Zero-shot Fault Detection for Manipulators through Bayesian Inverse Reinforcement Learning

Hanqing Zhao[1], Xue Liu[1], Gregory Dudek[1]

***Abstract*— We consider the detection of faults in robotic manipulators, with particular emphasis on faults that have not been observed or identified in advance, which naturally includes those that occur very infrequently. Recent studies indicate that the reward function obtained through Inverse Reinforcement Learning (IRL) can help detect anomalies caused by faults in a control system (i.e. fault detection). Current IRL methods for fault detection, however, either use a linear reward representation or require extensive sampling from the environment to estimate the policy, rendering them inappropriate for safety-critical situations where sampling of failure observations via fault injection can be expensive and dangerous.**

To address this issue, this paper proposes a zero-shot and exogenous fault detector based on an approximate variational reward imitation learning (AVRIL) structure. The fault detector recovers a reward signal as a function of externally observable information to describe the normal operation, which can then be used to detect anomalies caused by faults. Our method incorporates expert knowledge through a customizable reward prior distribution, allowing the fault detector to learn the reward solely from normal operation samples, without the need for a simulator or costly interactions with the environment. We evaluate our approach for exogenous partial fault detection in multi-stage robotic manipulator tasks, comparing it with several baseline methods. The results demonstrate that our method more effectively identifies unseen faults even when they occur within just three controller time steps.

I. Introduction

Detecting and diagnosing faults is a fundamental requirement for robustness in robotic systems. Faults are unexpected changes in system function which hamper or disturb the normal operation, and they often cause unacceptable deterioration in performance [1]. Timely and accurate fault detection remains, however, a challenging task in the robotics context for two main reasons. Firstly, the complexity and volume of sequential data generated by robots require a data-driven fault detection model with a strong expressive power (e.g. recurrent neural networks). Secondly, in safety-critical environments, insufficient knowledge about the ramifications of faults and the costly or even unfeasible collection of fault samples necessitates a fault detection algorithm that can detect anomalies caused by previously unseen faults. In fact, some faults that occur in practice may never have been observed until they happen in the robot, yet they should still be detected.

A fundamental question in fault detection is how to understand and model the "normal operation" of the robotic system. Recent research suggests that all behaviours of intelligent agents can potentially be viewed as processes in which the agent seeks to maximize a reward function [2]. If a robot has faults that cause abnormal operations, it could not only fail to complete its task but also present a safety risk to its environment and human operators, thereby deviating from its intended objective. From this perspective, fault detection can be seen as an examination of the deviation of the actual behaviour of the robot from its intention (i.e. maximizing a reward signal), within a given task objective.

As such, a decrease in this reward can indicate a deviation from the normal operation of a manipulation task, assuming that the task environment and the robot's intention remain unchanged. Based on this idea, reinforcement learning (RL) has been shown to be an effective approach for detecting faults if the reward function is known. Most RL-based fault detection methods require, however, a simulator with an explicit dynamic model that has been constructed from previously observed data and a handcrafted reward function [3], [4]. As a result, they may not be suitable for detecting previously unseen faults, or may not work when the explicit dynamic model and reward function is unknown. Especially when the fault detector has a different observation space than the robot controller, learning explicit system dynamics or handcrafting a reward function based on externally observable information is usually non-trivial to overcome.

Inverse reinforcement learning (IRL) provides a way to infer a reward function from observational data that implicitly represents the intention of the intelligent agent. Most recent IRL algorithms are based on the maximum entropy principle [5], which computes a reward distribution that maximizes the entropy among all distributions that achieve at least the same total reward. Additionally, IRL approaches based on deep neural networks can learn to represent complex reward functions. The reward function recovered by IRL can be used to measure whether the current observation is consistent with the agent's intention (i.e. normal operation). A decrease in reward can thus indicate the existence of anomalies, offering a new approach to anomaly detection.

Recent advances in anomaly detection have leveraged the properties of IRL to enable data-driven anomaly detection without requiring knowledge of a predefined reward function that describes the agent's intention. While some research has been conducted in this field, such approaches often require intensive interaction with the environment to estimate the policy [6] or use a simple linear function to represent the reward [7]. As a result, such methods are hard to use in safety-critical scenarios where exemplars of faults are challenging or impossible to induce and observe.

For the deployment of IRL methods in anomaly detection without the presence of fault samples, we aim for the reward

[1] School of Computer Science, McGill University, Montréal, Canada

function to depict not just the task's progress, but also the difference between normal and faulty operations. In many industrial settings, during repetitive maneuvers such as device installation or object movement, a robotic manipulator may perform several stages of non-linear movements (e.g. installing a part onto a platform after moving it), while accessing only a limited set of its many possible configurations, leading to sparse sampling of configuration space. Therefore, achieving this goal is challenging without incorporating additional information, such as expert knowledge, into the IRL-based fault detection model.

In response to these challenges, we propose a zero-shot IRL method to detect anomalies in robot manipulation tasks caused by faults, without the need for faulty data in the training set or the explicit system dynamics model. The fault detector is based on what we refer to as the Approximate Variational Reward Imitation Learning (AVRIL) [8] structure, which approximates reward and Q-value distributions through variational inference. This makes it a completely off-policy IRL algorithm, as it doesn't require sampling for policy evaluation nor a solution to a forward RL problem. For a specific manipulation task, our fault detector can be trained offline using only the data observed during normal operation, while leveraging the helpful features of IRL to enhance fault detection.

