

Actionable Contextual Explanations for Cyber-Physical Systems

1st Sanjiv S. Jha

*Institute of Computer Science
University of St.Gallen
St. Gallen, Switzerland
sanjiv.jha@unisg.ch*

2nd Simon Mayer

*Institute of Computer Science
University of St.Gallen
St. Gallen, Switzerland
simon.mayer@unisg.ch*

3rd Kimberly Garcia

*Institute of Computer Science
University of St.Gallen
St. Gallen, Switzerland
kimberly.garcia@unisg.ch*

Abstract—Over the past two decades, Cyber-Physical Systems (CPS) have emerged as critical components in various industries, integrating digital and physical elements to improve efficiency and automation, from smart manufacturing and autonomous vehicles to advanced healthcare devices. However, the increasing complexity of CPS and their deployment in highly dynamic contexts undermine user trust. This motivates the investigation of methods capable of generating explanations about the behavior of CPS. To this end, Explainable Artificial Intelligence (XAI) methodologies show potential. However, these approaches do not consider *contextual* variables that a CPS may be subjected to (e.g., temperature, humidity), and the provided explanations are typically not actionable. In this article, we propose an Actionable Contextual Explanation System (ACES) that considers such contextual influences. Based on a user query about a behavioral attribute of a CPS (for example, vibrations and speed), ACES creates contextual explanations for the behavior of such a CPS considering its context. To generate contextual explanations, ACES uses a context model to discover sensors and actuators in the physical environment of a CPS and obtains time-series data from these devices. It then cross-correlates these time-series logs with the user-specified behavioral attribute of the CPS. Finally, ACES employs a counterfactual explanation method and takes user feedback to identify causal relationships between the contextual variables and the behavior of the CPS. We demonstrate our approach with a synthetic use case; the favorable results obtained, motivate the future deployment of ACES in real-world scenarios.

Index Terms—explainability, cyber-physical systems, knowledge graph, reasoning, context, XAI

I. INTRODUCTION

Artificial intelligence (AI) and the Internet of Things (IoT) have demonstrated significant potential to be used in a variety of fields, including industry, healthcare, education, and retail. It is expected that, in the future, Cyber-Physical Systems (CPS) that integrate AI and are connected with IoT will assist people in automating many more complex and labor-intensive processes across domains [1]. Smart vehicles, smart building solutions, intelligent industrial edge devices, and medical CPS are some examples of this combination. CPS already exhibits complex behavior as multiple cyber and physical components are involved in performing tasks, and the complexity of CPS keeps increasing as more and more processes across domains are automated. The complexity is further amplified when the context of a CPS is not static and some aspects of this context impact its behavior. For example, contextual variables such as

time, location, vibration, and the tasks assigned, as well as a broad range of further environmental factors (temperature, humidity, etc.) may influence the behavior of a CPS.

The manufacturer of a CPS may account for some of these *contextual* variables that affect the system based on knowledge gathered over time – this information is often gained through simulations or prototypes of the CPS that are exposed to different environmental contexts. However, such CPS might be deployed in a context where the manufacturer might not have tested the system against the CPS’ *Contextual Influences* (CI)— that is, *changes in the values of contextual variables that could potentially affect the behavior of the CPS* [2]. A location change, for example, may expose a CPS to CI that the manufacturer did not consider; For instance, a smart vehicle might have been tested in a controlled laboratory environment (e.g., a test chamber¹) for previously identified contextual variables that act at a certain altitude and air pressure level. However, the vehicle may behave differently when the end user takes it to higher or lower altitudes or air pressure levels. without an explanation regarding the changed behavior of the CPS the end user might not trust the CPS [3]. We argue that explanations of the behavior of a CPS should account not only for internal factors (e.g., a CPS’ programming or internal sensors) but also for CI. We posit that due to the increasing availability of contextual sensor data that can be obtained from pervasively deployed, low-cost, (Internet-)connected sensors (for example, battery-free floating [4] and passive sensors [5]) it is today becoming possible for CPS to make use of this measurable (by sensors) contextual information to better understand their behavior.

To explain the behavior of a CPS considering its context, we have implemented and demonstrated an *Actionable Contextual Explanation System* (ACES). ACES aims to improve the explainability of CPS by considering the context in which they operate. ACES provides a model-agnostic approach to explainability, which can be applied to a wide range of CPS regardless of the underlying model or algorithm used. ACES facilitates considering measurable CI to explain the CI of a CPS using a location-based context model and the

¹See <https://weiss-na.com/industry-applications/automotive/> for examples.

logs of contextual variables and the CPS' operations. ACES' contextual explanations are formulated through the integration of counterfactual explanations with standard semantic interface descriptions – concretely, W3C WoT Thing Description (TD)² – of the physical and virtual entities in the CPS' context. Moreover, different types of users might require different specifications of the explanations. Consider CPS' Expert users who know the CPS and its workings, and CPS Owners, who are aware of the context in which the CPS is deployed, but not of the internal workings of the CPS. These user types have distinct expectations of the explanations that should be provided about the behavior of a CPS: CPS Experts may have questions specific to some of the contextual changes that might not have been accounted for by the manufacturer in their limited tests (e.g., done in their test chambers), whereas CPS Owners may specifically be interested in understanding how this context might be adapted to optimize the operation of the CPS (e.g., by changing the ambient humidity). The main contributions of this article are:

