University of Bergen
Department of Informatics

Machine Teaching for Explainable AI in Industry: A Novel Approach for Time Series Classifiers

Author: Sebastian Einar Salas Røkholt
Supervisors: Jan Arne Telle & Kristian Flikka

UNIVERSITETET I BERGEN
Fakultet for naturvitenskap og teknologi

March, 2026

Abstract

Explainable Artificial Intelligence (XAI) for time series models remains difficult because many established explanation techniques are designed for static data and often produce outputs that are hard for humans to interpret when applied to temporal signals. This thesis investigates whether principles from Machine Teaching can be used to construct example-based explanations that improve understanding of a black-box time series classifier in an industrial setting.

The work is situated within the Machine Teaching for Explainable AI (MT4XAI) research project and uses electric vehicle charging-session data from Eviny as a practical case. A forecasting-based anomaly detection pipeline was developed in which a multi-horizon Long Short-Term Memory network predicts charging behaviour and session-level forecast errors are transformed into binary anomaly labels. To improve interpretability, Optimal Robust Simplifications (ORS) were computed for charging curves, producing piecewise linear representations that preserve the classifier's decision while reducing visual complexity. These simplified examples were then organised into a class-balanced teaching pool and selected into compact teaching sets using a facility-location objective designed to maximise behavioural coverage while limiting redundancy.

The thesis further operationalises MT4XAI through an experimental evaluation using multimodal LLMs as controlled proxy learners. Six experimental conditions were designed to compare different explanation modalities and teaching orders, including curriculum-based presentation, randomised teaching order, simplified-only examples, raw-only examples, and overlay-based representations. Performance was measured through pre-teaching and post-teaching simulate-the-model tasks under balanced exam conditions.

The results show that the forecasting model provides a sufficiently stable basis for anomaly scoring, that ORS can generate robust simplifications across a large set of charging sessions, and that teaching sets can be constructed as a reproducible end-to-end explanation pipeline. The MLLM experiment indicates that representation and ordering influence post-teaching performance, although the findings also highlight methodological limitations when using frozen language models as proxies for human learning. Overall, the thesis demonstrates how Machine Teaching can be translated from recent theory into a practical XAI pipeline for multivariate time series classification, while identifying important constraints for future human-centred evaluation.

Acknowledgements

I would first like to thank my wife, Camilla, for her patience and unwavering support throughout my work on this project. I am also deeply grateful to my supervisors, Jan Arne Telle and Kristian Flikka, for their continued guidance, encouragement, and valuable feedback. I further appreciate the support of the MT4XAI research group at the University of Bergen and Universitat Politècnica de València for generously sharing ideas and perspectives that helped shape this project. A particular thank you goes to Brigt Håvardstun, who offered feedback and guidance on multiple occasions despite having no formal obligation to do so. Finally, I would like to thank everyone who has shown interest in my thesis topic. Your enthusiasm has been both motivating and greatly appreciated!

Sebastian Einar Salas Røkholt

Monday 16th March, 2026

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and Motivation

Artificial intelligence (AI) has become a central tool in modern decision-making across industry, healthcare, and society at large. Systems based on deep learning, probabilistic models, and other machine learning (ML) approaches are now embedded in critical applications such as disease diagnosis, fraud detection, industrial process control, and energy management. What these domains have in common is the demand for high accuracy and efficiency, but also for transparency. If the reasoning of an AI system cannot be understood, it becomes difficult to trust, audit, or improve. This concern has been one of the main driving forces behind the increased interest for *Explainable AI (XAI)*, a field dedicated to providing human-understandable explanations for AI outputs [1, 4].

1.1.1 The challenge of time series data

While XAI research has achieved meaningful progress in areas such as image and text classification, *time series data* presents unique difficulties. Unlike images, which can be visually inspected, or text, which humans interpret naturally through language, time series are abstract sequences of values over time. The meaning of their patterns is often domain-specific and not intuitive to non-experts [49, 95]. The following examples from different domains illustrate this challenge:

- **Healthcare**: Electrocardiograms (ECGs) record a heart's electrical activity over time. Clinicians learn to recognise anomalies that indicate e.g. the early stages of heart attacks, while accurate interpretation of ECGs is difficult for those without specialised training.[19].
- **Finance**: Modern electronic markets continuously generate multivariate data through the limit-order book (LOB). Predicting short term price movements requires interpreting rapid changes across many interdependent variables such as bid-ask depths, order arrival rates, and liquidity imbalances. These interactions form a high-dimensional, rapidly evolving time series that is difficult for humans to interpret directly [97].
- **Cybersecurity**: Intrusion detection relies on monitoring network traffic patterns. Anomalous activity may only be detectable by subtle deviations in timing or frequency, which humans struggle to recognise unaided [6].
- **Energy systems**: Load curves representing electricity consumption evolve across days and weeks, where unusual fluctuations may signal inefficiency or faults [95].
- **Transportation**: Vehicle telemetry, including engine vibrations, brake pressure, and GPS trajectories, generates multivariate time series where anomalous events may be hidden within noisy signals [63].

In all these settings, humans are poorly equipped to interpret long, multidimensional sequences. People tend to focus on a few salient peaks or drops, and may overlook less obvious but crucial dependencies. This cognitive mismatch means that black-box time series models risk becoming tools whose outputs are accepted or rejected without true understanding [96].

There are several reasons why time series are particularly difficult for human interpretation:

1. **Temporal length**: Sequences often contain hundreds or thousands of points. Unlike images, where the whole object is visible at once, time series must be "read" across time, which strains working memory [65].

2. **Multivariate complexity**: Many applications involve multiple signals recorded simultaneously, e.g., power, state-of-charge, and ambient temperature in electric vehicle (EV) charging. The interplay between these variables can be non-linear and unintuitive.

3. **Lack of semantic anchors**: In text, words carry meaning, and in images, shapes can be labelled. In time series, the same rise or dip may have different implications depending on context. For example, as a battery approaches full charge, the power accepted by the battery decreases significantly. However, should the same pattern occur at 40% charge, it might indicate a fault.

4. **Hidden dependencies**: Important relationships may occur across distant time steps. Humans struggle to track these long-range dependencies without computational assistance [51].

1.1.2 Time series modelling and interpretability paradigms

This thesis studies sequence-level time series anomaly detection (SL-TSAD) on multivariate time series from EV charging sessions. Concretely, we build a system that assigns each session a binary label ('normal' or 'abnormal') by computing an anomaly score from forecasting errors and comparing it to a threshold. Although the underlying model is trained through self-supervised forecasting, the deployed system functions as a binary classifier once the anomaly score is thresholded. The formal definitions, modelling approaches and learning paradigms for time series classification (TSC) and SL-TSAD are provided in section 2.1.

One direction for explainability is to highlight which parts of the input drive a model's decisions, but for time series such explanations can be cognitively demanding and may feel fragmented. Common post-hoc approaches attempt this by assigning importance to individual time steps, variables, or subsequences, for example through feature attribution methods such as SHAP and LIME, discriminative subsequences such as shapelets, or counterfactual perturbations. While such methods can reveal locally important parts of an input, they often present isolated fragments rather than a coherent explanation of the full temporal behaviour. For long multivariate sequences, users must still reconstruct how these fragments relate to the model's overall decision, which can impose substantial cognitive load. This motivates simplified, example-based explanations that aim to preserve decision-relevant temporal structure while presenting it as an interpretable whole rather than as disconnected local cues [73, 48].

1.1.3 MT4XAI: Example-based explanations through machine teaching

The research project *Machine Teaching for Explainable AI (MT4XAI)* aims to address these interpretability challenges by reframing explanations of AI behaviour as a teaching problem [94, 111, 27]. Instead of overwhelming users with all possible examples or post-hoc rules, MT4XAI identifies a small, optimised *teaching set* of examples that illustrate the concept a model has learned.

This builds on the theory of *machine teaching (MT)*, which studies how a teacher can teach a concept to a learner as efficiently as possible using the most informative examples [98]. In the MT4XAI framework, the AI system plays the role of the "teacher", while the "learner" is the human user, and the "concept" to be taught is the decision boundary of a time series classifier. The challenge is to find teaching sets that are both simple and faithful: small enough for humans to grasp, but accurate enough to reflect the model's reasoning. See sections 2.3 and 2.4 for an overview of relevant MT concepts.

This master's thesis aims to meaningfully contribute to MT4XAI by applying the framework to a real-world industry use case in collaboration with a major operator of EV charging stations in Scandinavia and Germany. The project develops an anomaly detection system for EV charging sessions and explanation mechanisms that leverage recent advances in time series simplification and machine teaching. By simplifying complex charging curves and selecting optimised teaching sets, the thesis aims to provide explanations that are both accurate and cognitively accessible. This practical case study serves as a validation of MT4XAI's theoretical ideas and as a step toward their adoption in industrial practice.

1.2 Problem Statement

Deep learning models have been proven effective for time series classification and forecasting tasks [52, 66], but their opacity limits adoption in high-stakes applications in industries such as energy, health care and defence [4, 38]. Traditional XAI methods are not well suited to time series, as they often fail to produce explanations that are faithful to the model while also being accessible to a wide audience of non-experts [96, 86, 74]. Session-level time series anomaly detection systems for EV charging cannot provide customers or domain experts with actionable insights without also justifying their decisions

with explanations. This creates a gap between predictive performance and practical usability, and motivates explanation methods that can justify anomaly detections in a way that is both faithful to the underlying model and accessible to non-experts.

1.3 Research Questions and Objectives

The overarching research question for this project is formulated as follows:

> **"How can techniques from Machine Teaching for XAI be applied to time series classifiers in order to generate simple, understandable and faithful explanations of model decisions in a real world industry setting?"**

In order to validate the interpretability of the generated explanations, I conduct a study with Multimodal Large Language Model (MLLM)-based "participants" acting as proxies for human participants. The study is designed to test the following hypotheses, where improvement is measured as the change in simulation accuracy from the pre-teaching exam to the post-teaching exam (post minus pre):

- **H1 (Effect of simplification):** Participants in **Group B** (simplified overlay examples, unordered teaching sequence) will show greater post-minus-pre accuracy improvement than participants in **Group C** (raw-only examples, unordered teaching sequence).
- **H2 (Effect of curriculum ordering):** Participants in **Group A** (simplified overlay examples, curriculum-ordered teaching sequence) will show greater post-minus-pre accuracy improvement than participants in **Group B** (the same simplified overlay modality, but unordered sequence).
- **H3 (Effect of teaching vs no teaching):** Participants who receive a teaching phase (**Groups A**–**E**) will show greater post-minus-pre accuracy improvement than participants in the no-teaching baseline condition (**Group F**).

I answer the research question and test the hypotheses by pursuing the following objectives:

1. **Wrangle**: Load, clean, and enrich the EV charging dataset.

2. **Analyse**: Conduct exploratory data analysis (EDA) to analyse the statistical characteristics and limitations of the dataset.
3. **Model**: Design, train, tune, and evaluate multiple machine learning models for time series forecasting. Select the highest performing model for downstream tasks.
4. **Detect anomalies**: Implement, evaluate, and select the overall anomaly detection approach by integrating and evaluating different error metrics.
5. **Simplify**: Adapt, implement and evaluate curve simplification techniques.
6. **Teach**: Design and implement a machine teaching-based system that provides example-based explanations, and evaluate it on MLLM participants.

Objectives 1–5 establish the deployed decision system and its explanation pipeline, while Objective 6 evaluates whether the resulting teaching setup improves simulateability for a proxy learner.

1.4 Thesis Structure

The remainder of this thesis is organised as follows:

- Chapter 2 introduces the theoretical background, covering time series data, time series classification including modelling and learning approaches, explainable AI, machine teaching fundamentals, and learnability theory.
- Chapter 3 reviews related work, focusing on XAI methods for time series classification, simplification algorithms, and recent contributions in MT4XAI.
- Chapter 4 describes the methodology, including data collection, preprocessing, model design, training and evaluation, simplification algorithms, teaching-set construction, and the MT4XAI experiment with MLLMs.
- Chapter 5 reports the main empirical and engineering results from forecasting-model selection, anomaly-scoring calibration, curve simplification, and the MT4XAI MLLM experiment.
- Chapter 6 discusses the findings in relation to the research question, the broader MT4XAI project, and the main methodological limitations and contributions of the thesis.
- Chapter 7 concludes the thesis and outlines directions for future work.

Chapter 2

Background and Theoretical Foundations

2.1 Time Series Data and Time Series Classification

Time series data are sequences of observations ordered in time. They occur in diverse domains such as healthcare, finance, energy, and transportation, and are characterised by temporal dependencies, autocorrelation, and often non-stationarity [65]. In the context of this thesis, the primary focus is on electric vehicle (EV) charging sessions, which generate multivariate time series (hereafter referred to as "sequences") capturing variables such as the charger's power output (kW) and the battery's state of charge (SOC) percentage over time for a charging session. The EV charging dataset provided by a large Norwegian renewable energies company allows us to explore the practical challenges that arise from applying novel XAI techniques to a time series classification system operating in a real-world industry context.

2.1.1 Structure of Panel Datasets

The data used in this project falls in the category of a panel dataset, also referred to as longitudinal data. Panel data has been defined as a collection of many unit-indexed and discrete time series where each series corresponds to a unique recording event [43, 21, 13]. In the case of the EV charging session dataset, each session contains temporally ordered observations of variables such as charger power P_t and battery state of charge SOC_t.

Panel datasets are different from a single, continuous monitoring stream because they comprise repeated observations of similar processes across units rather than one long sequence / series [43, 21]. This structure supports analyses that exploit both within-sequence temporal dynamics and across-sequence regularities (population-level structure), as is common in grouped or hierarchical collections of related sequences [46]. In our setting, we assume negligible interaction across sequences (EV charging sessions) and treat each charging event as conditionally independent given contextual variables, which is consistent with panel systems being modelled as independent stochastic processes linked by shared parameters [13].

Formally, each charging session is a finite multivariate sequence

$$X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,T_i}\}, \quad x_{i,t} \in \mathbb{R}^d.$$

Here, d denotes the dimensionality of the data, i.e. the number of measured quantities observed at each time step. Throughout this thesis, *variable* is used as the default term when discussing data dimensions generally or statistically. Additionally, to better align the subject matter with supporting literature, the term *feature* is reserved to specify model inputs and the term *channel* is used when discussing structured signal dimensions in a mathematical tensor-representation context. Thus, power, battery state of charge (SOC), and ambient temperature are treated as variables in the dataset, as features when used as model inputs, and as channels when represented inside neural network tensors.

The complete dataset is defined as

$$D = \{X_i\}_{i=1}^{N},$$

where t is the current time step, T_i is the variable length of a sequence with index i and N is the number of charging sessions in the dataset. The assumption of independence across sessions and temporal dependence within sessions reflects standard panel data practice [43, 13].

Practically, time series from the same system or process may vary in length, cadence, and available variables, which makes specialised per-sequence modelling inefficient and limits the ability to exploit regularities shared across sessions. This motivates training a single global model across many sequences, allowing the model to learn cross-sequence structure while preserving temporal dynamics within each individual sequence. Recent forecasting literature shows that such global models can generalise effectively across heterogeneous panel datasets when the underlying sequences share sufficient statistical regularity [42].

2.1.2 Time Series Classification and Anomaly Detection

Time series classification (TSC) problems require constructing decision functions that assign a discrete label to an entire temporal sequence. Formally, given a sequence space $\mathcal{X}$ and a label space $\mathcal{Y}$, a classifier learns a mapping $f : \mathcal{X} \to \mathcal{Y}$, where each input sequence $X_i \in \mathcal{X}$ receives one label $y_i \in \mathcal{Y}$ [52, 96]. In the classical supervised setting, labels are provided during training and the model learns discriminative temporal patterns that separate predefined classes. TSC is therefore fundamentally a sequence-level decision problem regardless of whether the sequence is univariate or multivariate, fixed-length or variable-length, as the output is a single class assignment for the full sequence.

The set of time series anomaly detection (TSAD) problems is broader. Rather than predicting semantic classes, TSAD aims to identify observations or temporal behaviours that deviate from learned normality patterns. In recent years, the field of time series anomaly detection (TSAD) has seen increased interest due to its wide applicability [106]. Unlike TSC, TSAD may operate at multiple temporal granularities: anomaly decisions can be produced for individual timestamps, contiguous subsequences, or complete sequences. For this reason, TSAD cannot in general be treated as a subset of TSC.

In addition, this thesis uses the convenient term *sequence-level time series anomaly detection* (SL-TSAD). To the best of my knowledge, SL-TSAD is not an established term in the literature. It is introduced here to clearly delimit the subset of problems studied in this thesis. The distinction (and overlap) between TSC, TSAD and SL-TSAD is important because methodological choices, explanation strategies and evaluation results may transfer differently across related problem classes.

We will explicitly distinguish the following relationships:

$$\text{SL-TSAD} \subset \text{TSAD}, \qquad \text{SL-TSAD} \subset \text{TSC}, \qquad \text{TSAD} \not\subset \text{TSC}.$$

In SL-TSAD, a temporal model trained with self-supervised objectives (see subsection 2.1.3) is converted into a binary decision system by aggregating prediction errors into an anomaly score $s(X_i)$ and thresholding it at some τ. Once τ is fixed, the overall system induces a decision function

$$f_{\text{AI}}(X_i) = \begin{cases} 1 & \text{if } s(X_i) > \tau \quad \text{(anomalous)} \\ 0 & \text{if } s(X_i) \leq \tau \quad \text{(normal)} \end{cases} \tag{2.1}$$

which maps entire sequences to discrete labels. Therefore, it behaves as a TSC at the sequence level, even though the core machine learning component has not been explicitly trained to output a single target from sequence-level samples. This distinction is only important for conceptual clarity, as the paradigm used for training does not need to coincide with the functional form of the deployed decision system.

A common approach in TSAD is to train forecasting models on sequences assumed to represent normal behaviour. The model is expected to learn the dynamics of the non-anomalous distribution and to exhibit elevated prediction errors when confronted with out-of-distribution (OOD) samples, which are deemed anomalous [72, 44, 15]. The training data should predominantly be non-anomalous but may contain a small fraction of anomalous samples, provided that this contamination is insufficient to bias the learned representation [84]. In evaluation and deployment, the model's forecasts are compared to actual observations, and deviations beyond a threshold can indicate anomalies [11].

2.1.3 Learning Paradigms for Training Time Series Classifiers

Time series models can be trained under different learning paradigms depending on the availability of labels and the nature of the training objective. In the context of explaining the theoretical foundations of TSC, TSAD and SL-TSAD it is important to distinguish between supervised, unsupervised, and self-supervised learning, as these paradigms imply different assumptions about the data and different interpretations of the resulting model.

Supervised learning
In supervised TSC, each sequence $X_i \in \mathcal{X}$ in a dataset $D = \{(X_i, y_i)\}_{i=1}^{N}$ is associated with a ground-truth label $y_i \in \mathcal{Y}$, where $\mathcal{X}$ denotes the space of admissible input sequences, typically

$$\mathcal{X} = \bigcup_{T \in \mathbb{N}} \mathbb{R}^{T \times d},$$

i.e., the set of multivariate time series of arbitrary length T and dimensionality d. The model is trained to minimise a classification loss (e.g., cross-entropy) between predicted and true labels. In this setting, the classifier directly learns a mapping $f : \mathcal{X} \to \mathcal{Y}$ from a carefully constructed set of labelled examples.

Unsupervised learning
In unsupervised learning, no ground-truth labels are available. The objective is instead to uncover structure in the data, such as clusters, latent representations, or calculate density

estimates. For TSC, this may involve clustering sequences (e.g. using K-means with dynamic time warping), learning generative models (e.g. using variational autoencoders), or estimating the probability density of normal behaviour (e.g. with Hidden Markov Models or autoregressive likelihood models). Purely unsupervised models do not directly optimise for a classification objective. However, their outputs can easily be post-processed to yield a classifier, for example by automatically classifying members of the three largest clusters as normal and all others as anomalous.

Self-supervised learning
Self-supervised learning sits somewhere in between, as the prediction targets are automatically derived from the structure of the data itself rather than from externally provided labels. Predictive self-supervision has proven to be highly effective for learning structure in unlabelled temporal data [108, 26]. In time series modelling, common self-supervised objectives include forecasting future values from past observations, reconstructing masked segments, or predicting transformations applied to the sequence [11, 106]. The model is trained to solve an auxiliary task that encourages it to learn meaningful representations of temporal dynamics. Importantly, self-supervised learning does not require labelled anomalies or semantic class annotations. Instead, the structure of the sequence provides the learning signal.

Once trained, a self-supervised model can support downstream TSC and TSAD tasks in various ways [29]. In SL-TSAD, we convert the self-supervised forecasting errors into an anomaly score and apply a threshold to induce a binary decision rule (defined in subsection 2.1.2). Here, self-supervision provides the learning signal, while the deployed system behaves as a sequence-level classifier.

2.1.4 Modelling Approaches for Time Series Classification

A wide array of machine-learning techniques have been applied to time series classification (TSC) tasks, ranging from traditional statistical models to modern deep-learning architectures [52, 75, 51].

Traditional models. Classic machine-learning methods remain an important baseline for TSC. For example, logistic regression estimates class probabilities and is most effective when series are linearly separable or after suitable feature transformations [8]. Decision-tree methods permit non-linear interactions and rule-based decision boundaries, though

they are prone to overfitting in high-dimensional or highly temporal data. The K-nearest neighbours (kNN) classifier assigns classes based on closest training series in the feature space. It is intuitive and often competitive despite its scalability challenges and high-dimensionality limitations [62].

Deep learning models. Deep learning architectures have achieved state-of-the-art (SOTA) results in TSC by learning hierarchical representations directly from raw sequences [52].

- Convolutional Neural Networks (CNNs): Originally developed for image tasks (e.g., the seminal work by ImageNet Classification with Deep Convolutional Neural Networks [58]), CNNs have been adapted to time series where convolutional filters capture local temporal pattern structure and parameter sharing aids generalisation [99].
- Long Short-Term Memory (LSTM) networks: Introduced by Long Short-Term Memory (1997) and widely used for sequence modelling, LSTMs excel at capturing long-range dependencies in sequential data via four different gating mechanisms, making them especially suited for TSC tasks where a early temporal context matters for late-stage predictions.
- Gated Recurrent Unit (GRU): Introduced by Cho et al. (2014), the GRU is a streamlined variant of the recurrent memory cell which combines the forget and input controls into a single "update" gate, and uses a "reset" gate to decide how much of the previous hidden state to forget [17]. On short to medium-length sequence modelling tasks, it tends to achieve performance comparable to LSTMs with approximately 30% fewer parameters.
- Temporal Convolutional Networks (TCNs): The architecture proposed by An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling [7] demonstrates that fully convolutional, dilated causal networks can outperform recurrent models in sequence tasks, prompting more use in TSC.
- Transformers: The well-known Attention Is All You Need (2017) paper introduced self-attention mechanisms that capture global dependencies without recurrence. More recently, adaptations to time series are surveyed in works such as Wen et al. (2022) [102], showing strong promise in TSC and beyond. It also worth noting that attention mechanisms can be added to recurrent architectures, such as adding attention heads to a GRU network in order to improve long-term context retention [92, 61, 64].

2.1.5 Forecasting Techniques

The prediction horizon H defines how far into the future the model must forecast. The choice of prediction horizon H and forecasting techniques matters greatly for model performance and how evaluation metrics are calculated. In general, shorter horizons are easier to predict than longer ones, since errors compound with time. A trivial solution to time series forecasting is to always predict $y_{t+1} = y_t$, and in practice, we often see such a "lagging" effect during model training and inference as the penalty for making the safe "lag" prediction is low. In fact, such models might even perform surprisingly well on highly persistent series [45, 90].

Multi-step and multi-horizon forecasting can be framed in several ways, each with different bias–variance and error-accumulation properties. Recursive strategies propagate one-step errors forward, so errors tend to compound with horizon. Direct or multi-output strategies avoid compounding but require the model to produce stable multi-horizon outputs, which can be challenging. Empirical and theoretical comparisons of these strategies are well studied in the forecasting literature [90, 82, 45].

A common formalisation of the multi-horizon forecasting problem is as follows. Let $y_t \in \mathbb{R}^d$ denote the univariate or multivariate observation at discrete time t. Given the history $\mathcal{H}_t = \{y_t, y_{t-1}, \dots\}$ up to time t, the goal of an H-step (multi-horizon) forecaster is to produce forecasts for the next H timesteps:

$$\hat{y}_{t+1|t},\ \hat{y}_{t+2|t},\ \dots,\ \hat{y}_{t+H|t},$$

or, with vector notation:

$$\hat{\mathbf{y}}_{t+1:t+H|t} = \big(\hat{y}_{t+1|t},\ \hat{y}_{t+2|t},\ \dots,\ \hat{y}_{t+H|t}\big)^\top \in \mathbb{R}^{H\times d}.$$

There are multiple well-established strategies for constructing $\hat{\mathbf{y}}_{t+1:t+H|t}$ from data. The literature commonly distinguishes the following [90, 91]:

- **Recursive (iterative) strategy:** Train a one-step ahead model $m(\cdot)$ to predict y_{t+1} from $\mathcal{H}_t$. Multi-step forecasts are obtained by iteratively feeding previous predictions back into the model: $\hat{y}_{t+2|t} = m(\hat{y}_{t+1|t}, \dots)$, and so on. This approach is computationally cheap but may accumulate error across horizons, causing error compounding particularly when the underlying dynamics are nonlinear [90].

- **Direct strategy:** Train a separate model $m_h(\cdot)$ for each horizon $h = 1, \ldots, H$ so that $\hat{y}_{t+h|t} = m_h(\mathcal{H}_t)$. Direct models avoid iterative error accumulation but can suffer higher estimation variance because each horizon uses fewer effective training targets [90].
- **Multi-output (MIMO) / simultaneous strategy:** Train a single model that directly outputs the whole horizon vector $\hat{\mathbf{y}}_{t+1:t+H|t}$ in one pass. This multiple-output formulation preserves stochastic dependence between future steps and is often referred to as MIMO (multiple-input, multiple-output) in the forecasting literature [91].

Because the different strategies trade off bias and variance, hybrid schemes have been proposed that attempt to combine their advantages. For example, the "rectify" approach adjusts recursive forecasts using horizon-specific corrections [90]. In modern deep learning practise, the multi-horizon problem is often handled with one of the above paradigms implemented as either (a) encoder–decoder or sequence-to-sequence networks that produce an entire forecasting path, (b) multi-output heads that predict all horizons jointly, or (c) direct heads per horizon [66].