The AVRIL algorithm assumes that the reward is a latent variable associated with the observed behaviour and that the reward distribution during normal operation can be inferred from a pre-defined reward prior distribution, taking into account the observations obtained from normal operations. By using a standard Gaussian distribution as the reward prior, the sparsity of the reward can be regularized, which helps to stabilize the reward signal and avoid over-fitting to the policy [9]. A Gaussian prior may, however, not always be the best option to serve as the reward prior that describes normal operations, especially for fault detection in repeat maneuvers of robotic manipulators.

In this study, we explore the use of a reward prior represented by a Beta distribution. This distribution adjusts the magnitude and sparsity of reward towards a predefined level within fixed bounds, which helps the AVRIL fault detector capture the characteristics of a repetitive maneuver performed by a robotic manipulator. Furthermore, we also adopt variance regularization in the reward space to regularize the variability of the expected reward posterior over a mini-batch of samples towards a set value, in order to further prevent over-fitting.

We validate our framework by conducting exogenous online fault detection in two simulated scenarios. In each scenario, a Panda robotic manipulator, controlled by a Soft Actor-Critic (SAC) [10] reinforcement learning controller, attempts to accomplish a manipulation task, (i.e. door opening or block lifting), while two types of faults are injected into different numbers and positions of the robotic manipulator's joints. The fault detector itself has no prior knowledge (i.e. no training data) related to these faults. The purpose of the fault detector is to detect the presence of faults within a short time window following their injection, to prevent catastrophic damage if the fault persists.

This paper gives four contributions:

1) A zero-shot and exogenous fault detection framework based on the AVRIL structure, which recovers a reward function that describes the agent's intention under normal operation as a function of externally observable information, where the decrease in the reward enables timely detection of anomalies caused by faults.
2) A method to incorporate expert knowledge about the characteristics of the manipulation task into the fault detector, this is achieved by selecting a Beta-family prior and variance regularization target value over the reward space, so as to detect unseen fault based solely on the information observed from normal operations.
3) A solution using the proposed framework, for exogenous partial fault detection in the door opening and block lifting tasks. The trained fault detector effectively distinguishes various unseen faults in different tasks by giving different reward outputs.
4) A quantitative evaluation of different fault detectors, with respect to their effectiveness and speed in unseen fault detection.

II. Related Work

Fault detection techniques aim to detect whether a fault has occurred when the robot undergoes an abnormal operation [11]. Although some faults that significantly disturb the behaviour of the robot can be easily detected (e.g., a robot stops working), most abnormalities in robot operations are due to partial faults [12].

Partial faults in robotic manipulators do not result in a complete failure of the system (e.g. the entire manipulator stops working), but rather cause the system to operate at a degraded level (e.g. one joint becoming unresponsive). These partial faults may initially result in only slight deviations from the robot's intended normal operation. Therefore, task-specific approaches that employ dedicated models designed for the task at hand are typically necessary to detect partial faults [13].

Moreover, Fault detection approaches in robotics can be distinguished into two types: *endogenous* and *exogenous*. Endogenous fault detection relies on a centralized fault detector that is integrated into the robot's control system, while exogenous fault detection uses a detector that is independent of the robot and observes the robot from the outside, which allows for the detection of a wider range of faults, especially partial faults [14].

Most endogenous and some exogenous fault detection approaches rely on monitoring the robot's internal variables (e.g. motor torque and circuit current). However, this typically requires additional sensing and communication modules to reliably report these variables [15], [16], as well as the need for an explicit dynamics model of the manipulator based on these internal variables [17]–[19]. Incorporating these components raises the level of complexity within the robot system and the fault detector is compelled to presume

that these components will remain fault-free, which is not always feasible in real-world applications. To address these challenges, Christensen et al. [20] designed an exogenous fault detection protocol for multi-robot systems that detect faults in other robots from externally observable information, without relying on internal sensors. That approach, however, requires the robots to repeatedly perform a predefined behavioural protocol created for the diagnosis of specific faults, and as a result, it is not suitable for detecting unseen faults.

Data-driven machine learning approaches are sometimes used in fault detection, especially for scenarios with multi-dimensional observation space and difficult-to-solve dynamics models [21]. While most data-driven approaches are designed for supervised learning with known (expected) faults [22], some recent work has considered detecting unseen faults using zero-shot methods (i.e. samples indicative of faults are absent or only partially present in the training set). Zero-shot learning approaches either apply transfer learning techniques to adapt a model learned on a different yet sufficiently similar system [23], or integrate expert knowledge by selecting a set of attributes from an interpretable attribute space [24]–[26]. In these approaches, selecting the set of attributes is important for the performance of the fault detector and it is not a straightforward task, particularly for multi-stage maneuvers. This is because motion characteristics are often difficult to describe through interpretable attributes (e.g. semantic words), and the description can vary between different stages (e.g., the robot moves "straight forward" in the first stage and then "rotates" in the next stage).

Our framework can be seen as a zero-shot and exogenous approach to detecting anomalies caused by faults. Unlike other approaches for reward inference from observations (e.g. normalizing flows [27]), our approach, based on an AVRIL structure, employs a customizable reward prior. This serves as a tool for integrating expert knowledge of normal operational characteristics through a prior distribution in a one-dimensional reward space, thus enabling zero-shot learning only from normal operation samples.

While our approach is not very sensitive to the choice of the two parameters for the reward prior distribution within the predefined surrogate distribution family, and we believe it can be readily to be adjusted automatically, in this work, we simply delegate it to a human expert's assessment of task observations. Compared to other zero-shot learning approaches that involve attribute selection from a high-dimensional interpretable space [28], our framework significantly reduces the number of parameters that experts need to determine.