- (i) *A Contextual Explanation System*: Conceptualization and implementation of a novel contextual explanation system for CPS—ACES (see Section III)—that can explain the behavior of a CPS while discovering and taking into account information about the CPS' context.
- (ii) *A Framework for Drop-in Contextual Explanation Generation*: Through ACES' context discovery mechanism (detailed in Section III-A) and its model-agnostic nature, the system becomes a *drop-in* (or plug and play style) explanation system for discovering CI and relating them to the behavior of a CPS given that a contextual model is available and that time-series data about the CI can be obtained.
- (iii) *A Knowledge Discovery and Actionable Explanation Module*: The contextual explanations that ACES provides about the behavior of a CPS are directly tied to controls that the user can use to immediately reconfigure the context of the CPS, by actuating on connected devices that expose standardized descriptions of their programming interfaces. Finally, ACES stores the derived explanations in a machine-readable format so that they can be reused to hint at the probable CI in similar events in the future. Here, generated explanations are enriched with user feedback – a positive or negative rating – which permits evaluating ACES explanations' effectiveness and over time increases user trust.

The rest of this article is structured as follows: in Section II, we discuss the background and related work. Next, in Section III we detail the architecture and implementation of ACES and clarify each step that ACES takes to deliver contextual explanations to users. In Section IV, we describe the deployment of ACES in a laboratory scenario and evaluate it in this context. Finally, in Section V, we discuss the effectiveness—how much an explanation aids users in making good decisions [6]—and actionability capabilities—

the capacity of generated explanations to assist users in taking appropriate action [7]—of ACES as an explanation tool.

II. RELATED WORK

In this section, we look into recent research on understanding and explaining the behavior of CPS and the approaches used to model the context of a CPS. Here, we are not only interested in context modeling per se but also in the strategies that are used to present the computed explanations to human users. This is relevant, since communication strategies contribute to the human users' understanding of the explanations and eventually to the trust in the CPS.

A. Contextual Modeling

Understanding how context affects a CPS and using that knowledge to explain a CPS's behavior is not an unusual problem. For instance, in industries such as manufacturing and automotive, *Test Chambers* are employed to better understand the effects of contextual variables on CPS, for example, the vehicular reverberation chamber used for cars in [8]. Using test chambers, it is possible to correlate behavioral changes in machinery when it is subjected to simulated contextual variations (e.g., temperature or humidity differences). However, the concrete physical implementation of a test chamber influences the results of the experiments: For instance, the position of sensors in a test chamber can influence the sensors' responses, which is critical in correlation-based observations [9]. Therefore, for every new experiment and machinery, customization of the test chamber is necessary, which might make usage of test chambers specific to individual systems. As a result of their high resource demand, cost of maintenance, and low flexibility, small and medium-sized enterprises are typically reluctant to employ test chambers. As another option to complement rigid and costly test chambers for explaining the behavior of a CPS in its context, we lean towards the recently presented strategies for model-based and model-free data analysis methods for inference and automated reasoning, including feature extraction with limit checking, clustering, classification, and knowledge-based methods [10], [11].

To derive contextual explanations about a CPS, an explanation system first needs to access sources of information about the CPS context. To model a (system's) context, *Location* plays a central role since physical interactions are highly dependent on the physical vicinity of entities [12] – this is especially true for CPS and environmental factors that might affect them. Hence, for explaining the behavior of a CPS based on CI, it is particularly relevant that the sensors are located near to supply the explanation system with relevant input data. Regarding context modeling, ontologies are increasingly employed in several projects [13], [14].

B. Explainable Artificial Intelligence Methods

Towards achieving better explainability across CPS application domains, several XAI approaches are employed in current projects [15]. Existing methods such as *Local Interpretable Model-agnostic Explanations* (LIME) [16], *SHapley Additive exPlanations* (SHAP) [17], *Deep Learning Important Features*

²<https://www.w3.org/TR/wot-thing-description/>

(DeepLift) [18], *Anchors* [19] derive explanations from input features. These methods are collectively referred to as *intrinsic* explanation methods [20]. When applied to CPS, these intrinsic XAI methods are useful to explain the internal decisions of a CPS, but they do not account for external influences on the system's behavior; hence, we argue that intrinsic XAI methods are not *effective* to provide explanations about complex CPS that are influenced by their context. In the proposed solution to create contextual explanations of the behavior of the CPS, context modeling using co-located sensors is proposed to add contextualization and actionability, unlike the existing intrinsic explanation methods.

C. Expert Knowledge and Explanation Interpretation

The reasoning about and determination of possible CI often relies on expert knowledge that can not be transferred across domains and that is certainly not effective in unfamiliar contexts. Recent research on explainable CPS proposes using *expert* knowledge to explain the effects of contextual changes – for instance, the effect of weather on an energy demand-response system [21]. This is because human experts often make good guesses regarding the causes of observed behavioral irregularities in a particular context due to their experience from the (implicit or explicit) analysis of large amounts of information generated by a CPS and its context. However, finding actionable explanations for unexpected behaviors in an unfamiliar context remains a challenge even for experts.