For training and evaluation it is common to aggregate per-horizon losses using a weighted sum. Let $\ell_{\text{pred}} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_{\geq 0}$ be a per-step forecasting error metric / loss function (see subsection 2.1.6). A horizon-weighted objective for a single training example at time t can be written as

$$\mathcal{L}_t \;=\; \sum_{h=1}^{H} w_h \, \ell_{\text{pred}}\big(y_{t+h},\, \hat{y}_{t+h|t}\big),$$

where $w_h \in [0, \infty]$ are horizon weights that allow the practitioner to emphasise near-term accuracy (larger w_h for small h) or to treat all horizons equally [66, 91]. Horizon weighting is a straightforward and widely-used mechanism for reflecting the differing importance and difficulty of short versus long horizons in both model training and hyperparameter selection.

2.1.6 Forecasting Error Metrics

The choice of error metric directly impacts what kinds of time series are classified as anomalous. We will therefore review some commonly used metrics for time series forecasting, along with the specialised metrics used in the project's SL-TSAD implementation.

- Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

- Mean Squared Error (MSE):

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

- Root Mean Squared Error (RMSE):

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

- Weighted Root Mean Squared Error (W-RMSE): ...
- Huber Loss (SmoothL1): Quadratic for small errors, linear for large errors. This offers robustness against outliers.

$$\ell_\delta(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2, & \text{if } |y - \hat{y}| \le \delta \\ \delta\big(|y - \hat{y}| - \frac{1}{2}\delta\big), & \text{otherwise} \end{cases}$$

- Robust Weighted Standardised Error (RWSE): Residuals are first computed per time step, horizon, and channel:

$$r_{t,h,c} = y_{t+h,c} - \hat{y}_{t+h|t,c}.$$

From validation residuals, we compute the median $m_{h,c}$ and median absolute deviation $\text{MAD}_{h,c}$, then standardise:

$$z_{t,h,c} = \frac{|r_{t,h,c} - m_{h,c}|}{\text{MAD}_{h,c}},$$

with clipping (e.g. $z_{t,h,c} \le 5.0$). To combine across horizons and channels, we apply horizon weights w_h (exponential decay, normalised to 1) and channel weights w_c (uniform, sum to 1). The session-level RWSE is

$$s_{\text{RWSE}}(X_i) = \frac{1}{T'}\sum_{t=1}^{T'}\sum_{h=1}^{H}\sum_{c=1}^{C} w_h w_c \, z_{t,h,c}$$

where $s_{\text{RWSE}}(X_i) \equiv s(X_i)$ discussed in subsection 2.1.2, T' is the number of evaluated time steps, H is the number of horizons, and C is the number of channels.

Each anomaly scoring metric variant $s(X_i)$ emphasises different aspects of error: RMSE heavily penalises large spikes, MAE balances across magnitudes, Huber Loss introduces robustness to outliers, and RWSE highlights systematic deviations rather than isolated spikes.

2.1.7 Statistical vs. Perceptual Anomalies in Applied SL-TSAD

All approaches to anomaly detection must deal with the tension between statistical and perceptual anomalies. What a model flags as unusual based on high variance or prediction error may or may not align with what *humans* perceive as surprising or suspicious. Although anomalies in time series are often described as deviations from an expected temporal pattern, what counts as "expected" depends on domain context, operating conditions, and the cost of false alarms. As a result, anomalies can be statistically rare under a learned model without being operationally meaningful, and vice versa. Ultimately, anomaly labels often carry a degree of subjectivity [60]. In industrial applications such as EV charging, this subjectivity is particularly salient, since deviations that are statistically rare may nonetheless represent acceptable operational variations.

This thesis focuses on SL-TSAD, where the objective is to judge whether an entire charging session deviates from learned regularities, rather than detecting isolated anomalous points. This choice aligns with the setting of customer service for EV charging, where the practical question is whether a charging session is "normal" or "abnormal" as a whole. Because anomaly judgements are context dependent and may be interpreted differently by different stakeholders (e.g. customers, technicians, Data Scientists), explainability becomes particularly important as users need to understand why a session was flagged in order to determine whether it reflects an actual issue or an acceptable variation.

2.2 Explainable Artificial Intelligence

The modelling approaches explored in subsection 2.1.4 may achieve strong predictive performance but vary in interpretability. Deep learning models are often criticised as

"black boxes" because their internal decision processes are difficult to inspect directly [83]. This becomes especially relevant in multivariate time series settings, where several interdependent variables evolve over time while static contextual features influence the observed temporal dynamics. For example, ambient temperature is a static variable that may act as a proxy for battery temperature, which in turn affects battery chemistry and charging behaviour.

Explainable Artificial Intelligence (XAI) seeks to make the behaviour of AI systems understandable to humans. In practice, this includes models that are interpretable by design, such as sparse or prototype-based models, as well as post-hoc methods that explain already-trained systems through local or global surrogates, feature attribution, counterfactuals, examples, or combinations of these approaches [69, 24, 38]. Generating explanations is therefore not only a technical task, but also a human-AI interaction problem in which people form and revise mental models of how a system behaves [74].

In the early 1970s, expert systems such as *MYCIN* included built-in explanation facilities that returned rule-based justifications and traces of inference when users asked "why" or "how" questions [87, 14, 89]. After the deep learning resurgence in the 2010s, general-purpose post-hoc methods such as LIME [81], SHAP [70], and Integrated Gradients [88] became widely used. Around the same period, researchers argued for clearer definitions and stronger scientific standards for interpretability [69, 24]. Others emphasised that good explanations should reflect how people actually ask for and use explanations. In particular, human explanations are often contrastive, selective, and shaped by context rather than exhaustive descriptions of all causes [74]. More recent work models explanation as a process of belief updating, where users revise their understanding of a model through interaction with explanatory information [104].

2.2.1 Why XAI?

Although the conceptual motivation for XAI follows from the opacity of many modern learning systems, its practical importance becomes clearest when models are used in settings where predictions influence human decisions, operational processes, or critical infrastructure. In such settings, limited interpretability can reduce oversight, make model errors harder to diagnose, and complicate accountability when systems behave unexpectedly. This has led to growing demand for XAI methods that provide human-understandable insight into model behaviour and predictions [4, 39].

Explanations also support practical work during model development and deployment. They help practitioners detect spurious correlations, inspect whether predictions rely on meaningful patterns, and judge when model outputs should be trusted. Without such support, users may either reject useful systems or place too much confidence in unreliable predictions [10]. Explanation quality therefore depends strongly on the user's task. Developers auditing a model need different information from domain experts trying to understand decision boundaries. Interactive explanations that allow users to probe predictions, compare alternatives, and test counterfactual "what-if" cases can help users build more accurate mental models, while poor visual or interaction design may weaken trust instead [10, 74, 104].

2.2.2 Families of XAI methods

These practical needs have led to several broad families of XAI methods, each based on different assumptions about what should be explained and how explanations should be presented. Feature attribution methods use gradient-based or perturbation-based scores to estimate how input features contribute to a prediction [107, 88]. They are widely used, but their outputs are sensitive to implementation choices and often difficult to interpret without strong domain knowledge [2, 34, 74, 96].

Other widely used approaches rely on surrogate models. Methods such as LIME [81] approximate the behaviour of a black-box model locally with a simpler interpretable model [38]. Similar ideas can also be applied globally, although explanatory fidelity often decreases as the explained region grows [76].

Example-based explanations communicate model decisions through concrete instances rather than abstract feature scores. This often matches how people naturally reason about decisions [76]. Common techniques include prototypes, criticisms, and counterfactual examples, all of which are increasingly used in time series explanation research [96, 33, 22, 5].

Concept-based methods explain predictions through human-understandable concepts rather than raw input dimensions. *Testing with Concept Activation Vectors* (TCAV) measures how user-defined concepts influence hidden-layer activations, while related methods such as *Automated Concept Extraction* (ACE) and concept bottleneck models aim to discover or enforce concept-level representations inside the model [56, 35, 57].

A different line of work studies internal model mechanisms directly. Mechanistic interpretability seeks causal explanations by identifying neurons or circuits that implement specific computations inside the model rather than only analysing inputs and outputs [9, 78, 18]. This remains rare in time series applications [96]. More recently, large language models have also been used to generate natural-language explanations or translate technical explanations into human-readable form, although concerns remain about factual accuracy and faithfulness [71, 16, 59, 53].

2.2.3 Limitations and critiques

Despite rapid progress, important limitations remain. A central critique is that post-hoc explanations may create an illusion of transparency without faithfully showing how a model actually computes its predictions. Rudin [83] argues that, particularly in high-stakes settings, explaining a black-box model after training may be misleading because the explanation is only an approximation of the real decision process and may therefore fail precisely where trust is most needed. She also challenges the common assumption that interpretability must always be traded for predictive performance, arguing that for many structured data problems, inherently interpretable models can achieve comparable accuracy without requiring a separate explanatory layer.

Technical evaluation remains difficult. Metrics such as faithfulness or fidelity do not necessarily show whether users truly understand a model or make better decisions with explanations, while rigorous human-centred evaluation remains scarce and expensive [24]. Saliency and perturbation methods may be unstable, dataset-dependent, and vulnerable to confirmation bias when visually plausible patterns are mistaken for causal evidence [2, 34, 60]. Explanations may also over-persuade users by creating confidence that is not justified by the underlying model behaviour [74, 10, 83].

These concerns also reflect a broader point from the social sciences. Human explanations are usually selective and contrastive rather than complete. Miller [74] argues that people typically ask why one outcome occurred instead of another, which means that explanation quality depends on context, audience, and purpose rather than technical completeness alone. No single explanation can therefore fully capture the behaviour of a high-capacity model, and explanation methods must ultimately be judged relative to the human task they are meant to support.

2.2.4 XAI for time series models

Time series data introduces additional challenges. Long sequences place demands on working memory, multiple variables interact over time, and similar temporal patterns may have different meanings depending on context. Methods originally developed for images or tabular data therefore often transfer poorly to temporal settings [96].

Many existing time series explanation methods adapt general XAI ideas, but the temporal structure introduces both cognitive and methodological difficulties. A broader taxonomy of explanation methods for TSC, together with representative methods and evaluation practices, is reviewed in section 3.1.

2.2.5 Implications for this thesis

Three points from the literature have shaped my approach to implementing XAI in this work. First, explanations for TSC should be contrastive and example-based, as this aligns with human reasoning and reduces cognitive load [74, 47]. Secondly, interactivity supports agency but must be carefully designed to avoid over- or under-trust [10]. Third, curve simplification tailored to the classifier can offer a principled path to cognitively accessible yet faithful explanations for TSC's [95].

2.3 Machine Teaching Fundamentals

Machine Teaching (MT) treats explanation as a constructive act. A teacher selects examples so that a learner arrives at a target concept using as few and effective examples as possible [111]. In contrast to passive learning, where examples are sampled from an underlying distribution, MT optimises both which examples to provide and how they are organised. This section summarises the key theoretical ideas that underpin later chapters.

2.3.1 Origins and evolution of Machine Teaching

MT originates in learnability theory. In Valiant's *Probably Approximately Correct (PAC)* framework, learning is feasible when sufficiently many random samples are available [98]. MT inverts this: given a target concept, the goal is to design an optimal set of labelled examples (often called *witnesses*) that uniquely identify the target concept to a particular learner [110, 36].

Early formulations introduced the *teaching dimension* (TD), the smallest number of labelled instances that force an idealised learner to recover the target concept. Later work generalised this to the *teaching size* (TS), which replaces cardinality with a length or complexity measure over examples, making it suitable for structured objects such as sequences or programs [93].

Beyond batch teaching, subsequent developments examined *interactive* and *curriculum-based* variants. Here examples are adaptively chosen or ordered to reflect the learner's evolving state, attention, or cognitive constraints [79, 32]. These refinements reflect the shift from worst-case theoretical guarantees to more realistic models of human learning and explanation.

2.3.2 Setting and notation

Let $\mathcal{X}$ denote an input domain and $\mathcal{Y}$ a label set. A concept class $\Theta \subseteq \mathcal{Y}^{\mathcal{X}}$ contains possible target behaviours. A learner L maps a labelled set

$$S = \{(x_i, y_i)\}_{i=1}^{n} \subseteq \mathcal{X} \times \mathcal{Y}$$

to a hypothesis $\hat{\theta} = L(S) \in \Theta$. MT asks for a teaching set S such that $L(S)$ identifies the target $\theta^* \in \Theta$ while minimising a cost $C(S)$ reflecting the size or complexity of the examples:

$$\min_{S} C(S) \quad \text{s.t.} \quad L(S) = \theta^*.$$

A common formalisation is

$$J(S) = \sum_{(x,y)\in S} \delta(x) + \mu\, \lambda\big(\theta^*, L(S)\big),$$

where $\delta(x)$ measures example complexity and λ penalises misalignment between the learner's inferred rule and the target. Different teaching models instantiate L, C, and admissible S according to representational or cognitive constraints.

2.3.3 Concepts, representations and witness sets

MT distinguishes between *concepts*, *representations*, and *examples*. A representation language $\mathcal{R}$ and a mapping $\pi : \mathcal{R} \to \Theta$ relate descriptive/symbolic descriptions to conceptual behaviours. Multiple representations may encode the same concept.

A labelled set S is a *witness set* for θ^* if any representation r consistent with S induces the target concept, i.e. $\pi(r) = \theta^*$. Teaching therefore selects witnesses that isolate θ^* from competing hypotheses. Teaching sets are typically chosen to minimise an objective such as $J(S)$.

2.3.4 Teacher protocols and representation teaching

Teacher protocols formalise how witness sets are constructed. *Eager* protocols select minimal witnesses for each target, while *greedy* protocols reuse examples across targets to reduce overall teaching effort. Other protocols optimise over a consistency graph whose nodes are representations and whose edges encode witness relations [27, 32].

MT distinguishes *concept teaching* from *representation teaching*. Concept teaching is about identifying the correct equivalence class of representations, while representation teaching aims at a particular $r^* \in \mathcal{R}$. Many real-world explanatory tasks require representation teaching because the learner must internalise not only the correct labels but also the structural features (characteristics, behaviours) that give rise to them.

2.3.5 Teaching infinite and expressive concept classes

Classical TD often becomes infinite for rich or expressive concept classes, including those induced by modern ML models. Two interesting refinements overcome this limitation by shifting focus from eliminating all competing hypotheses to describing how a learner chooses among those that remain.

In *finite biased teaching*, the learner has a *learning bias*, i.e. a prior distribution favouring simpler or more plausible hypotheses (e.g. Occam's Razor). A teaching set succeeds when the target concept becomes the most likely consistent hypothesis [41]. Interestingly, the *expected* biased TD can be finite even for Turing-complete hypothesis spaces when both the learning and sampling biases favour simplicity.

A complementary perspective is *preference-based teaching*, where the learner possesses a preference relation over hypotheses. Rather than eliminating all competitors, teaching makes the target more preferred than any consistent alternative [31]. This framework yields finite teaching complexity for many infinite classes, including geometric and threshold-based concept families.

Together, these perspectives justify the feasibility of teaching complex or infinite model-induced behaviours through small, well-chosen example sets, provided that teacher and learner share aligned simplicity or preference structures.

2.4 Learnability theory and teaching complexity

This section connects classical learnability to teaching complexity, then discusses practical limitations with fallible human learners and structured data such as time series.

2.4.1 From PAC learnability to teaching guarantees

In PAC learning, a hypothesis class with finite complexity admits sample-efficient learning from random data [98]. A typical complexity measure is the Vapnik–Chervonenkis (VC) dimension, which counts the size of the largest set that the hypothesis class can *shatter*. A hypothesis class shatters a set of data points if it can realise every possible assignment of labels to those points. In other words, no matter how the points are labelled, there is a hypothesis in the class that fits them perfectly. Teaching inverts the sampling process and asks for *constructive* sample design for a specific target. For a learner L and target θ^*, the *teaching dimension* is

$$\mathrm{TD}(\theta^*, L) = \min\{|S| \ : \ L(S) = \theta^*\},$$

the smallest number of labelled instances that forces L to infer θ^* [36]. This quantifies an ideal sample complexity under a helpful teacher rather than under random sampling [111].

For structured objects, how large an example is can matter more than how many examples there are. The *teaching size* replaces cardinality with an additive length function ℓ:

$$\mathrm{TS}(\theta^*, L, \ell) = \min\Big\{ \sum_{(x,y)\in S} \ell(x,y) \ : \ L(S) = \theta^* \Big\}.$$

Teaching size can remain bounded even when teaching dimension is high. This motivates simplification for sequences [93].

2.4.2 Fidelity, simplicity and robust teaching

Going forward, we will bridge MT and TSC notation with

$$\theta_{\mathrm{AI}} \equiv f_{\mathrm{AI}} \in \Theta, \qquad f_{\mathrm{AI}} : X \to Y,$$

where $\theta_{\mathrm{AI}} \equiv f_{\mathrm{AI}}$ is the deployed decision system.

In XAI the teacher's target is the *model's* behaviour rather than an external ground truth. We therefore balance faithfulness to θ_{AI} with human-centred simplicity. The objective $J(S)$ above captures this trade-off by penalising both cognitive cost and misalignment [40]. Teaching sets must also be robust to realistic human lapses in attention. Redundancy across features or time windows can stabilise learning, reduce misinterpretation and increase simulateability [27]. When the teacher cannot tell the learner directly which features are relevant, example choice must implicitly highlight relevance.

2.4.3 Conditional teaching, redundancy and human-oriented curricula

In realistic teaching scenarios the learner rarely starts from scratch. They arrive with prior knowledge and with examples they have already seen. To capture this, MT4XAI introduces the idea of *conditional teaching size*. Let B denote background examples or previously taught concepts. The conditional teaching size for a new target θ^* can be written as

$$\mathrm{TS}(\theta^* \mid B, L, \ell) = \min\Big\{ \sum_{(x,y)\in S} \ell(x,y) \ : \ L(B \cup S) = \theta^* \Big\},$$

which measures the additional teaching effort needed given B [32, 93]. This is particularly relevant for curriculum teaching and for multi-session human studies, where participants see a sequence of examples over time.

Teaching may interact with prior representations through *interposition*: previously learned concepts can interfere with or facilitate subsequent teaching, depending on ordering and learner preferences. Redundancy interacts with these ideas in a subtle way. If

there are many redundant representations or examples for a concept, an idealised teacher might try to avoid redundancy altogether. However, MT results show that when learners build internal representations and are fallible, adding some redundancy can *reduce* conditional teaching size. Repeated or overlapping examples can stabilise learning and help the learner recover from confusion or forgetting [27].

Finally, human-oriented teaching models incorporate fallibility, memory limitations and attentional constraints. Human learners forget, form wrong intermediate representations and may project their own categorisation onto the AI. Bayesian Teaching explicitly models such behaviour and selects examples to optimally change the explainee's posterior over the AI's decision rule [103]. Interactive teaching frameworks plan sequences that adapt to observed errors [79]. Curriculum heuristics can further reduce confusion when time is short [32].

2.4.4 Implications for XAI

The MT perspective reframes example-based XAI as a teaching problem. When the target is a fixed model θ_{AI}, an explanation set plays the role of a teaching set, as it is a small collection of labelled instances selected to induce in the user a hypothesis that closely matches the model's behaviour [111, 40]. Complexity measures such as teaching dimension and teaching size then provide idealised lower bounds on how many, and how complex, such examples must be. Conditional teaching size highlights that explanation design should account for prior exposure and context, because additional examples are interpreted against what the user has already learned [93, 32].

From this viewpoint, two design principles for XAI follow. First, explanations should be evaluated not only on faithfulness but also on how they affect a learner's simulateability and residual uncertainty about the model, in line with the idea that teaching success is measured by the learner's induced hypothesis rather than by internal model access [103]. Second, some redundancy and curriculum structure are desirable for fallible human learners: MT results show that overlapping examples and carefully chosen teaching sequences can reduce conditional teaching size and increase robustness to forgetting, misinterpretation, and noise [27, 32].

For structured domains such as time series, teaching size further suggests that explanation sets should not only be small in cardinality but also in representational complexity.

When examples are long sequences, cost terms like $\delta(x)$ naturally favour concise representations that preserve the model's discriminative behaviour while reducing cognitive load. This provides a theoretical justification for integrating sequence simplification and model-aware selection into example-based XAI for TSC. [93, 96, 95, 49].

2.5 Summary and research gap

This section has established the key theoretical tools needed to understand the methodology in Chapter 4. To summarise, Machine Teaching provides a principled way to explain "black box" models through examples. It offers three core ingredients. First, it defines objectives that balance simplicity and faithfulness. Second, it introduces complexity measures such as teaching dimension and teaching size. Third, it offers interactive and curriculum-based strategies that account for the fact that human learners are fallible and have limited attention [111, 36, 93, 79].

For TSC, cognitive load is a main bottleneck in human interpretability. This, together with the idea that the teaching size of examples should be small, motivates the need for sequence simplification algorithms that preserve the classifier's output. Such simplifications reduce the cost $\delta(x)$ of inspecting examples and improve human simulateability [95, 47]. This connects Machine Teaching directly to the problem of explaining high-dimensional, structured data.

The broader research gap that motivates the MT4XAI research project concerns the lack of example-based explanation methods that combine: (i) a principled link between teaching objectives and human cognitive constraints, (ii) simplification techniques that make complex data more accessible without harming faithfulness, and (iii) teaching strategies that support human learners through curricula, redundancy and interaction. These ideas form the foundation of the MT4XAI research project, as described in the project proposal submitted to the Research Council of Norway [94]. MT4XAI investigates how these elements can be combined into a coherent framework for XAI. The next chapter reviews recent research that contributes to this direction and positions the work presented in this thesis within that line of research.

Chapter 3

Related Work

Building on the theoretical foundations outlined in chapter 2, this chapter reviews the research contributions that are most directly relevant to the research questions and methodological choices of this thesis. Whereas Chapter 2 introduced the conceptual foundations of TSC, TSAD, XAI, and machine teaching, the present chapter focuses on concrete methods, empirical findings, and evaluation practices. Its purpose is to situate this thesis within the broader scientific landscape, identify the most relevant methodological precedents, and clarify the remaining gaps that motivate the MT4XAI approach. In particular, the following sections review related work on (i) explainability methods for TSC and SL-TSAD (section 3.1), (ii) time series simplification algorithms with a focus on class-preserving simplifications (section 3.2), (iii) MT frameworks applied to XAI (section 3.3), and (iv) LLM- and MLLM-based evaluation paradigms relevant to early-stage XAI research (section 3.4). section 3.5 then positions the present thesis within this literature.

3.1 Explainable AI for Time Series Classification

Related work on explainability for TSC is best understood as a specialised extension of the broader XAI families introduced in subsection 2.2.2. A survey by Theissler et al. [96] groups the literature into three main families, reflecting where the explanatory signal is located and how it is presented to the user:

- *time-point-based* attributions

- *subsequence-based* methods (e.g., shapelets, wavelet components)
- *instance-based* approaches using prototypes and counterfactuals.

These families address a common problem: temporal data is difficult both to explain and to inspect. Time-point-based methods adapt attribution ideas (e.g. feature importance) to temporal inputs, but their outputs can become visually noisy and unstable over long multivariate sequences [96, 86]. Subsequence-based methods trace back to shapelets, which identify short, discriminative patterns within longer series that can be used directly for classification or explanation [105]. Instance-based approaches instead present representative, prototypical, or counterfactual series to support comparison between cases [33, 23, 5]. In practical interfaces, such explanations are often paired with linked plots or contrastive example selection rather than attribution maps alone.

Despite this methodological variety, empirical human studies for TSC remain limited. Surveys have repeatedly called for more rigorous evaluation protocols tailored to temporal data, where explanation quality is shaped not only by faithfulness but also by cognitive burden [86, 96]. A recent controlled study by Håvardstun et al. [47] shows that users can benefit from model inspection for TSC, but also that time series remain cognitively demanding: results depend strongly on task framing, users' willingness to invest effort, and whether they can mentally simulate the classifier's behaviour on unseen instances. To support such evaluation, the authors propose *forward simulation* as a practical protocol in which users are shown explanations or labelled examples and are then asked to predict the classifier's outputs on new cases, yielding a task-grounded measure of simulateability.

This aligns with cognitive accounts of explanation that emphasise contrastive questions and selective presentation over exhaustive detail [74]. For long and noisy sequences, the central challenge is therefore to reduce visual and cognitive load while preserving the decision-relevant signal. Because many TSAD systems in practice are built on forecasting or reconstruction models, adjacent work on explaining forecasting models is also relevant. Surveys of XAI on time series report that most methods in this space adapt attribution or perturbation techniques to temporal windows and profiles, but rigorous benchmarks and user studies remain scarce, especially when the target is a forecasting model rather than a classifier [82, 86, 96]. This recurring trade-off between informativeness and cognitive burden motivates simplification methods that preserve decision-relevant structure while reducing visual complexity.

3.2 Time Series Simplification Algorithms

Recent work has explored whether simplifying a time series before explanation can reduce visual and cognitive load without removing the structure required to understand a classifier's behaviour. In the MT4XAI line of work, simplification is not treated merely as a compression problem, but as part of the explanation itself [95]. Complementary work has proposed metrics for evaluating simplification algorithms for interpretability, showing that simplified series can improve simulateability relative to raw inputs across datasets with different temporal characteristics [49]. Together, these studies suggest that simplification can act as a bridge between model behaviour and human perception.

Curve simplification has a long history in cartography and signal processing. Classical algorithms include the bottom-up approach, the top-down Ramer–Douglas–Peucker (RDP) method and the dynamic-programming (DP) algorithm of Imai and Iri, which optimises a piecewise-linear approximation under an error budget or a segment budget [54, 80, 25, 50]. Piecewise approximation methods such as Piecewise Linear Approximation (PLA) and Piecewise Aggregate Approximation (PAA) reduce dimensionality for efficient indexing and retrieval [55]. Symbolic Aggregate approXimation (SAX) extends PAA by converting segment values into discrete symbols [68]. These methods control reconstruction error, yet they do not account for how a downstream TSC will react to the simplified input. A simplification that is optimal under ℓ_2 error can therefore still flip the classifier's label.