III. AVRIL Fault Detection Framework

We model the exogenous fault detection process as a Partially Observable Markov Decision Process (POMDP) [29], consisting of a 7-tuple $(\mathbb{S}, \mathbb{A}, \mathbb{T}, R, \Omega, O, \gamma)$. The POMDP includes robot states $s \in \mathbb{S}$, robot actions $a \in \mathbb{A}$, state transitions $\mathbb{T}$, a reward function R, an observation space for the exogenous fault detector Ω, an observation model O that

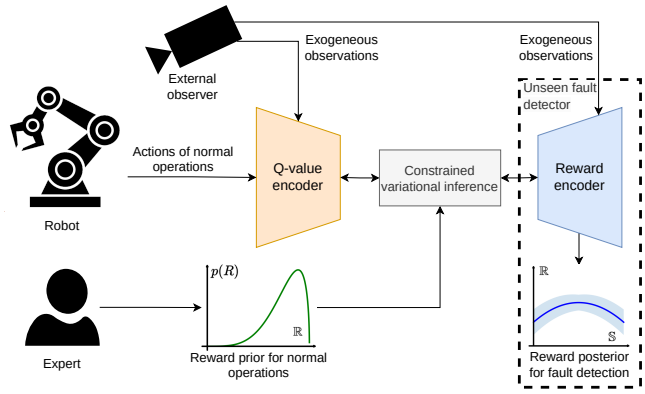

Fig. 1. Illustration of the AVRIL fault detection framework, in which the model is trained from exogenous observations from an external sensor, and actions taken during normal operation. Once trained, the framework detects previously unseen faults in an exogenous manner, using the reward encoder that takes exogenous observations as input. Expert knowledge about the manipulation task's characteristics is implicitly conveyed through a reward prior distribution.

defines the relationship between o and the internal state s, and a discount factor $\gamma \in (0, 1]$. We assume that during normal operation, the manipulator follows an underlying policy π_n, and the fault detector receives observation $o \in \Omega$ from an external sensor. The fault detector is trained from a dataset of normal operation D, which consists of pairs of adjacent observations and actions $(o_t, a_t, o_{t+1}, a_{t+1}), t \in \mathbb{N}$, where a is sampled from $\pi_n(s)$.

Unlike many IRL approaches [30], [31], the AVRIL fault detector learns a reward function that describes the agent's intention solely from normal operation. AVRIL considers the reward as a latent representation of the action, allowing the use of variational inference to solve the Q function and the reward in the same loop. Specifically, AVRIL aims to minimize the KL divergence between a surrogate distribution over reward q_ϕ and the posterior distribution of the reward given data D, as shown below:

$$min_{q_\phi \in \mathcal{Q}} \{D_{KL}(q_\phi(R)||p(R|D))\}\,, \tag{1}$$

where $\mathcal{Q}$ denotes the family of surrogate distributions, and ϕ represents the parameters of the reward encoder network. In our implementation, we restrict the reward space to be $\mathbb{R} \cap [0, 1]$, and we set $\mathcal{Q}$ to be the Beta distribution family $Beta(a, b)$ satisfies that $a \leq b \leq 1$. In our experiments, the parameters a, b are manually selected. Our result is not significantly affected by these parameters, but its detailed optimization is outside the scope of this paper. The reward encoder generates a posterior reward distribution within $\mathcal{Q}$ based on the reward prior and exogenous observation in Ω.

Similarly, the Q-function encoder maps observations and actions to Q values, and its parameters are denoted as θ. In this variational inference setting, we optimize the reward and Q-value encoders using the Evidence Lower Bound (ELBO) objective:

$$F_{ELBO} = \sum_{(o,a)\in D} \frac{exp(\psi Q_\theta(o,a))}{\sum_{b\in\mathbb{A}} exp(\psi Q_\theta(o,b))} - D_{KL}(q_\phi(R(o))|p(R)) \,, \tag{2}$$

in the ELBO objective function, we assume that the normal operation policy π_n is Boltzmann-rational over Q values [32]. Subsequently, the conditional probability of the robot's behavioural data D, given the reward signal $p(D|R)$ can be approximated via a Boltzmann distribution over Q values. Where ψ is the inverse temperature in the Boltzmann distribution formula. Moreover, $p(R)$ denotes the reward prior, and the choice of $p(R)$ and the surrogate distribution family $\mathcal{Q}$ reflects the expert's understanding of the characteristics inherent to the normal operation.

In addition, to jointly update reward and Q-value encoders, AVRIL introduces a consistency constraint in addition to the ELBO objective function. This objective couples the output of the two encoders by maximizing the likelihood of the temporal difference reward given the reward posterior. Rewriting the constraint under KTT conditions with a Lagrange multiplier λ, the consistency constraint brings an additional term to the objective function:

$$F_{consistancy} = \sum_{(o,a,o',a')\in D} \lambda log(q_\phi(Q_\theta(o,a) - \gamma Q_\theta(o',a'))) \,. \tag{3}$$