Expert knowledge is often implicit and experiential, and therefore hard to access in a more scalable way, it might furthermore be biased by the expert's mental model of the present conditions [22], and may not apply to unknown scenarios. Experts might be supported with tools that simplify observing a CPS and, specifically, interactions between its components. One such system is [23], where a magic lens system for displaying interactions between multiple linked devices through augmented reality is demonstrated. This supports users in monitoring the system, but the interpretation of the visualization is left to the human, which could lead to misunderstandings by users despite the causal ordering of interactions that are imposed by the underlying monitoring system [24]. To reduce misunderstandings of the explanations provided by an explanation system, a contextualized counterfactual (*what-if-not*) explanation of the system may support users' interpretations.

Counterfactual-based explanation methods are used in current research [25], [26] to offer causal explanations considering possible world scenarios. As a result, a counterfactual explanation may help users see the influence that the change in the value of an input feature or set of input features has on the final output of a machine-learning model. Hence, using counterfactual methods for discovering the relationship between a feature of the CPS and the system's behavior in a dynamic context could help explain the cause of CPS behavior in a specific context. Multiple algorithms are available to find the best-fitting counterfactual from the pool of possibilities for changing the decision of a machine learn-

ing model; these include the *Diverse set of Counterfactual Explanations* (DiCE) [27] and *Nearest Instance Counterfactual Explanations* (NICE) [28]. Even though counterfactual explanations are deemed to provide higher reasoning ability, one of their main drawbacks is that such methods tend to provide *multiple* counterfactuals to explain the behavior of a system. Several possibilities may confuse users and decrease their trust in the explanations. To overcome this, the so-called Rashomon Effect [29] proposes a game theory-based approach for assisting users in selecting the best explanation. In [29] users play a Rashomon Game by weighing the costs and benefits of changing suggested feature values. This game can help evaluate how practical a generated counterfactual explanation is for changing the underlying model's decision using the users' domain knowledge in the proposed contextual explanation method.

III. SYSTEM OVERVIEW

In this section, we introduce the architecture and implementation of ACES which derives actionable contextual explanations about a target CPS. In response to a user query for an explanation, ACES discovers sensors in the context of a target CPS (Section III-A). Thereafter, it accesses sensor data to compute correlations (Section III-B) and, based on a surrogate model (Section III-C), it creates counterfactual explanations of the behavior of the CPS (Section III-D). Derived explanations are stored in a knowledge base (Section III-E) to enable ACES to recall past responses to user queries and keep user feedback. In the following, we elaborate on these main components of the proposed system and show how they work together.

A. Context Discovery

To answer user queries regarding explanations of the behavior of a CPS, ACES' *Context Discovery* component identifies resources (e.g., sensors) that permit the system to access data about contextual variables that might affect the behavior of the CPS. To do this, ACES relies on the interfaces of these resources being described using a standardized Interface Description Language (IDL). While ACES is not in principle constrained to using a specific IDL, our prototype assumes these interfaces use the W3C WoT Thing Descriptions (TD). TDs, like other IDLs, provide a uniform representation of a device API. Thus, TDs abstract away from the concrete underlying communication protocol to uniformly describe interactions with devices connected via a variety of protocols. By integrating with TD, the ACES system hence becomes interoperable with a wide range of powerful (via HTTP(s)) as well as constrained devices (via CoAP(s)), and TD furthermore supports MQTT as well as industrial communication standards such as OPC Unified Architecture (OPC UA).

ACES discovers IDLs based on a semantically described context model of the CPS' environment. ACES explores this environment using a URI of the context model and a set of semantic predicates given by the user. Since this resembles the bootstrapping of semantic search engines, we call this procedure *seeding* [30]. A predicate is a component of the triple (subject, predicate, and object) that defines a

Listing 1: SPARQL query to discover entities that are co-located with a CPS and their W3C WoT Thing Descriptions

```

1 PREFIX ex:<http://.../livingcampus#>
2 PREFIX brick:<https://brickschema.org/.../Brick#>
3 SELECT ?TD
4 WHERE {
5   ex:targetCPS brick:hasLocation ?location .
6   ?thing brick:hasLocation ?location;
7           ex:hasTD ?TD .
8 }

```

logical statement in semantic technologies, e.g. `ex:Luna foaf:knows ex:Carol`.

Specifically, when the TD of an entity that can provide information about the CI of a CPS (e.g., a sensor) has been discovered, ACES automatically uses the entity's API to obtain log data. Moreover, the context model allows ACES to keep functioning in dynamic environments, in which sensors can be added or removed. Through this seeding process, ACES remains applicable to a wide range of scenarios: The system may be seeded with a context model expressed in any ontology. For instance, to discover TDs of entities that are *spatially co-located with the target CPS*, ACES may be seeded with the predicate `brick:hasLocation`, which is a property from the BRICK Ontology³. This property expresses that the subject is physically located in the location given by the object. Hence, it can be used to find co-located entities. Another possible seed is the property `ex:hasTD`, which is introduced in this work to link the representation of an entity (subject) to its TD (object).

Given a context model, a semantic query formulated in SPARQL is used to discover available TDs in the surroundings of the target CPS; a concrete query for the previous example is given in Listing 1. Considering the relevant ontologies used by the context model (Lines 1-2), this query returns all the URIs of the TDs (Line 7) that describe the entities co-located (Lines 5-6) with the target CPS.