The Optimal Robust Simplification (ORS) algorithm presented in Telle et al. [95] addresses this gap by searching for piecewise-linear series that preserve the classifier's decision and remain stable under perturbations. Given a trained classifier f_{AI}, an original sequence X and its simplified candidate sequence $\tilde{X}$ with k segments, ORS formulates a multi-term objective that balances reconstruction error, complexity and fragility:

$$\min_{\tilde{X}} \ \alpha||X - \tilde{X}||_2^2 + \beta k + \gamma\big(1 - \mathrm{RP}(\tilde{X})\big) \quad \text{s.t. } f_{\mathrm{AI}}(\tilde{X}) = f_{\mathrm{AI}}(X)$$

Here $\mathrm{RP}(\tilde{X})$ denotes the robust probability that the classifier's label remains unchanged when $\tilde{X}$ is perturbed by noise drawn from a specified distribution. Fragility is $1 - \mathrm{RP}(\tilde{X})$ and the constraint $f_{\mathrm{AI}}(\tilde{x}) = f_{\mathrm{AI}}(x)$ enforces faithfulness. In contrast to reconstruction-driven simplifiers, ORS is classifier-aware: it searches for simplified curves that remain faithful to the classifier while trading off approximation quality, complexity and robustness. The algorithm first generates promising candidates and then evaluates a small

subset more expensively using Monte Carlo estimates of RP. In controlled experiments, ORS yields shorter and more robust explanations than standard simplifiers at comparable fidelity [95].

The authors further argue that two ORS properties are especially relevant for human-facing tools. First, the complexity term k acts as a proxy for the cognitive cost of visual inspection. Second, robustness attenuates accidental label flips caused by small drawing or resampling errors, which makes explanations more stable in interactive settings [95]. Telle et al. also identify several open directions for future work, including broader notions of fidelity, faster search procedures, and human-subject validation of whether ORS simplifications genuinely improve interpretability and decision support in practice [95]. Simplification alone, however, does not determine which examples should be shown, in what order, or with what degree of redundancy, which leads directly to machine-teaching approaches for XAI.

3.3 Machine Teaching in XAI

Building on the MT concepts introduced in section 2.3 and section 2.4, recent MT4XAI research has begun applying teaching-theoretic principles directly to explainability problems, where the target concept is not an external rule but the behaviour of a fixed AI system. In this setting, the central question is not whether a learner can recover a concept class in theory, nor whether they can uncover details about the AI's technical implementation, but whether carefully selected examples improve human simulateability of black-box model decisions in practice.

Several MT4XAI studies connect MT to XAI in this more human-centred sense. When irrelevant features are present and users are not told about them, carefully chosen examples still improve identification of the model's decision rule(s), and informing users about irrelevance priors further improves simulateability [40]. Teaching sets that include some redundancy can be *beneficial* for human learners, because partial repetition stabilises learning in the presence of memory limits and attention lapses; theory and simulations support this claim for representation teaching [27]. The ordering of examples also matters. Heuristic search over curricula can reduce total teaching effort by ordering examples from simpler to harder or by interleaving typical and borderline cases, which aligns with both classroom practice and human-subject results in XAI settings [32]. Bayesian Teaching and decision-theoretic tutors formalise interaction by planning example choices that

move a fallible learner's beliefs towards the target rule or by adapting examples to observed errors [103, 79]. These strands share a common aim, namely to improve human understanding of complex models by choosing few, simple, and well-ordered examples that are aligned with how people learn.

A closely related methodological question is how candidate examples should be represented and selected once explanation is framed as a teaching problem. Outside XAI, feature-based time series classification has long shown that temporal sequences can be mapped to fixed length vectors of descriptive characteristics, enabling comparison in a common representation space rather than only through raw point-wise alignment [30]. In parallel, work on submodular subset selection has shown that facility-location and related coverage-based objectives provide a principled way to choose representative yet non-redundant subsets under a similarity measure, with efficient greedy optimisation and well-studied approximation properties [67, 101, 77]. Although these methods were not developed specifically for MT4XAI or time-series explanations, they are highly relevant to example-based XAI because they provide a natural basis for constructing compact teaching sets that cover the behavioural variability of a candidate pool.

Taken together, this literature suggests that successful example-based XAI should not only select representative examples, but also manage simplicity, redundancy, and ordering in ways that reflect human learning constraints. For TSC explanations, this yields two immediate design priorities. First, individual examples should be visually concise yet faithful to the classifier, which motivates class-preserving simplification methods such as ORS. Second, teaching sets should balance diversity with limited redundancy and curriculum structure to support fallible learners. Once explanations are reframed as teaching problems, a separate question arises: how should such teaching strategies be evaluated efficiently before committing to full human-participant studies?

3.4 LLMs and MLLMs as Evaluators and Proxy Learners in XAI Research

Recent work has begun exploring two related uses of foundation models in explainability research: as proxy learners in forward-simulation settings, and as automated evaluators of explanations or model outputs. A particularly relevant example appears in the first arXiv version of Martí-Pérez et al. [73], where the authors tested whether an MLLM could recover the class label of simplified time series produced by a black-box TSC model.

In that setup, the MLLM acts as a fallible learner rather than as a judge: the task is to simulate the classifier's decision from the simplified input. The study varies the simplification strength through the number of segments k and reports a non-monotonic pattern in simulateability, with performance first improving as more salient temporal structure is preserved and then degrading as the representation becomes overly detailed. This evidence is preliminary, and should be interpreted as such, because the later version of the same paper replaced the MLLM section with a human forward-simulation study [49]. Even so, the version history is informative because it illustrates a methodological progression from proxy-based pilot testing to human-centred validation rather than a claim that MLLMs can replace human participants.

A broader LLM-as-a-judge literature provides a related but distinct precedent. Strong LLM judges can align reasonably well with human preferences in some open-ended evaluation tasks, yet the same literature also documents important limitations, including position bias, verbosity bias, self-enhancement bias, and the need for careful prompt design and calibration [109, 37]. In time-series-adjacent work, Aksu et al. [3] introduce simulateability-based metrics for evaluating natural-language explanations of forecasting models and show agreement with human judgements, while Fiori et al. [28] use LLMs to compare candidate explanation approaches for smart-home activity recognition and report preliminary alignment with user surveys. Taken together, these studies suggest that LLMs and MLLMs can be useful as low-cost screening instruments for early-stage hypothesis testing and protocol refinement in XAI research. They do not, however, remove the need for human-grounded evaluation when the objective is to make claims about end-user understanding [47, 49].

3.5 Positioning of the Thesis

The MT4XAI research project proposes example-based explanations for complex AI systems by combining MT objectives with simplification for structured data and with interactive teaching strategies for human users [94]. Within that agenda, this thesis focuses on an industrial SL-TSAD use case in which the deployed anomaly-detection system behaves as a sequence-level classifier. The work therefore sits at the intersection of three strands reviewed above: TSC explanation methods that remain understandable under temporal complexity, classifier-aware simplification methods that preserve behaviour while reducing visual burden, and MT-inspired example selection strategies that account for human fallibility.

Concretely, this thesis extends that line of work in four ways. First, it instantiates the MT4XAI agenda on EV charging sessions, a real-world multivariate time-series domain where end-user interpretability is practically important. Second, it adapts and improves ORS to generate large numbers of class-preserving and robust simplifications of charging sessions. Third, it constructs teaching sets and curricula that organise examples by simplicity, robustness and margin, using feature-based embeddings of simplified power curves together with a facility-location coverage objective to select a compact yet diverse set of teaching examples. This brings MT principles into a structured time series setting while linking them to broader literature on time-series representations and representative subset selection. Finally, it pilots an MLLM-based forward-simulation evaluation as a low-cost pre-human screening step for the MT4XAI industry implementation. The methodological details of these contributions are presented in next chapter.

Chapter 4

Methodology

This chapter describes the end-to-end methodological pipeline used in the thesis. I first transformed raw EV charging telemetry into a modelling-ready panel time-series dataset, then trained and calibrated a forecasting-based anomaly-detection pipeline, simplified classifier-relevant charging curves, constructed a teaching set for MT4XAI, and finally evaluated the resulting explanation workflow with multimodal large language models (MLLMs) acting as proxy learners. The purpose of this chapter is to explain how each step of the pipeline was implemented. Comparative outcomes, final engineering choices, and implementation trade-offs are reported separately in Chapter 5 in order to keep the methods/results boundary clear.

4.1 Data Collection and Preprocessing

This section describes how the EV charging-session dataset was transformed from raw telemetry into a modelling-ready panel time-series table. Data governance, software availability, and reproducibility are reported separately in section 4.7.

4.1.1 EV Charging Session Data

The empirical basis of this thesis is a large real-world dataset of EV charging sessions associated with Audi e-tron 55 vehicles. A charging session is defined as one contiguous charging event at one location and is uniquely identified by `charging_id`. Going forward,

the terms *session* and *sequence* will be used interchangeably. The raw dataset contained 1,643,654 time-indexed rows across 62,422 sessions before any cleaning was performed on it.

The raw dataset was treated as panel data, where each panel entity corresponded to one charging session and each panel time index corresponded to one minute within that session. The downstream anomaly-detection system used `power` (kW) and `soc` (state of charge, %) as the primary dynamic variables. Additional variables, such as `energy`, `charger_category`, `nominal_power`, `lat`, and `lon`, were retained when they were useful for feature engineering, contextualisation, or traceability.

At ingestion, I retained only the variables required for modelling and explainability: `charging_id`, `timestamp`, `power`, `soc`, `energy`, `charging_duration`, `charger_category`, `nominal_power`, `lat`, and `lon`. The table was then sorted by `charging_id` and `timestamp` so that all later transformations operated on stable within-session temporal order.

Complete charging sessions were the unit of analysis throughout the thesis, which is consistent with the SL-TSAD and XAI for TSC objectives introduced in chapter 2 and chapter 3.

4.1.2 Data Wrangling and Feature Engineering

Loading, cleaning and transforming the EV charging dataset was implemented in the Jupyter notebook `01__Data_Wrangling_and_FE.ipynb` [85]. The raw charging-session extract was transformed into a session-aware panel time-series table in which each row represented one one-minute observation within a charging session and each `charging_id` defined one panel entity. The final modelling table combined dynamic charging measurements, engineered temporal features, charger-context variables, and weather-derived contextual information so that each session could be processed as a multivariate sequence while preserving within-session temporal order.

Table 4.1 summarises the variables retained after preprocessing and feature engineering.

1. Structural validation and type normalisation. The selected columns were inspected for explicit null-like values and cast to appropriate data types before any temporal cleaning was performed. This established a consistent schema for the later session-level operations.

Variable	Role in final modelling table
charging_id	Unique session identifier used as panel entity.
timestamp	One-minute time index within session.
power	Charging power in kW (primary prediction target).
soc	Battery state of charge in %.
energy	Cumulative delivered energy.
minutes_elapsed	Minutes since session start.
progress	Log-scaled temporal progress feature in $[0, 1]$.
rel_power	Power normalised by nominal charger capacity.
d_power, d_soc	First-order temporal differences.
d_power_ema3, d_soc_ema3	Smoothed short-span temporal trends.
nominal_power	Charger rated power.
charger_cat_*	One-hot charger-category indicators.
temp	Ambient temperature proxy at session start.
lat, lon	Charging-site coordinates retained for contextual traceability.
charger_category	Original charger category label.
nearest_weather_station	Weather-source traceability identifier.

Table 4.1: Variables retained in the final session-level modelling table after preprocessing

2. Handling edge zero-power artefacts. Readings with power equal to zero were not treated uniformly. Leading and trailing zeroes within a session were removed because they typically reflected connection and disconnection boundaries rather than active charging behaviour. Zeroes in the middle of a session were retained because they could still represent informative irregular behaviour.

3. Enforcing one-minute cadence. For each session, expected and observed sample counts were compared. Let session i start at $t_i^{\min}$ and end at $t_i^{\max}$. The expected number of one-minute readings was defined as

$$n_i^{\exp} = \left\lfloor \frac{t_i^{\max} - t_i^{\min}}{60 \text{ s}} \right\rfloor + 1.$$

Observed rows were denoted n_i^{obs}, and the missing-count difference was $n_i^{\exp} - n_i^{\text{obs}}$. Sessions with more than 20% missing readings relative to $n_i^{\exp}$ were excluded in order to avoid heavy interpolation artefacts.

For retained sessions, the data were resampled to one-minute cadence. Dynamic numeric channels (soc, power, energy) were aggregated by one-minute mean and then linearly interpolated. Session-level static fields were forward-filled so that they remained constant within session.

4. Duration filtering and unreliable metadata removal. After cadence harmonisation, sessions shorter than 8 minutes and longer than 60 minutes were removed to stabilise the modelling conditions. The original `charging_duration` field was then dropped because it was not fully consistent with timestamp-derived duration and was therefore not treated as a trusted feature.

5. Temporal progress features. Within each session, elapsed minutes from the first observation were computed as

$$\texttt{minutes\_elapsed}_{i,t} = \frac{t - t_i^{\min}}{60\ \text{s}}.$$

A log-scaled progression feature was then defined as

$$\texttt{progress}_{i,t} = \min\left(\frac{\log(1 + \texttt{minutes\_elapsed}_{i,t})}{\log(1 + 120)},\ 1\right).$$

This feature allocated finer temporal resolution to the early part of each charging session, where changes in power and SOC were often strongest, while compressing the upper range of longer sessions. The resulting representation remained monotonic and bounded in $[0, 1]$.

6. Scale-aware and dynamic-shape features. To make charging power more comparable across charger capacities, relative power was defined as

$$\texttt{rel\_power}_{i,t} = \frac{\text{clip}\left(\frac{\texttt{power}_{i,t}}{\texttt{nominal\_power}_i},\ 0,\ 1.2\right)}{1.2}.$$

First differences were then computed per session:

$$\texttt{d\_power}_{i,t} = \Delta\texttt{power}_{i,t}, \qquad \texttt{d\_soc}_{i,t} = \Delta\texttt{soc}_{i,t},$$

with initial values set to zero. Short-span exponential moving averages were applied to both derivatives, giving `d_power_ema3` and `d_soc_ema3` with span = 3, in order to reduce local noise while retaining short-term trend information.

7. Charger-category encoding. Nominal charger power was binned into low, mid, and high regimes (≤ 75 kW, 75 to 200 kW, and > 200 kW), then represented as one-hot indicators: `charger_cat_low`, `charger_cat_mid`, and `charger_cat_high`.

8. Weather enrichment at session start. Ambient temperature was attached per session using nearest-station weather observations from Norwegian, Swedish, and Danish meteorological services. The modelling variable was the temperature nearest to the session start hour, rounded to integer degrees Celsius and stored as `temp`. The station identifier was retained as `nearest_weather_station` for traceability. I treated ambient temperature as a useful proxy for starting battery temperature, which could plausibly affect charging behaviour.

9. Taper-regime features informed by EDA. Based on the exploratory analysis described in subsection 4.1.3, a taper-onset threshold of 71% SOC was used to derive two regime features: `in_taper`, a binary flag for `soc` ≥ 71, and `dist_to_taper`, a normalised distance above the taper threshold.

After preprocessing, the EV charging dataset was stored in Parquet format for downstream modelling, anomaly scoring, simplification, and teaching-set construction.

4.1.3 Exploratory Data Analysis

Exploratory data analysis (EDA) acts as a methodological bridge between cleaned data and downstream design choices. The analysis is implemented in `02__EDA.ipynb` [85] and is used to produce explicit evidence for decisions in modelling, anomaly detection, curve simplification, and machine teaching.

The cleaned panel contains 1,520,027 measurements across 60,914 charging sessions, spanning 11 January 2020 to 3 November 2024 (Table 4.2). The schema check shows complete non-null coverage for all 23 variables used in the forecasting and explanation pipeline. This supports the retention of engineered temporal derivatives and contextual covariates without additional imputation.

Dependence analysis confirmed short-horizon predictability while showing progressive uncertainty with increasing horizon. The persistence baseline error increases from 7.39 to 36.97 kW for power between horizons $h = 1$ and $h = 14$ (Table 4.3), while power autocorrelation declines from 0.83 to -0.10 over the same range (Table 4.4). SOC remains strongly autocorrelated at approximately 0.99 to 1.00, indicating highly structured (linear) charging progression, as expected. These patterns motivate a power forecasting setup that emphasises short to medium horizons.

Feature relevance analysis further supports the selected representation. Across horizons $h = 1$ to $h = 5$, `rel_power` remains the strongest mutual-information driver of future power, followed by derivative and state features such as `d_soc_ema3`, `d_power`, and `soc` (Table 4.5). This empirical ranking justifies retention of relative-power scaling, gradient channels, and charger-capacity context in the predictive feature set. The engineered `temp` feature had very low forecasting relevance.

Charging-phase analysis identifies a median taper onset at 71% SOC with interquartile range 68% to 79% (Table 4.6). The median taper onset remains 71% across temperature quartiles, which supports representing taper as a stable regime transition rather than a narrowly temperature-specific effect. This directly motivates the taper-regime features used in the forecasting and anomaly-detection stack, and informs simplification constraints that preserve post-peak decline structure.

Operational diagnostics also inform model configuration. Label availability falls by 16.7% from $h = 1$ to $h = 5$, and session length has median 23 minutes with interquartile range 17 to 31 minutes (Table 4.7). This supports bounded horizon choices and sequence handling that avoids overextending long-horizon supervision on shorter sessions.

In aggregate, EDA is not treated as a separate results component. It functions as an operational design stage where descriptive outputs are converted into traceable methodological choices.

Table 4.2: Dataset overview from EDA summary statistics.

Metric	Value
Measurements (rows)	1,520,027
Charging sessions	60,914
Variables (columns)	23
Time span start	2020-01-11 12:37:00
Time span end	2024-11-03 19:32:00
Unique charging locations	285
Unique nominal power levels	22
Charger category levels	2 (Rapid, Ultra)
Missing values in listed variables	0

Table 4.3: Persistence baseline RMSE by forecast horizon.

Horizon h (min)	$\mathrm{RMSE}_{\mathrm{persist,power}}$ (kW)	$\mathrm{RMSE}_{\mathrm{persist,SOC}}$ (pp)
1	7.39	1.99
2	10.12	3.84
3	12.80	5.73
4	15.40	7.62
5	17.94	9.52
6	20.40	11.40
7	22.77	13.28
8	25.06	15.13
9	27.28	16.97
10	29.41	18.78
11	31.46	20.57
12	33.42	22.33
13	35.28	24.07
14	36.97	25.77

Table 4.4: Mean within-session autocorrelation by horizon.

Horizon h (min)	$\mathrm{ACF}_{\mathrm{power}}(h)$	$\mathrm{ACF}_{\mathrm{SOC}}(h)$
1	0.83	1.00
2	0.74	1.00
3	0.65	1.00
4	0.57	1.00
5	0.47	0.99
6	0.39	0.99
7	0.32	0.99
8	0.25	0.99
9	0.20	0.99
10	0.13	0.99
11	0.07	0.99
12	0.02	0.99
13	-0.04	0.99
14	-0.10	0.99

Table 4.5: Top mutual-information drivers of future power across short horizons.

Feature	MI@h1	MI@h2	MI@h3	MI@h4	MI@h5
rel_power	2.34	1.85	1.59	1.40	1.26
d_soc_ema3	1.05	0.97	0.90	0.83	0.77
d_power	0.76	0.73	0.70	0.69	0.68
soc	0.58	0.61	0.64	0.68	0.71
d_power_ema3	0.56	0.55	0.54	0.54	0.53
nominal_power	0.47	0.47	0.48	0.48	0.48
d_soc	0.36	0.35	0.34	0.32	0.30
charger_cat_low	0.21	0.21	0.21	0.21	0.21

Table 4.6: Taper onset summary from session-level SOC drop analysis.

Metric	Value
Sessions with detected taper onset	43,164
Share of all sessions	70.86%
Mean taper SOC	69.34%
Median taper SOC	71.00%
Standard deviation	13.62 pp
Interquartile range	68.00% to 79.00%
Minimum to maximum	4.00% to 100.00%
Median taper SOC by temp quartile Q1	71.00%
Median taper SOC by temp quartile Q2	71.00%
Median taper SOC by temp quartile Q3	71.00%
Median taper SOC by temp quartile Q4	71.00%

Table 4.7: Label availability by horizon and session-length distribution.

Horizon h (min)	$N_{\text{power labels}}$	$N_{\text{SOC labels}}$
1	1,459,113	1,459,113
2	1,398,199	1,398,199
3	1,337,285	1,337,285
4	1,276,371	1,276,371
5	1,215,457	1,215,457
Session length summary (minutes)		
Mean	24.95	
Standard deviation	10.53	
Minimum	8	
25th percentile	17	
Median	23	
75th percentile	31	
Maximum	60	

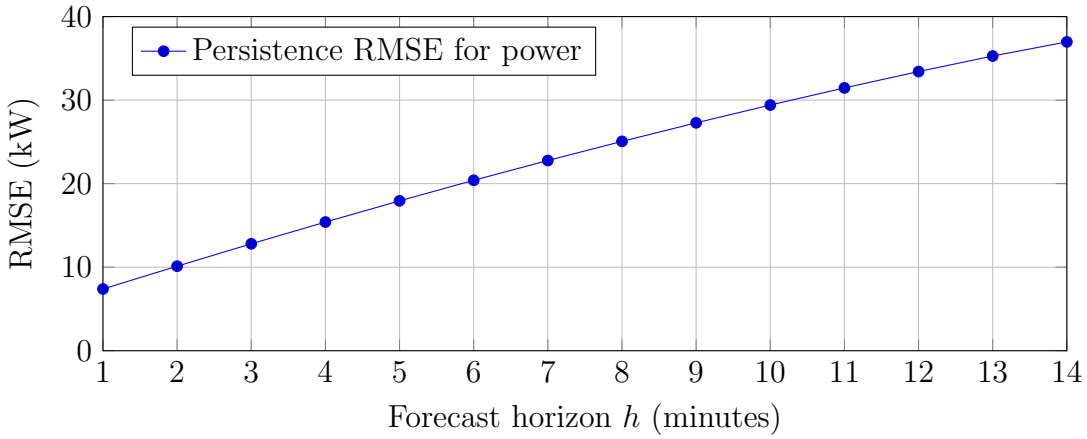

Figure 4.1: Persistence-baseline error growth with forecast horizon for charging power.

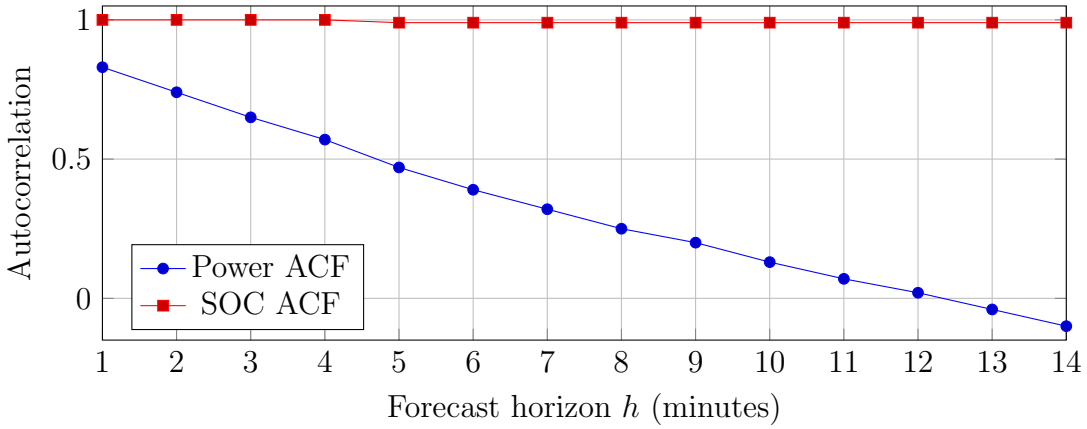

Figure 4.2: Mean within-session autocorrelation by horizon for power and SOC.

4.2 Anomaly Detection Methodology

The modelling task of mapping EV charging sessions to one of `normal, abnormal` sits well within the problem set of SL-TSAD, where the deployed AI system ultimately behaves as a binary sequence classifier. I first trained a forecasting model for expected charging behaviour, then converted forecasting error into a session-level anomaly score, and finally applied thresholding to obtain one of the two operational labels. This design allowed the pipeline to learn from unlabelled industrial time-series data while still producing classifier-like outputs that could be used in the MT4XAI framework [96, 49].

4.2.1 Forecasting Objective and Metrics

The forecasting model used a multivariate input vector and a univariate target. For each charging session and each minute t, the model received static context and dynamic charging-state features, while the prediction target was future `power`. Table 4.8 summarises the features used during training.

Feature	**What it represents**
`temp`	Ambient temperature associated with the session context.
`nominal_power`	Charger rated power capacity for the charging point.
`power`	Measured charging power at time t (kW).
`soc`	Battery state of charge at time t (%).
`progress`	Log-scaled session progress in $[0, 1]$, derived from elapsed minutes.
`rel_power`	Power normalised by charger capacity.
`d_power`	First-order temporal change in power.
`d_soc`	First-order temporal change in state of charge.
`d_power_ema3`	Smoothed short-term trend of `d_power`.
`d_soc_ema3`	Smoothed short-term trend of `d_soc`.
`in_taper`	Binary indicator for taper regime.
`dist_to_taper`	Distance to the estimated taper point.

Table 4.8: Input features used for multi-horizon forecasting.

Rather than predicting absolute future power directly, the model was trained on residual targets. For horizon $h \in \{1, \ldots, H\}$,

$$r_{t,h} = p_{t+h} - p_t,$$

where p_t denotes charging power at time t. At inference time, absolute forecasts were reconstructed as

$$\hat{p}_{t,h} = p_t + \hat{r}_{t,h}.$$

This residual formulation improved numerical stability and aligned neighbouring horizons through a shared base value.

The forecasting setup was multi-horizon with $H = 5$, so one forward pass produced forecasts for all horizons $1, \ldots, 5$. Horizons were weighted with exponential decay,

$$w_h = \frac{\exp(-\alpha(h-1))}{\sum_{j=1}^{H} \exp(-\alpha(j-1))},$$

where $\alpha > 0$ controlled the balance between short-horizon and longer-horizon accuracy.

Training used a horizon-weighted Huber loss over valid time-horizon pairs,

$$\mathcal{L} = \frac{1}{|\mathcal{V}|} \sum_{(t,h)\in\mathcal{V}} w_h \, \mathrm{Huber}_\delta(\hat{p}_{t,h} - p_{t+h}),$$

with masking to exclude padded regions and invalid forecast positions. Huber loss was chosen for robustness to occasional large residuals.

Validation and model selection were based on Macro-RMSE in physical units (kW). For each session, per-horizon RMSE values were first aggregated by the horizon weights and then macro-averaged across sessions. This prevented any single long session or single horizon from dominating the evaluation [66]. In the anomaly-detection stage, I also computed RWSE as a robust diagnostic metric based on median and median absolute deviation scaling of residuals.

4.2.2 Candidate Forecasting Architectures

I implemented and compared two sequence-modelling families, a multi-layer LSTM and a causal WaveNet-style TCN. Both architectures emitted residual forecasts with shape (B, T, H, C), where B denotes batch size, T sequence length, $H = 5$ forecast horizons, and $C = 1$ target channel.