Recent developments in deep generative models [33], [34] suggest that incorporating constraints on the generator and discriminator's output variance can enhance the model's generalization ability. In AVRIL, the Q-values estimated by the Q-function encoder are evaluated under the reward posterior distribution from the reward encoder, making it a generative model in the reward space. From this perspective, to enhance the generalizability of the retrieved reward signal across different unseen faults while avoiding overfitting the data from normal operations, we suggest an additional regularization objective. This objective aimed at modulating the expected value of the estimated reward posterior over the normal operation dataset D, to align it more closely with a predefined target variance v:

$$F_{regularization} = -(Var_{o\in D}[\mathbb{E}_{q_\phi}(R(o))] - v)^2 \,, \tag{4}$$

where the target variance v represents the expert's comprehension of the difference among attained observations across different stages of normal operation. In all experiments introduced in this paper, we set $v = 0.01$. Finally, by combining all three aforementioned objectives, we obtain an objective for the AVRIL fault detector as follows:

$$F = F_{ELBO} + F_{consistancy} + F_{regularization} \,. \tag{5}$$

Fig. 1 provides a general overview of the framework. From a graphical model standpoint, the relationships between variables in our approach can be summarized as shown in Fig. 2, where the dotted line signifies parameterized neural network representations of the conditional probability of the internal state given the external observation.

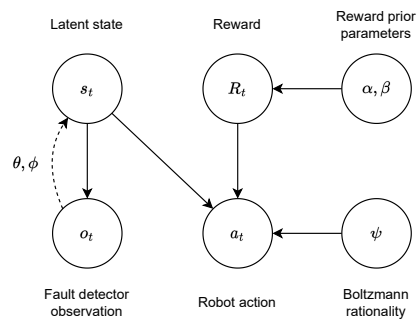

Fig. 2. A graphical model representation of the AVRIL exogenous fault detector, for a time step t, the reward R_t is considered as a latent variable that drives the behaviour of the robot a_t. θ, ϕ represent the parameters of the Q value and reward distribution encoders respectively. It is assumed that both encoders include a parameterized representation of an inverse observation model, which translates external observations o_t into the robot's internal state s_t (represented by the dotted line).

IV. Experimental Studies

A. Problem Statement and Fault Injection

To evaluate the effectiveness and limitations of our proposed approach, we concentrate on the timely detection of partial faults in manipulation tasks. Our experimental scheme is inspired by the fact that in industrial settings, robotic manipulators often perform the same routine operations for a given task. When a partial fault occurs, the state of the robot may not be drastically altered immediately, but over time, the robot may complete the task with less efficiency/accuracy, and in certain cases, a partial fault can even lead to disastrous consequences. As illustrated in the first part of Fig. 3, where a robotic manipulator is shown performing a door opening task, one of its joints receives zero action since time step 3 due to a stuck-at-zero partial fault. At time step 6, despite the presence of the fault for three time steps, there is no visible difference between the robot's pose and the pose it would have in normal operation. By the end of the episode, however, the robotic manipulator with the fault is stuck in the door handle, leading to a catastrophic failure in accomplishing the task's objective.

To tackle the challenges associated with real-time exogenous partial fault detection, our focus is on partial fault detection within three controller time steps in simulated manipulation environments, as shown in Fig. 3. During normal operation, the robotic manipulator is controlled by a SAC controller, which repeatedly executes a maneuver, such as opening a door or lifting a block. The SAC controller makes proprioceptive observations of all robot joints and manipulated objects as input and generates the desired end effector pose. An Operation Space Control (OSC) solver [35] then translates the desired end effector position pose into the torque command for each joint, which is finally sent to the robot manipulator as the action. As depicted in Fig. 1, the AVRIL fault detector is trained on the readings from

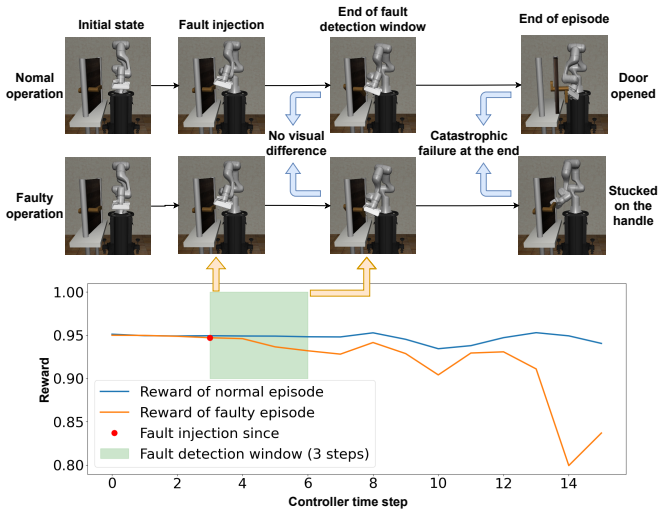

Fig. 3. An example of a normal and a faulty episode of the door opening task, and reward signal estimated by the AVRIL fault detector over the course of each episode. In the faulty episode, a stuck-at-zero partial fault is injected into one of the joints since time step 3.

external sensors (e.g., RGB-D camera or an IMU strapped to the manipulator) and the discretized actions recorded during normal operations. Precisely, the action of each joint a is discretized according to its rotational torque direction $a \in \{\text{counterclockwise}, \text{no torque}, \text{clockwise}\}$. After training, the AVRIL fault detector detects the presence of faults in an exogenous manner—i.e., solely through externally-observable information from external sensors [36], [37]. In this paper, we assume that the AVRIL fault detector detects faults from an external virtual sensor. This sensor reliably tracks the following externally observable information.

1) **Joint positions:** the position of the four joints farthest from the base, each represented as a three-dimensional vector in the Cartesian world coordinates.
2) **Joint velocities:** the rotation speed of each joint relative to its adjacent joint, each represented by an angular velocity value in rad/s.
3) **Target position:** the relative position of the manipulation target (e.g. door handle, the centre of the block) to the end effector of the manipulator.