B. Cross Correlation

ACES explanation process is triggered by a user query that specifies a behavioral attribute of the CPS (e.g., the speed at which a robotic arm moves). Given a contextual model, ACES retrieves the TDs of contextual entities that might affect the behavioral attribute of the CPS. ACES uses these TDs to collect log data from the discovered entities using the interaction methods (e.g., the HTTP protocol's GET method) specified in their TDs. ACES then checks for changes in the behavioral attribute of the CPS that coincide with the changes in the CI. To this end, ACES utilizes Pearson coefficients to determine highly correlated CI. Pearson coefficients are effective techniques for computing cross-correlation in time-series applications. They help lower the number of CI by removing non-correlated continuous CI based on a threshold range, for example, +1 to -1, where + shows a positive correlation and - is a negative correlation. ACES preserves positive and negative correlations and discards all other CI with a 0 (no correlation) Pearson coefficient.

³<https://brickschema.org/ontology>

C. Surrogate Model

ACES next uses a surrogate model to simulate the behavior of a CPS in its context using highly correlated features between the CPS and its contextual variables. It is useful to understand how the queried attribute of a CPS is affected by the combined or individual effect of the input characteristics created using the log of a CPS's operations and its context at a certain time. To do this, we employ the random forest (RF) binary classification model [31]. Since RF uses the majority of decisions made by all the trees for higher accuracy in the model, the risk of over-fitting is lower. However, a higher number of decision trees increases the time taken to train the model. This supervised learning-based binary classification model uses labeled sensor data to create a classification function. There are various methods for labeling continuous data, such as sliding windows, trailing mean, etc. However, ACES currently depends on the users to have accurate labels for supervised classification. Users provide the class labels for the behavior of the CPS to be explained (for example, providing binary class labels like the low or high vibration level of a vehicle). An example of labeling is shown in our case study in Section IV.

D. Counterfactual Explanation

Finally, ACES uses a counterfactual explanation method to locate causal influences using this surrogate model. ACES uses the DiCE method [27] to find relationships between CI and the behavior of a CPS, as simulated by the surrogate model. The binary classification function of the surrogate model helps create *possible* multiple counterfactuals for the user's query using DiCE. To choose a feasible counterfactual value (an example, updating the temperature of the room that would not affect the working of other machines in the same workspace), users are expected to reflect on the following costs (which have been customized for CPS but are based on [29]) using their knowledge of the CPS and its context:

- *The difficulty of changing a feature*: Consider the possibility and easiness of changing a feature (e.g., weather conditions), where some features may even be immutable.
- *Willingness to change a feature*: Users might be reluctant to change the value of certain features. To define such willingness, users can make use of the following subcategories:
 - *The magnitude of the value change of a feature*: To what extent can the value of a feature be changed to change the behavior of the CPS?
 - *Interdependence of the feature with other CPS*: How does the value of a feature of a CPS depend on the value of features of other CPS in the proximity?
- *The amount of time to change the features*: How long does it take to change the feature (e.g., if a third-party service is needed to change the feature)? Additionally, if other features have already deviated from their current values by the time the feature is changed, it might fail to achieve the desired effect on the CPS behavior.

The evaluation of different possible CI by users serves

Listing 2: Update the context model with the discovered contextual influence and explanations

```

1 PREFIX brick:<https://brickschema.org/.../Brick#>
2 PREFIX prov:<http://www.w3.org/ns/prov#>
3 PREFIX ex:<http://.../livingcampus#>
4 INSERT{
5   ?ml a brick:Temperature ;
6   a prov:Entity .
7   ex:example_lamp_1 brick:hasTag ?t1 .
8   ?t1 a ex:brightness_level ;
9   a prov:Entity ;
10  prov:wasInfluencedBy ?ml ;
11  prov:qualifiedInfluence [
12    a prov:EntityInfluence ;
13    prov:influencer ?ml ;
14    ex:influenceType "positiveInfluence" ;
15    ex:rating 1 ] .
16 }
17 WHERE {
18   SELECT ?ml ?t1
19   WHERE {
20     BIND( IRI( CONCAT( "https://things.example#context"
21                       , strUUID() ) ) as ?ml ) .
22     BIND( IRI( CONCAT( "https://things.example#context"
23                       , strUUID() ) ) as ?t1 ) .
24   }
25 }

```

ACES towards being able to produce *actionable* explanations. For example, a behavioral attribute of a CPS (e.g., the speed of a robot) might exhibit correlations with both, *outside humidity* as well as *inside humidity*. A user would evaluate that actuating upon inside humidity is the only viable option.

E. Update Knowledge Graph

The relationships learned through counterfactual explanations in a new scenario are used to enrich a knowledge base that tracks discovered CI on attributes of a CPS to generate explanations in the future, given a similar context. ACES uses the insert SPARQL statement in Listing 2, which uses the PROV-O⁴ ontology to specify the influenced entity (ex:brightness_level) and the relationship between the influencer (brick:Temperature) and the influenced entity using the prov:qualifiedInfluence predicate, which defines:

- The *relationship* between the behavior of the CPS and the CI. It is stored as a semantic triple i.e. (subject, predicate, and object). For example, lines 5 – 15 in Listing 2 show that the brightness level of the ex:example_lamp_1 was influenced by Temperature. Here, the subject is a behavioral attribute of a CPS, and the object is its CI.
- The *type of influence* (see line 14 in Listing 2) is seen as either positive or negative. The influence is positive if there is a direct proportionality between the value of a contextual influence and the value of the behavioral attribute of the CPS, and the indirect proportionality is stored as a negative influence.
- To gradually improve the contextual explanation's credibility and trustworthiness using user feedback (line 15 in Listing 2), the KG stores the explanations and the users' binary ratings (based on the results of the actions taken).