The LSTM used packed variable-length sequences, recurrent hidden-state processing, and a linear output head that emitted all horizons jointly at each time step. The TCN used an input 1×1 projection followed by stacked causal dilated depthwise-separable convolution blocks with residual and skip connections, and then a 1×1 prediction head. The purpose of this comparison was not standalone benchmarking, but methodological model-family selection under a shared forecasting objective.

4.2.3 Model Training and Validation

The data were split by `charging_id` using grouped shuffle splitting in order to prevent session leakage across partitions. The split was 70% train, 10% validation, and 20% test, corresponding to 42,639 training sequences, 6,092 validation sequences, and 12,183 test sequences. The test set size was set to double that of the validation set size because the overall dataset was sufficiently large for robust model fitting with 70% training data, while a larger held-out test partition improves the statistical stability of final performance estimates and supports more reliable error analysis across heterogeneous charging sessions.

Scaling was fitted on the training set only and applied unchanged to validation and test data. Separate Min-Max scalers were used for `power`, `soc`, static context features, and delta features. Predictions were reconstructed and inverse-transformed before metric computation so that evaluation remained in kW.

Each session was represented as a variable-length sequence, and residual targets were constructed as $y_{t+h} - y_t$. Batching used length-bucket sampling and a custom collate function with padding and sequence-length tensors. This reduced padding waste and supported mask-aware loss computation on heterogeneous session lengths.

The optimisation stack used AdamW, automatic mixed precision, gradient clipping, and ReduceLROnPlateau scheduling. Validation was performed after each epoch with Macro-RMSE, and this validation metric was also the optimisation target for hyperparameter tuning.

4.2.4 Hyperparameter Tuning and Architecture Selection

Hyperparameter tuning used Ray Tune with Bayesian Optimisation with HyperBand (BOHB) search and HyperBand scheduling. Each architecture was tuned with 64 sampled configurations, and each trial reported validation Macro-RMSE as the objective. BOHB was selected because it combines model-based search with multi-fidelity resource allocation, which is well suited to this setting where individual training runs are computationally expensive and performance can be estimated reliably from partial training. The Bayesian component improves sample efficiency by prioritising promising regions of a mixed hyperparameter space, while HyperBand prunes weak trials early and reallocates budget to stronger candidates. This yields a better trade-off between exploration

breadth and compute cost than grid search, random search, or Bayesian search without early-stopping control, and supports a fair architecture comparison under a fixed tuning budget.

For the LSTM, the tuned space included hidden dimension, number of layers, dropout, learning rate, weight decay, batch size, gradient clipping norm, and horizon-decay parameter α_h. For the TCN, the tuned space included hidden dimension, number of layers, kernel size, dropout, learning rate, weight decay, batch size, gradient clipping norm, and α_h.

The best checkpoint from each architecture was exported to a clean `.pth` artefact. Architecture selection was then based on the validation set only. After I had selected the best model, I evaluated that model once on the held-out test set. The comparative outcomes and the final selected hyperparameters are reported in section 5.1.

4.2.5 Inference Protocol for Anomaly Detection

At inference time, the selected forecaster was applied sequence-wise to produce residual forecasts, reconstruct absolute power trajectories, and compute session-level error scores. Preprocessing and scaling were matched exactly to the modelling stage in order to avoid train/inference skew.

Anomaly detection thresholds were estimated from validation-set error distributions and then applied unchanged to unseen sessions. The decision rule is identical to Equation 2.1.

At this stage, several candidate design choices were systematically explored, including the session-level error metric (Macro-RMSE or RWSE), the percentile threshold, the horizon-decay parameter used in anomaly scoring, and the evaluation start index $t_{\text{min,eval}}$. To support quantitative calibration of these parameters, a lightweight interactive inspection tool was developed in the notebook `04__Anomaly_Detection.ipynb` [85]. The tool enabled rapid inspection of session-level forecasts, reconstructed trajectories, and resulting anomaly scores under alternative parameter configurations, making it possible to compare behavioural effects directly across charging sessions. Because no external anomaly labels were available, and because collecting expert annotations was not considered necessary for the aims of this thesis, parameter selection was based on internal quantitative analysis rather than supervised validation. This is justified by the fact that anomaly detection performance itself is not the primary research objective, but rather a supporting component in the construction of an MT4XAI system around the classifier output. The final configuration and its empirical behaviour are reported in section 5.2.

4.3 Simplifying Examples for the Teaching Pipeline

This section describes how raw charging-session trajectories were simplified before example selection for machine teaching. The goal was to reduce visual and structural complexity while preserving the behaviour of the deployed anomaly classifier. In this work, simplification was treated as a constrained optimisation problem over piecewise-linear representations of full charging sessions.

Let x denote one charging session and let $f_{\mathrm{AI}}(x) \in \{\text{normal}, \text{abnormal}\}$ denote the anomaly classifier defined in section 4.2. For a simplified curve $\tilde{x}$, the primary constraint was decision preservation,

$$f_{\mathrm{AI}}(\tilde{x}) = f_{\mathrm{AI}}(x),$$

while the optimisation target reduced a complexity measure $\delta(\tilde{x})$, implemented here as the number of line segments k.

In the final teaching workflow, simplification was applied primarily to the `power` channel because a charging session's "forecastability" is largely driven by power behaviour, as seen in subsection 4.1.3. Simplification of `soc` was also implemented to provide additional context about charging phase, but the anomaly classifier itself did not operate on a separately simplified SOC decision rule. The SOC simplification was included in the machine teaching system to remove uncertainty about whether minor fluctuations in SOC were important, when they are in fact irrelevant. The final presentation choices are reported in section 5.3.

4.3.1 Optimal Robust Simplifications

I adopted the Optimal Robust Simplifications (ORS) framework proposed by Telle et al. [95] and instantiated it for the forecasting-based anomaly detector developed in this thesis.

For one session of length T, let the original power trajectory be $y \in \mathbb{R}^T$. A candidate simplification is represented by ordered pivot indices

$$P = (p_0, \ldots, p_k), \qquad 0 \le p_0 < \cdots < p_k = T - 1,$$

and a dense piecewise-linear reconstruction $s_P \in \mathbb{R}^T$ obtained by interpolation between pivots.

Stage 1: candidate generation. Stage 1 generated candidate pivot sets that traded reconstruction error against complexity. For the dynamic-programming-based modes, the stage-1 score was

$$C_{\mathrm{S1}}(P) = \alpha\, E(P) + \beta\, k,$$

where $E(P) = \sum_{t=0}^{T-1}(y_t - s_{P,t})^2$ is squared reconstruction error and $k = |P| - 1$. The implementation followed the two-stage ORS structure with top-q candidate retention.

Stage 2: label-preserving robust selection. Each stage-1 candidate was reconstructed and re-scored by the deployed classifier. Only candidates that preserved the original session label were feasible:

$$\mathcal{P}_{\mathrm{feasible}} = \{P \mid f_{\mathrm{AI}}(s_P) = f_{\mathrm{AI}}(y)\}.$$

For each feasible candidate, robustness was estimated by perturbing pivot values with uniform noise in $[-\varepsilon, +\varepsilon]$, rebuilding R perturbed curves, and reclassifying each one. Fragility was defined as the label-flip rate,

$$\mathrm{frag}(P) = \frac{1}{R}\sum_{r=1}^{R} \mathbb{1}\left[f_{\mathrm{AI}}(s_P^{(r)}) \neq f_{\mathrm{AI}}(y)\right].$$

The final objective became

$$J(P) = C_{\mathrm{S1}}(P) + \gamma\, \mathrm{frag}(P),$$

and the selected simplification was

$$P^* = \arg\min_{P \in \mathcal{P}_{\mathrm{feasible}}} J(P).$$

In this pipeline, f_{AI} was the forecasting-based anomaly classifier from section 4.2, so ORS was optimised directly against the same decision rule used in downstream teaching and evaluation.

4.3.2 Extensions to ORS

I extended the baseline ORS workflow in several ways to improve runtime and to align simplification more closely with the anomaly-detection objective.

First, I implemented a prefix-sum-accelerated dynamic-programming variant for stage 1. Prefix sums over $\{1, t, t^2, y, ty, y^2\}$ allowed constant-time segment SSE queries when building error tables, while preserving the same stage-1 objective and candidate semantics.

Second, I implemented an RDP-based stage-1 heuristic that swept over candidate ε values and kept the best candidate per k under the same reconstruction-complexity trade-off. This provided a faster alternative when exact dynamic programming was unnecessarily expensive.

Third, I adapted endpoint handling to the anomaly-scoring setup. Because forecasting errors were evaluated only from $t \geq t_{\text{min,eval}}$, I allowed candidate search on the evaluated suffix with right-end anchoring and slope-preserving extrapolation of the left prefix. This avoided forcing unnecessary breakpoints in parts of the curve that the anomaly score did not actually use, which avoids unnecessary confusion about which part of the simplified curve was important for the anomaly detection system's decision.

Fourth, I vectorised robustness estimation by scoring batches of perturbed curves in one forward-pass loop, and I implemented practical fallbacks for cases where no feasible candidate survived the label-preservation and k-range constraints.

Across these extensions, I treated simplification as an engineering component that had to balance fidelity, compactness, and tractable runtime. The comparative outcomes and final operational choices are reported in section 5.3.

4.4 Machine Teaching for XAI Framework

Chapter 2 framed Machine Teaching (MT) as the problem of selecting a small set of labelled witnesses that helps a learner recover a target behaviour while minimising teaching cost. Chapter 3 then argued that, for time-series XAI, this requires jointly managing four design choices: faithfulness to the deployed classifier, representational simplicity of individual examples, coverage of the classifier's behavioural variation, and the ordering in which examples are shown to a fallible learner. The MT4XAI pipeline implemented in this thesis instantiates those four elements for EV charging-session anomaly detection.

Although the implementation in this thesis was instantiated for SL-TSAD on EV charging sessions, the MT4XAI pipeline was model-agnostic at the framework level. It

requires only four components: a deployed time-series classifier f_{AI}, a label-preserving simplification operator, a representation map for comparing candidate examples, and a selection policy for constructing a compact teaching set. In this implementation, f_{AI} is the forecasting-based anomaly detector from section 4.2, the simplification operator is extended ORS from section 4.3, the representation map is a fixed-length embedding of simplified power curves, and the selection policy is a stratified facility-location objective with curriculum-aware serving.

4.4.1 Adapting the MT4XAI Formalism

Let $\mathcal{X}$ denote the space of charging sessions and let $f_{\text{AI}} : \mathcal{X} \to \{0, 1\}$ be the deployed sequence-level classifier, where 0 denotes `normal` and 1 denotes `abnormal` (anomalous). In MT terms, the target concept is therefore the deployed decision behaviour itself, $\theta_{\text{AI}} \equiv f_{\text{AI}}$, and the explanatory task is to construct a teaching set of labelled charging sessions that helps a learner form a hypothesis $\hat{\theta}$ that approximates f_{AI}.

The implementation is best understood as *representation teaching* rather than only concept teaching. The learner never observes the classifier's full multivariate input space directly. Instead, for each selected session x_i, the teacher first constructs a faithful simplified surrogate $\tilde{x}_i$, and then renders a learner-visible witness (example)

$$w_i^{(c)} = \rho_c(x_i, \tilde{x}_i), \tag{4.1}$$

where ρ_c is the presentation policy for condition $c \in \{A, B, C, D, E\}$. Thus, the same underlying session can be taught through several representations: raw trajectories, raw-plus-simplification overlays, or simplified-only views. This mirrors the distinction between concepts, representations, and witness sets introduced in section 2.3, and aligns the pipeline with the representation-teaching perspective that redundancy and representational choice matter for what is actually learned [27].

Each candidate example is represented as

$$u_i = \big(x_i, \tilde{x}_i, w_i, y_i, k_i, z_i, m_i, r_i\big), \tag{4.2}$$

where $y_i = f_{\text{AI}}(x_i) = f_{\text{AI}}(\tilde{x}_i)$ is the AI label, k_i is simplification complexity measured as the number of line segments, z_i is the selection embedding, m_i is the margin (i.e.

the signed distance to the anomaly threshold), and r_i is the simplification robustness probability. In implementation terms,

$$m_i = \tau - s(\tilde{x}_i), \qquad r_i = 1 - \mathrm{frag}(\tilde{x}_i),$$

where $s(\tilde{x}_i)$ is the simplified session's Macro-RMSE anomaly score, τ is the deployed threshold, and $\mathrm{frag}(\tilde{x}_i)$ is the label-flip rate under bounded perturbations of the simplification pivots.

This formulation makes simplification part of the MT objective rather than a cosmetic post-processing step. Only simplifications that preserve the deployed label are admitted to the teaching pool. In the language of teaching size from Chapter 2, ORS reduces the per-example complexity term by replacing a dense charging trajectory with a shorter piecewise-linear surrogate while keeping the witness faithful to the deployed classifier.

The forecasting model itself operates on a richer multivariate feature space than the learner ever sees. However, the teaching interface intentionally restricts learner-visible information to the most salient trajectory-level channels. Depending on condition, participants are shown raw or simplified `power` and `soc` trajectories, while internal model features such as temperature, taper indicators, and derivative features remain hidden. This follows the MT4XAI principle that explanation quality depends not only on faithfulness, but also on what information is made visible to the learner [40].

Selection is performed in a separate representation space, and similarity comparisons play an important part. I therefore mapped each simplified power curve to a fixed-length vector using techniques adapted from Fulcher and Jones [30]. Let $\tilde{p}_i$ denote dense simplified power for session i. Let $\tilde{p}_i$ denote the dense simplified `power` curve for session i. The embedding map ϕ is defined as

$$\phi(\tilde{p}_i) = \left[z_{1:L},\ \Delta z_{1:L},\ \pi_{1:P},\ \omega_{1:P} \right] \in \mathbb{R}^{2L+2P}, \tag{4.3}$$

where L is the embedding vector length, z is the resampled z-scored curve, Δz its first difference, and (π, ω) the top-P peak prominences and widths. The operational embedding configuration uses $L = 128$ and $P = 4$. This means that the pipeline distinguishes between four related but non-identical objects: the original model input x_i, the faithful simplified surrogate $\tilde{x}_i$, the embedding $z_i = \phi(\tilde{p}_i)$ used for subset selection, and the learner-facing witness $w_i^{(c)}$ used in the teaching session.

4.4.2 Teaching Set Construction

Teaching-set construction is implemented as a four-stage pipeline.

Stage 1: constructing a faithful candidate pool. First, the anomaly detection system is run (inference) for all charging sessions in the validation and test sets using the same Macro-RMSE thresholding rule as in section 4.2. The validation set served as the source for the teaching pool, while the test set served as the source for exam items used later in the MLLM evaluation. The teaching pool was class-balanced by retaining all abnormal sessions and randomly sampling an equal number of normal sessions. For each sampled session, the pipeline ran ORS, stored raw and simplified arrays, and attached metadata including k, simplified error, fragility, robustness probability, and signed threshold margin. The resulting pool therefore contained only classifier-faithful simplified surrogates.

Stage 2: stratification and budget assignment. Pool examples were then stratified by class and simplification complexity k using class-conditional k-bins. In the main notebook workflow, fixed binning was used in order to keep bin edges stable across runs. Per-bin budgets were derived from per-class targets through even allocation, followed by a spillover rule that reassigned infeasible demand to nearby bins within the same class. This stage operationalised the Chapter 2 and 3 requirement that compact teaching sets should still preserve coverage of rare or pedagogically important representation regimes rather than being dominated by the most common low-k examples.

Stage 3: selecting a compact set with coverage and controlled redundancy. Let U denote the candidate pool and $S \subseteq U$ the selected teaching set. Selection followed a facility-location objective over the embedding space:

$$F(S) = \sum_{i \in U} \max_{s \in S} \mathrm{sim}(z_i, z_s), \tag{4.4}$$

where sim is cosine similarity on L2-normalised embeddings. In implementation, this objective was optimised independently within each class-bin stratum under explicit budgets, and the resulting subsets were then unioned into the final teaching set.

At each greedy step, the score for a candidate $x \in U \setminus S$ was

$$\text{score}(x) = \Delta F(x \mid S) + \lambda_m \, m(x) + \lambda_r \, r(x), \tag{4.5}$$

where $\Delta F(x \mid S) = F(S \cup \{x\}) - F(S)$, $m(x)$ is signed threshold margin, and $r(x)$ is robustness probability. The operational setting used $\lambda_m = 0.10$ and $\lambda_r = 0.05$.

This stage balanced diversity against controlled redundancy. Facility-location discouraged near-duplicate examples in the embedding space, while class-conditional stratification, spillover, and a deterministic `min_per_k` seeding rule prevented rare complexity levels from disappearing entirely. The implementation did not explicitly optimise conditional teaching size or a cognitive model of forgetting, but it operationalised the same MT intuition from Chapter 2: for a fallible learner, some structured overlap is preferable to collapsing the teaching set to only the most central examples.

Stage 4: serving examples from S. Selection and curriculum were separated deliberately. The selected set determined *which* sessions were taught, while the serving policy determined *how* the learner accumulated them over time. All conditions were therefore served from the same selected session IDs, while modality and order varied by group policy. Groups A and B used overlay witnesses, Group C used raw-only witnesses, and Groups D and E used simplified-only witnesses, with Group E reusing Group D assets.

Curriculum was implemented at the example-serving stage rather than at teaching-set construction. For curriculum-enabled conditions, examples were ordered class-conditionally by simplification complexity k and decision margin. In implementation terms, normal sessions were ordered by ascending (k, m), while anomalous (abnormal) sessions were ordered by descending (k, m), and the final stream alternated between normal and anomalous examples. This policy reflected the operational assumption that visually simple normal sessions would be easier starting points, whereas highly structured anomalous sessions could provide salient counterexamples. The intuition was consistent with the discussion of human default expectations in Yang et al. [104]. Non-curriculum conditions used randomised order over the same selected session IDs.

Taken together, these four stages yield a coherent MT4XAI pipeline: ORS reduces per-example teaching cost while preserving classifier behaviour, the embedding defines a representation space for comparing candidates, facility-location selects a compact witness set with controlled redundancy, and the serving policy realises the curriculum and modality manipulations evaluated in the later chapters.

4.5 Evaluating MT4XAI with MLLMs

The explanation pipeline was evaluated through forward simulation, following the paradigm of Martí-Pérez et al. [73], where the MLLM-based learner first observes labelled examples from the deployed AI system and then predicts the AI's labels for unseen examples. In this thesis, each MLLM participant corresponded to one independent trial in the experiment runner.

I used `gpt-5-nano` as the proxy learner MLLM because it offered a practical balance between cost, speed, and reasoning capability, which made it feasible to run many trials under multiple conditions.

4.5.1 Experimental Conditions

The experiment automatically assigned MLLM participants into one of six groups using balanced random allocation. Each group encountered a separate experimental condition:

- **Group A (Overlay + Curriculum):** Overlay modality in teaching and post-exam. Teaching order follows curriculum metadata order.
- **Group B (Overlay + Unordered):** Same overlay modality as A, but teaching order is randomised per participant.
- **Group C (Raw-only + Unordered):** Raw-only modality, teaching order randomised per participant.
- **Group D (Simplified-only + Curriculum):** Simplified-only modality, curriculum order.
- **Group E (Simplified-only + Curriculum + Frequent Rule Updates):** Reuses Group D teaching assets and order, but enforces per-example rule-of-thumb updates with strict JSON schema.
- **Group F (No Teaching Baseline):** No teaching phase, raw-only pre and post exams.

To preserve the behavioural hypotheses inferred by the MLLM participants during the teaching phase, Groups A–D were required at regular intervals to externalise their current interpretation of the AI classifier's behaviour (see Appendix C for full prompt templates). These structured descriptions included the fields `normal_cues`, `abnormal_cues`,

`exceptions`, `confidence`, and `rule_of_thumb`, where the latter represented a concise provisional decision rule describing how the participant believed the AI classifier distinguished normal from abnormal charging sessions.

This intervention served two methodological purposes. First, it reduced information loss caused by the rapidly expanding conversational context, since each charging-curve visualisation consumed a substantial number of tokens and earlier examples could otherwise become less salient within the active context window. Second, it operationalised learning as explicit belief revision rather than parameter adaptation. Deployed LLMs perform inference using fixed model parameters and do not update their internal weights during interaction unless explicitly retrained or fine-tuned offline [12, 100]. For this reason, learning in the present experiment was simulated through structured belief updating within the prompt context rather than through any underlying change in model parameters.

Group E followed a stricter variant of this design. After the first teaching example, participants were required to state an initial `rule_of_thumb`, and after every subsequent example they were explicitly instructed either to retain or revise that rule. This created a controlled approximation of iterative rule refinement after each new observation. Group F did not produce intermediate descriptions or rule updates.

This design enabled controlled contrasts for simplification effects, curriculum effects, teaching effects, and structured rule-maintenance effects, while keeping the selected teaching examples identical across all teaching conditions.

4.5.2 Evaluation Structure

Each participant trial consisted of three phases.

In the **pre-teaching exam**, the model was shown one of two disjoint exam sets and asked to predict the AI's label for each item. All groups used raw-only modality in this phase.

In the **teaching phase**, Groups A–E received labelled teaching items from the selected teaching set, while Group F skipped teaching and functioned as a baseline. The visual modality depended on group condition. Curriculum and rule-maintenance policies were applied only where specified by condition.

In the **post-teaching exam**, the participant received the opposite exam set. The same item never appeared in both pre- and post-exam for the same participant. For the teaching groups, the post-exam evaluated application of the rule state learned during teaching rather than further rule editing.

The complete prompts and JSON schemas are documented in Appendix C. They are not reproduced in full here because the methodological focus of this chapter is the experimental design, not the low-level prompt syntax.

4.5.3 Controlling Prior Information and Leakage

Each participant began from a fresh conversation state with fixed system instructions. Exam and teaching images were anonymised and shown as static plots. During exams, labels were never provided. During teaching, the AI label was supplied explicitly in the prompt. Pre- and post-exam sets were disjoint within participant, and the post-exam for the teaching groups used the rule state produced during teaching rather than allowing unrestricted continued updating.

These controls were intended to reduce item leakage, reduce memory confounds between exam phases, and isolate the effect of the teaching intervention as far as possible within an LLM-based experimental setup.

4.5.4 Outcome Measures

The primary participant-level metric was accuracy in each phase,

$$\text{Accuracy}_{\text{phase}} = \frac{\sum \text{is\_correct}}{30},$$

and the main improvement metric was

$$\Delta\text{Accuracy} = \text{Accuracy}_{\text{post}} - \text{Accuracy}_{\text{pre}}.$$

I also computed a relative teaching effect versus the no-teaching baseline:

$$\text{RelativeEffect}_g = \left(\frac{1}{n_g}\sum_{i=1}^{n_g}\Delta\text{Accuracy}_{g,i}\right) - \left(\frac{1}{n_F}\sum_{j=1}^{n_F}\Delta\text{Accuracy}_{F,j}\right), \quad g \in \{A, B, C, D, E\}.$$

At the software level, all item-level and participant-level outputs were persisted to structured files so that experiments could be resumed and audited consistently.

4.6 Methodological Assumptions and Validity Constraints

This section consolidates the assumptions and validity constraints that shape the interpretation of the methodology.

4.6.1 Data and preprocessing

The pipeline assumed that each `charging_id` corresponded to one coherent charging event and that sorting by timestamp recovered a valid within-session temporal process. It also assumed that owner-reported EV model identity was sufficiently reliable for constructing a large single-vehicle dataset centred on the Audi e-tron 55. This assumption was operational rather than absolute. Some reporting error was possible, but I judged it unlikely to dominate the downstream modelling behaviour.

The preprocessing also assumed that leading and trailing zero-power segments mainly reflected boundary artefacts, that sessions shorter than 8 minutes or longer than 60 minutes were less suitable for the modelling objective, and that session-start ambient temperature from the nearest available weather station was a reasonable proxy for local charging conditions.

4.6.2 Forecasting and anomaly scoring

The forecasting-based anomaly-detection pipeline assumed that the unlabelled training corpus was dominated by normal charging behaviour, so that a forecaster trained on the full dataset would primarily learn expected behaviour. It also assumed that large session-level forecasting errors corresponded, at least operationally, to behaviour that was useful to treat as abnormal.

A central validity limitation is that the anomaly labels used later in the pipeline were pseudo-labels induced by the forecasting-and-thresholding rule. They were not externally validated fault labels. I did not conduct a separate expert-labelling study for the anomaly-detection subsystem because that would have required substantial domain-expert time, and optimising a production-quality anomaly detector was outside the scope of the thesis.

The central thesis question was not whether I could build the best possible EV charging anomaly detector, but whether a machine-teaching explanation pipeline could be built on top of a non-trivial time-series classifier and whether its behaviour could be simulated by an MLLM. The anomaly-detection subsystem was therefore treated as the deployed AI system to be explained, not as the primary optimisation target of the thesis.

4.6.3 Simplification and teaching-set construction

The simplification stage assumed that classifier-relevant charging-power morphology could be represented adequately with a piecewise-linear curve, provided that the simplified curve preserved the classifier's decision. It also assumed that local robustness under bounded pivot perturbations was a useful proxy for simplification stability.

A further design assumption was that anomaly decisions were driven primarily by charging-power behaviour, while SOC mainly contributed contextual information about charging phase. This assumption was informed by qualitative inspection of many anomaly-detection plots using the interactive inspection tool, not by a formal ablation study. I therefore treat it as an empirically informed engineering assumption rather than a proven property of the classifier.

The teaching-set construction stage further assumed that the embedding $\phi(\tilde{p})$ preserved enough shape information for diversity-based selection, that simplification complexity k and decision margin could act as useful proxies for pedagogical difficulty, and that keeping the underlying selected session IDs fixed across conditions improved the interpretability of modality comparisons.

4.6.4 MLLM evaluation

The MLLM evaluation assumed that forward simulation with a multimodal language model could function as a useful proxy for controlled, scalable comparison of teaching conditions. This does not imply that MLLMs are equivalent to human participants. Rather, the MLLM experiment was intended as an intermediate evaluation layer that could reveal whether a consistent teaching signal existed at all under tightly controlled experimental conditions.

The results from this part of the thesis must therefore be interpreted as evidence about proxy-learner simulateability under the given prompt and image protocol, not as a substitute for a full human-subject study.

4.7 Data, Software and Reproducibility

4.7.1 Data

The EV charging-session dataset was provided by an industrial partner who has requested to remain anonymous for the purpose of this thesis. Due to data-ownership constraints, the original raw dataset is not redistributable. However, an anonymised dataset is available through the public project repository, subject to a non-commercial licence agreement. The anonymised dataset is identical in modelling content to the one used in the thesis, except that `timestamp`, `lat`, `lon`, and `nearest_weather_station` have been removed to protect the identity of the dataset owner. Individual EV owners are not identifiable in either the original or the cleaned datasets, as neither direct personal identifiers nor indirect proxy variables that could enable re-identification are present among the recorded features.