Inspired by previous studies [13], we simulate two kinds of partial faults through fault injection to the joint torque command, which represent two commonplace and representative faults in robotic manipulators.

1) **Stuck-at-constant:** The input command torque of one or more joints is stuck at a random value. This simulates a controller software crash or a communication problem between the controller and the motor.
2) **Stuck-at-zero:** The input command torque of one or more joints becomes zero. This usually occurs with a physical failure during command transmission and execution of the motor in the joint.

Note that since the fault injection is made in the action space and due to the inertia of the manipulator, neither fault will immediately change the joint velocity to a fixed value. Additionally, the stuck-at-zero fault can be considered as an extreme and special case of the stuck-at-constant fault. In our experiments, we randomly inject one kind of fault into one to three different joints at a random time point t in each episode. To assess whether the reward retrieved by the AVRIL fault detector can serve as a fault detection score to distinguish between observations from faulty and normal operations, we examine the averaged fault score r_a over a three-step fault detection window:

$$r_a(o_t) = \frac{\sum_{i\in[0,3)} R_{\text{AVRIL}}(o_{t+i})}{3}, \tag{6}$$

where R_{AVRIL} denotes the mean of the reward posterior estimated by the AVRIL fault detector.

The dataset for each round of evaluation consists of two sets of observations. The first set, D_n^o, consists of sequential observations and actions gathered during normal operation. The second set includes T sub-sets of observations collected during faulty operations $D_f^o(t), t \in [1, T]$. In each episode of a sub-set $D_f^o(t)$, the same kind of fault is injected from the tth time step, and T denotes the latest possible time step at which faults can be injected in the experiment configuration. Given the estimated reward over both sets of observations, we construct the average area under the receiver operating characteristic curve (AUC_a) as an evaluation metric, the AUC_a evaluates the effectiveness of a fault detection score (e.g. reward signal in AVRIL fault detector) in differentiating between normal and faulty observations within the fault detection window:

$$AUC_a = \frac{\sum_{t\in[1,T]} AUC(\{r_a(o_t^n)|o^n \in D_n^o\}, \{r_a(o_t^f)|o^f \in D_f^o(t)\})}{T}, \tag{7}$$

where r_a denotes the averaged fault score defined in Equation 6, and $AUC(\mathbb{A}^+, \mathbb{A}^-)$ represents the AUC computed from the fault detection scores of two sets of observations associated with two different categorical labels (e.g. normal episodes versus episodes where faults are injected since the third timestep).

B. Experiment Setup

We assess the effectiveness of our approach in two simulated manipulation tasks – the *Door opening* and *Block lifting* tasks, using a Panda arm model in the Robosuite simulator [38]. As depicted in Fig. 4, both tasks involve non-linear movements and can be split into two stages. Furthermore, the Block lifting task generates different and sparser observations than the Door opening task.

For each task, we train a SAC to control the robot to complete the tasks. Following this, we collect 1,000 episodes of normal operation data for each task by repeatedly executing the pre-trained SAC controller. An episode is considered to be under normal operation when the robot successfully completes the task at the end of the episode, such as opening the door more than 17 degrees or lifting the block higher than 15 cm.

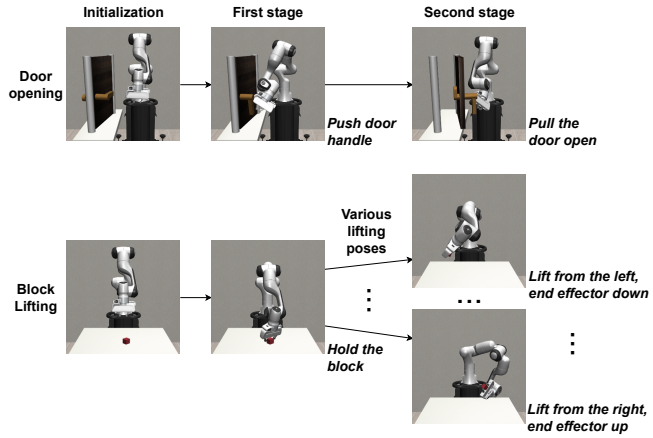

Fig. 4. Illustration of the two stages division of normal operations of Door opening and Block lifting tasks. In the Block Lifting task, the robotic manipulator experiences fewer motion restrictions from the environment compared to the Door opening task. As a result, the observations it attained are more thinly dispersed across the observation space.

In addition, we collect 1,000 episodes of faulty operations for each task. In each episode, a stuck-at-constant or a stuck-at-zero fault as described above is induced in between one and three randomly selected joints. The fault injection time step is chosen randomly between the first and fifth time step in each episode from a uniform distribution.

We compare our approach to two unsupervised anomaly detection models, namely the One-class Support vector machine (O.-c. SVM) [39] and Long short-term memory autoencoder (LSTMAE) [40]. The O.-c. SVM constructs a hyperplane in Ω to describe the distribution of normal observations based on training data containing only normal operations. When presented with an unseen observation $o_t \in \Omega$, the O.-c. SVM calculates the fault detection score $R_{\text{SVM}}(o_t)$ of the observation based on its distance from the separating hyperplane, as shown in Equation 8,

$$R_{\text{SVM}}(o_t) = sigmoid(\frac{|W \cdot \varphi(o_t) + b|}{||W||}) \,, \tag{8}$$

where SVM parameters W and b are used together to define the hyperplane, and the function φ maps the observations to a higher dimensional space. The dot product of two observations that have been transformed by φ is characterized by the SVM kernel function. In our experiment, we use a 5-degree polynomial kernel for the O.-c. SVM, as it shows the best performance in all experiment scenarios.