⁴<https://www.w3.org/TR/prov-o/>

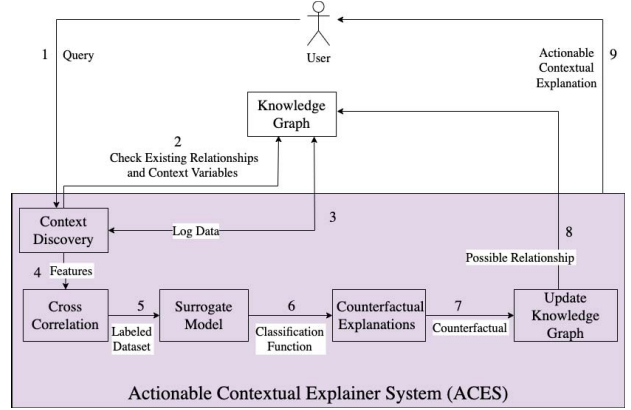


Fig. 1: Based on queries submitted by users, ACES generates contextual explanations.

These binary values give the average user rating of the gathered CI which is expressed as the credibility of the explanation according to the user's feedback.

F. The Data Flow of an ACES Prototype

We summarize the operation of ACES, using Figure 1 to illustrate the data flows in the system:

- 1) ACES receives a user query composed of the name of the target CPS (for example, ex:example_lamp_1), and a behavioral attribute (for example, ex:brightness_level).
- 2) Given a user query specifying the target CPS, a behavioral attribute to be explained, and a seed for exploring the context model; the *Context Discovery* component discovers entities in the CPS' context and gathers their TDs. Furthermore, ACES checks the knowledge base for already existing information about discovered CI on the CPS and displays these relationships to the user. Here the user either accepts the explanation about the discovered CI as an explanation or ACES looks for other CI.
- 3) If no such relationships are found in the knowledge base, the system starts to generate other possible explanations. For this, using the TDs, ACES collects log data of the CPS and its context. In case the context and CPS log data are unavailable, ACES waits for at least 1000 entries while collecting the data itself.
- 4) Thereafter, ACES sends the collected data to the *Cross Correlation* component, to select several related continuous context data with the CPS's behavioral attribute; this dataset is then labeled by the user (for example, light brightness level class (High/Low) in the given example case study detailed in Section IV).
- 5) Using the labeled dataset and the correlated features from the previous step, a surrogate model is created to generate a binary classification function.
- 6) Given the classification function and the user query, ACES creates diverse counterfactuals using DiCE. These counterfactuals help to find possible relationships between the behavior of the CPS and its context. Since there can be multiple possible counterfactuals to explain the

behavior of the CPS, ACES asks the user to help select a feasible counterfactual explanation by calculating the cost of selecting a counterfactual explanation as mentioned in Section III-D.

- 7) ACES then checks the user-selected counterfactual for the type of influence a contextual variable (one of the features used to generate the counterfactuals) has over the behavioral attribute of the CPS. Then, the user is prompted to provide binary feedback (agree or not) for the discovered relationship between the CPS and its context.
- 8) This feedback and a SPARQL Query that represents it (based on this counterfactual-based correlation) are preserved in the knowledge base to be used in the future for increasing the rating of an explanation of a similar CPS in a similar context.
- 9) Finally, a text-based contextual explanation is offered to the user.

IV. CASE STUDY: BRIGHTNESS LEVEL OF A SMART LAMP

We illustrate ACES's capabilities through a smart lamp use case. Recently, smart building solutions have focused on using contextual variables for adjusting the brightness and color of the luminaries to promote energy savings, comfort for human users' eyes, etc. Such smart lighting installations get enhanced with additional capabilities, often in combination with machine learning, which increases their behavioral complexity. We have therefore selected such a smart lighting system for an initial demonstration of the ACES system. While realistic, this use case is simple enough to demonstrate ACES' capabilities. We deployed the ACES prototype in a controlled setup to explain the behavior of a smart lamp in our office. This space is equipped with an existing sensor network that allows us to access contextual variables, whose interfaces are described in TDs. In our scenario, we have set up a smart lamp that has been programmed to increase or decrease its brightness based on the current temperature of the space.

We logged the power status and brightness level of the smart lamp controlled through a dimmer switch. For collecting the context data of the space, we deployed Thunderboard sensor kits⁵ that log temperature, ambient light, and humidity. Apart from the sensors installed in the office space, we also collected external (weather) data. A user (in this scenario, the CPS owner) unfamiliar with the working of the smart lamp, might not understand why the brightness of this lamp changes at what might look like arbitrary times. Thus, the user asks ACES to explain the variations in the brightness level and to provide an actionable explanation for the behavior of the smart light. For this, ACES needs the following data:

- The local name (or URI) of the target CPS; in this case, this is the lamp called `example_lamp_1`.
- The name of the behavioral attribute (i.e., feature) to be explained about the target CPS; in this case, the `Light Brightness Level`.

