Sensitivity and Ontology-Guided Counterfactual Explanation Generation for Battery State of Charge Estimation

Slimane Arbaoui[a], Ali Ayadi[b], Ahmed Samet[c], Tedjani Mesbahi[d] and Romuald Boné[e]
Université de Strasbourg, INSA Strasbourg, ICube laboratory UMR 7357 and CNRS, Strasbourg, 67000, France

Keywords: Counterfactual Explanations, Multivariate Time Series, Explainable AI, Ontology-Based Validation.

Abstract: Deep learning models have demonstrated exceptional performance in estimating battery parameters such as State Of Charge (SOC) and State Of Health (SOH) in electric vehicles. However, their black-box nature hinders interpretability, which is critical in safety-sensitive applications. In this work, we introduce *a Sensitivity and Ontology-Guided Counterfactual Explanation method* (*SOCEG*) for generating counterfactual explanations in multivariate time series regression tasks. *SOCEG* combines a gradient-based technique to identify key data segments for modification with an ontology-driven validator to enforce domain constraints and preserve feature correlations, ensuring plausibility and realism. Unlike conventional methods, it does not require an initial population and progressively refines counterfactual quality across iterations. We evaluate *SOCEG* on SOC estimation for Lithium Iron Phosphate (LFP) cells and benchmark it against four baselines, including GENO-TOPSIS and NSGA-II. Results demonstrate that *SOCEG* consistently surpasses existing approaches in dissimilarity, implausibility, instability, and runtime, while providing actionable insights into feature modifications, thereby enhancing the model interpretability.

1 INTRODUCTION

Deep learning models have demonstrated remarkable accuracy across a wide range of applications, particularly in domains involving complex and high-dimensional data such as image processing (He et al., 2015; Bakator and Radosav, 2018) and time series analysis (Lim and Zohren, 2021). With the rapid advancements in the Internet of Things (IoT) and the deployment of 5G networks, these models are increasingly being integrated into critical real-world applications. A prominent example is the transportation sector, especially in the context of Electric Vehicles (EVs), where deep learning techniques are widely employed to estimate key battery parameters. Among these, the State Of Charge (SOC) and the State Of Health (SOH) are of particular importance, as accurate estimation of these indicators directly impacts the efficiency, reliability, and safety of battery management systems.

Although deep learning models are highly accurate and effective at uncovering complex patterns, their black-box nature makes it difficult to understand how predictions are generated (Chennam et al., 2022). This lack of transparency becomes particularly critical in safety-sensitive domains such as EVs, where it is essential to assess how models will behave under unseen scenarios. To address this concern, some researchers prefer to adopt inherently interpretable models, such as linear Support Vector Machines (SVMs) (Álvarez Antón et al., 2013), decision trees (Qian et al., 2023), linear regression (Castanho et al., 2022), and Naïve Bayes methods (Ng et al., 2014), despite their potential trade-off in predictive performance compared to deep learning approaches.

Post hoc explanation methods provide an alternative approach for interpreting model predictions without modifying the underlying architecture (Kok et al., 2022). These methods can be broadly categorized into three types:

1. **Feature importance methods**, such as Local Interpretable Model-Agnostic Explanations (LIME) (Ribeiro et al., 2016), which employs local surrogate models, and SHapley Additive exPlanations (SHAP) (Lundberg, 2017), which calculates Shapley values to quantify each feature's contri-

[a] https://orcid.org/0000-0002-9805-6103
[b] https://orcid.org/0000-0003-1660-4100
[c] https://orcid.org/0000-0002-1612-3465
[d] https://orcid.org/0000-0003-3808-6519
[e] https://orcid.org/0000-0002-0106-6576

Arbaoui, S., Ayadi, A., Samet, A., Mesbahi, T. and Boné, R.
Sensitivity and Ontology-Guided Counterfactual Explanation Generation for Battery State of Charge Estimation.
DOI: 10.5220/0014236600004052

In *Proceedings of the 18th International Conference on Agents and Artificial Intelligence (ICAART 2026) - Volume 2*, pages 1210-1221
ISBN: 978-989-758-796-2; ISSN: 2184-433X

bution.

2. **Rule-based methods**, which generate simple if-then rules to describe the model's behavior (Ribeiro et al., 2018).
3. **Hybrid approaches**, combining feature attribution with visualization, such as saliency-based techniques that highlight input regions most relevant to the outputs (Saadallah et al., 2021).

In the context of explaining SOC and SOH estimation models, most studies have focused on SHAP (Hidouri et al., 2024; Qian et al., 2025; Arbaoui et al., 2024) and LIME (Chen et al., 2025). While these approaches can highlight which features are important and their contributions to predictions, they do not reveal the internal mechanisms of the model. For instance, (Hidouri et al., 2024) demonstrates that the model assigns higher importance to current and voltage, with lower current values negatively impacting SOC and a positive correlation observed between voltage and SOC, while temperature is largely ignored. This is particularly limiting for time series data, where temporal dependencies play a crucial role. These limitations are partly due to method-specific issues: in the case of SHAP, importance scores can be misleading, potentially indicating significance for features that are actually irrelevant (Young et al., 2019; Huang et al., 2023; Huang and Marques-Silva, 2024). For LIME, the reliance on synthetically generated local data may lead to explanations that do not accurately reflect the true model behavior (Rahnama and Boström, 2019).

While these methods provide valuable insights into feature importance, they are inherently limited in that they describe "what" influences the prediction rather than "how" the prediction would change under different conditions. This limitation has motivated researchers to explore alternative approaches that provide more actionable and intuitive explanations. In this context, Counterfactual (Cf) explanations have recently gained increasing attention (Wachter et al., 2018). Cf methods indicate what specific changes in an input instance would be necessary to alter the model's prediction, offering a more direct understanding of the model's decision boundaries and potential behavior under hypothetical scenarios. Formally, they can be defined as a function f_k that takes as input a deep learning model $\mathcal{M}$ and a query instance qr from a given dataset X used for training. The function f_k then returns a set of valid Cf examples that offer alternative modifications to qr, leading to a different model prediction (Guidotti, 2022): $C = f_k(qr, \mathcal{M}, X) = \{x'_1, x'_2, \ldots, x'_h\}$ where $h \leq k$, with k representing the number of Cf examples required by the user.

To generate high-quality Cf explanations, several key properties must be considered. A good Cf should suggest minimal changes to the query instance, both in terms of the magnitude of changes and the number of features altered, while ensuring that the prediction is modified to the desired outcome and that the resulting instance remains plausible. While most existing works focus primarily on minimal change and prediction alteration, to the best of our knowledge, few studies attempt to generate Cfs for multivariate time series data, and even fewer rigorously evaluate the realism of the generated Cfs. In battery applications, realism is defined by adherence to electrochemical constraints, such as the interdependence of current, voltage, and temperature during charging and discharging. Cfs that violate these relationships, for example, suggesting that SOC can increase solely through a change in temperature while voltage and current remain constant, are physically implausible and compromise the reliability of explanations in practical battery management systems.