On the other hand, the LSTMAE generates a reconstruction for the observation o_t based on observations from prior time steps within the current episode. We then construct a fault detection score for each observation by calculating the mean-squared reconstruction error between the reconstructed and actual observation, as defined below, in which n_o denotes the dimension of observation o_t:

$$R_{\text{LSTMAE}}(o_t) = sigmoid(\frac{||\hat{o}_t - o_t||^2}{n_o}) \,. \tag{9}$$

We train and evaluate all the approaches on the data collected from each combination of task and fault respectively. In each round, we randomly shuffle the 1000 episodes of normal operation data and divide them into two sets of 500 episodes. The first half is reserved for training and validation, and the second half is used for evaluation. During the training of deep learning-based methods, such as AVRIL and LSTMAE, we set aside 100 of the 500 episodes in the first half of the normal data as a validation set to prevent over-fitting. Training stops when the loss on the validation set stops decreasing. For the training of the O.-c. SVM baseline, all 500 episodes are used as the training set. We use the remaining 500 episodes of normal operation data and all faulty operation data as the evaluation set. Given the evaluation set, we compare the efficiency of the fault detection score from baselines and the mean of reward posterior from our approach in differentiating between observations sampled from faulty and normal operations in the evaluation set, using the AUC_a metric as defined in Equation 7. Throughout both training and evaluation, we set AVRIL discount factor $\lambda = 0.5$ and inverse temperature $\psi = 1$.

Apart from comparing our approach with baselines, we also investigate the impact of incorporating expert knowledge on the fault detection outcome. To achieve this, we construct an AVRIL fault detector without expert knowledge, where the Beta reward prior is replaced by a standard normal distribution $\mathcal{N}(0, 1)$. In this configuration, if the consistency constraint in Equation 3 is satisfied, the KL-divergence term in Equation 2 will become a simple regulator over the variance of the retrieved reward [8].

C. Result Analysis

The averaged AUC of the AVRIL fault detector and two baselines under different fault and task configurations are shown in Fig. 5. We find that the best AVRIL reward prior configuration varied for the two tasks with different observation sparsity, as discussed in Fig. 4. Specifically, for the Door opening task where the normal operation produces observations with lower sparsity, the $Beta(5, 1.5)$ prior achieved the best results. In contrast, for the Block lifting task that produces sparser observations, the reward prior $Beta(15, 1.5)$ with lower variance, which constrains the subspace of higher rewards closer to the attained observations from normal operation, yields better results.

The results in Fig. 5 indicate that the AVRIL fault detector outperforms the LSTMAE and O.-c. SVM baselines for the more general stuck-at-constant fault, provided the prior distribution is appropriately configured. This advantage becomes particularly evident in the Door-opening task, where observation sparsity is relatively low, thus increasing the risk of over-fitting for fault detectors.

The stuck-at-zero fault, however, is a special case of the more general stuck-at-constant fault, although AVRIL again outperforms the baselines in the Door opening task. However, in the Block lifting task, the performance of the AVRIL fault detector is not superior to the baselines. This observation suggests that while an appropriately chosen reward prior can

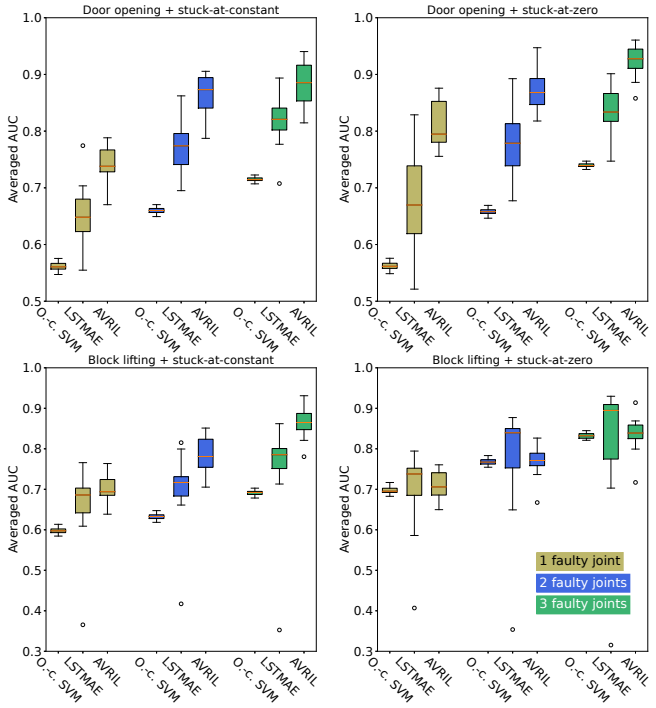

Fig. 5. Averaged AUC of the AVRIL fault detector and baseline approaches across various combinations of tasks and faults, based on 20 repetitions with distinct random seeds. Box plot colours indicate different numbers of faulty joints. The AVRIL fault detector outperforms baselines in all task and fault combinations except in the detection of a stuck-at-zero fault in the Block lifting task.

help the model improve performance on more general tasks, it may not be able to perfectly describe all of its specific sub-tasks, given its highly abstract representation of expert knowledge.