⁵<https://www.silabs.com/development-tools/thunderboard/thunderboard-g22-kit>

- The URI of the space's knowledge graph (context) that contains the TDs of the contextual variables, in this case, `http://.../livingcampus#`.
- A seed predicate to navigate the semantically described context model of the space and to access the log data of relevant contextual entities, in this case, `brick:hasLocation`.

Based on the given input, ACES produces a list of the possible CI it has discovered in the past for the behavior of the CPS mentioned in the query of the user. ACES first lists stored CI for the CPS's behavioral attribute, then seeds the context model to find the contextual entities in the location of the target CPS and access their data using its *Context Discovery* component. ACES then generates a data set of correlated contextual variables with the specified behavioral attribute of the smart light (i.e., the light brightness level in this case) using its Cross-Correlation component. It next trains its surrogate model where we set a threshold to label the brightness level of the smart lamp as the lowest or highest brightness. For this scenario, we configure the brightness level as *low* if it is lower than 0.4, and otherwise as *high*. Using the labeled data (we recorded all the data for six months and labeled the feature – brightness level of the smart lamp as *Light Brightness Class*), in our scenario the surrogate model exhibits an accuracy of 98.93% (with 20 trees), on a maximum number of splits in decision trees of 5. The resulting surrogate model predicts the light brightness level as *Low* (in line 1 of Listing 3) when the temperature of the space was 17 degrees Celsius.

ACES offers a variety of counterfactual explanations then using the DiCE algorithm (two examples are shown in Table I for a given query) to explain how the brightness level of the smart light changes when changing the contextual variables. For instance, in the first counterfactual explanation in Table I, to change the light brightness level of the smart light to 0.7, users must increase the room temperature to 22.6 degrees Celsius. Similarly, if the user does not want to (the willingness cost in Section III-D) change the room temperature but has access to increase the humidity, they could select the 3rd counterfactual and increase the value by 16.6 (i.e., 60.5-43.1). This would potentially change the brightness level of the CPS if there is a causal relationship (for the given setup it is not true). Accordingly, users estimate the costs and benefits (i.e., difficulty, willingness, and time sensitivity) of changing a contextual variable based on a counterfactual explanation and their knowledge of the context and the smart lamp. Listing 3 shows that the user chose the 2nd counterfactual (line 3) given their willingness and the short time it takes to change the value of the room temperature compared to the other contextual variables in their context. For the given scenario it is also true and there is a real causal relationship. With the feedback from the CPS owner's domain knowledge and the chosen counterfactual, ACES treats the low room temperature as one of the CI for the low light brightness levels in the scope of the given data. ACES learns that updating the room temperature to achieve a light brightness level of 0.7 (high) could be an effective and actionable explanation for query 1

TABLE I: Examples of query instances and counterfactual explanations to switch the expected behavior of the smart lamp. For simplification, only perturbed values are shown in the table.

1	Light Brightness Level	External Temperature	External Light Level	Room Temperature	Room Humidity	Light Brightness Class
Query (brightness level < 0.4)	-	0.0	6548.48	19.0	43.1	Low
Counterfactuals (brightness level >= 0.4)	1.0	-	258.60	-	-	High
	0.7	-	-	22.6	-	High
	0.4	-	-	-	60.5	High

2	Light Brightness Level	External Temperature	External Light Level	Room Temperature	Room Humidity	Light Brightness Class
Query (brightness level > 0.4)	-	21.0	4321.0	25.8	45.0	High
Counterfactuals (brightness level <= 0.4)	0.3	-	71873.3	-	-	Low
	0.1	-	-	17.8	-	Low
	0.2	-	48762.0	-	52.0	Low

Listing 3: A simplified version of the Contextual explanation from ACES explaining the room temperature is directly proportional to the Light Brightness Level of the smart lamp

```

1 Light Brightness: Low (the binary classification
  prediction of the surrogate model for the query
  instance)
2 Surrogate model accuracy: 98.93%
3 Select a counterfactual: 2 (index of the counterfactual,
  Light brightness level 0.7)
4 Contextual Explanation:
5 Room_Temperature has positiveInfluence on light
  brightness Level of example_lamp_1
6 The preserved influences between ex:example_lamp_1
  and brick:Temperature have been reviewed 16
  times by the users as positiveInfluence
7 Average ratings for the preserved contextual
  influence:
8 { 'positiveInfluence': 0.75, 'negativeInfluence':
  0.0 }
9 Please provide your feedback for future reference by the users:
10 Do you agree with the found relationship? yes/no: yes

```

shown in Table I. ACES now provides a contextual explanation to the user, which is shown in Listing 3. In its explanation (see lines 4 to 8), ACES describes the type of relationship between the contextual variable (Room_Temperature) and the Light Brightness Level of the CPS (example_lamp_1). It shows for the given query that the relationship is a *positiveInfluence* as the temperature has to be increased to raise the brightness level of the smart lamp. These entities are then linked (shown with underlines) with the set of related TDs to take respective actions as per the counterfactual values. To substantiate the current finding of ACES, it presents the number of times this relationship has been discovered and presented to the users and their average rating (in terms of agreement or disagreement with the presented explanation) on line 10. Hence, it verifies that the ACES is capable of detecting CI on the behavior of a CPS – in this case, the programmed behavior of a smart lamp.