In this paper, we propose a *Sensitivity and Ontology-Guided Counterfactual Explanation method* (*SOCEG*) for generating counterfactual explanations for multivariate time series data, applied specifically to an SOC estimation model. Our method leverages a gradient-based technique to identify the parts of the data that need to be modified. This approach offers several advantages, most notably that it does not require an initial population. In practical scenarios, users may have limited or low-quality data for generating Cfs, and the choice of the initial population can significantly impact the runtime of traditional methods. Moreover, in optimization-based approaches, the selection of the loss function strongly influences the quality of the generated Cfs, even though it provides an effective solution for addressing the *plausibility* problem.

SOCEG ensures that the proposed changes respect the correlations between features, thereby maintaining realistic Cfs. To further address the *plausibility* issue, we incorporate a domain ontology reasoning process that verifies compliance with domain-specific constraints and rules derived from the physical and electrical behavior of the battery cell.

The performance of the proposed method is evaluated using several metrics derived from (Guidotti, 2022), which will be detailed in the following section. Additionally, it is compared against four baseline methods: GENO-TOPSIS, NSGA-II, CoMTE*m* (Ates et al., 2021), and AB-CF*m* (Li et al., 2023), with the latter two representing modified versions of the original methods that were initially designed for classification tasks.

To summarize, our contributions are as follows:

- The formulation of counterfactual explanations for multivariate time series data in a sequence-to-sequence prediction context.
- The proposal of a method for generating Cfs without requiring an initial population, while ensuring plausibility through a dedicated ontology reasoner that encodes datasheet information and physical rules of Lithium Iron Phosphate (LFP) cells.
- The evaluation of the approach on a real scenario of SOC estimation for LFP cells and comparison against several baseline methods.

2 STATE OF THE ART

To generate Cf explanations, key properties have been defined to measure their quality. These properties are based on the work of Verma et al. (Verma et al., 2020), Mothilal et al. (Mothilal et al., 2020a), Wachter et al. (Wachter et al., 2018), and Guidotti et al. (Guidotti, 2022), and they are as follows:

- *Validity*: A Cf must change the model's prediction to a desired outcome. In a classification task, this means flipping the predicted class, while in a regression task, the prediction should be adjusted to fall within a specific target range.
- *Similarity (Proximity)*: A Cf should be as close as possible to the query instance in terms of distance.
- *Minimality (Sparsity)*: Only a minimal set of features should be changed, rather than modifying all features.
- *Diversity*: Counterfactuals generated for a given query instance should follow different paths to change the model's output. No two counterfactuals should be identical while modifying the same set of features.
- *Plausibility (Feasibility, Reliability)*: Cfs should resemble existing data, ensuring that the generated instances are realistic.
- *Actionability*: A Cf must modify only actionable features, i.e., those that the user permits to change.

Four categories of methods can be identified to generate Cfs explanations (Guidotti, 2022):

- *Optimization*: Methods that construct a loss function by incorporating some of the key properties mentioned above and seek to minimize it (Wachter et al., 2018; Dhurandhar et al., 2018; Dhurandhar et al., 2019).
- *Heuristic Search Strategy*: Methods that iteratively update specific parts of the query instance to minimize a cost function (Vermeire and Martens, 2020; Gomez et al., 2020).
- *Instance-based Strategies*: The simplest approach, where Cfs are directly selected from a given population by identifying instances that minimize the distance to the query instance while yielding a different prediction (Poyiadzi et al., 2020; Brughmans et al., 2022).
- *Decision Trees*: This approach approximates the black-box model locally using a decision tree, where the extracted rules are then utilized to generate Cfs (Guidotti et al., 2020).

Most existing methods focus on tabular data and classification tasks (Guidotti, 2022), with Diverse Counterfactual Explanations (DiCE) (Mothilal et al., 2020b) being one of the most widely used frameworks. Only a few studies have explored counterfactual explanations for time series data (Wang et al., 2023; Delaney et al., 2021), and even fewer have addressed multivariate time series (Ates et al., 2021; Li et al., 2023). These approaches typically formulate the problem as an optimization task and define a loss function, but they focus only on classification problems and often neglect key properties such as *diversity* and *plausibility*.

To the best of our knowledge, only one approach (Arbaoui et al., 2025) has applied Cf explanations to interpret a deep learning model for SOC estimation of LFP batteries. This approach represents the only method to successfully generate Cf explanations for multivariate time series data in a regression task (sequence-to-sequence prediction). In their study, Cf explanations were generated using two methods based on classical genetic algorithm (GA) steps:

- *GENO-TOPSIS*, which employs the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) as a selection method.
- *NSGA-II*, the Non-Dominated Sorting Genetic Algorithm.

Both methods focus on identifying optimal Cfs based on three key properties: *validity, proximity, and diversity*. An ontology is used to filter Cfs that do not respect domain constraints. However, the main limitation is that the generation process and ontology validation are performed separately. If all generated Cfs are filtered out, new ones must be regenerated, which is time-consuming due to the GA-based process. Additionally, the method does not incorporate learning after each iteration; if a Cf is generated by modifying a specific subsequence of the data, this information is not used to improve subsequent generations. As a

result, the process remains inefficient and leads to increased computational costs.

3 THEORETICAL FOUNDATIONS

3.1 Counterfactual Properties

In this section, we provide a detailed presentation of the Cf properties used in this work (*validity*, *proximity*, *sparsity* and *diversity*). Consider a multivariate time series query instance $qr \in \mathbb{R}^{m \times n}$, where m represents the sequence length and n is the number of features. This query instance is taken from an initial population X and fed into a black-box model $\mathcal{M}$, where $\mathcal{M}(qr) = pr \in \mathbb{R}^t$ denotes the model's prediction for the query instance, with t representing the output dimension. The set $C = \{x'_1, x'_2, \ldots, x'_h\}$ represents the Cfs generated using our approach, and $\mathcal{M}(C) = \{pr'_1, pr'_2, \ldots, pr'_h\}$ denotes the corresponding model predictions for the generated Cfs. Let k represent the number of Cfs desired by the user.