Furthermore, we conduct an analysis of the impact of the prior distribution on fault detection outcomes. Where we substitute the Beta prior distribution with a standard normal distribution $\mathcal{N}(0, 1)$. Results show that the appropriate selection of the reward prior is crucial for the performance of the AVRIL fault detector. Specifically, when using the $\mathcal{N}(0, 1)$ reward prior, the averaged AUC results for both tasks become worse. This degradation is more noticeable in the Door-opening task, which produces lower observation sparsity and therefore a higher risk of over-fitting.

In general, the reward signal retrieved by our approach can be used as a more effective indicator for identifying previously unseen faults in most experiment scenarios. Comparing two tasks with varying degrees of observation sparsity, our approach performs better on the task with lower observation sparsity, where normal operation observations represent only a small fraction of the entire observation space. This is evidenced by the higher performance gain achieved using the customized reward prior and the greater outperformance over the baselines in terms of the averaged AUC.

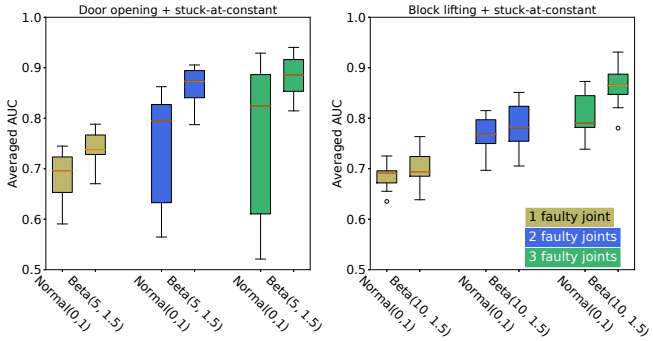

Fig. 6. Averaged AUC of AVRIL fault detectors with different reward priors over different tasks with stuck-at-constant fault (experiments over 20 repetitions). Results indicate that incorporating the Beta prior, which incorporates expert knowledge, significantly improves the fault detector's performance across both tasks.

V. Conclusion

We propose a zero-shot exogenous fault detection method using the AVRIL structure and assessed its efficacy within the context of detecting faults during recurrent robotic manipulator tasks. In this context, the fault detector is trained solely on data from normal operations and is capable to detect unseen faults through the retrieved reward function. The customizable reward prior distribution in AVRIL enables the incorporation of high-level expert knowledge about the manipulation task in an interpretable manner. Through our experiments, we find that a $Beta(a, b)$ prior where $a \leq b \leq 1$ is a suitable description for the manipulation tasks, as it enables the exogenous fault detector to identify faults from externally observable information more efficiently.

Our approach is evaluated within two distinct two-stage manipulation tasks with different observation sparsity, using the Robosuite simulator. Results show that the reward signal from our approach better distinguishes most unseen faults compared to the reconstruction error in LSTMAE and the distance to the hyperplane in O.-c. SVM baselines. Moreover, the use of the Beta reward prior improved performance in all tasks compared to the standard Gaussian prior.

The automated selection of the parameters for the Beta reward prior distribution is a topic for future work. In addition, we seek to explore the applicability of this generic framework to other categories of fault detection tasks in robotics, especially in monitoring real-time movements of robotic manipulators with high-dimensional sensor inputs, such as camera images.

Acknowledgement

H. Zhao and G. Dudek acknowledge support from the NSERC Canadian Robotics Network (NCRN), where they are a graduate student member and the scientific director respectively. H. Zhao acknowledges support from the Fonds de recherches du Québec – Nature et technologies (FRQNT).

REFERENCES

[1] R. Patton, J. Chen, and S. Nielsen, "Model-based methods for fault diagnosis: some guide-lines," *Transactions of the Institute of Measurement and Control*, vol. 17, no. 2, pp. 73–83, 1995.

[2] D. Silver, S. Singh, D. Precup, and R. S. Sutton, "Reward is enough," *Artificial Intelligence*, vol. 299, p. 103535, 2021.

[3] Y. Ding, L. Ma, J. Ma, M. Suo, L. Tao, Y. Cheng, and C. Lu, "Intelligent fault diagnosis for rotating machinery using deep q-network based health state classification: A deep reinforcement learning approach," *Advanced Engineering Informatics*, vol. 42, p. 100977, 2019.

[4] L. Junhuai, W. Yunwen, W. Huaijun, and X. Jiang, "Fault detection method based on adversarial reinforcement learning," *Frontiers in Computer Science*, vol. 4, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fcomp.2022.1007665

[5] S. Guiasu and A. Shenitzer, "The principle of maximum entropy," *The mathematical intelligencer*, vol. 7, pp. 42–48, 1985.

[6] M.-h. Oh and G. Iyengar, "Sequential anomaly detection using inverse reinforcement learning," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & data mining*, 2019, pp. 1480–1490.

[7] B. Lian, Y. Kartal, F. L. Lewis, D. G. Mikulski, G. R. Hudas, Y. Wan, and A. Davoudi, "Anomaly detection and correction of optimizing autonomous systems with inverse reinforcement learning," *IEEE Transactions on Cybernetics*, 2022.

[8] A. J. Chan and M. van der Schaar, "Scalable bayesian inverse reinforcement learning," in *International Conference on Learning Representations*, 2021.

[9] B. Piot, M. Geist, and O. Pietquin, "Boosted and reward-regularized classification for apprenticeship learning," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, 2014, pp. 1249–1256.

[10] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[11] M. L. Visinsky, J. R. Cavallaro, and I. D. Walker, "Robotic fault detection and fault tolerance: A survey," *Reliability Engineering & System Safety*, vol. 46, no. 2, pp. 139–158, 1994.

