IoT Anomaly Detection Via Device Interaction Graph: Technical Report

1 Evaluation

In this section, we post our evaluation result on the CASAS testbed.

1.1 Experimental Setup

CASAS collected 42,388 device events during a 30-day living experience, and the device information is shown in Table 1. One can check that most devices are presence sensors, and there is one contact sensor installed on the front door. Similar with the ContextAct testbed, there is no automation rules installed. As a result, we first generate three automation rules: (1) If M002 is activated, unlock D002; (2) If M004 is activated, unlock D002, and (3) If M011 is deactivated, unlock D002. By doing so, 15,911 events which simulate the automation executions are generated. We further preprocess the 58,299 events and partition the generated time series. Eventually, 80% of them are used as the training data (for DIG construction), while 20% are used for testing data (anomaly detection).

For the ground truth construction, since most devices are presence sensors, we mainly check the user movement traces during the 30 days. Then we identify the neighboring sensors, and construct the ground-truth interactions correspondingly. As a result, we identify 54 ground-truth interactions, which record user movement traces.

1.2 Interaction Mining Evaluation

With the training data, CausalIoT initiate the TemporalPC algorithm with $\tau = 2$ and $\alpha = 0.001$. The result shows that it identifies 53 ground-truth interactions with 100% precision and 98.2% recall. In particular, for all the installed three automation rules, CasualIoT successfully identifies them. Moreover, CausalIoT successfully rejects 10 spurious interactions, and all of them stem from the intermediate factor. For example, it rejects the M011 $\rightarrow$ M003, since M004 is an intermediate node in the path, i.e., M011 $\rightarrow$ M004 $\rightarrow$ M003. For the missing interaction D002 $\rightarrow$ M006, it is due to the low frequency. That is, users seldom move along this path, and as a result, CausalIoT cannot identify the association relationships between the two device states.

Table 1: Overview of device information

Abbr.	**Attributes**	**# devices (CASAS)**	**# devices (ContextAct)**	**Value type**	**Description**
M	Presence Sensor	7	5	Discrete	Movement detection
D	Contact Sensor	1	2	Discrete	Door/window state

1.3 Anomaly Detection Evaluation

With the inferred interactions, we initiate the contextual anomaly detection evaluation. Recall that in our paper, we simulate 4 malicious cases for contextual anomaly. Unfortunately, since the CASAS testbed contains limited types of devices, we can only simulate case 2 (Burglar Intrusion) for evaluation purpose. Specifically, we inject 4,000 spoofed events of the presence and contact sensors, and use the precision and the recall as the metric. The evaluation results show that CausalIoT achieves a high detection accuracy with 94.2% precision and 98.8% recall. We also compare with the three baselines. In particular, the precision for Markov/OCSVM/HAWatcher is 35.2%/24.5%/19.7%, and the recall is 54.3%/77.4%/32.9%. One can check that compared with the three baselines, our CausalIoT achieves the best performance.

Finally, we initiate the collective anomaly detection evaluation. Similar with the simulation of contextual anomalies, we only simulate case 1 (Burglar Wandering) for collective anomaly generation. Specifically, we first randomly select 1,000 positions for injection of spoofed contact/presence sensor events, which simulate the case of burglar intrusion. Then we propagate them based on the ground truth, and eventually add 1,000 movement traces to the testing data. The length of these traces is bounded by the parameter $k_{\max}$, and it ranges from 2 to 4. Evaluation results show that when $k_{\max}$ ranges from 2 to 4, the average length of injected anomalies is 2/2.476/2.979. CausalIoT can successfully detect 98.6%, 98.2%, 94.4% anomaly chains. For all the detected chains, CausalIoT can also fully reconstruct them.

2 Discussion

While the evaluation result shows that CausalIoT outperforms the existing work, we consider it the first step towards interaction-aware anomaly detection at smart homes. As a result, we list some limitations that we plan to address.
Unmeasured confounder. In Section VI, we highlight that some unmeasured factors (e.g., the sunrise) come from the environment. These factors result in spurious interactions, which eventually reduce the detection accuracy and interpretation capability. The best solution is collecting environmental information about smart homes (e.g., the weather). Since CausalIoT can be easily extended, we can introduce additional nodes that represent the environmental factors, and study their relationship with the device. By doing so, the generated device interaction graph will encode more fruitful information, and the

inferred interactions among devices will be more accurate.
Stationary assumption. While TemporalPC achieves good detection accuracy, it is initiated in an offline way. The inferred device interaction graph only describes the interactions during the collection period, and the stationary assumption is needed for applying the graph to handle the runtime device event. As a result, it is susceptible to the scenario where the interaction graph changes. For example, there may be behavioral deviations, or users may add/remove a device. While CausalIoT can use a new training dataset collected within a short period and efficiently re-train the interaction graph, other alternatives are worth exploiting, e.g., runtime inference of the interaction graph.
Multi-user scenarios. Finally, our work focuses on the interaction graph construction in a single-user setting. However, when extending to the multi-user scenario, the graph construction can be complicated as different users may have different preferences about smart home usage. Even worse, some preferences may be conflicted (e.g., user A deploys an automation rule "Turn on the light when the TV is on", while user B deploys a conflicted rule "Turn off the light when the TV is on"). Some auxiliary information (e.g., temporal information) may help to distinguish the device usage for different users and assist in the interaction mining for each user.