The *validity* property can be calculated depending on the *direction* specified by the user, to either maximize (greater) the output or minimize it (smaller). It is calculated in our case based on Equation 1.

$$\text{validity}(x'_i) = \begin{cases} \|pr'_i - pr\|_2 & \text{if direction = “greater” and } pr'_i[d] > pr[d],\ \forall d \in \{1,\ldots,t\} \\ \|pr'_i - pr\|_2 & \text{if direction = “less” and } pr'_i[d] \leq pr[d],\ \forall d \in \{1,\ldots,t\} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

For the *proximity*, we calculate the distance between a Cf and the query instance as shown in Equation 2

$$proximity(x'_i) = \|x'_i - qr\|_2 \tag{2}$$

For the *sparsity* property, we measure the feature-wise distance S_j between a generated Cf and the query instance. If the distance is greater than a predefined threshold ε, the feature is considered changed. The number of features changed is computed as shown in Equation 3:

$$sparsity(x'_i) = \sum_{j=1}^{n} 1_{S_j > \varepsilon} \tag{3}$$

Diversity, on the other hand, calculates the minimum distance between a Cf and its nearest neighbor in C, using Equation 4

$$diversity(x'_i) = \min_{\substack{z \in C \\ z \neq x'_i}} \|x' - z\|_2 \tag{4}$$

There are properties that need to be maximized such as *validity* and *diversity*, and those that need to be minimize like *proximity* and *sparsity*.

3.2 Ontologies

An ontology provides a structured framework for representing knowledge by defining concepts and their relationships within a specific domain, and it is represented as a four-element tuple (Cao et al., 2022):

$$O = \langle Cl, Prts, \text{Sub}, \text{Applic} \rangle$$

where: Cl denotes the set of classes that define the domain concepts. $Prts$ represents the set of properties used to describe instances of these classes. Sub : $Cl \rightarrow 2^C$ is the subsumption function, where for any two classes $cl_1, cl_2 \in Cl$, cl_1 is a superclass of cl_2 if $\forall x \in cl_2, x \in cl_1$. Applic : $Cl \rightarrow 2^{Prts}$ maps each class to its associated properties, including both object and data properties. In the context of counterfactual explanations, ontologies play a crucial role in ensuring the *actionability* and *plausibility* of generated explanations. By leveraging domain-specific ontologies, counterfactual generation methods can incorporate logical constraints and background knowledge to produce meaningful and realistic alternative scenarios.

For instance, in battery SOC estimation, an ontology can define relationships between voltage, temperature, and charge levels, ensuring that generated counterfactuals respect physical constraints and domain knowledge. This is achieved using a reasoner to evaluate Semantic Web Rule Language (SWRL) rules, if-then statements specifying conditions for modifying ontology instances. The reasoner thus determines whether a counterfactual candidate satisfies domain constraints.

The key advantage of using ontologies is transparency: expert knowledge is directly embedded into the counterfactual generation process. Rejected counterfactuals can be traced to specific violated rules, enhancing interpretability. Moreover, all valid counterfactuals are stored within the ontology and can be queried via SPARQL (spr, 2025), supporting structured analysis and systematic insight extraction.

4 SENSITIVITY AND ONTOLOGY-GUIDED COUNTERFACTUAL EXPLANATION GENERATION

SOCEG generates Cf candidates based on the model's sensitivity, which will be described in the fol-

lowing subsection. These candidates, along with their properties, are added to an ontology that defines domain constraints. In our case, datasheet constraints for an LFP cell, as well as Cf properties expressed through SWRL rules.

The method initially generates candidates that aim to change the model prediction, without ensuring the realism of the proposed modifications. These candidates are then sent to the ontology, which classifies them into two categories: *realistic* and *unrealistic*. Candidates deemed realistic are fed back into the method to guide the generation of additional counterfactuals. This iterative process continues until the number of realistic counterfactuals in the ontology matches the number of counterfactuals requested by the user.

4.1 Counterfactual Generation Guided by Model Sensitivity

Instead of adding random noise to a random subsequence of qr or relying on the classical steps of a GA that require an initial population, our approach leverages model gradients to guide counterfactual generation. Specifically, we compute the gradient of the model's output with respect to its input, yielding a weight matrix W of the same shape as qr (m,n), which highlights the regions of the input that contribute most strongly to the prediction. These weights are then used to introduce noise l, which follows a Gaussian distribution. To preserve feature dependencies, we generate this noise using the multivariate normal distribution [1], where the mean values for each feature are derived from qr, and the covariance matrix is computed accordingly.

At each iteration, a Cf candidate is generated by either adding or subtracting a perturbation δ. This perturbation is determined based on the weight matrix W, the noise l, and the standard deviation std of each feature in qr, ensuring that δ remains stable when feature values are constant. The model then predicts the outcome for the generated counterfactual and compares it to the original prediction pr. Depending on the user-specified direction, the algorithm aims to either increase or decrease the prediction value.

To accelerate convergence and improve learning, the ontology provides a population of validated Cfs (*realistic*). If available, a random selection from this population serves as the basis for generating new candidates. However, rather than modifying the entire instance, we introduce noise only to the feature subset that was not originally used for estimation and that by updating W. If the ontology does not return a population, previously generated, yet unvalidated Cfs are refined following the same approach.

Algorithm 1 details the entire process step by step and returns a list of Cf candidates that satisfy four key properties:

- **Validity.** Ensured by selecting only Cfs that successfully alter the prediction.
- **Proximity & Sparsity.** Achieved by leveraging gradients to ensure minimal modifications and targeting only relevant features.
- **Diversity.** Maintained by iteratively modifying not just the original qr, but also by selecting Cfs randomly from the ontology-provided population. These candidates adhere to all constraints, including domain-specific constraints.

Algorithm 1: Generate Counterfactual Candidates (*SOCEG*).

Require: Model $\mathcal{M}$, query instance qr, its prediction pr, direction of perturbation, Ontology $onto$, number of Cfs requested k
Ensure: A set of realistic Cfs C

Initialize $C \leftarrow \{\}$
Get realistic candidates from ontology: $P \leftarrow onto.RealisticCf()$
while $|P| < k$ **do**
  if P is not empty **then**
    Select x randomly from P
    Compute gradient matrix $W = \frac{\partial \mathcal{M}(x)}{\partial qr}$
    Update W to focus on unused features: $W = \max(W) + \min(W) - W$
  else if C contains non-validated Cfs **then**
    Select x from non-validated Cfs
    Compute and update W as above
  else
    $x \leftarrow qr$
    Compute $W = \frac{\partial \mathcal{M}(x)}{\partial qr}$
  end if
  Compute mean μ and covariance matrix Σ from x
  Generate noise $l \sim \mathcal{N}(\mu, \Sigma)$
  Compute standard deviation std of features in x
  for each feature i **do**
    $\delta_i = 0.5 \cdot l_i \cdot W_i \cdot std_i$
    Randomly select $sign \in \{+1, -1\}$
    $x_i \leftarrow x_i + sign \cdot \delta_i$
  end for
  Clip x to valid range $[0,1]$
  Compute new prediction $Y = \mathcal{M}(x)$
  if Y changes in the required direction **then**
    Add x to C
    Send x to ontology and run reasoner
    Update $P \leftarrow onto.RealisticCf()$
  end if
end while
return P

[1] https://numpy.org/doc/2.2/reference/random/generated/numpy.random.multivariate_normal.html

All counterfactual candidates generated by this method will be sent to the ontology module to verify the *plausibility* property and ensure that they adhere to the domain constraints specified in the domain ontology. For each counterfactual candidate, we will compute its properties before sending them all to the ontology. While these properties could also be computed directly within the ontology, we prefer to precompute them to reduce the complexity of the ontology, ensuring that the reasoner does not crash due to the large volume of data points being processed.