V. DISCUSSION

We assess ACES for generating contextual explanations on common metrics used for evaluating explanation systems, some of which are inspired by [27]. The findings based on the metrics are as follows:

Validity represents the fraction of output that constitutes the actual counterfactuals (brightness level greater or equal to the threshold 0.4). Our randomly selected examples of counterfactuals (examples in Table I) created using DiCE and

using the user’s help in selecting a counterfactual, ACES showed 100% valid contextual explanation for the given use case.

Diversity is the difference between the two counterfactuals that the system produced. It is calculated as the average difference between two (or many) counterfactuals’ perturbed values. ACES produced a diverse set of counterfactuals to explain the behavior of a CPS. This diversity could help users choose a feasible counterfactual explanation for making one actionable contextual explanation.

Proximity and Sparsity: With the definition from [27], we distinguish sparsity from proximity. Sparsity is the number of features that should be changed to achieve a desired behavior of the CPS. ACES currently uses only continuous features for generating the counterfactuals. Hence for proximity, we only look for continuous feature proximity evaluation, which is the mean of the feature-wise distance between the counterfactual and the input data. In the case of explaining CPS using contextual variables, it is desirable to change a smaller number of features to update the context of the CPS. However, due to the existing confounders in the dataset (for example, external light level), the average change between the input data and the counterfactual was high in this case. For example, if a user wants to change multiple context variables to update the context of the CPS, it might not be feasible, given their costs. However, a single contextual variable that does not affect the working of other systems may be possible to change significantly. Furthermore, ACES delivered a single CI – Room_Temperature with a significant value change for updating the context of CPS. Thus, ACES showed low sparsity and high proximity in the discussed use case.

Actionability is the ability to take immediate action (to update the context of the CPS) based on an explanation. ACES in this scenario helps users to take actions (when an actual causal relationship is present). Following the given explanation, users could find an actuator that changes the room temperature (e.g., a thermostat) to increase or decrease the room temperature (context) for changing the light brightness (behavioral attribute of the smart light).

VI. LIMITATIONS AND FUTURE WORK

ACES lacks a causal model to strengthen the assertions that a CPS (the smart light) and its contextual influence (the room temperature) are causally related in reality. ACES, at present,

does not account for any potential confounding factors that are not identified by its context discovery component based on the supplied context model but that may still affect the behavior of the target CPS. Furthermore, based on the results from the synthetic use case, the effectiveness and feasibility of the explanations cannot be generalized to more complex scenarios.

In order to assess the effectiveness of the explanations that ACES generates, real-world scenarios will be tested. For this, we could use a similar method to [32] for approximating the human user's understanding of the explanation with the simplicity of another machine learning model such as the 1-NN classifier.

VII. CONCLUSIONS

In this article, we presented ACES, a contextual explanation system that uses time-series log data generated by a CPS and by entities in its context, and a counterfactual explanation method to derive explanations about the behavior of the system. Unlike test chambers and other context-effect explanation methods, ACES is a cost-effective and scalable solution for creating explanations that account for the context of a CPS. It is thus positioned as an extension of conventional explanation approaches that consider only the internal workings of a CPS. Through deploying and testing ACES using a simple smart light scenario, we show that ACES may thus be helpful in discovering known or unknown contextual influences and help users anticipate the effects that changes in the context might have on a CPS. This can help improve the user's contextual understanding of the space by relating contextual influences to the behavior of the CPS, thereby reinforcing a more trustworthy relationship between users and their CPS.