The primary modelling dataset is stored as `etron55-charging-sessions.parquet` [85], produced by the preprocessing pipeline described earlier in this chapter. Intermediate and final artefacts for teaching-pool construction and MLLM evaluation are also persisted in deterministic project directories.

4.7.2 OS and Hardware Specifications

All experiments were run in a local Linux environment (Fedora 40 under WSL2) on a single workstation with the following specifications:

- GPU: NVIDIA GeForce RTX 4070 Laptop GPU (8 GB VRAM)
- CPU: Intel Core i9-13900HX
- RAM: 16 GB

GPU acceleration was used for model training and for the more computationally intensive simplification and MLLM experiment stages.

4.7.3 Software

The software developed for this thesis is publicly available as a GitHub repository [85]. The implementation followed a notebook-plus-package workflow. Ordered notebooks documented the main methodological stages, while reusable logic was packaged under `src/` in the `mt4xai` and `mllm_experiment` Python packages.

The repository also contains a pinned dependency snapshot (`linux_requirements.txt`), package metadata (`pyproject.toml`), and a central configuration file (`config.yaml`). Installation and execution steps are documented in the project `README.md`.

The software environment targeted Python 3.12 and used pinned dependencies from `linux_requirements.txt`. The project's reusable Python asset libraries were packaged via `pyproject.toml`. Core methodology settings were centralised in `config.yaml`, including paths, random seeds, inference defaults, and teaching-related settings.

Randomness was controlled through explicit seeds. A project-wide seed (`random_seed=42`) was used in the main pipeline configuration, and MLLM trials also exposed a CLI seed parameter with logged effective seed values for reruns.

Environment variables were used for external services. These included `OPENAI_API_KEY` for MLLM trials and weather-service credentials for optional re-running of the feature engineering pipeline. The credentials are personal and are therefore kept out of source code, however, new functional credentials are easily obtainable by creating personal accounts for the external services. See the setup guide in the repository `README.md` [85] for details.

Chapter 5

Results and Design Choices

This chapter reports the comparative experiments, final engineering choices, and empirical results that instantiated the methodology from Chapter 4. I focus on the four parts of the pipeline where practical trade-offs and observed outcomes mattered most: forecasting-model selection, anomaly-scoring calibration, curve simplification, and the MT4XAI MLLM evaluation. The aim is not to repeat the methodology chapter, but to report which options I explored, how I compared them, which choices were ultimately carried forward into the MT4XAI pipeline, and how the resulting teaching conditions performed in the proxy-learner experiment.

5.1 Forecasting Model Experimentation

In my modelling work I compared two model families under the same multi-horizon residual-forecasting objective: an LSTM and a causal TCN. I treated this as an engineering and architecture selection problem rather than a pure model benchmarking exercise. Both families were trained with the same data split, the same loss definition, and the same validation metric, and both were tuned with Ray Tune using 64 BOHB configurations.

Figures 5.1–5.4 show that both architectures learn the forecasting task quickly, with the largest reductions in Huber loss and validation Macro-RMSE occurring in the early epochs before the gains taper off. The LSTM reaches a lower and earlier validation plateau, reducing the validation Macro-RMSE from about 4.8 kW to roughly 2.88 kW, whereas the TCN improves more gradually and stabilises at a higher level of about 3.13

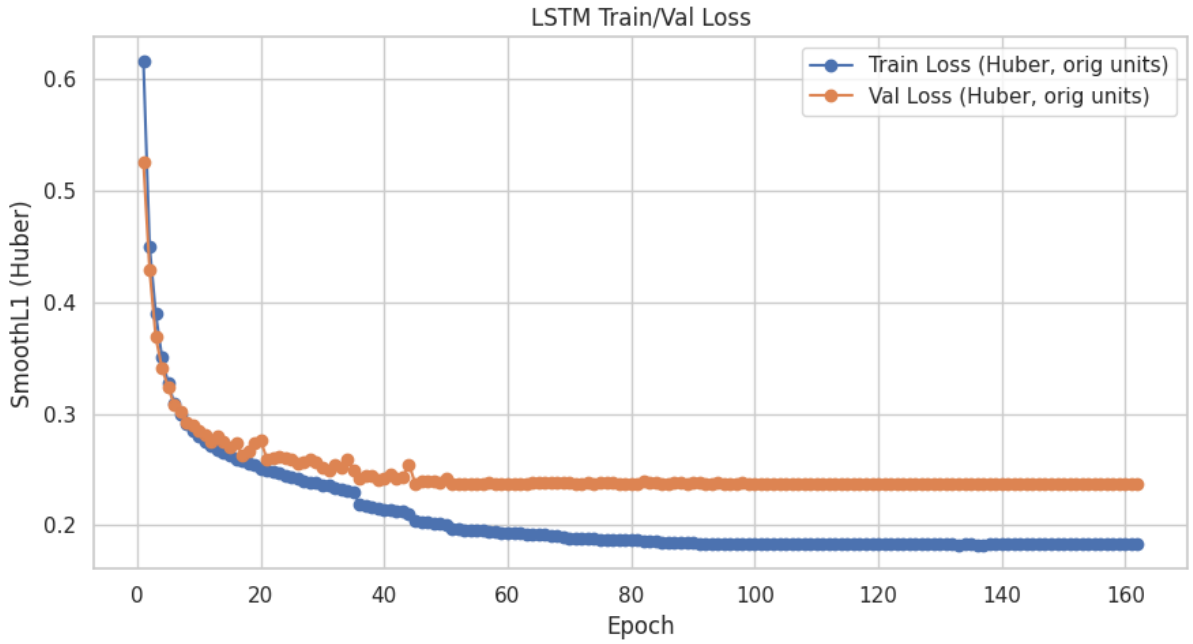

Figure 5.1: Best LSTM model's training and validation loss

kW. The LSTM also shows a more visible separation between training and validation Huber loss, while the TCN's two loss curves remain much closer throughout training, which is consistent with the stronger regularisation of the selected TCN configuration. Taken together, the loss and validation-RMSE plots indicate stable convergence for both model families, but a consistently better validation-time fit for the LSTM, which justifies its selection as the forecasting model used in the anomaly detection system.

Architecture	**Search budget**	**Tuning time**	**Outcome**
LSTM	64 BOHB trials	1 h 37 min	Achieved the lowest validation Macro-RMSE and was selected.
TCN	64 BOHB trials	2 h 43 min	Remained competitive, but did not outperform the best LSTM checkpoint on validation

Table 5.1: Summary of the architecture-tuning runs.

The LSTM's final hyperparameter configuration was `hidden_dim=256`, `num_layers=4`, `dropout=0.0027575`, `lr=0.00050155`, `weight_decay=1.1078e-06`, `batch_size=32`, `grad_clip_norm=5.0`, `alpha_h=0.518759`, and `horizon=5`. This model was then exported as `Models/final/final_model.pth` [85] and used in all downstream stages of the MT4XAI pipeline.

On the held-out test set, the selected LSTM achieved a Macro-RMSE of 3.5138 kW. Qualitative inspection of forecast trajectories like the one in Figure 5.5 suggested that

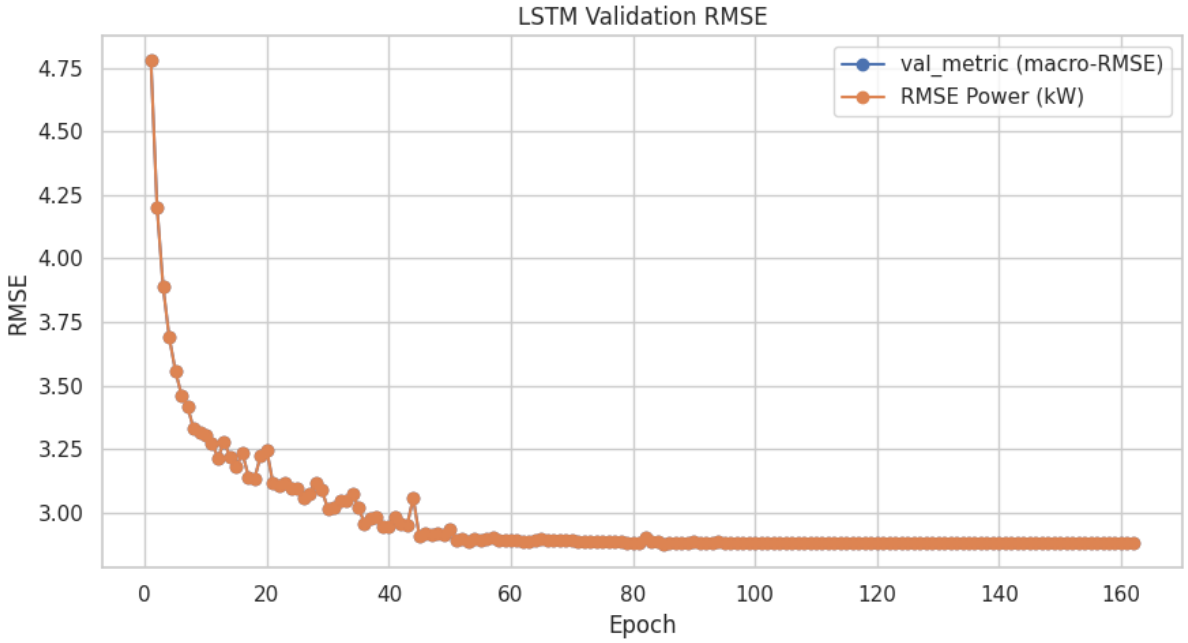

Figure 5.2: Best LSTM model's partial validation RMSE per epoch

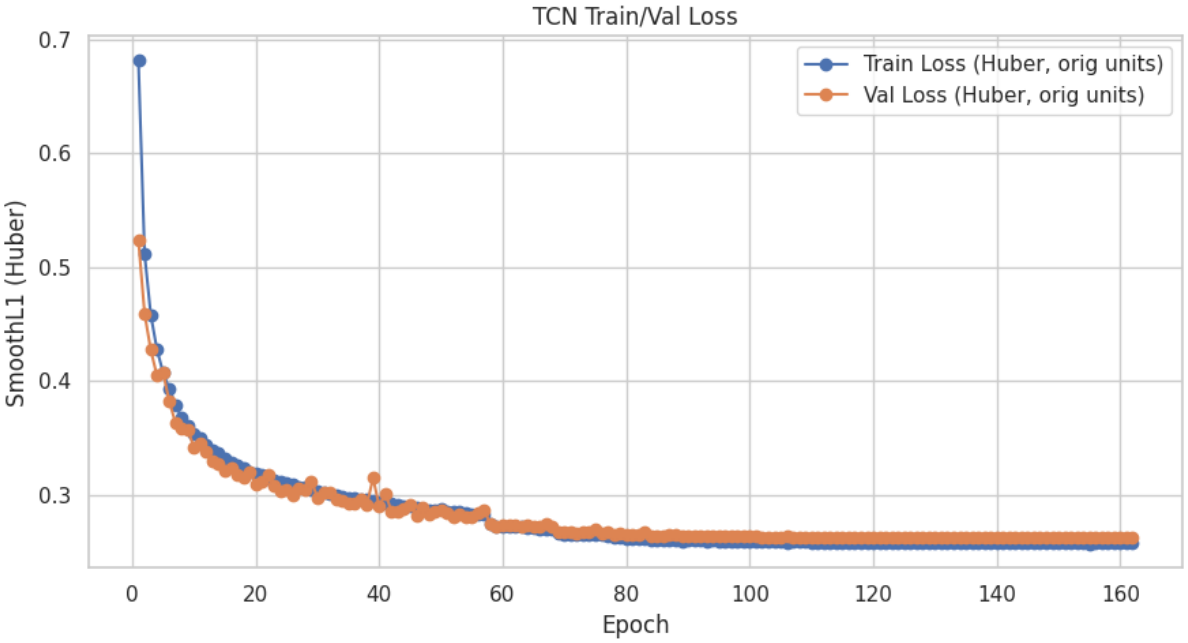

Figure 5.3: Best TCN model's training and validation loss

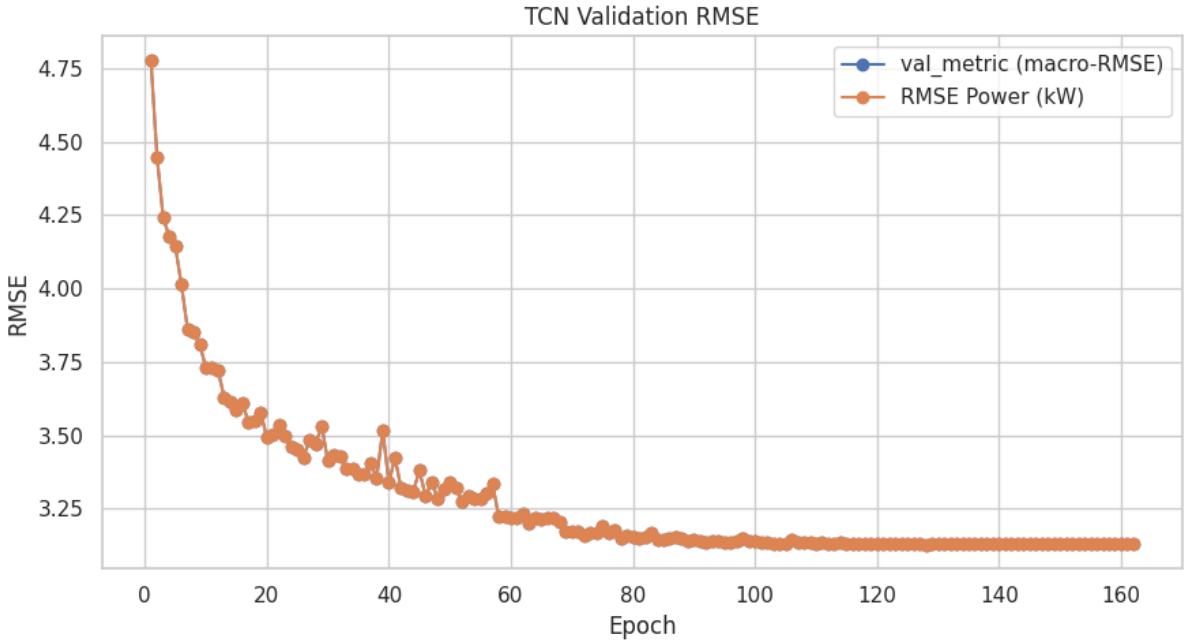

Figure 5.4: Best TCN model's partial validation RMSE per epoch

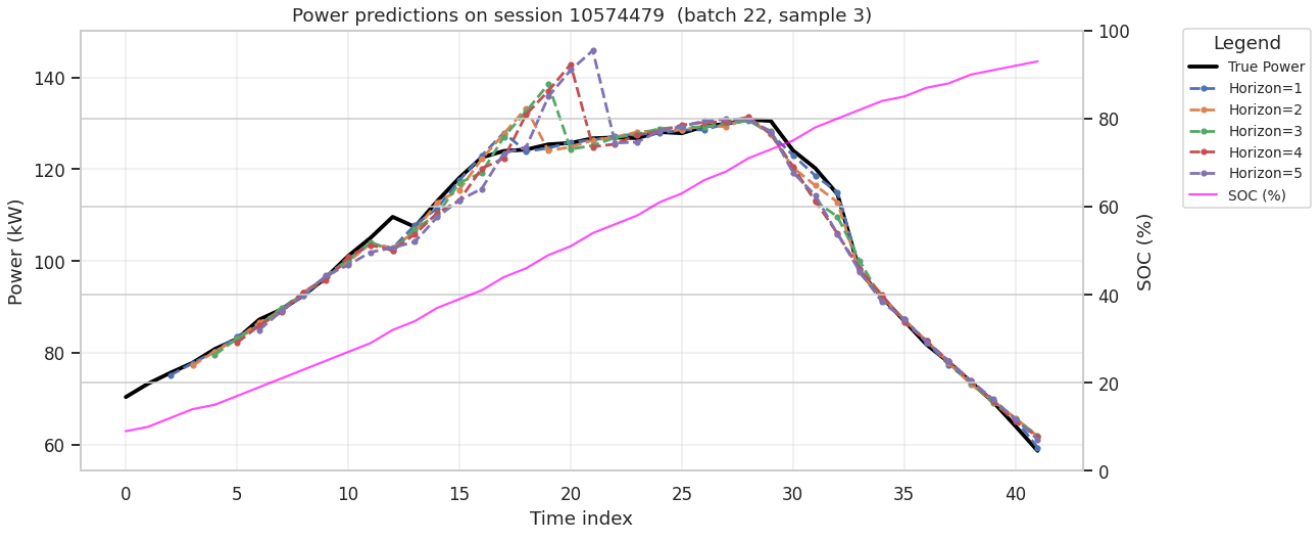

Figure 5.5: LSTM model forecasting on a single session

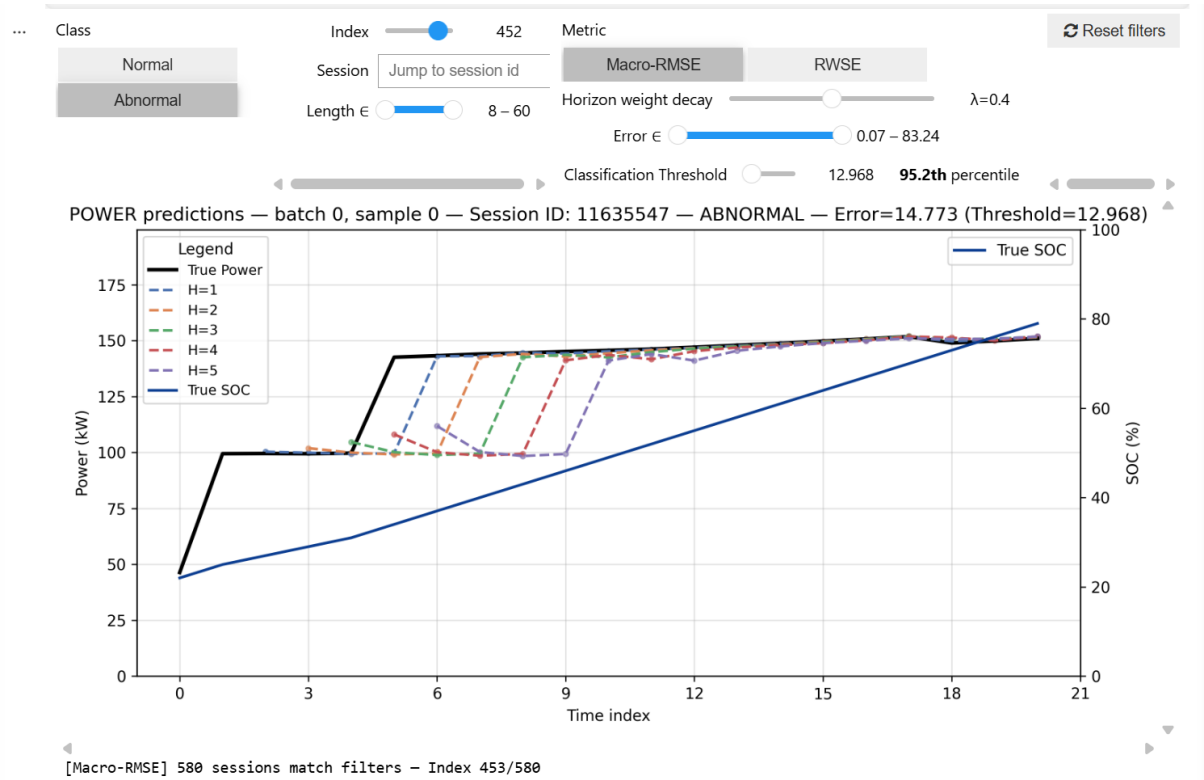

Figure 5.6: Interactive anomaly inspection tool

the model captured the dominant charging dynamics well enough for the downstream anomaly-scoring task. Importantly, the purpose of this subsystem was not to maximise forecasting performance in isolation, but to provide a stable and non-trivial deployed classifier whose behaviour could later be simplified, taught, and simulated.

5.2 Anomaly Detection Results and Design Choices

After the forecasting model had been fixed, I converted session-level forecasting error into a binary anomaly label. This stage involved several practical design choices: which error metric to use, how strongly to weight near-term versus longer-term forecast horizons, where to set the decision threshold, and how to inspect the consequences of each choice without ground-truth anomaly labels.

5.2.1 Compared anomaly detection design options

I compared two candidate session-level error metrics, Macro-RMSE and RWSE, and I explored horizon-decay settings

$$\lambda \in \{0.0, 0.2, 0.4, 0.6, 2.0\}.$$

I also examined high-percentile thresholds on the validation-set error distribution and used a lightweight self-built interactive anomaly-inspection tool (see Figure 5.6 and `04_Anomaly_Detection.ipynb` [85]) to browse individual charging sessions, reconstructed forecasts, and resulting session scores under various parameters and settings.

This inspection workflow mattered because the dataset did not contain trusted session-level anomaly labels. Instead of measuring precision, recall, or AUROC directly, I evaluated candidate settings qualitatively by asking whether the resulting anomalous sessions looked like meaningful departures from expected charging behaviour, whether the metric over-reacted to isolated local errors, and whether the resulting abnormal set was small enough to remain plausible as a high-confidence subset.

5.2.2 Final anomaly scoring configuration

The final operational configuration used Macro-RMSE as the session-level scoring metric, a 95th-percentile threshold estimated on validation errors, horizon-decay $\lambda = 0.4$, and `t_min_eval=1`. The corresponding Macro-RMSE threshold was $\tau = 12.97$.

When applied to the 12,183-session test set, this yielded 580 sessions labelled as abnormal and 11,603 sessions labelled as normal. I retained Macro-RMSE rather than RWSE because it aligned better with the later interpretability objective. In practice, Macro-RMSE produced a set of abnormal sessions whose deviations were easier to inspect visually and easier to reason about in the teaching pipeline. RWSE remained useful as a robustness-oriented diagnostic, but not as the main operational score.

5.2.3 Absence of anomaly detection scoring metrics

I did not collect domain-expert labels for the anomaly-detection subsystem, and I therefore did not compute standard supervised anomaly-detection metrics such as precision, recall, F1, or AUROC. There were two reasons for this.

Design aspect	Options explored	Final choice
Session-level score	Macro-RMSE, RWSE	Macro-RMSE
Horizon-decay parameter	$0.0, 0.2, 0.4, 0.6, 2.0$	0.4
Thresholding rule	high validation percentiles	95th percentile
Evaluation start index	low $t_{\text{min,eval}}$ settings	$t_{\text{min,eval}} = 1$

Table 5.2: Final anomaly scoring choices after qualitative calibration.

First, reliable labelling would have required substantial time from domain experts with knowledge of EV charging behaviour, battery limitations, and charger-site conditions. That annotation effort would have been significant in its own right.

Second, building the best possible anomaly detector is not the main goal of the thesis. The central research question is whether a machine-teaching explanation pipeline could be constructed on top of a non-trivial time-series classifier and whether the behaviour of that classifier could be taught to a proxy learner. For that purpose, the anomaly-detection pipeline needed to be realistic, coherent, reproducible, and behaviourally inspectable. It did not need to be fully optimised as a production anomaly-detection system. I therefore treat the abnormal labels used in later chapters as operational pseudo-labels induced by the forecasting-and-thresholding rule.

5.3 Curve Simplification Results and Design Choices

The simplification stage shifts the emphasis from predictive modelling to classifier-aware optimisation and visual representation design. Here I had to decide which signals should be shown to the learner, which curve-simplification method should be used for the decision-relevant power channel, and how to balance fidelity against computational cost.

5.3.1 Presentation-level design choices

The teaching interface did not expose the full multivariate feature space used internally by the forecasting model. Instead, I chose to show power and SOC trajectories only. Temperature, nominal charger power, taper flags, and derivative features remained internal modelling features and were omitted from the learner-facing plots.

This was a deliberate simplification choice. Power carried the most visible and decision-relevant signal in the anomaly plots I inspected, which was expected from the

feature importance analysis in the EDA (see subsection 4.1.3). However, while changes in SOC do not provide an important forecasting signal, the SOC trajectory gives easily interpretable contextual information about charging phase and tapering behaviour. Including additional model inputs such as ambient temperature in the teaching visualisations would have increased clutter and cognitive demand without obviously improving the learner's ability to approximate the classifier. I therefore kept the visual interface deliberately narrow, with power as the primary explanatory signal and SOC as supporting context.

5.3.2 Comparing stage-1 candidate-generation methods

I compared three practical options for stage-1 candidate generation:

- Ramer–Douglas–Peucker (RDP) simplifications,
- ORS with dynamic programming (DP),
- ORS with prefix-sum-accelerated dynamic programming (DP-prefix).

The comparison focused on two criteria: wall-clock efficiency and the compactness of feasible simplifications, measured operationally through the number of line segments k. In exploratory runs, the DP-based variants tended to find feasible simplifications with lower k than plain RDP, which was desirable for teaching because simpler decision-preserving curves are easier to inspect and compare. However, vanilla DP was noticeably slower and became expensive when simplification had to be run repeatedly across a large teaching pool.

My DP-prefix variant of ORS preserves the DP search objective while reducing the computational burden of stage 1. In practice, this gave the best trade-off for the power channel: it retained the stronger search behaviour associated with DP, but was much more practical for large-scale pool construction. I therefore selected ORS with DP-prefix as the default power-simplification method.

For the SOC channel, the situation was different. SOC served mainly as contextual support in the plots, and exact preservation of the classifier's decision through the SOC simplification itself was not the central requirement. Furthermore, the highly linear nature of SOC suggested that those simplifications would be much more trivial than for power. I therefore used RDP directly for SOC overlays to save time and computational resources.

5.3.3 Comparing additional simplification parameters

In order to investigate how the remaining ORS simplification parameters should be set for the MT4XAI pipeline, I built an interactive curve simplification inspection tool largely by reusing code assets from the interactive anomaly detection tool presented in subsection 5.2.1. The main purpose of the tool was to support qualitative and semi-structured comparison of parameter settings across individual charging sessions, while keeping the underlying anomaly classification logic fixed. To ensure this consistency, the dashboard uses the same Macro-RMSE-based session scoring, horizon-decay setting, and anomaly thresholding procedure as the anomaly detection analysis.