4.2 Ontology for Guiding Counterfactual Selection

To ensure that the generated Cfs respect both domain constraints and desired counterfactual properties, we defined a general ontology (comprising 234 axioms in total, including 383 logical axioms and 135 declaration axioms, as well as 47 classes, 44 object properties, 42 data properties), illustrated in Figure 1, which integrates two main components:

- **Component 1:** Encodes all concepts related to Cfs and the representation of multivariate time series data.
- **Component 2:** Encodes all domain-specific concepts, in this case the constraints derived from the datasheet of the LFP cells used to train the model.

The first component captures all concepts associated with Cfs, including the values of each feature measured over time, the counterfactual properties, and the thresholds linked to these properties. These thresholds act as selection criteria for classifying a counterfactual candidate into two categories: *RealisticCf* and *UnRealisticCf*. Moreover, the decision-making process is further refined by the domain ontology, as it extends the *Validity Criterion* through the *Domain Specification*.

Below are some examples of rules coming from the first component:

Rule 1: Counterfactual selection based on domain and counterfactual properties.

```
Counterfactual(?cf) ∧ Follow(?cf, ?criterion1) ∧ DomainSpecification(?criterion1) ∧ isRespected(?criterion1, true) ∧ Follow(?cf, ?criterion2) ∧ ProximityProperty(?criterion2) ∧ isRespected(?criterion2, true) ∧ Follow(?cf, ?criterion3) ∧ ValidityProperty(?criterion3) ∧ IsRespected(?criterion3, true) ∧ Follow(?cf, ?criterion4) ∧ SparsityProperty(?criterion4) ∧ isRespected(?criterion4, true) ∧ Follow(?cf, ?criterion5) ∧ DiversityProperty(?criterion5) ∧ isRespected(?criterion5, true) → RealisticCF(?cf)
```

Rule 2: If a counterfactual does not respect domain specification or counterfactual properties, it is categorize as *UnrealisticCf*.

```
Counterfactual(?cf) ∧ Follow(?cf, ?criterion1) ∧ DomainSpecification(?criterion1) ∧ isRespected(?criterion1, false) → UnrealisticCF(?cf)
```

```
Counterfactual(?cf) ∧ Follow(?cf, ?criterion2) ∧ CounterfactualProperties(?criterion2) ∧ isRespected(?criterion2, false) → UnrealisticCF(?cf)
```

Rule 3 (*Proximity*): If the distance is less than the threshold, the *proximity* criterion is respected.

```
Counterfactual(?cf) ∧ Follow(?cf, ?criterion) ∧ ProximityProperty(?criterion) ∧hasProximityDistance(?cf, ?distance) ∧ hasProximityThresholdValue(?criterion, ?threshold) ∧ lessThan(?distance, ?threshold) → isRespected(?criterion, true)
```

Similarly, we define rules for *sparsity*, *validity*, and *diversity*. These rules follow the same principle as above, differing only in whether the goal is to minimize (e.g., proximity and sparsity) or maximize (e.g., diversity and validity) the respective property.

The second component represents the domain ontology and contains all rules that ensure the *plausibility* of Cfs. In our case, we adopted a modified version of the Battinfo ontology (Bat,), since the original ontology does not include SWRL rules. We extended it by adding rules to enforce the recommended ranges of current, voltage, and temperature, as described in (Arbaoui et al., 2025). Furthermore, we incorporated rules to capture the positive correlation between voltage and SOC, based on the open circuit model (Yan et al., 2025), thereby ensuring that the generated Cfs comply with physical and operational constraints necessary for the safe and optimal functioning of the cell.

Rule 5: Ensures that the *isRespect* data property for *DomainSpecification* is updated correctly if Cf respect all domain-constraints.

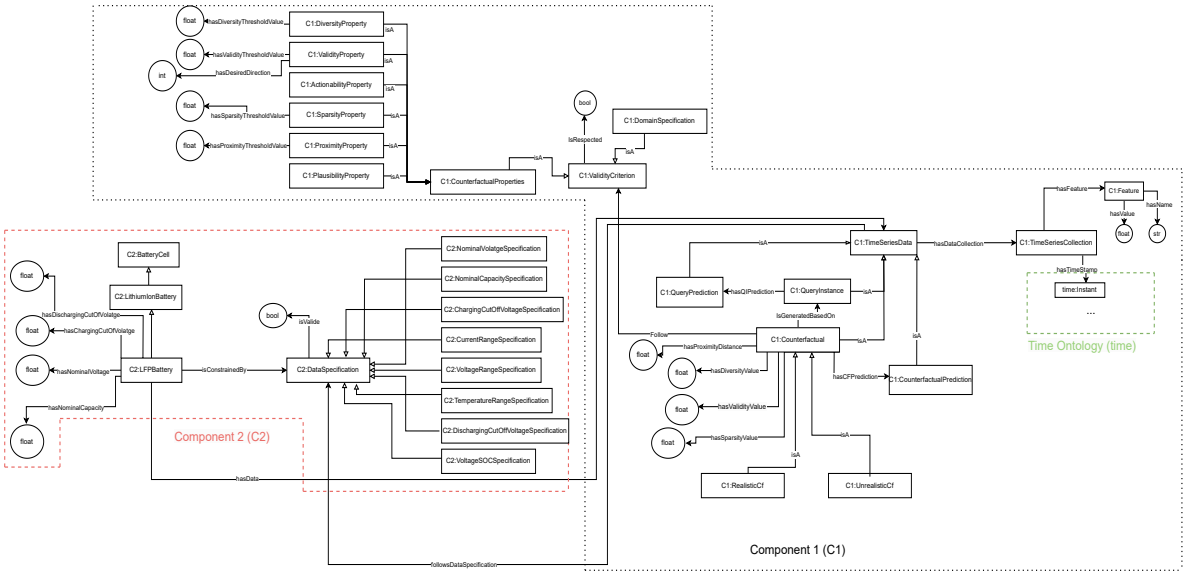

Figure 1: Ontology structure combining counterfactual and domain-specific knowledge.

```
LFPBattery(?bc)∧Counterfactual(?cf)
    ∧hasData(?bc,?cf)∧
    DomainSpecification(?ds)∧Follow
    (?cf,?ds)∧
    TemperatureRangeSpecification(?
    trsp)∧VoltageRangeSpecification
    (?vrsp)∧
    CurrentRangeSpecification(?crsp)
    ∧VoltageSOCSpecification(?vcs)∧
    followsDataSpecification(?cf,?
    vrsp)∧followsDataSpecification(?
    cf,?trsp)∧
    followsDataSpecification(?cf,?
    crsp)∧followsDataSpecification(?
    cf,?vcs)∧isValide(?vcs,true)∧
    isValide(?crsp,true)∧isValide(?
    vrsp,true)∧isValide(?trsp,true)
    →isRespected(?ds,true)
```