REFERENCES

- [1] P. Radanliev, D. De Roure, M. Van Kleek, O. Santos, and U. Ani, "Artificial intelligence in cyber physical systems," 2021, <https://doi.org/10.1007/s00146-020-01049-0>.
- [2] S. Jha, S. Mayer, and K. García, "Poster: Towards explaining the effects of contextual influences on cyber-physical systems," 2021, <https://doi.org/10.1145/3494322.3494359>.
- [3] B. Y. Lim, A. K. Dey, and D. Avrahami, "Why and why not explanations improve the intelligibility of context-aware intelligent systems," in *Proceedings of the SIGCHI conference on human factors in computing systems*, 2009, <https://doi.org/10.1145/1518701.1519023>.
- [4] V. Iyer, H. Gaensbauer, T. L. Daniel, and S. Gollakota, "Wind dispersal of battery-free wireless devices," 2022, <https://doi.org/10.1038/s41586-021-04363-9>.
- [5] R. Bhattacharyya, C. Floerkemeier, and S. Sarma, "Rfid tag antenna based sensing: Does your beverage glass need a refill?" 2010, <https://doi.org/10.1109/RFID.2010.5467235>.
- [6] N. Tintarev and J. Masthoff, "Evaluating the effectiveness of explanations for recommender systems," 2012, <https://doi.org/10.1007/s11257-011-9117-5>.
- [7] B. Murray, D. T. Anderson, and T. C. Havens, "Actionable xai for the fuzzy integral," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2021, <https://doi.org/10.1109/FUZZ45933.2021.9494563>.
- [8] P.-S. Kildal, A. A. Glazunov, J. Carlsson, and A. Majidzadeh, "Cost-effective measurement setups for testing wireless communication to vehicles in reverberation chambers and anechoic chambers," in *2014 IEEE Conference on Antenna Measurements & Applications (CAMA)*. IEEE, 2014, pp. 1-4.
- [9] L. Parellada-Monreal, G. Mandayo, A. Martín-Mayor, I. Souto-Canteli, and M. M. Bou-Ali, "Influence of the test-chamber shape on the performance of conductometric gas sensors," 2022, <https://doi.org/10.1016/j.snb.2022.131694>.
- [10] F. Lopez, M. Saez, Y. Shao, E. C. Balta, J. Moyne, Z. M. Mao, K. Barton, and D. Tilbury, "Categorization of anomalies in smart manufacturing systems to support the selection of detection mechanisms," 2017, <https://doi.org/10.1109/LRA.2017.2714135>.
- [11] M. Balduccini, E. Griffor, M. Huth, C. Vishik, M. Burns, and D. Wollman, "Ontology-based reasoning about the trustworthiness of cyber-physical systems," 2018, <https://doi.org/10.1049/cp.2018.0012>.
- [12] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," 2013, <https://doi.org/10.1109/SURV.2013.042313.00197>.
- [13] L. Petnga and M. Austin, "Ontologies of time and time-based reasoning for mbse of cyber-physical systems," 2013, <https://doi.org/10.1016/j.procs.2013.01.042>.
- [14] M. Aziez, S. Benharzallah, and H. Bennoui, "An ontology based context model for the discovery of iot services in the internet of things," 2017, <https://doi.org/110.1109/MATHIT.2017.8259719>.
- [15] E. Daglarli, "Explainable artificial intelligence (xai) approaches and deep meta-learning models for cyber-physical systems," 2021, <http://dx.doi.org/10.4018/978-1-6684-3666-0.ch015>.
- [16] M. T. R. Singh, and C. Guestrin, "'why should I trust you?': Explaining the predictions of any classifier," 2016, <https://doi.org/10.1145/2939672.2939778>.
- [17] S. M. Lundberg and S. Lee, "A unified approach to interpreting model predictions," 2017. [Online]. Available: <http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf>
- [18] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," 2017.
- [19] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," 2018, <https://doi.org/10.1609/aaai.v32i1.11491>.
- [20] T. Weber and S. Wermter, "Integrating intrinsic and extrinsic explainability: The relevance of understanding neural networks for human-robot interaction," 2020, <https://doi.org/10.48550/arXiv.2010.04602>.
- [21] P. R. Aryan, F. J. Ekaputra, M. Sabou, D. Hauer, R. Mosshammer, A. Einfalt, T. Miksa, and A. Rauber, "Explainable cyber-physical energy systems based on knowledge graph," 2021, <https://doi.org/10.1145/3470481.3472704>.
- [22] I. E. Dror, V. Pascual-Leone, Aand Ramachandran *et al.*, "The paradox of human expertise: why experts get it wrong," 2011.
- [23] S. Mayer, Y. N. Hassan, and G. Sörös, "A magic lens for revealing device interactions in smart environments," in *SIGGRAPH Asia 2014 Mobile Graphics and Interactive Applications*, 2014, <https://doi.org/10.1145/2669062.2669077>.
- [24] S. Mayer, C. Beckel, B. Scheidegger, C. Barthels, and G. Sörös, "Demo: Uncovering device whispers in smart homes," 2012, <https://doi.org/10.1145/2406367.2406435>.
- [25] S. Mueller, A. Li, and J. Pearl, "Causes of effects: Learning individual responses from population data," 2021, <https://doi.org/10.48550/arXiv.2104.13730>.
- [26] C. Wang, X. Li, H. Han, S. Wang, L. Wang, C. C. Cao, and L. Chen, "Counterfactual explanations in explainable ai: A tutorial," 2021, <https://doi.org/10.1145/3447548.3470797>.
- [27] R. K. Mothilal, A. Sharma, and C. Tan, "Explaining machine learning classifiers through diverse counterfactual explanations," 2020, <https://doi.org/10.1145/3351095.3372850>.
- [28] D. Brughmans and D. Martens, "Nice: an algorithm for nearest instance counterfactual explanations," *arXiv preprint arXiv:2104.07411*, 2021, <https://doi.org/10.48550/arXiv.2104.07411>.
- [29] M. G. M. M. Hasan and D. Talbert, "Mitigating the rashomon effect in counterfactual explanation: A game-theoretic approach," in *The International FLAIRS Conference Proceedings*, 2022, <https://doi.org/10.32473/flairs.v35i.130711>.
- [30] A. Ciortea, S. Mayer, S. Bienz, F. Gandon, and O. Corby, "Autonomous search in a social and ubiquitous web," <http://www.alexandria.unisg.ch/260427/>.
- [31] M. Pal, "Random forest classifier for remote sensing classification," 2005, <https://doi.org/10.1080/01431160412331269698>.
- [32] U. Ehsan and M. O. Riedl, "Explainability pitfalls: Beyond dark patterns in explainable ai," <https://doi.org/10.48550/arXiv.2109.12480>.