The tool allowed me to filter sessions by original classifier label and sequence length, inspect specific charging-session IDs, and recompute ORS simplifications under different parameter combinations. In particular, it exposed the main ORS trade-offs that were relevant for downstream teaching-set construction: the stage-1 candidate generation method (`DP`, `DP_prefix`, or `RDP`), the stage-2 candidate selection metric (`L2` or Macro-RMSE), the robustness and fragmentation weights α, β, and γ, the admissible deviation parameter ε, the allowed simplification range k, and the search-budget parameters q, R, and `n_candidates`. This made it possible to compare not only whether a simplification remained label-preserving, but also how the chosen parameters affected visual faithfulness, abrupt artefacts, fragmentation, and the achieved simplicity level.

Figure 5.7 shows the interactive dashboard for exploring different tunable parameters and algorithmic implementations for simplifying charging sessions. The plot overlays the original charging curve, the model prediction horizons, and the resulting simplified curve, while also reporting the original and simplified Macro-RMSE values, the decision threshold, the resulting label, the ORS objective value, and the number of retained segments. This combination of controls and visual feedback was useful for identifying parameter settings that produced simpler and visually plausible charging curves without changing the classifier decision, which was the main requirement for the explanation pipeline.

5.3.4 Final operational configuration

For large-scale teaching pool and exam pool construction, the final power-simplification configuration used `stage1_mode=dp_prefix`, `dp_alpha=0.0075`, `dp_q=300`, `beta=4.0`,

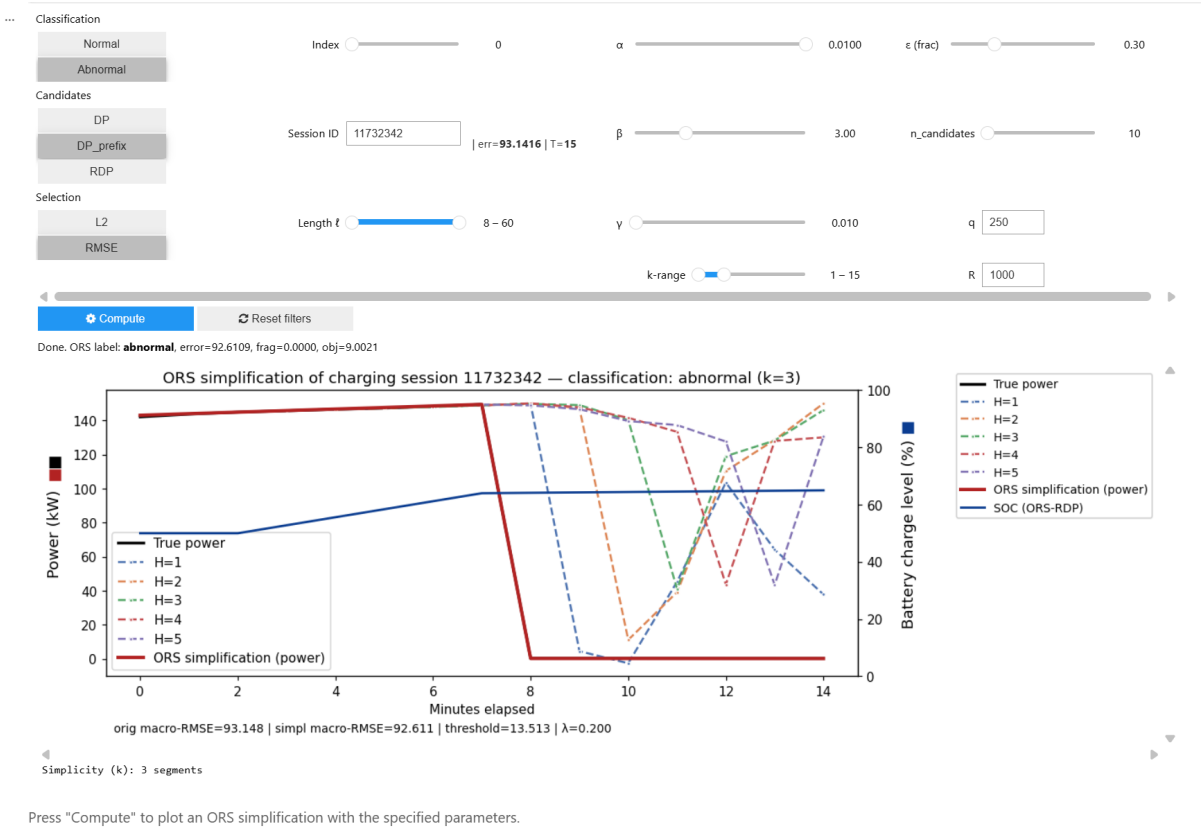

Figure 5.7: Interactive simplification tool

`gamma=0.05`, `R=3500`, `epsilon_mode=fraction`, `epsilon_value=0.2`, `t_min_eval=2`, `anchor_endpoints=last`, and `min_k=1`, `max_k=12`. For SOC, I used `soc_stage1_mode=rdp` with `soc_rdp_epsilon=0.75`.

These choices reflected the practical role of simplification in the thesis. The power simplification had to remain classifier-faithful and computationally manageable across many sessions. The SOC simplification had to remain visually useful, but it did not need to carry the full burden of decision preservation.

5.3.5 Asymptotic improvement of the prefix-sum DP variant

The key theoretical reason for preferring the DP-prefix variant was that, in a fully prefix-sum-based implementation of ORS stage 1, it reduces error-table initialisation from cubic to quadratic in the sequence length T, while leaving the original heap-based q-best DP and stage 2 robustness screening unchanged.

Let a candidate segment over indices $a, \dots, b$ be approximated by a line

$$\ell_{a,b}(t) = u_{a,b}t + v_{a,b}.$$

The segment reconstruction error is then

$$\mathrm{SSE}(a,b) = \sum_{t=a}^{b} \big(y_t - \ell_{a,b}(t)\big)^2.$$

Expanding the square gives

$$\mathrm{SSE}(a,b) = \sum_{t=a}^{b} y_t^2 - 2u_{a,b}\sum_{t=a}^{b} ty_t - 2v_{a,b}\sum_{t=a}^{b} y_t + u_{a,b}^2\sum_{t=a}^{b} t^2 + 2u_{a,b}v_{a,b}\sum_{t=a}^{b} t + v_{a,b}^2\sum_{t=a}^{b} 1.$$

Each term depends only on interval sums of the form

$$\sum 1, \qquad \sum t, \qquad \sum t^2, \qquad \sum y_t, \qquad \sum ty_t, \qquad \sum y_t^2.$$

If prefix sums for these six quantities are precomputed once, every interval query $[a,b]$ can be answered in $O(1)$ time. It follows that the SSE for any candidate segment can also be computed in $O(1)$ time. The expansion above implies that, once the line through a candidate pair of pivots is fixed, each ORS error term can be evaluated from six cumulative statistics. This applies not only to interior segments $\mathrm{err}(i,j)$, but also to the

endpoint-extended terms $\mathrm{err}(1, i, j)$, $\mathrm{err}(i, j, T)$, and $\mathrm{err}(1, i, j, T)$, since these only change the queried interval.

Using cumulative sum tables to answer interval aggregate queries in constant time is a standard optimisation, classically exemplified by summed-area tables [20]. Here, the same idea is specialised to the one-dimensional sufficient statistics needed for ORS segment SSE computations.

In the original ORS paper, the $O(T^2)$ error terms are initialised in $O(T)$ time each, yielding an $O(T^3)$ preprocessing cost and hence an $O(T^3 + Tq \log T)$ stage-1 runtime [95]. If all four error tables are instead backed by prefix-sum queries, the preprocessing cost falls to $O(T^2)$, so stage 1 becomes

$$O(T^2 + Tq \log T).$$

Stage 2 is unaffected and still requires classifier evaluation of the retained candidates and their perturbations, which can be written explicitly as

$$O(q(R + 1)C_f(T))$$

up to lower-order constants. The DP-prefix variant therefore leaves the ORS objective, candidate semantics, and optimality logic unchanged, but accelerates the deterministic candidate-generation step relative to the original cubic implementation.

At present, the repository implementation applies constant-time prefix queries only to the interior error table $\mathrm{err}(i, j)$, while the endpoint-extended tables are still accumulated directly. The current code therefore realises a constant-factor speed-up, but not yet the full $O(T^2 + Tq \log T)$ stage-1 bound.

5.4 MT4XAI MLLM Experiment Results

This section reports the empirical results of the MLLM evaluation introduced in section 4.5. The experimental protocol, condition design, and validity scope are therefore not repeated here, since those aspects are already discussed in section 4.5 and subsection 4.6.4. The focus of the present section is on the observed outcome patterns in the proxy-learner experiment, while their broader interpretation is deferred to chapter 6.

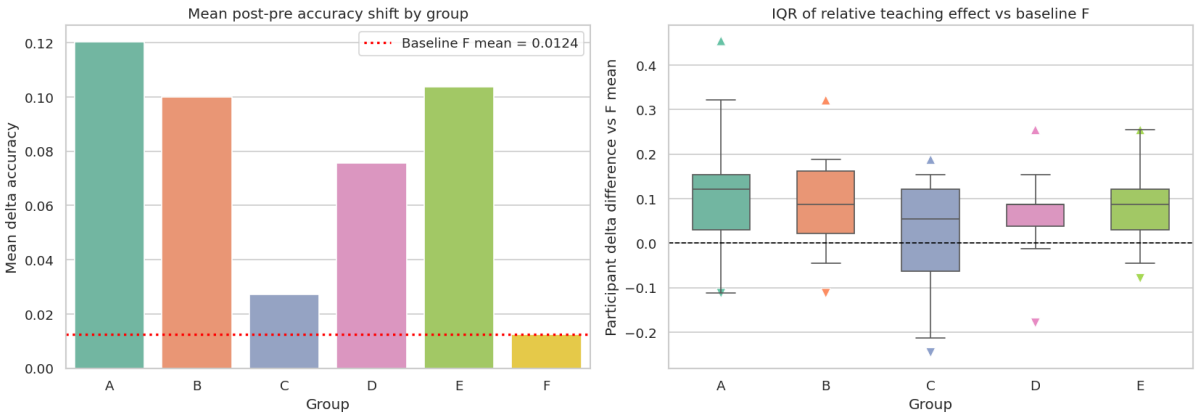

Figure 5.8: Teaching effect per group and condition for `gpt-5-nano` MLLM participants. The left panel shows the mean post-pre accuracy shift for each group, with the dotted red line marking the baseline mean shift for Group F. The right panel shows relative teaching effect versus Group F. The box plot is configured so that the median is the central horizontal line, while the top and bottom of the box are the 75th and 25th percentiles, respectively. The whiskers show the 95th and 5th percentiles, while the dots show the minimum and maximum observations.

5.4.1 Experimental sample and condition balance

The experiment conducted with `gpt-5-nano` MLLM participants yielded 101 completed participants after excluding incomplete and failed trials. The distribution of completed participants remained close to balanced across conditions, with 18 participants in Groups A, E, and F, 16 in Groups B and C, and 15 in Group D.

5.4.2 Empirical Results

Figure 5.8 summarises both the mean post-pre accuracy shifts and the relative teaching effects against the no-teaching baseline.

Several patterns are immediately visible in Figure 5.8. First, the no-teaching baseline in Group F shows only a very small mean improvement, which suggests that most of the larger shifts in the teaching groups are unlikely to be explained by exam-set differences alone. Second, the pattern suggests that simply being exposed to more exam questions or examples is unlikely to generate a large positive teaching effect on its own. Third, the strongest mean improvement is observed in Group A, followed closely by Groups E and B. Fourth, Group C produces only a small net gain, which is why the post-pre change remains the more informative summary than post-exam accuracy alone.

The right panel of Figure 5.8 makes the same pattern clearer after baseline adjustment. The ranking by relative teaching effect is $A > E > B > D > C$. Groups A, E, and B cluster as the strongest conditions, Group D remains positive but weaker, and Group C is close to the baseline with an IQR that crosses zero.

Group	**Welch p**	**Bonf. p**	**Bonf. rej.**	**Holm p**	**Holm rej.**
A	0.0093	0.0466	True	0.0373	True
E	0.0028	0.0139	True	0.0139	True
B	0.0103	0.0514	False	0.0373	True
D	0.0377	0.1886	False	0.0754	False
C	0.6918	1.0000	False	0.6918	False

Table 5.3: Significance results for the relative teaching effect comparisons against baseline Group F in the main `gpt-5-nano` experiment.

The significance pattern in Table 5.3 broadly supports the same descriptive ranking. In the uncorrected Welch comparisons against Group F, Groups A, B, D, and E all outperform the baseline, whereas Group C does not. After Holm correction for multiple comparisons, Groups A, B, and E remain significant. Under the more conservative Bonferroni correction, only Groups A and E remain below the 0.05 threshold, while Group B narrowly misses it ($p = 0.0514$). The strongest evidence therefore concentrates around the overlay curriculum condition (A) and the simplified-only condition with frequent rule updates (E), with Group B close behind.

Chapter 6

Discussion

This chapter interprets the results reported in chapter 5, relates them back to the research question and the broader MT4XAI agenda, and discusses the main limitations and contributions of the thesis.

6.1 Interpretation of Key Findings

6.1.1 From methodology to a working MT4XAI pipeline

A central contribution of this thesis is not only that MT4XAI components were combined end-to-end, but that several of them required adaptation before they could operate coherently on industrial charging-session data. The MT4XAI framework can be instantiated as a coherent end-to-end pipeline for a real industrial time series use case. In practice, this meant building not only an explanation method, but an entire chain of components: a forecasting-based anomaly detector, a classifier-aware simplification procedure, a teaching-set construction pipeline, and a forward-simulation evaluation protocol. The main value of this result is therefore not only in any single metric, but in showing that MT4XAI can move beyond controlled benchmark settings and operate on noisy, multivariate EV charging-session data.

This is important because the industrial setting introduces constraints that are easy to abstract away in theory. The anomaly labels used later in the pipeline are not expert-validated fault labels, but operational pseudo-labels induced by forecasting error and

thresholding. The charging sessions are long enough, multivariate enough, and variable enough that a purely raw-example explanation strategy would be difficult to interpret. The fact that the pipeline still produces stable teaching pools, simplified examples, and meaningful condition differences in the MLLM study suggests that the overall MT4XAI idea is viable even when the data is messy and the classifier is only indirectly defined through forecasting behaviour.

The software-engineering results also matter in their own right. The prefix-sum dynamic-programming variant of ORS does not change the underlying simplification objective, but it makes large-scale simplification substantially more practical by reducing the stage-1 candidate-generation cost from cubic to quadratic in sequence length for fixed K. That does not remove the cost of stage-2 robustness checking, but it does remove one of the main bottlenecks in building large teaching and exam pools. In that sense, the contribution is not just theoretical neatness. It is one of the reasons why the broader MT4XAI pipeline becomes executable at all on a realistically sized dataset.

6.1.2 What the MLLM experiment supports

The strongest empirical pattern from the MLLM experiment is that the teaching conditions are not equally effective. Groups A, E, and B produce the largest post-minus-pre gains, Group D is positive but weaker, and Group C remains close to the no-teaching baseline. This pattern matters because Group C still receives labelled examples. The weak performance of Group C therefore suggests that simply showing more examples is not enough. How the examples are represented seems to matter at least as much as how many are shown.

Although these results are still preliminary, they already suggest a meaningful structure in the teaching conditions. The two overlay-based groups perform strongly in both ordered and unordered form, and Group A is modestly better than Group B. This points to a possible benefit of curriculum order, although the gap between A and B is not large enough to support a strong claim on its own. The significance results in Table 5.3 strengthen this interpretation to a degree, because both A and B survive Holm correction, while only A also survives the stricter Bonferroni correction.

The simplified-only conditions are also encouraging. Group E produces a relative effect that is close to Group A and larger than Group D, which suggests that simplification does not destroy the teaching signal and may work especially well when the learner is

required to maintain an explicit rule-of-thumb throughout teaching. This interpretation is reinforced by Table 5.3, where Group E is significant under both Holm and Bonferroni correction. Group D is still above the baseline, but the evidence for that condition is less secure after multiplicity correction, since it does not remain significant once the family-wise corrections are applied.

By contrast, the raw-only teaching condition C contributes very little beyond the no-teaching baseline once improvement is measured relative to the participant's own pre-exam performance. This is an important result, because it suggests that merely showing additional labelled examples is not sufficient. The way the examples are presented appears to matter. In the present results, both overlay-based and simplified-only representations appear more effective than raw-only presentation for helping the proxy learner infer the classifier's decision behaviour.

This is one of the clearest takeaways of the thesis. The stronger conditions all reduce cognitive burden in some way. Groups A and B use overlays that directly compare raw and simplified behaviour, while Groups D and E use simplified-only views that remove much of the visual clutter. Group E adds explicit rule maintenance, which appears to help the proxy learner retain and refine a usable decision heuristic throughout the teaching phase. Taken together, the results suggest that MT4XAI-style explanation benefits do not come from example exposure alone. They seem to come from combining faithful simplification, structured presentation, and a teaching protocol that encourages the learner to consolidate a rule.

Taken together, the preliminary `gpt-5-nano` results provide the first empirical indication that the MT4XAI-generated teaching sets contain a usable teaching signal for an MLLM proxy learner. The gains are not uniform across all conditions, but the strongest conditions improve substantially more than the no-teaching baseline and do so in a pattern that is consistent with the intended hypotheses behind the explanation design.

The hypothesis-level interpretation should, however, remain cautious. H1 receives directional support, but not enough evidence to be accepted conclusively. Group B improves more than Group C on average (0.100 versus 0.027 post-minus-pre accuracy change), which is consistent with a benefit from simplification, but the direct B-versus-C contrast is still statistically uncertain on the current sample ($p \approx 0.10$). H2 is weaker still. Group A outperforms Group B numerically (0.120 versus 0.100), but the gap is small and does not provide convincing direct evidence for a curriculum effect on its own ($p \approx 0.65$). The safest interpretation is therefore that curriculum ordering may help, but this thesis does not establish that strongly.

H3 is the most convincing of the three hypotheses. When the teaching groups are considered together, they outperform the no-teaching baseline clearly (0.087 versus 0.012 mean improvement, $p \approx 0.0007$). At the condition level, Groups A, E, and B also show the clearest gains beyond Group F. I therefore treat H3 as broadly supported, while also noting that the size of the teaching effect depends strongly on the teaching format. In other words, teaching helps, but some ways of teaching help much more than others.

6.1.3 Research question, MT4XAI, and the XAI picture

The overarching research question asks how MT4XAI techniques can be applied to time series classifiers to generate simple, understandable, and faithful explanations in a real-world industry setting. Based on the results of this thesis, I would answer that question positively, but only at proof-of-concept level. The thesis shows one workable way of doing this: by treating the deployed anomaly detector as the target behaviour to be taught, simplifying charging curves in a classifier-aware way, selecting a compact and diverse teaching set, and evaluating whether a proxy learner becomes better at simulating the classifier after teaching. What the thesis does *not* show is that this is already a finished or fully validated industrial explanation solution.

Within the broader MT4XAI research project, this thesis contributes empirical grounding in three places. First, it shows that simplification is not merely a visual convenience, but a practical ingredient in explanation design for time series. Second, it operationalises the teaching-set idea in a realistic pipeline that includes balancing, stratification, diversity, robustness, margin, and curriculum. Third, it pilots MLLM-based forward simulation as a scalable pre-human evaluation method. That last point is especially relevant for MT4XAI, because human studies are expensive and slow, while explanation design often requires many rounds of refinement.

More broadly for XAI, the thesis supports a shift away from evaluating explanations only in terms of local faithfulness or visual plausibility. The results suggest that explanation quality should also be judged by whether it helps a learner simulate the model's behaviour. This is closely aligned with the MT view of explanation as teaching. The negative or weak result for raw-only Group C is especially informative here. It suggests that explanation research for time series should care not only about preserving model-relevant information, but also about presentation, cognitive load, and how information is sequenced over time.

6.2 Limitations

6.2.1 Classifier and anomaly-labelling limitations

The anomaly-detection subsystem is a necessary foundation of the thesis, but it is also a major source of uncertainty. The labels explained later in the pipeline are pseudo-labels induced by forecasting error and thresholding, not externally validated fault labels. As a result, the thesis explains the behaviour of the implemented classifier, not ground truth in any strict operational sense. This is acceptable for a proof-of-concept MT4XAI study, but it limits any strong claim about the practical fault-detection quality of the underlying anomaly detector.

A related limitation is that anomaly-scoring choices were calibrated qualitatively rather than through standard supervised metrics such as precision, recall, F1, or AU-ROC. This was a reasonable decision given the lack of trusted labels and the scope of the project, but it means that the chosen configuration should be understood as an operationally plausible classifier for explanation research, not as a production-optimised anomaly detector.

6.2.2 Simplification and teaching-pipeline limitations

The simplification pipeline preserves classifier behaviour by design, but that does not guarantee that it preserves all semantically important aspects of a charging session for a human viewer. In particular, the decision to focus visually on power and use SOC mainly as context is an informed engineering judgement rather than a formally proven property of the classifier. It seems reasonable from the qualitative inspection work, but it remains a simplification of a richer multivariate decision process.

There are also practical limitations in the ORS pipeline itself. Some sessions are harder to simplify robustly than others, and the current implementation is still computationally expensive despite the prefix-sum speed-up. In the exam-pool workflow, at least one session exhausted the ORS fallbacks and had to be skipped. That does not invalidate the overall approach, but it is a reminder that the current system is research software rather than a universally reliable simplification engine.

Finally, the teaching-set construction pipeline uses one embedding design, one coverage objective, and one curriculum heuristic. These choices are well motivated and internally coherent, but they should not be treated as uniquely optimal for human learning. They are plausible operational approximations of MT principles, not final answers.

6.2.3 Limitations of the MLLM experiment

The MLLM study is best viewed as a pre-human evaluation rather than as a substitute for human-participant validation. The strongest claim it supports is that the explanation sets contain enough teaching signal for a fallible proxy learner to improve under some conditions. It does not show that domain experts, customers, or other real users would learn in the same way.

The experiment is also sensitive to prompt design, response schemas, context management, and model-specific behaviour. Group E is a good example of this. Its strong performance is interesting, but Group E differs from Group D not only in simplification modality but also in how aggressively it forces rule maintenance. This makes it a valuable experimental condition, but also a less clean comparison than a single-factor design. More generally, the current sample is strong enough to support the claim that teaching can help, but weaker when it comes to separating closely related teaching policies such as simplification versus curriculum in a definitive way.

6.3 Contributions

This thesis makes four main contributions. First, it instantiates the MT4XAI agenda on a real industrial EV charging use case and shows that the full pipeline can be executed end to end on multivariate time series. Second, it extends ORS with a prefix-sum-accelerated dynamic-programming variant that preserves the original search objective while making large-scale candidate generation much more practical. Third, it operationalises MT4XAI for time series through a reusable pipeline for building faithful simplification pools, selecting compact teaching sets, and serving examples under controlled modality and curriculum conditions. Fourth, it pilots an MLLM-based forward-simulation evaluation that appears useful as a low-cost screening step before human studies.

Taken together, these contributions do not amount to a finished industrial XAI product. What they do provide is a credible proof of concept that MT4XAI can be adapted

to a difficult real-world time-series setting, and that its core ideas, simplification, structured example selection, and learner-centred evaluation, remain meaningful outside small benchmark demonstrations.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

This thesis investigates whether techniques from Machine Teaching for Explainable AI (MT4XAI) can be applied to time-series classifiers in order to generate explanations that are simple, understandable, and faithful in a real-world industry setting. Based on the work presented in the previous chapters, the overall answer to this research question is cautiously positive.

The thesis shows that it is possible to build an end-to-end MT4XAI pipeline for a realistic time-series anomaly-detection problem based on EV charging sessions. The resulting system combines four main components: a forecasting-based anomaly detector, classifier-aware curve simplification, teaching-set construction, and forward-simulation evaluation. Taken together, these components form a coherent explanation pipeline rather than a collection of isolated experiments. This is an important result in itself, because much of the earlier MT4XAI and time-series XAI literature remains closer to controlled benchmarks, narrower methodological contributions, or purely theoretical proposals. In contrast, this thesis demonstrates how the ideas can be connected and implemented in a practically motivated setting with noisy multivariate data, engineering constraints, and a non-trivial deployed classifier.

The results also indicate that simplification is a meaningful part of explanation design for time-series classifiers. In the MLLM experiment, the weakest teaching condition was the raw-only condition, even though it still exposed the learner to labelled examples.

By comparison, the stronger conditions used either overlaid or simplified representations. This suggests that the value of MT4XAI in this setting does not come only from choosing representative examples, but also from presenting them in a way that reduces visual complexity while preserving the decision-relevant structure.

At the software and algorithmic level, the thesis further shows that ORS can be adapted in ways that make it more practical for large-scale use in a teaching pipeline. The prefix-sum dynamic-programming variant introduced here does not change the ORS objective itself, but it reduces the computational burden of stage-1 candidate generation and therefore makes large-scale teaching-pool and exam-pool construction much more feasible. In practice, this matters because MT4XAI for time series depends on being able to generate many faithful simplifications efficiently enough to support repeated experimentation.

The MLLM study provides a preliminary but useful signal about the explanatory quality of the produced teaching sets. The strongest support is for the broader claim that structured teaching improves simulateability relative to a no-teaching baseline. The evidence for more specific claims is more tentative. The simplified overlay condition performs better on average than the raw-only condition, which is consistent with the hypothesis that simplification helps, but the current experiment does not establish that difference strongly enough to treat it as conclusive. Likewise, the curriculum-ordered overlay condition performs somewhat better than the unordered overlay condition, but the gap is small and does not support a strong claim on its own. The clearest conclusion is therefore that teaching helps, but that the effect depends strongly on how the teaching is designed.

More broadly, the thesis supports a learner-centred view of explainability in which explanation quality is judged not only by local faithfulness or visual plausibility, but also by whether an explainee can learn to simulate the system better after being taught. In that sense, the thesis does not only apply an existing framework to a new domain. It also reinforces one of the central ideas of the MT4XAI perspective.

At the same time, these conclusions should be kept within scope. The anomaly labels explained throughout the pipeline are operational pseudo-labels induced by the forecasting-and-thresholding rule, not expert-validated fault labels. Similarly, the MLLM experiment is best understood as a scalable pre-human evaluation step rather than as a replacement for human-participant studies. For these reasons, the thesis should be read as a proof of concept for MT4XAI in industrial time-series explainability, not as a finished industrial solution.

Taken together, the thesis makes four main contributions. First, it instantiates MT4XAI in a realistic EV charging-session anomaly-detection setting. Second, it improves the practical usability of ORS through a prefix-sum dynamic-programming variant. Third, it operationalises a complete pipeline for simplification-aware teaching-set construction and controlled teaching conditions. Fourth, it pilots an MLLM-based forward-simulation experiment that appears useful for early-stage evaluation of example-based explanations. These contributions do not fully solve the problem of explainable time-series anomaly detection, but they do show that the MT4XAI approach is both implementable and promising in a difficult applied setting.