Rule 6: Ensures that the voltage remains within the specified range defined by the charging and discharging cut-off voltages. The same principle applies to current and temperature, ensuring that their values stay within the respective minimum and maximum limits.

```
LFPBattery(?bc)∧
    hasChargingCutOfVoltage(?bc,?ccv
    )∧hasDischargingCutOfVoltage(?bc
    ,?dcv)∧Counterfactual(?cf)∧
    hasData(?bc,?cf)∧
    VoltageRangeSpecification(?vrsp)
    ∧VoltageRangeSpecification(?
    vrspCf)∧isConstrainedBy(?bc,vrsp
    )∧followsDataSpecification(?cf,?
    vrspCf)∧TimeSeriesCollection(?c)
    ∧hasDataCollection(?cf,?c)∧
    hasFeature(?c,?f)∧hasName(?f,"
    Voltage")∧hasValue(?f,?voltage)∧
    lessThanOrEqual(?voltage,?ccv) ∧
     greaterThanOrEqual(?voltage, ?
    dcv)→isValide(?vrspCf,true)
```

This rule ensures that the SOC is consistent with the battery voltage:

- For low voltage ($V \leq 2.7$ V), SOC should be close to (0).
- For high voltage ($V \geq 3.5$ V), SOC should be close to (1).
- For intermediate voltage ($2.7 < V < 3.5$ V), SOC should lie within a moderate range ($0.2 \lesssim$ SOC $\lesssim 0.8$).

```
LFPBattery(?bc) ∧ Counterfactual(?
    cf) ∧ hasData(?bc, ?cf) ∧
    VoltageSOCSpecification(?vss) ∧
    VoltageSOCSpecification(?vssCf)
    ∧isConstrainedBy(?bc, ?vss) ∧
    followsDataSpecification(?cf, ?
    vssCf) ∧TimeSeriesCollection(?c1
    ) ∧ hasDataCollection(?cf, ?c1)
    ∧hasFeature(?c1, ?f1) ∧ hasName
    (?f1, "Voltage") ∧ hasValue(?f1,
     ?voltage) ∧hasCFPrediction(?cf,
     ?cfp) ∧
    CounterfactualPrediction(?cfp) ∧
    TimeSeriesCollection(?c2) ∧
    hasDataCollection(?cfp, ?c2) ∧
    hasFeature(?c2, ?f2) ∧ hasName(?
    f2, "SOC") ∧ hasValue(?f2, ?soc)
     ∧((lessThanOrEqual(?voltage,
    2.7) ∧ lessThanOrEqual(?soc,
    0.4)) ;(greaterThanOrEqual(?
```

```
    voltage, 3.5) ∧
    greaterThanOrEqual(?soc, 0.6))
    ;(greaterThan(?voltage, 2.7) ∧
    lessThan(?voltage, 3.5) ∧
    greaterThanOrEqual(?soc, 0.2) ∧
    lessThanOrEqual(?soc, 0.8)))→
    isValide(?vssCf, true).
```

5 EXPERIMENTS

To assess the performance of the proposed approach, we applied it to the use case of estimating the SOC for LFP cells, as presented in (Arbaoui et al., 2025). We adopted the same model and dataset, which consist of a simple 1D Convolutional Neural Network (CNN) followed by a 1D max-pooling layer and four dense layers. It is important to emphasize that the specific architecture of the predictive model is not critical to our method.

The model takes as input three time series signals: current (I), voltage (V), and temperature (T), each of length 100, and predicts the SOC for the next two minutes. We used the same query instances as in (Arbaoui et al., 2025), which correspond to four representative points: Early phase of charging (Q1), Mid charging phase (Q2), Near full charge (Q3) and Discharging with constant current (Q4). These query instances are sufficient to understand how the model works, as they cover all possible scenarios encountered in the SOC estimation task. For each experiment, we generated a predefined number of Cfs, starting from one and progressively increasing up to ten. The quality of the explanations was evaluated using five metrics defined in (Guidotti, 2022):

- *Dissimilarity*: Measures the average distance between Cfs and the query instance, as shown in Equation 5

$$dissimilarity(C) = \frac{1}{h}\sum_{i=1}^{h} proximity(x'_i) \qquad (5)$$

- *Implausibility*: Assesses the plausibility of the generated Cfs by computing the distance between each Cf and its nearest neighbor in X, as shown in Equation 6

$$implausibility(C) = \frac{1}{h}\sum_{i=1}^{h} \min_{x\in X} \|x'_i - x\|_2 \qquad (6)$$

- *Actionability*: Counts the number of Cfs that apply changes only to permitted features ($\mathcal{F}_{\text{perm}}$) specified by the user, as shown in Equation 7, where $\mathbf{fx}'_i$ represents the set of modified features in the i-th counterfactual instance, In our case, this metric was not applied since all features were considered changeable.

$$actionability(C) = \sum_{i=1}^{h} 1_{(\mathbf{fx}'_i \subseteq \mathcal{F}_{\text{perm}})} \qquad (7)$$

- *Instability*: Evaluates whether the explainer provides same Cfs (C and $\overline{C}$) when given two query instances (qr and $\overline{qr}$) that are close to each other and yield the same prediction, as shown in Equation 8

$$instablity(qr,\overline{qr},C,\overline{C}) = \frac{1}{1+\|qr-\overline{qr}\|_2}\sum_{x'\in C}\sum_{x''\in\overline{C}}\|x'-x''\|_2 \qquad (8)$$

In addition to these metrics, we also consider the *running time* to further assess and refine the proposed approach.

For each case, we computed the evaluation metrics. Figure 2 illustrates the results across the four query instances. In addition, we compare our approach with two dedicated methods, *GENO-TOPSIS* (Arbaoui et al., 2025) and *NSGA-II* (Arbaoui et al., 2025), as well as two adapted methods, CoMTE$_m$ (Ates et al., 2021) and AB-CF$_m$ (Li et al., 2023). The latter were originally designed for binary classification tasks and thus required modifications to their loss functions and stopping conditions in order to be applied to our regression setting. The evaluation was conducted using the first query instance (Q1) with $k = 10$. Results are reported in Table 1.

