment-in-firm-behind-chatgpt.

[69] J. Jongerius. Zorgen om AI: 'Ik hoop toch dat kranten en tijdschriften gebruik blijven maken van authentiek werk', 2023. URL: https://www.villamedia.nl/artikel/zorgen-om-ai-ik-hoop-toch-dat-kranten-en-tijdschriften-gebruik-blijven-maken-van-authentiek-werk.

[70] J. Schellevis and S. Moerland. ChatGPT glipt langs docenten: 'Ik gebruik het om snel huiswerk te maken', 2023. URL: https://nos.nl/artikel/2460020-chatgpt-glipt-langs-docenten-ik-gebruik-het-om-snel-huiswerk-te-maken.

[71] Lorna McGregor, Daragh Murray, and Vivian Ng. International human rights law as a framework for algorithmic accountability. *International & Comparative Law Quarterly*, 68(2):309–343, 2019. URL: https://www.cambridge.org/core/journals/international-and-comparative-law-quarterly/article/international-human-rights-law-as-a-framework-for-algorithmic-accountability/1D6D0A456B36BA7512A6AFF17F16E9B6, doi:10.1017/S0020589319000046.

[72] Joel Laity. Principal component analysis: pictures, code and proofs, 2018. URL: https://joellaity.com/2018/10/18/pca.html.

[73] Cosma Shalizi. Principal Components Analysis, 2011.

[74] Bernhard Schölkopf, Alexander Smola, Klaus-Robert Müller, and Sungsu Lim. Introduction PCA in Feature Space Kernel PCA Experiments Nonlinear Variants of Other Algorithms Nonlinear Component Analysis as a Kernel Eigenvalue Problem. 1996.

[75] O. Chapelle, P. Haffner, and V.N. Vapnik. Support vector machines for histogram-based image classification. *ieeexplore.ieee.org*, 10(5), 1999. URL: https://ieeexplore.ieee.org/abstract/document/788646/.

[76] Kilian Q Weinberger, Fei Sha, and Lawrence K Saul. Learning a kernel matrix for nonlinear dimensionality reduction. 2004. URL: https://repository.upenn.edu/cis_papersThisconferencepaperisavailableatScholarlyCommons:https://repository.upenn.edu/cis_papers/2, doi:https://doi.org/10.1145/1015330.1015345.

[77] Rajat Mittal. Semidefinite programming, 2014. URL: https://www.cse.iitk.ac.in/users/rmittal/prev_course/s14/course_s14.php.

[78] Raviteja Vemulapalli. Maximum Variance Unfolding (MVU). 2013.

[79] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S Yu. Transfer Feature Learning with Joint Distribution Adaptation. 2014. doi:10.1109/ICCV.2013.274.

[80] Srikanth Ryali, Kaustubh Supekar, Daniel A. Abrams, and Vinod Menon. Sparse logistic regression for whole-brain classification of fMRI data. *NeuroImage*, 51(2):752–764, 6 2010. doi:10.1016/j.neuroimage.2010.02.040.