7.2 Future Work

The most important next step is human evaluation. Although the MLLM experiment provides useful early evidence, the ultimate question for MT4XAI is whether real users can learn the classifier's behaviour more effectively from the generated teaching sets. A forward-simulation user study with humans would therefore be the most natural continuation of this work. Such a study could compare domain experts and non-experts, examine whether the same condition ranking appears as in the MLLM experiment, and investigate not only post-teaching accuracy but also confidence, response time, and subjective understanding. This would provide a much stronger basis for claims about interpretability in practice.

A second important direction is stronger validation of the anomaly-detection subsystem itself. In the present thesis, the anomaly detector functions as the deployed AI system to be explained, and this is sufficient for a proof-of-concept MT4XAI study. However, future work should ideally complement this with expert-labelled anomalies or other forms of operational validation. That would make it possible to distinguish more clearly between explaining the behaviour of the implemented classifier and explaining genuinely meaningful faults or abnormal charging events. It would also strengthen the practical value of the resulting explanations for industrial stakeholders.

A third direction concerns the teaching process itself. The current teaching pipeline uses fixed teaching sets, fixed modalities, and relatively simple curriculum heuristics. This is a reasonable starting point, but machine teaching theory suggests richer possibilities. One promising direction is to model learning more explicitly through Bayesian updating, as in the work of Yang et al. [104]. This could make it possible to estimate how much

each teaching example changes the learner's belief about the classifier's decision rule, and thereby measure teaching benefit more directly rather than only through pre- and post-exam accuracy. It could also support adaptive teaching strategies in which the next example depends on what the learner currently seems to misunderstand.

Related to this, future work could investigate whether ORS can do more than merely simplify examples while preserving the classifier's label. One especially interesting question is whether ORS-style methods can help identify the exact parts of a sequence that matter most for a given classification. In other words, instead of using simplification only to reduce cognitive load, one might also use it to localise decision-relevant temporal regions or transitions. If this proves feasible, it could create a bridge between example-based MT4XAI explanations and more local explanatory signals, while still keeping the representation visually coherent.

There is also substantial room for improving the simplification and teaching-set construction pipeline. Even with the prefix-sum speed-up, ORS remains computationally expensive because robustness checking still requires repeated classifier evaluations. Future work could therefore focus on faster robustness estimation, better candidate pruning, or alternative approximations that preserve most of the benefit at lower cost. It would also be valuable to study more systematically how the current embedding design, facility-location objective, margin term, robustness term, and curriculum heuristic affect downstream learning. The present choices are reasonable and empirically useful, but they are not guaranteed to be optimal for human learners.

Another natural direction is broader generalisation. This thesis focuses on one EV charging-session dataset, one forecasting-based anomaly detector, and one industrial use case. Future work should therefore test whether the same MT4XAI pipeline transfers to other types of time-series classifiers, other anomaly-detection settings, and other domains such as healthcare, energy forecasting, manufacturing, or cyber-physical systems. Such transfer studies would help clarify which parts of the pipeline are domain-specific engineering choices and which parts are more general MT4XAI principles.

Finally, the MLLM evaluation itself can be extended. The current experiment uses one main model and one main trial design. Future studies could compare several MLLMs, vary the prompt protocol more systematically, and replicate the direct condition contrasts with larger samples. This would help separate robust explanation effects from model-specific prompt effects and would make the proxy-learner methodology more reliable as a screening tool before human studies.

In summary, the future of this line of work lies in moving from proof of concept to stronger validation, broader generalisation, and more adaptive teaching. The thesis shows that MT4XAI is a plausible and promising approach for explainable time-series classification in industry. The next challenge is to make that promise more rigorous, more human-grounded, and more broadly applicable.

Taken together, the thesis shows that MT4XAI can be operationalised beyond controlled benchmark settings and applied coherently to noisy industrial multivariate time-series data, while also highlighting the methodological work still required before such explanations can be evaluated with human end users at scale.

Bibliography

[1] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[2] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *NeurIPS*, 2018.

[3] Taha Aksu, Chenghao Liu, Amrita Saha, Sarah Tan, Caiming Xiong, and Doyen Sahoo. Xforecast: Evaluating natural language explanations for time series forecasting. *arXiv preprint arXiv:2410.14180*, 2024. doi: 10.48550/arXiv.2410.14180.

[4] Alejandro Barredo Arrieta et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

[5] Emre Ates, Burak Aksar, Vitus J. Leung, and Ayse K. Coskun. CoMTE: Counterfactual explanations for multivariate time series. In *2021 International Conference on Applied Artificial Intelligence (ICAPAI)*. IEEE, 2021.

[6] Ammar Athar et al. A time-series classification approach for detecting anomalies in network traffic. *Journal of Network and Computer Applications*, 133:35–45, 2019.

[7] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

[8] Edwin Baidoo. An analysis of accuracy using logistic regression and time series models. *Kennesaw State University Digital Commons*, 2016.

[9] David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6541–6549, 2017.

URL: https://openaccess.thecvf.com/content_cvpr_2017/papers/Bau_Network_Dissection_Quantifying_CVPR_2017_paper.pdf.

[10] Emma Beauxis-Aussalet, Michael Behrisch, Rita Borgo, Duen Horng Chau, Christopher Collins, David Ebert, Mennatallah El-Assady, Alex Endert, Daniel A. Keim, Jörn Kohlhammer, Daniela Oelke, Jaakko Peltonen, Maria Riveiro, Tobias Schreck, Hendrik Strobelt, and Jarke J. van Wijk. The role of interactive visualization in fostering trust in ai. *IEEE Computer Graphics and Applications*, 41(6): 7–11, 2021. doi: 10.1109/MCG.2021.3107875.

[11] Ane Blázquez-García, Alberto Conde, Ugo Mori, and Jose A Lozano. A review on outlier/anomaly detection in time series data. *ACM Computing Surveys (CSUR)*, 54(3):1–33, 2021.

[12] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

URL: https://crfm.stanford.edu/assets/report.pdf.

[13] Carles Bretó, Edward L. Ionides, and Aaron A. King. Panel data analysis via mechanistic models. *Journal of the American Statistical Association*, 115(531): 1178–1188, 2020. doi: 10.1080/01621459.2019.1604367.

[14] Bruce G. Buchanan and Edward H. Shortliffe. *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project*. Addison-Wesley, Reading, MA, 1984.

[15] Raghavendra Chalapathy and Sanjay Chawla. Deep learning for anomaly detection: A survey. 2019.

URL: https://arxiv.org/abs/1901.03407.

[16] Chi Yan Chan, Yi Ting Tan, et al. Chatgpt for explainable machine learning: An empirical study. In *Proceedings of the 32nd International Joint Conference on Artificial Intelligence (IJCAI)*, 2023.

URL: https://arxiv.org/abs/2304.03442.

[17] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the*

2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1724–1734, 2014. doi: 10.3115/v1/D14-1179.

[18] Arthur Conmy, Ameya Ramkumar, Neel Nanda, Ben Marks, Nelson Elhage, Jonathan Mu, Roger Grosse, Jacob Steinhardt, and Chris Olah. Towards automated circuit discovery for mechanistic interpretability. *arXiv preprint arXiv:2304.14997*, 2023.
URL: `https://arxiv.org/abs/2304.14997`.

[19] DA Cook, S Oh, and MV Pusic. Accuracy of physicians' electrocardiogram interpretations: A systematic review and meta-analysis. *JAMA Intern Med*, 2020. doi: 10.1001/jamainternmed.2020.3989.
URL: `https://jamanetwork.com/journals/jamainternalmedicine/fullarticle/2771093`.

[20] Franklin C. Crow. Summed-area tables for texture mapping. *ACM SIGGRAPH Computer Graphics*, 18(3):207–212, 1984. doi: 10.1145/964965.808600.

[21] Angus Deaton. Panel data from time series of cross-sections. *Journal of Econometrics*, 30(1-2):109–126, 1985. doi: 10.1016/0304-4076(85)90134-4.
URL: `https://www.sciencedirect.com/science/article/abs/pii/0304407685901344`.

[22] Eoin Delaney, Derek Greene, and Mark T. Keane. Instance-based counterfactual explanations for time series classification. *arXiv preprint arXiv:2009.13211*, 2020.

[23] Eoin Delaney, Derek Greene, and Mark T. Keane. Instance-based counterfactual explanations for time series classification. In Antonio A. Sánchez-Ruiz and Michael W. Floyd, editors, *Case-Based Reasoning Research and Development*, Lecture Notes in Computer Science, pages 32–47. Springer, 2021. doi: 10.1007/978-3-030-86957-1_3.

[24] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.
URL: `https://arxiv.org/abs/1702.08608`.

[25] David H. Douglas and Thomas K. Peucker. Least-number of points for approximating a digitized curve. *Cartographica*, 10(2):112–122, 1973. doi: 10.3138/FM57-6770-U75U-7727.

[26] Emadeldeen Eldele, Mohamed Ragab, Min Chen, Min Wu, Chee-Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series anomaly detection: A deep self-supervised perspective. In *Proceedings of the 30th International Joint Conference on Artificial Intelligence (IJCAI 2021)*, pages 4746–4753, 2021. doi: 10.24963/ijcai.2021/647.

[27] Cèsar Ferri, Darío Garigliotti, Brigt Arve Toppe Håvardstun, José Hernández-Orallo, and Jan Arne Telle. When redundancy matters: Machine teaching of representations, 2024.
URL: https://arxiv.org/abs/2401.12711.

[28] Michele Fiori, Gabriele Civitarese, and Claudio Bettini. Using large language models to compare explainable models for smart home human activity recognition. In *Companion of the 2024 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2024. doi: 10.1145/3675094.3679000.

[29] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems (NEURIPS)*, volume 32, pages 4650–4661. Curran Associates, Inc., 2019.
URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/53c6de78244e9f528eb3e1cda69699bb-Paper.pdf.

[30] Ben D. Fulcher and Nick S. Jones. Highly Comparative Feature-Based Time-Series Classification . *IEEE Transactions on Knowledge & Data Engineering*, 26(12):3026–3037, December 2014. ISSN 1558-2191. doi: 10.1109/TKDE.2014.2316504.
URL: https://doi.ieeecomputersociety.org/10.1109/TKDE.2014.2316504.

[31] Ziyuan Gao, Christoph Ries, Hans Ulrich Simon, and Sandra Zilles. Preference-based teaching. *arXiv preprint arXiv:1702.02047*, 2017.

[32] Miguel García-Piqueras and José Hernández-Orallo. Heuristic search of optimal machine teaching curricula. *Machine Learning*, 112:5159–5191, 2023. doi: 10.1007/s10994-023-06508-3.

[33] Alan H. Gee, Diego Garcia-Olano, Joydeep Ghosh, and David Paydarfar. Explaining deep classification of time-series data with learned prototypes. *arXiv preprint arXiv:1904.08935*, 2019. Also appeared in CEUR Workshop Proceedings (IJCAI 2019, Vol. 2429, pp. 15–22).

[34] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *AAAI*, pages 3681–3688, 2019.

[35] Amirata Ghorbani, James Wexler, James Y. Zou, and Been Kim. Towards automatic concept-based explanations. *arXiv preprint arXiv:1902.03129*, 2019.
URL: https://arxiv.org/abs/1902.03129.

[36] Sally A. Goldman and Michael J. Kearns. On the complexity of teaching. *Journal of Computer and System Sciences*, 50(1):20–31, 1995. doi: 10.1006/jcss.1995.1003.

[37] Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, Saizhuo Wang, Kun Zhang, Yuanzhuo Wang, Wen Gao, Lionel Ni, and Jian Guo. A survey on LLM-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024. doi: 10.48550/arXiv.2411.15594.

[38] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM Computing Surveys*, 51(5), 2018. doi: 10.1145/3236009.

[39] David Gunning and David Aha. Darpa's explainable artificial intelligence (xai) program. *AI Magazine*, 40(2):44–58, Jun. 2019. doi: 10.1609/aimag.v40i2.2850.
URL: `https://ojs.aaai.org/aimagazine/index.php/aimagazine/article/view/2850`.

[40] Brigt Arve Toppe Håvardstun, Cèsar Ferri, José Hernández-Orallo, Pekka Parviainen, and Jan Arne Telle. Xai with machine teaching when humans are (not) informed about the irrelevant features. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2023), Research Track, Part III*, pages 378–393. Springer, 2023. doi: 10.1007/978-3-031-43418-1_23.

[41] José Hernández-Orallo and Jan Arne Telle. Finite biased teaching with infinite concept classes. *Journal of Artificial Intelligence Research*, Unpublished manuscript / preprint, 2024. Available as preprint.

[42] Hansika Hewamalage, Christoph Bergmeir, and Kasun Bandara. Global models for time series forecasting: A simulation study. *Pattern Recognition*, 124:108441, 2022. doi: 10.1016/j.patcog.2021.108441.

[43] Cheng Hsiao. Panel data analysis: advantages and challenges. *TEST*, 16(1):1–22, 2007. doi: 10.1007/s11749-007-0046-x.
URL: `https://link.springer.com/article/10.1007/s11749-007-0046-x`.

[44] Kyle Hundman, Valentin Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *KDD '18: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 387–395, 2018. ISBN 9781450355520. doi: 10.1145/3219819.3219845.

[45] Rob J. Hyndman and George Athanasopoulos. *Forecasting: Principles and Practice*. OTexts, Melbourne, Australia, 2 edition, 2018.
URL: https://otexts.com/fpp2/.

[46] Rob J. Hyndman, Roman A. Ahmed, George Athanasopoulos, and Hanlin L. Shang. Optimal combination forecasts for hierarchical time series. *Computational Statistics & Data Analysis*, 55(9):2579–2589, 2011. doi: 10.1016/j.csda.2011.03.006.

[47] Brigt Håvardstun, Cèsar Ferri, Kristian Flikka, and Jan Arne Telle. Xai for time series classification: Evaluating the benefits of model inspection for end-users. In *Proceedings of ECML PKDD*, 2024.

[48] Brigt Håvardstun, Cèsar Ferri, Carlos Monserrat, and Jan Arne Telle. An interactive tool for interpretability of time series classification. In *Proceedings of ECML PKDD*, 2024.

[49] Brigt Håvardstun, Felix Martí-Pérez, Cèsar Ferri, Carlos Monserrat, and Jan Arne Telle. Evaluating simplification algorithms for interpretability of time series classification. *arXiv preprint arXiv:2505.08846*, 2025. version 2.

[50] Hiroshi Imai and Masao Iri. Polygonal approximations of curves and surfaces. *Graphical Models and Image Processing*, 50(2):127–141, 1988. doi: 10.1016/0895-6111(88)90021-9.

[51] H Ismail Fawaz, B Lucas, G Forestier, C Pelletier, DF Schmidt, J Weber, GI Webb, L Idoumghar, P-A Muller, and F Petitjean. Benchmarking deep learning models on large time-series classification. In *Data Mining and Knowledge Discovery*, volume 34, pages 914–960. Springer, 2020.

[52] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33:917–963, 2019.

[53] Alon Jacovi, Matan Shlain, Roy Schwartz, and Yoav Goldberg. Towards transparent ai: Assessing, harnessing, and improving explainability of large language models. *arXiv preprint arXiv:2301.13212*, 2023.

[54] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. *Segmenting Time Series: A Survey and Novel Approach*, pages 1–21. doi: 10.1142/9789812565402_0001.
URL: https://www.worldscientific.com/doi/abs/10.1142/9789812565402_0001.

[55] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. In *Proc. of the 10th International Conference on Knowledge Discovery and Data Mining*, pages 122–131, 2001. doi: 10.1145/502512.502529.

[56] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2668–2677. PMLR, 2018.
URL: `https://proceedings.mlr.press/v80/kim18d.html`.

[57] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5338–5348. PMLR, 2020.
URL: `https://proceedings.mlr.press/v119/koh20a.html`.

[58] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS)*, pages 1097–1105, 2012. doi: 10.1145/3065386.

[59] Andrew Lampinen, Ishita Dasgupta, Stephanie C.Y. Chan, and et al. Can language models learn to explain? *Transactions on Machine Learning Research*, 2022.
URL: `https://openreview.net/forum?id=Jw1XQoSK6J`.

[60] Sebastian Lapuschkin, Stefan Wäldchen, Alexander Binder, Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Unmasking clever hans predictors and assessing what machines really learn. *Nature Communications*, 10:1096, 2019. doi: 10.1038/s41467-019-08987-4.

[61] M. C. Lee, J. Lee, and S. Lee. Research on the feasibility of applying gru and attention mechanism. *Applied Sciences*, 12(3):1007, 2022. doi: 10.3390/app12031007.

[62] Y. H. Lee. Nearest-neighbour-based approach to time-series classification. *Expert Systems with Applications*, 39(15):12138–12144, 2012. doi: 10.1016/j.eswa.2012.04.128.

[63] J Li et al. Time series analysis for vehicle sensor data: fault detection and predictive maintenance. *Mechanical Systems and Signal Processing*, 162:107995, 2022.

[64] Zhen Li, Chen Lei, Peng Zou, Ding Ding, Shuyue Hu, and Jie Gao. Attention with long-term interval-based gated recurrent units (ali-gru) for modeling sequential user behaviors. In *Proceedings of the International Conference on Knowledge Science, Engineering and Management (KSEM)*, pages 72–83, 2019. doi: 10.1007/978-3-030-29563-9_7.

[65] TW Liao. Clustering of time series data — a survey. In *Pattern Recognition*, volume 38, pages 1857–1874. Elsevier, 2005.

[66] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.

[67] Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 510–520. Association for Computational Linguistics, 2011.
URL: https://aclanthology.org/P11-1052/.

[68] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pages 2–11, 2003.

[69] Zachary C. Lipton. The mythos of model interpretability. *arXiv preprint arXiv:1606.03490*, 2016.
URL: https://arxiv.org/abs/1606.03490.

[70] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 4765–4774. Curran Associates, Inc., 2017.
URL: https://proceedings.neurips.cc/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html.

[71] Scott M. Lundberg, Harsha Nori, Samuel Jenkins, and et al. Programmatic interpretable explanations with large language models. arXiv preprint arXiv:2310.04877, 2023.
URL: https://arxiv.org/abs/2310.04877.

[72] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Lstm-based encoder-decoder for multi-sensor anomaly detection. 2016.
URL: https://arxiv.org/pdf/1607.00148.

[73] Felix Martí-Pérez, Brigt Håvardstun, Cèsar Ferri, Carlos Monserrat, and Jan Arne Telle. Evaluating simplification algorithms for interpretability of time series classification. *arXiv preprint arXiv:2505.08846*, 2025. version 1. Note: Section 7 "Interpretability of TCS by MLLMs" was removed in version 2 in favour of "End-user evaluation with forward simulation".

[74] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38, 2019.

[75] Navid Mohammadi Foumani, Lynn Miller, Chang Wei Tan, Geoffrey I. Webb, Germain Forestier, and Mahsa Salehi. Deep learning for time series classification and extrinsic regression: A current survey. *arXiv preprint arXiv:2302.02515*, 2023.

[76] Christoph Molnar. *Interpretable Machine Learning*. 2019. `https://chistoph.github.io/interpretable-ml-book\`.

[77] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14:265–294, 1978. doi: 10.1007/BF01588971.
URL: `https://doi.org/10.1007/BF01588971`.

[78] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3), 2020. doi: 10.23915/distill.00024.
URL: `https://distill.pub/2020/circuits/zoom-in`.

[79] Anna N. Rafferty, Emma Brunskill, Thomas L. Griffiths, and Patrick Shafto. Faster teaching by POMDP planning. *Cognitive Science*, 40(6):1290–1332, 2016. doi: 10.1111/cogs.12290.

[80] Urs Ramer. An iterative procedure for the polygonal approximation of plane curves. *Computer Graphics and Image Processing*, 1(3):244–256, 1972. doi: 10.1016/S0146-664X(72)80017-0.

[81] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 1135–1144, New York, NY, USA, 2016. ACM. doi: 10.1145/2939672.2939778.
URL: `https://dl.acm.org/doi/10.1145/2939672.2939778`.

[82] T. Rojat, R. Puget, D. Filliat, J. Del Ser, R. Gelin, and N. Díaz-Rodríguez. Explainable artificial intelligence (xai) on time series data: A survey. *arXiv preprint*, 2021.

[83] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *arXiv preprint arXiv:1811.10154*, 2018.
URL: `https://arxiv.org/abs/1811.10154`. version 3, 22.09.2019.

[84] Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021. doi: 10.1109/JPROC.2021.3052449.

[85] Sebastian E. S. Røkholt. Machine-teaching-for-xai–timeseries-models, 2026.
URL: `https://github.com/SebastianRokholt/Machine-Teaching-for-XAI--TimeSeries-Models`. GitHub repository. Accessed 16 March 2026.

[86] Udo Schlegel, Daniela Oelke, and Daniel A. Keim. Towards rigorous evaluation of XAI methods on time series. In *2019 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1437–1444. IEEE, 2019. doi: 10.1109/ICDMW.2019.00198.

[87] Edward H. Shortliffe. *Computer-Based Medical Consultations: MYCIN*. American Elsevier, New York, 1976.

[88] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, pages 3319–3328. PMLR, 2017.

[89] William R. Swartout. Xplain: A system for creating and explaining expert consulting programs. *Artificial Intelligence*, 21(3):285–325, 1983. doi: 10.1016/0004-3702(83)90003-9.

[90] Souhaib Ben Taieb and Rob J. Hyndman. Recursive and direct multi-step forecasting: the best of both worlds. Technical report, Monash University / University of Melbourne working paper, 2012.
URL: `https://robjhyndman.com/papers/rectify.pdf`. working paper; introduces the "rectify" strategy and compares recursive, direct and hybrid multi-step forecasting approaches.

[91] Souhaib Ben Taieb, Antti Sorjamaa, and Gianluca Bontempi. Multiple-output modelling for multi-step-ahead time series forecasting. *Neurocomputing*, 73(10–12): 1950–1957, 2010. doi: 10.1016/j.neucom.2009.11.030.

[92] Qi Tan and et al. Data-gru: Dual-attention time-aware gated recurrent units for unreliability-aware data streams. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 6063–6070, 2020. doi: 10.1609/aaai.v34i04.6063.

[93] Jan Arne Telle, José Hernández-Orallo, and Cèsar Ferri. The teaching size: computable teachers and learners for universal languages. *Machine Learning*, 108(8–9): 1653–1675, 2019. doi: 10.1007/s10994-019-05821-2.

[94] Jan Arne Telle, Cèsar Ferri, Jose Hernández-Orallo, and Pekka Parviainen. Machine teaching for explainable ai: Research project proposal for the research council, 2022. Unpublished project proposal.

[95] Jan Arne Telle, Cèsar Ferri, and Brigt Håvardstun. Optimal robust simplifications for explaining time series classifications. In *Proceedings of ECML PKDD*, 2024.

[96] Andreas Theissler, Francesco Spinnato, Udo Schlegel, and Riccardo Guidotti. Explainable ai for time series classification: A review, taxonomy and research directions. *IEEE Access*, 10:100700–100724, 2022. doi: 10.1109/ACCESS.2022.3207765.

[97] Avraam Tsantekidis, Nikolaos Passalis, Anastasios Tefas, Juho Kanniainen, Moncef Gabbouj, and Alexandros Iosifidis. Forecasting stock prices from the limit order book using convolutional neural networks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(4):300–312, 2018. doi: 10.1109/TETCI.2017.2787060.

[98] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11): 1134–1142, 1984.

[99] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1585, 2017. doi: 10.1109/IJCNN.2017.7966039.

[100] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine*

Learning Research, 2022. ISSN 2835-8856.
URL: https://openreview.net/forum?id=yzkSU5zdwD. Survey Certification.

[101] Kai Wei, Rishabh Iyer, and Jeff Bilmes. Submodularity in data subset selection and active learning. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1954–1963. PMLR, 2015.
URL: https://proceedings.mlr.press/v37/wei15.html.

[102] Qingsong Wen and et al. Transformers in time series: A survey. *IJCAI Workshop on Time Series*, 2022. doi: 10.24963/ijcai.2022/795.

[103] Scott Cheng-Hsin Yang, Wai Keen Vong, Ravi B. Sojitra, Tomas Folke, and Patrick Shafto. Mitigating belief projection in explainable artificial intelligence via bayesian teaching, 2021.

[104] Scott Cheng-Hsin Yang, Tomas Folke, and Patrick Shafto. A psychological theory of explainability. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 25123–25145. PMLR, 2022.
URL: https://proceedings.mlr.press/v162/yang22a.html.

[105] Lexiang Ye and Eamonn Keogh. Time series shapelets: A new primitive for data mining. In *Proc. of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 947–956, 2009. doi: 10.1145/1557019.1557122.

[106] Zahra Zamanzadeh Darban, Geoffrey I. Webb, Shirui Pan, Charu C. Aggarwal, and Mahsa Salehi. Deep learning for time series anomaly detection: A survey. *ACM Computing Surveys*, 57(1):1–42, 2024. doi: 10.1145/3691338.

[107] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833. Springer, 2014.

[108] Ailing Zhang, Jian Yang, Dongjin Zhang, and Jinqiang Sun. Self-supervised learning for time-series analysis: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(12):3655–3670, 2019. doi: 10.1109/TPAMI.2019.2904202.

[109] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023. doi: 10.48550/arXiv.2306.05685.

[110] Xiaojin Zhu. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. AAAI Blue Sky Ideas Track, 2015.
URL: `https://pages.cs.wisc.edu/~jerryzhu/machineteaching/`.

[111] Xiaojin Zhu, Adish Singla, Sandra Zilles, and Anna Rafferty. An overview of machine teaching. *arXiv preprint arXiv:1801.05927*, 2018.

Appendix A:

Generative AI Statement

The contents of this thesis, together with all related software assets, are the result of my own independent work. Generative AI tools were used selectively as supportive instruments during the project, primarily in the following ways:

- To assist with revision of the thesis text, including feedback on structure, flow, grammar, and sentence formulation.
- To recommend potentially relevant academic literature based on the evolving thesis draft and bibliography.
- To support debugging of Python and Latex code, drafting Python docstrings for functions and classes, minor refactors/cleanup limited to specific functionality, and, on a limited number of occasions, describing Jupyter Notebook outputs in Markdown.
- To answer general questions about programming (e.g. "I forgot to add a file to a commit I just pushed to remote, what are the Git commands to fix this again?")

Generative AI tools were not used to generate core scientific contributions, research findings, methodological decisions, experimental results, or final interpretations presented in this thesis.