The choice of GENO-TOPSIS and NSGA-II is motivated by the fact that they are, to the best of our knowledge, the only existing approaches explicitly developed for generating counterfactual explanations in multivariate time series regression. Conversely, CoMTE$_m$ and AB-CF$_m$ were adapted from methods designed for classification tasks, which explains their performance differences.

Overall, our approach achieves the most balanced performance across all evaluation criteria, consistently outperforming competing methods in terms of *dissimilarity, implausibility, instability, and running time*. While AB-CF$_m$ and CoMTE$_m$ achieve relatively close results on some metrics, they fall short of *SOCEG*, particularly in plausibility and stability. This limitation is partly due to their original design, as both AB-CF$_m$ and CoMTE$_m$ were developed to generate only a single counterfactual. In contrast, *SOCEG* offers greater versatility by producing multiple high-quality counterfactuals, thereby ensuring both diversity and reliability in explanations.

The main advantage of our approach is that it does not require an initial population to generate counterfactuals, unlike the competing methods. Moreover,

Table 1: Comparison of counterfactual generation methods for Q1 with $k = 10$. The symbol ↓ indicates that smaller values are preferred, while ↑ indicates that larger values are preferred.

Evaluation Metric	*SOCEG*	GENO-TOPSIS	NSGA-II	CoMTE$_m$	AB-CF$_m$
Dissimilarity ($\times 10^{-3}$, ↓)	**0.18**	2.43	5.47	0.40	0.34
Implausibility ($\times 10^{-3}$, ↓)	**2.69**	1.25	3.61	6.40	3.42
Instability ($\times 10^{-3}$, ↓)	**2.96**	18.32	104.19	6.88	18.58
Running Time (s, ↓)	**45.04**	944.09	168.49	56.98	172.94

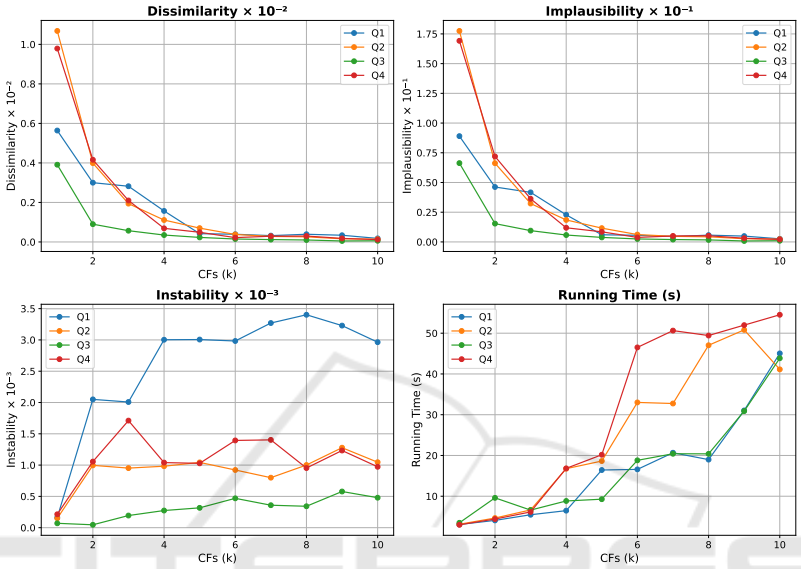

Figure 2: Evaluation of counterfactuals for different query instances.

it exhibits progressive improvement: after generating the first counterfactual, the *dissimilarity and implausibility* values decrease with each subsequent generation.

In terms of the information provided by the counterfactual explanations, our approach identifies both *which* features need to be altered and *how* to modify them in order to change the model's prediction to a desired outcome. In this use case, the results indicate that the model assigns greater importance to voltage when the current is stable, which occurs during Q2 and Q4. As expected, there is a strong correlation between voltage and SOC in these phases. The counterfactual explanations consistently suggest either increasing the voltage, as shown in Figures 4 and 5, to achieve higher SOC values, or decreasing it, as shown in Figures 6 and 7, to achieve lower SOC values.

It is also evident that the model infers the cell's charge level based on the beginning and end of the voltage time series, as modifications suggested by the counterfactuals predominantly occur during these periods. Additionally, the counterfactuals highlight the importance of temperature near 100% charge: they suggest reducing temperature while increasing current to lower the SOC, indicating that higher temperatures influence the model's predictions in this regime. Figure 3 presents the distribution of counterfactuals across all evaluated properties. As expected, the *sparsity* remains consistently equal to 3, since all features are considered and are correlated; the only variation lies in the magnitude of change applied to each feature.

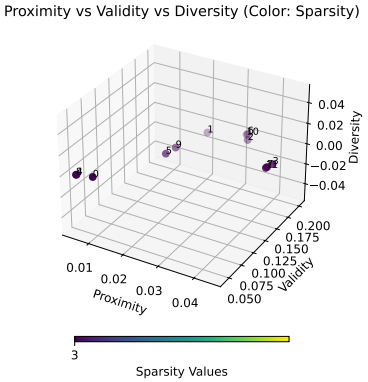

Figure 3: Distribution of generated counterfactuals across all properties based on query instance (Q1).

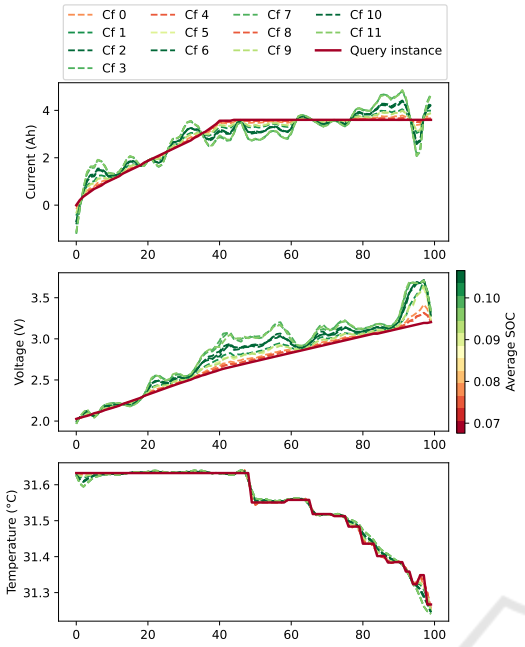

Figure 4: Comparison between a counterfactual data generated using our approach and the query instance (Q1) for the three feature.