[12] C. Lou, P. Huang, and S. Smith, "Understanding, detecting and localizing partial failures in large system software," in *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, 2020, pp. 559–574.

[13] A. L. Christensen, "Fault detection in autonomous robots," Ph.D. dissertation, Université Libre de Bruxelles, 2008.

[14] A. Khadidos, R. M. Crowder, and P. H. Chappell, "Exogenous fault detection and recovery for swarm robotics," *IFAC-PapersOnLine*, vol. 48, no. 3, pp. 2405–2410, 2015.

[15] B. Khaldi, F. Harrou, F. Cherif, and Y. Sun, "Monitoring a robot swarm using a data-driven fault detection approach," *Robotics and Autonomous Systems*, vol. 97, pp. 193–203, 2017.

[16] D. Tarapore, A. L. Christensen, and J. Timmis, "Generic, scalable and decentralized fault detection for robot swarms," *PLOS One*, vol. 12, no. 8, p. e0182058, 2017.

[17] M. Namvar and F. Aghili, "Fault diagnosis in robotic manipulators using joint torque sensing," *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 5330–5334, 2008.

[18] A. G. Millard, J. Timmis, and A. F. Winfield, "Run-time detection of faults in autonomous mobile robots based on the comparison of simulated and real robot behaviour," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2014)*. Piscataway, NJ, USA: IEEE Press, 2014, pp. 3720–3725.

[19] T. Ren, Y. Dong, D. Wu, and K. Chen, "Collision detection and identification for robot manipulators based on extended state observer," *Control Engineering Practice*, vol. 79, pp. 144–153, 2018.

[20] A. L. Christensen, R. O'Grady, and M. Dorigo, "From fireflies to fault-tolerant swarms of robots," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 754–766, 2009.

[21] H. Gu, H. Hu, H. Wang, and W. Chen, "Soft manipulator fault detection and identification using anc-based lstm," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 1702–1707.

[22] A. L. Christensen, R. O'Grady, M. Birattari, and M. Dorigo, "Fault detection in autonomous robots based on fault injection and learning," *Autonomous Robots*, vol. 24, pp. 49–67, 2008.

[23] Y. Gao, L. Gao, X. Li, and Y. Zheng, "A zero-shot learning method for fault diagnosis under unknown working loads," *Journal of Intelligent Manufacturing*, vol. 31, pp. 899–909, 2020.

[24] S. Zhang, H.-L. Wei, and J. Ding, "An effective zero-shot learning approach for intelligent fault detection using 1d cnn," *Applied Intelligence*, pp. 1–18, 2022.

[25] L. Feng and C. Zhao, "Fault description based attribute transfer for zero-sample industrial fault diagnosis," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 1852–1862, 2020.

[26] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 3174–3183.

[27] W.-D. Chang, J. C. G. Higuera, S. Fujimoto, D. Meger, and G. Dudek, "Il-flow: Imitation learning from observation using normalizing flows," *arXiv preprint arXiv:2205.09251*, 2022.

[28] Y. Guo, G. Ding, J. Han, and S. Tang, "Zero-shot learning with attribute selection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[29] M. T. Spaan, "Partially observable markov decision processes," *Reinforcement learning: State-of-the-art*, pp. 387–414, 2012.

[30] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, 2016.

[31] S. Arora and P. Doshi, "A survey of inverse reinforcement learning: Challenges, methods and progress," *Artificial Intelligence*, vol. 297, p. 103500, 2021.

[32] H. J. Jeon, S. Milli, and A. Dragan, "Reward-rational (implicit) choice: A unifying formalism for reward learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4415–4426, 2020.

[33] W. B. Kleijn, A. Storus, M. Chinen, T. Denton, F. S. Lim, A. Luebs, J. Skoglund, and H. Yeh, "Generative speech coding with predictive variance regularization," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6478–6482.

[34] Y.-F. Zhang, F.-M. Luo, and Y. Yu, "Improve generated adversarial imitation learning with reward variance regularization," *Machine Learning*, vol. 111, no. 3, pp. 977–995, 2022.

[35] O. Khatib, "Inertial properties in robotic manipulation: An object-level framework," *The international journal of robotics research*, vol. 14, no. 1, pp. 19–36, 1995.

[36] J. Rodziewicz-Bielewicz and M. Korzeń, "Comparison of graph fitting and sparse deep learning model for robot pose estimation," *Sensors*, vol. 22, no. 17, p. 6518, 2022.

[37] P. Neto, J. N. Pires, and A. P. Moreira, "3-d position estimation from inertial sensing: Minimizing the error from the process of double integration of accelerations," in *39th Annual Conference of the IEEE Industrial Electronics Society (IECON)*. IEEE, 2013, pp. 4026–4031.

[38] Y. Zhu, J. Wong, A. Mandlekar, and R. Martín-Martín, "robosuite: A modular simulation framework and benchmark for robot learning," *arXiv preprint arXiv:2009.12293*, 2020.

[39] H. J. Shin, D.-H. Eom, and S.-S. Kim, "One-class support vector machines—an application in machine fault detection and classification," *Computers & Industrial Engineering*, vol. 48, no. 2, pp. 395–408, 2005.

[40] P. Park, P. D. Marco, H. Shin, and J. Bang, "Fault detection and diagnosis using combined autoencoder and long short-term memory network," *Sensors*, vol. 19, no. 21, p. 4612, 2019.