Multimodal large language models also formed part of the empirical subject matter of the thesis itself, as a central experiment involved the use of `gpt-5-nano` as an experimental component.

Appendix B:

Examples from Teaching Sets

B.0.1 Teaching Set A

An arbitrary selection of examples from teaching set A follows. Each example is labeled with an index, that denotes its place in the curriculum of 60 examples (30 normal, 30 abnormal)

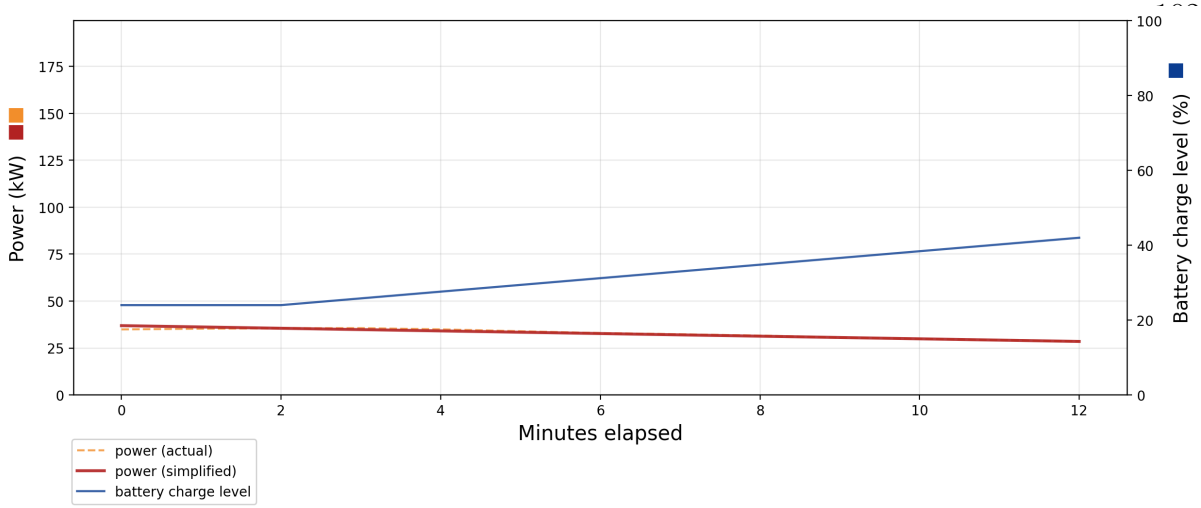

Figure B.1: Ex. 1, Normal, k=1

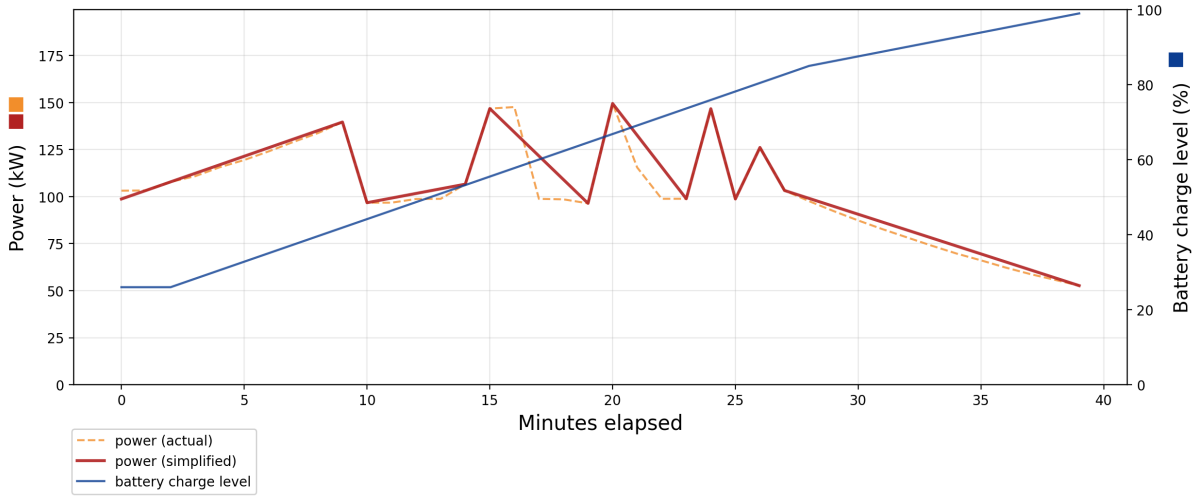

Figure B.2: Ex. 2, Abnormal, k=12

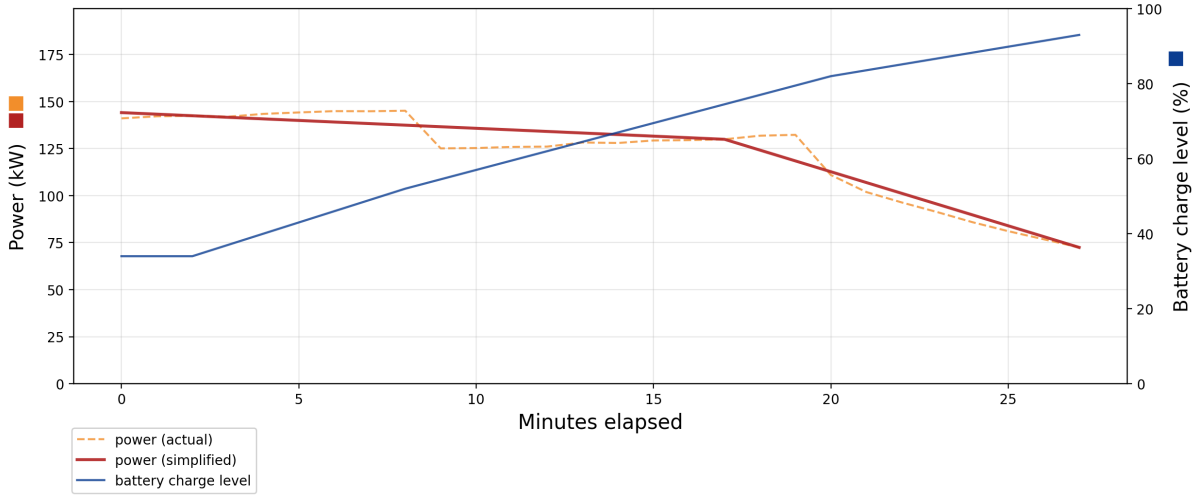

Figure B.3: Ex. 5, Normal, k=2

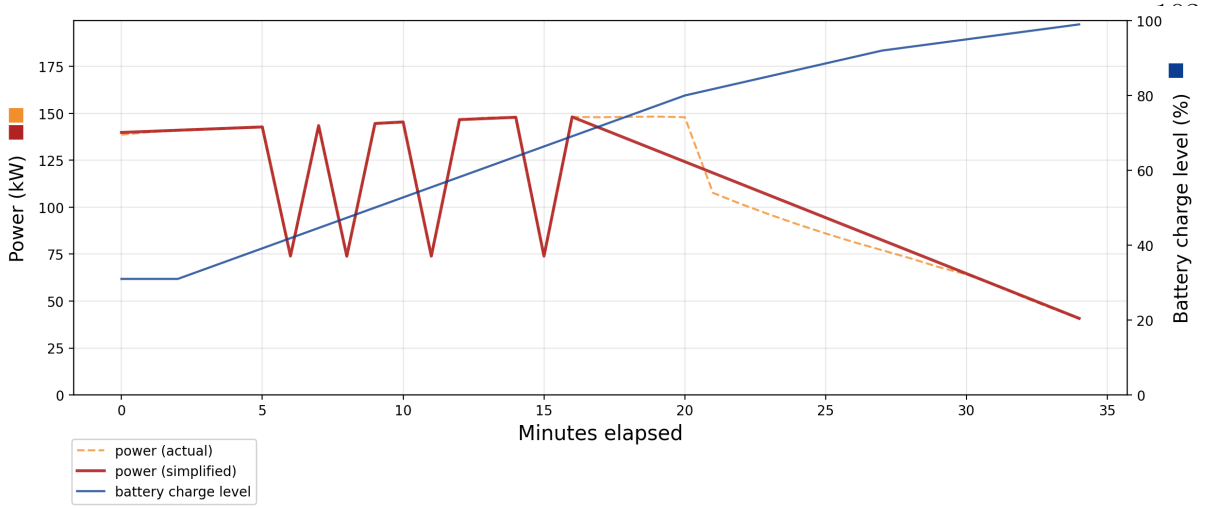

Figure B.4: Ex. 6, Abnormal, k=12

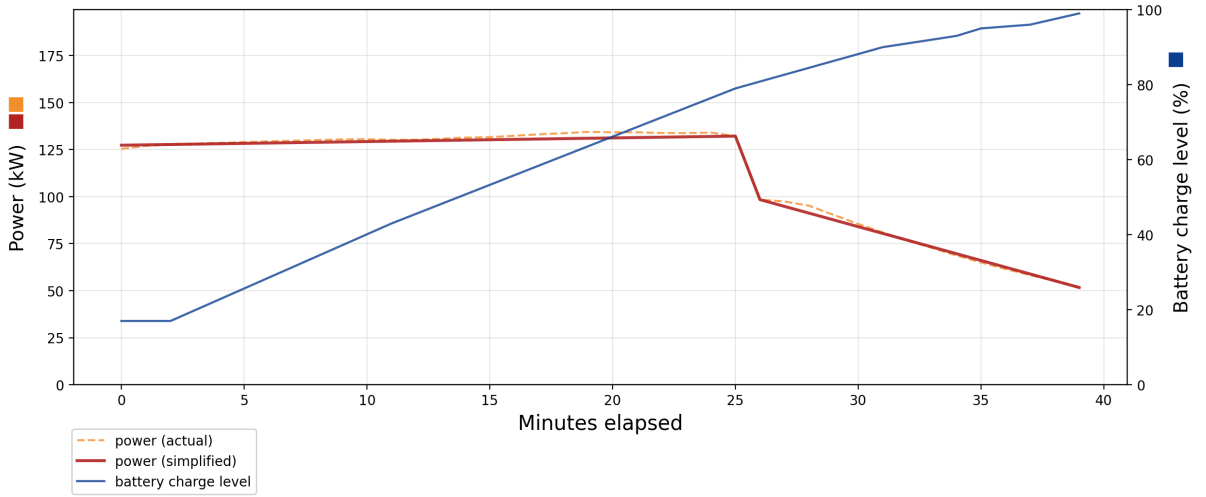

Figure B.5: Ex. 37, Normal, k=3

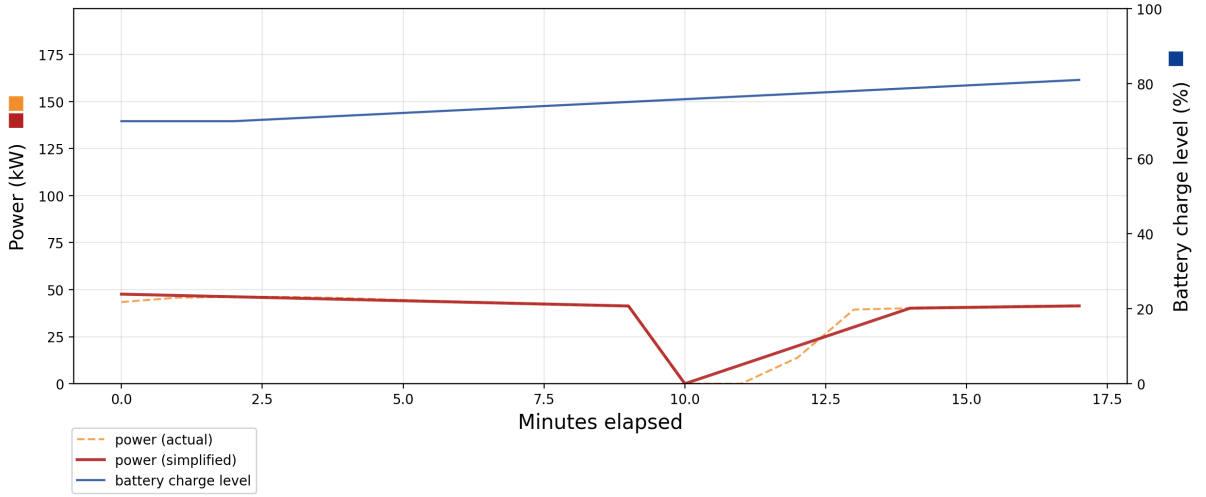

Figure B.6: Ex. 38, Abnormal, k=4

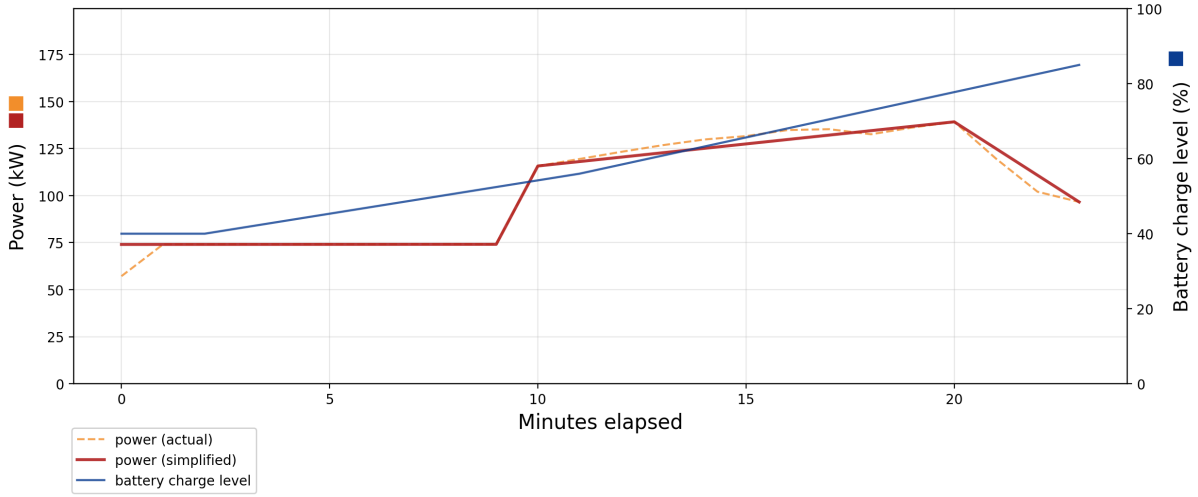

Figure B.7: Ex. 60, Abnormal, k=4

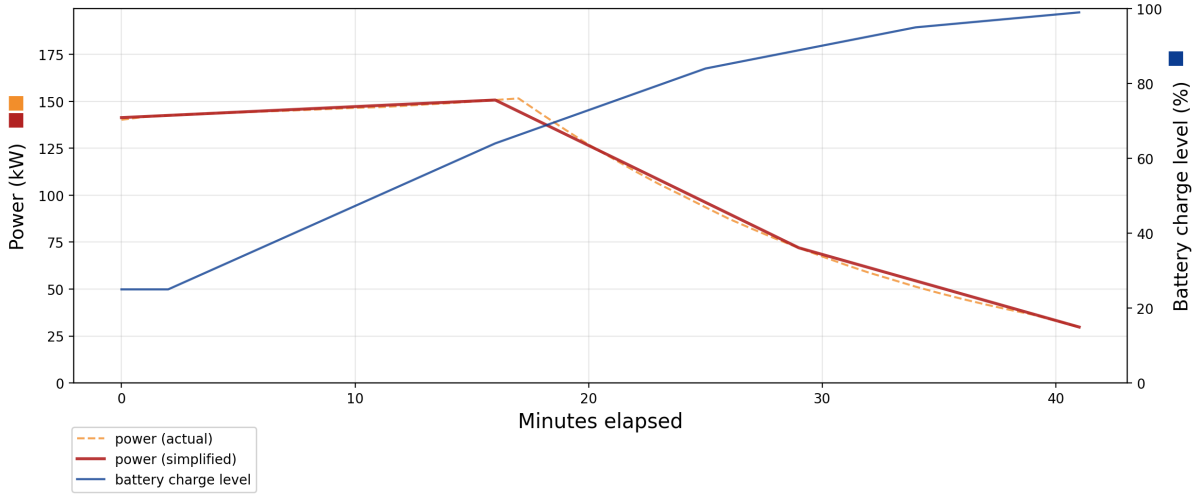

Figure B.8: Ex. 60, Normal, k=3

Appendix C:

Prompts Used in the MLLM Experiment

This appendix reports the exact prompt texts used in the implementation of the automated multimodal LLM experiment. Placeholders such as {`rule`}, {`item_id`}, and {`idx`} are runtime substitutions.

C.1 System Prompt (All Participants, All Phases)

```
You are participating in an Explainable AI study where you must infer the
behaviour of a fixed black-box AI model from examples and then simulate
its predictions. Your task is not to judge whether a charging session is
good or bad in the real world, but to mimic how this AI tends to label
sessions. For every task in this study you must respond with a single JSON
object that exactly matches the schema described in the user's
instructions. Do not include any additional explanations, commentary, code
fences, or extra keys. Return only machine-readable JSON.
```

C.2 Exam Prompts

C.2.1 Shared Intro Blocks

Raw-only intro

There is an AI that has been trained on a large dataset of electric vehicle (EV) charging sessions. The AI classifies each charging session as either 'normal' or 'abnormal'. It bases its decision only on patterns in the power transfer between charger and battery (in kW) over time, the battery's state of charge (SOC) over time, and a few technical features such as temperature and charger specifications. You do not know the exact rule the AI uses.
You will be shown line charts of individual charging sessions. In each chart, the orange line shows the power in kW over minutes since the start of charging and the blue line shows the battery's SOC in percent over time.
Each of these charts was shown to the AI. Your goal is to infer how this AI tends to behave and to simulate its 'normal'/'abnormal' decisions on new examples.

Overlay intro

There is an AI that has been trained on a large dataset of electric vehicle (EV) charging sessions. The AI classifies each charging session as either 'normal' or 'abnormal'. It bases its decision only on patterns in the power transfer between charger and battery (in kW) over time, the battery's state of charge (SOC) over time, and a few technical features such as temperature and charger specifications. You do not know the exact rule the AI uses.
You will be shown line charts of individual charging sessions. In each chart, the dashed orange line shows the original power in kW over minutes since the start of charging, the solid red line shows a simplified version of the same power curve, and the solid blue line shows the battery's SOC in percent over time. Each of these charts was shown to the AI. When we showed the AI only the simplified red power curve (without the orange line), it produced exactly the same classification as when it saw the original power curve. This means that, for this AI, the red simplified power curve together with the blue SOC curve contains all the information it needs to decide between 'normal' and 'abnormal'. Your goal is to infer how this AI tends to behave and to simulate its decisions on new examples.

Simplified-only intro

```
There is an AI that has been trained on a large dataset of electric vehicle
(EV) charging sessions. The AI classifies each charging session as either
'normal' or 'abnormal'. It bases its decision only on patterns in the power
transfer between charger and battery (in kW) over time, the battery's state
of charge (SOC) over time, and a few technical features such as temperature
and charger specifications. You do not know the exact rule the AI uses.
You will be shown line charts of individual charging sessions. In each chart,
the solid red line shows a simplified version of the charging power in kW over
minutes since the start of charging, and the solid blue line shows the battery's
SOC in percent over time. Each chart was shown to the AI as simplified power and
SOC only. Your goal is to infer how this AI tends to behave and to simulate its
decisions on new examples.
```

C.2.2 Post-exam prefix

```
Let us test what you have learned about this AI. You will now see new,
unlabelled charging session examples. For each example, you must guess how
the AI would classify it ('normal' or 'abnormal') based on the patterns you
observed during the teaching examples.
```

C.2.3 Locked-rule carry-over (Groups A-E, post-exam)

```
You finished the teaching phase with this locked rule snapshot: "{rule}".
Do not update, rewrite, refine, or shorten this locked rule during the exam.
Use this fixed rule to label every example in the batch.
```

C.2.4 Decision checklist (Groups A-E, post-exam)

```
Use this decision checklist for every item in the batch.
1. Do not rewrite, refine, or shorten the fixed rule.
2. Apply one consistent fixed rule across all examples in the batch.
3. {tie_breaker_text}
```

Default `tie_breaker_text` (groups A–D):

```
If the fixed rule is ambiguous for one item, choose the label
whose locked cue pattern is the closest match.
```

Group E `tie_breaker_text`:

```
If the fixed rule is ambiguous for one item, use this tie-break rule.
Choose the label whose locked cue pattern is the closest match to the current
example. Keep this tie-break behaviour consistent for all ambiguous items in
the batch.
```

C.2.5 Batch answer schema instruction

```
You will be shown a *batch* of exam examples. For each example you must
predict whether the charging session is 'normal' or 'abnormal'.Important:
 - For each batch, respond with a **single JSON object**.
 - Do not include any extra keys.
 - Do not add explanations, comments, or prose outside the JSON.
 - The JSON must have exactly this structure:
{'answers': [{'item_id': '<ITEM_ID_1>', 'guess': 'your_guess_label'},
             {'item_id': '<ITEM_ID_2>', 'guess': 'your_guess_label'},
             ...]}
 where you must include exactly one entry object per seen example/item_id.
 The value of 'guess' must be either 'normal' or 'abnormal' in lower case.
 Do not use any other labels.
```

C.2.6 Per-item line (exam)

```
Example {idx} (item_id={item_id}).
```

Then one image is attached as an `image_url` content block.

C.2.7 Repair prompt for missing exam answers

```
Your previous response does not include valid answers for all required
item IDs in this batch. Return exactly one JSON object in this shape:
{'answers': [{'item_id': '<ITEM_ID>', 'guess': '<normal|abnormal>'}, ...]}.
Include entries only for these missing item IDs: {missing_ids_csv}.
Do not include already answered IDs. Do not include any text outside JSON.
```

C.3 Teaching Prompts

C.3.1 Teaching phase intro (Groups A–C)

```
We will now show you labelled examples that reveal how the AI behaved on
specific charging sessions. For each example you will see one charging-chart
image together with the AI's classification ('normal' or 'abnormal'). Study
the relationship between the power curve(s) and the SOC curve carefully and
update your internal rule for how the AI seems to decide. Most examples require
an acknowledgement response. Every tenth example requires a structured checkpoint
JSON summary with normal cues, abnormal cues, exceptions, confidence and a
rule-of-thumb.
```

C.3.2 Teaching phase intro (Group D)

```
We will now show you labelled examples that reveal how the AI behaved on
specific charging sessions. For each example you will see one charging-chart
image together with the AI's classification ('normal' or 'abnormal'). The chart
shows only simplified power and SOC. Study the relationship between the
simplified power curve and the SOC curve carefully and update your internal rule
for how the AI seems to decide. Most examples require an acknowledgement response.
Every tenth example requires a structured checkpoint JSON summary with normal cues,
abnormal cues, exceptions, confidence and a rule-of-thumb.
```

C.3.3 Teaching phase intro (Group E)

```
We will now show you labelled examples that reveal how the AI behaved on
specific charging sessions. For each example you will see one charging-chart
image together with the AI's classification ('normal' or 'abnormal'). The chart
shows only simplified power and SOC. After each example, respond with a single
JSON object that includes exactly five keys:
{'description_sentence': '<one sentence>', 'rule_action': '<write|retain|rephrase>',
 'normal_cues': ['<cue>', '...'], 'abnormal_cues': ['<cue>', '...'],
 'rule_of_thumb': '<current rule>'}. The first example must use
'rule_action': 'write' to establish an initial rule. Keep both cue arrays
non-empty. If one side has no clear cue, include a placeholder cue string
instead of leaving that array empty.
```

C.3.4 Per-item label line (all teaching groups)

```
This is teaching example {index} of {total} (item_id={item_id}).
The AI classified this example as "{ai_class}".
```

Then one image is attached as an `image_url` content block.

C.3.5 Non-checkpoint response instruction (Groups A–D)

```
Once you have studied this example, respond with {"acknowledged": true}
and nothing else.
```

C.3.6 Checkpoint instruction (Groups A–D, every 10th example)

```
This example is a checkpoint. Respond with exactly one JSON object and
no other text. Use this copyable template exactly:
{"acknowledged":true,
 "normal_cues":["<normal cue>","..."],
```

```
 "abnormal_cues":["<abnormal cue>","..."],
 "exceptions":["<exception cue>","..."],
 "confidence":<float between 0 and 1>,
 "rule_of_thumb":"<current rule>"}.
Both cue arrays must be non-empty. If one side has no clear cue yet,
add a placeholder such as "no_clear_normal_cue_at_checkpoint" or
"no_clear_abnormal_cue_at_checkpoint".
```

If available, this sentence is appended:

```
The previous checkpoint rule-of-thumb is: "{prior_rule}".
```

C.3.7 Group E per-example response instruction

Example 1:

```
Respond with exactly one JSON object and no other text.
Use this copyable template exactly:
{"description_sentence":"<one sentence description of this example>",
 "rule_action":"write",
 "normal_cues":["<normal cue>","..."],
 "abnormal_cues":["<abnormal cue>","..."],
 "rule_of_thumb":"<your initial rule>"}.
Both cue arrays must be non-empty. If this single example gives no clear
cue for one side, include a placeholder such as
"no_clear_normal_cue_from_this_example" or
"no_clear_abnormal_cue_from_this_example".
```

Examples 2..N:

```
Your current rule-of-thumb before this example is: "{prior_rule}".
Respond with exactly one JSON object and no other text.
Use this copyable template exactly:
{"description_sentence":"<one sentence description of this example>",
 "rule_action":"<write|retain|rephrase>",
```

```
 "normal_cues":["<normal cue>","..."],
 "abnormal_cues":["<abnormal cue>","..."],
 "rule_of_thumb":"<your updated or retained rule>"}.
Both cue arrays must be non-empty. If this single example gives no clear
cue for one side, include a placeholder such as
"no_clear_normal_cue_from_this_example" or
"no_clear_abnormal_cue_from_this_example".
```

C.4 Final Rule-Lock Prompt (Groups A–E)

```
Teaching phase is complete. You have now seen {teaching_examples_seen}
labelled examples using {group_text}. Create one final locked rule snapshot
that will be reused unchanged during the post-exam. This lock is strict,
so do not leave out any required keys.
```

If available, this sentence is appended:

```
Current rule-of-thumb before lock: "{prior_rule}".
```

Then:

```
Respond with exactly one JSON object and no other text:
{'locked_rule_of_thumb': '<final fixed rule>',
 'normal_cues': ['<normal cue>', '...'],
 'abnormal_cues': ['<abnormal cue>', '...'],
 'exceptions': ['<exception cue>', '...'],
 'confidence': <float between 0 and 1>}.
```