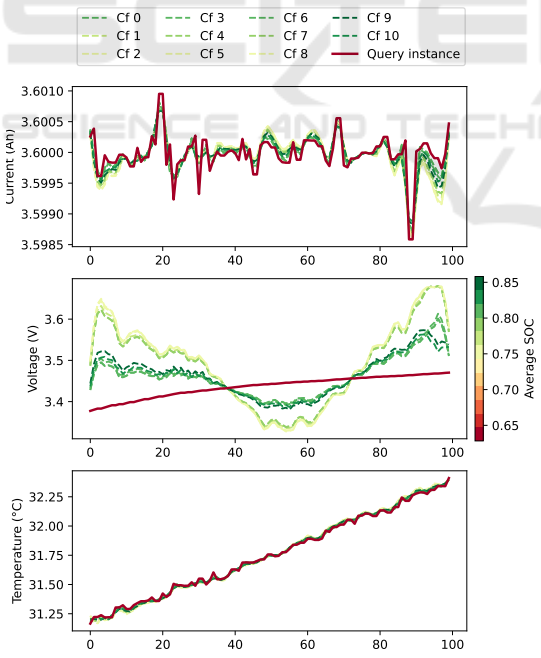

Figure 5: Comparison between a counterfactual data generated using our approach and the query instance (Q2) for the three feature.

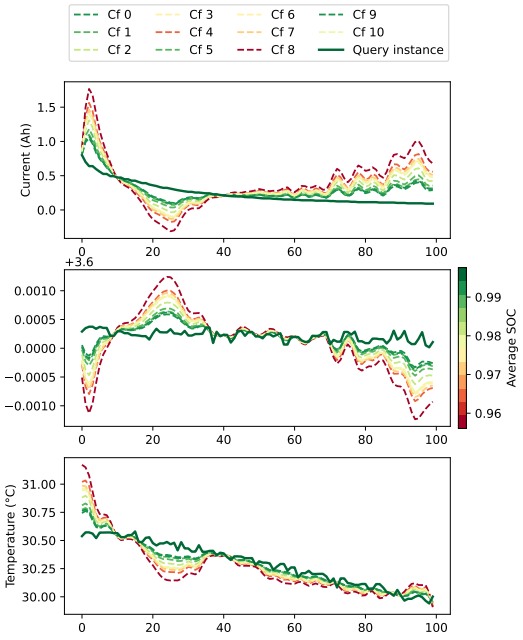

Figure 6: Comparison between a counterfactual data generated using our approach and the query instance (Q3) for the three feature.

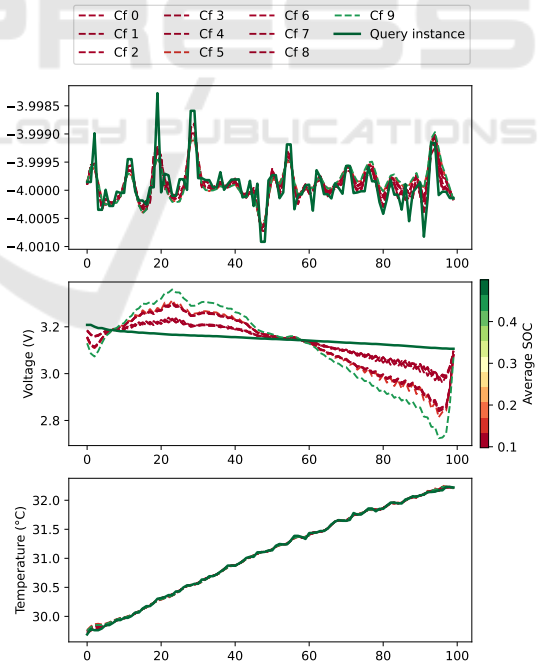

Figure 7: Comparison between a counterfactual data generated using our approach and the query instance (Q4) for the three feature.

6 CONCLUSION

In this paper, we proposed a method for generating counterfactual explanations for multivariate time series data in regression tasks. Our approach focuses on model sensitivity to noise and combines a gradient-based counterfactual candidate generator with an ontology-based validator to ensure that generated counterfactuals satisfy user-defined constraints while preserving essential properties. The generator selectively targets relevant data segments, adding controlled noise that respects feature correlations and variations, while the ontology-based validator guides generation using high-plausible examples.

We evaluated our method on the SOC estimation of LFP cells, comparing it with the only two existing methods designed for this task, GENO-TOPSIS and NSGA-II, using five metrics: *dissimilarity, implausibility, actionability, instability,* and *running time*. Experiments across four query instances, generating between 1 and 10 counterfactuals, further assessed the robustness of our approach. We also included comparisons with adapted methods, CoMTE$_m$ and AB-CF$_m$, originally developed for classification tasks. Results show that our approach consistently outperforms existing methods across all metrics while enhancing counterfactual quality in terms of *validity, proximity, sparsity, diversity,* and *plausibility*. The experiments show that the model captures the strong correlation between voltage and SOC, with Cfs suggesting voltage adjustments to achieve different SOC values. Temperature is influential near full charge, where lowering temperature while increasing current can reduce SOC predictions. The main contribution of this work is its adaptable framework, which can be extended to various use cases. The generation process in SOCEG is largely generalizable across tasks, but the ontology (particularly Component 2) and the rules enforcing domain constraints must be redefined for each domain. Future work will explore the generalizability of this approach across diverse tasks and datasets, including SOH estimation.

CODE AVAILABILITY

The source code used in this approach is available on GitHub at the following repository: SOCEG.

ACKNOWLEDGMENT

This research received partial funding from the French National Research Agency (ANR) under the project "ANR-22-CE92-0007-02". Additionally, support was provided by the European Union through the Horizon Europe program and the innovation program under "GAP-101103667".

REFERENCES

Battinfo. https://github.com/BIG-MAP/BattINFO. Accessed 26-September-2025.

(Accessed 19-September-2025). sprql. https://www.w3.org/TR/sparql11-query/.

Arbaoui, S., Ayadi, A., Samet, A., Mesbahi, T., and Boné, R. (2025). Validation ontologique des explications contrefactuelles pour les séries temporelles : Application aux batteries lithium-ion. *Revue des Nouvelles Technologies de l'Information*, Extraction et Gestion des Connaissances, RNTI-E-41:39–50.

Arbaoui, S., Samet, A., Ayadi, A., Mesbahi, T., and Boné, R. (2024). Data-driven strategy for state of health prediction and anomaly detection in lithium-ion batteries. *Energy and AI*, 17:100413.

Ates, E., Aksar, B., Leung, V. J., and Coskun, A. K. (2021). Counterfactual explanations for multivariate time series. In *2021 International Conference on Applied Artificial Intelligence (ICAPAI)*, pages 1–8.

Bakator, M. and Radosav, D. (2018). Deep learning and medical diagnosis: A review of literature. *Multimodal Technologies and Interaction*, 2(3).

Brughmans, D., Leyman, P., and Martens, D. (2022). Nice: An algorithm for nearest instance counterfactual explanations.

Cao, Q., Zanni-Merk, C., Samet, A., Reich, C., de Bertrand de Beuvron, F., Beckmann, A., and Giannetti, C. (2022). Kspmi: A knowledge-based system for predictive maintenance in industry 4.0. *Robotics and Computer-Integrated Manufacturing*, 74:102281.

Castanho, D., Guerreiro, M., Silva, L., Eckert, J., Antonini Alves, T., Tadano, Y. d. S., Stevan, S. L., Siqueira, H. V., and Corrêa, F. C. (2022). Method for soc estimation in lithium-ion batteries based on multiple linear regression and particle swarm optimization. *Energies*, 15(19).

Chen, O., Reid, J., and Meier, A. (2025). Explainable ai for battery degradation prediction in evs: Toward transparent energy forecasting. *Journal of Advances in Engineering and Technology*, 2(3).

Chennam, K., Swapna, M., Viswanadhula, U. M., Aluvalu, R., and Rao, K. (2022). *Black Box Models for eXplainable Artificial Intelligence*, pages 1–24. Springer International Publishing.

Delaney, E., Greene, D., and Keane, M. T. (2021). Instance-based counterfactual explanations for time series classification. In *International conference on case-based reasoning*, pages 32–47. Springer.

Dhurandhar, A., Chen, P.-Y., Luss, R., Tu, C.-C., Ting, P., Shanmugam, K., and Das, P. (2018). Explanations based on the missing: Towards contrastive explanations with pertinent negatives.

Dhurandhar, A., Pedapati, T., Balakrishnan, A., Chen, P.-Y., Shanmugam, K., and Puri, R. (2019). Model agnostic contrastive explanations for structured data.

Gomez, O., Holter, S., Yuan, J., and Bertini, E. (2020). Vice: Visual counterfactual explanations for machine learning models.

Guidotti, R. (2022). Counterfactual explanations and how to find them: literature review and benchmarking. *Data Mining and Knowledge Discovery*, 38:1–55.

Guidotti, R., Monreale, A., Matwin, S., and Pedreschi, D. (2020). Black box explanation by learning image exemplars in the latent feature space.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

Hidouri, A., Arbaoui, S., Samet, A., Ayadi, A., Mesbahi, T., Boné, R., and de Beuvron, F. d. B. (2024). Socxai: Leveraging cnn and shap analysis for battery soc estimation and anomaly detection. In *Computational Science – ICCS 2024: 24th International Conference, Malaga, Spain, July 2–4, 2024, Proceedings, Part VII*, page 177–191. Springer-Verlag.

Huang, X., Cooper, M. C., Morgado, A., Planes, J., and Marques-Silva, J. (2023). Feature necessity & relevancy in ml classifier explanations.

Huang, X. and Marques-Silva, J. (2024). On the failings of shapley values for explainability. *International Journal of Approximate Reasoning*, 171:109112. Synergies between Machine Learning and Reasoning.

Kok, I., Okay, F. Y., Muyanli, O., and Ozdemir, S. (2022). Explainable artificial intelligence (xai) for internet of things: A survey.

Li, P., Bahri, O., Boubrahimi, S. F., and Hamdi, S. M. (2023). Attention-based counterfactual explanation for multivariate time series. In Wrembel, R., Gamper, J., Kotsis, G., Tjoa, A. M., and Khalil, I., editors, *Big Data Analytics and Knowledge Discovery*, pages 287–293, Cham. Springer Nature Switzerland.

Lim, B. and Zohren, S. (2021). Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200209.

Lundberg, S. (2017). A unified approach to interpreting model predictions. *arXiv preprint arXiv:1705.07874*, page 4768–4777.

Mothilal, R. K., Sharma, A., and Tan, C. (2020a). Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, FAT* '20, page 607–617. ACM.

Mothilal, R. K., Sharma, A., and Tan, C. (2020b). Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*. ACM.

Ng, S. S., Xing, Y., and Tsui, K. L. (2014). A naive bayes model for robust remaining useful life prediction of lithium-ion battery. *Applied Energy*, 118:114–123.

Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., De Bie, T., and Flach, P. (2020). Face: Feasible and actionable counterfactual explanations. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, AIES '20, page 344–350. ACM.

Qian, K., Li, Y., Zou, Q., Cao, K., and Li, Z. (2025). Soh and rul estimation for lithium-ion batteries based on partial charging curve features. *Energies*, 18(13).

Qian, L., Xuan, L., and Chen, J. (2023). Battery soh estimation based on decision tree and improved support vector machine regression algorithm. *Frontiers in Energy Research*, 11:1218580.

Rahnama, A. H. A. and Boström, H. (2019). A study of data and label shift in the lime framework.

Ribeiro, M., Singh, S., and Guestrin, C. (2016). "why should I trust you?": Explaining the predictions of any classifier. In DeNero, J., Finlayson, M., and Reddy, S., editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 97–101, San Diego, California. Association for Computational Linguistics.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Saadallah, A., Jakobs, M., and Morik, K. (2021). Explainable online deep neural network selection using adaptive saliency maps for time series forecasting. In Oliver, N., Pérez-Cruz, F., Kramer, S., Read, J., and Lozano, J. A., editors, *Machine Learning and Knowledge Discovery in Databases. Research Track*, pages 404–420, Cham. Springer International Publishing.

Verma, S., Boonsanong, V., Hoang, M., Hines, K., Dickerson, J., and Shah, C. (2020). Counterfactual explanations and algorithmic recourses for machine learning: A review. *ACM Computing Surveys*.

Vermeire, T. and Martens, D. (2020). Explainable image classification with evidence counterfactual.

Wachter, S., Mittelstadt, B., and Russell, C. (2018). Explanations without opening the black box: Automated decisions and the gdp. *Harvard Journal of Law & Technology*, 31 (2):169–180.

Wang, Z., Miliou, I., Samsten, I., and Papapetrou, P. (2023). Counterfactual explanations for time series forecasting. In *2023 IEEE International Conference on Data Mining (ICDM)*, pages 1391–1396. IEEE.

Yan, L., Peng, J., Zhu, Z., Li, H., Huang, Z., Sauer, D. U., and Li, W. (2025). Data-driven modeling of open circuit voltage hysteresis for lifepo4 batteries with conditional generative adversarial network. *Energy and AI*, 20:100478.

Young, K., Booth, G., Simpson, B., Dutton, R., and Shrapnel, S. (2019). *Deep Neural Network or Dermatologist?*, page 48–55. Springer International Publishing.

Álvarez Antón, J., García Nieto, P., de Cos Juez, F., Sánchez Lasheras, F., González Vega, M., and Roqueñí Gutiérrez, M. (2013). Battery state-of-charge estimator using the svm technique. *Applied Mathematical Modelling*, 37(9):6244–6253.