Technical Disclosure Commons

Defensive Publications Series

January 2021

SELF-ADAPTIVE ANOMALY DETECTION WITH DEEP REINFORCEMENT LEARNING AND TOPOLOGY

Qihong Shao

Carlos M. Pignataro

Follow this and additional works at: https://www.tdcommons.org/dpubs_series

Recommended Citation
Shao, Qihong and Pignataro, Carlos M., "SELF-ADAPTIVE ANOMALY DETECTION WITH DEEP REINFORCEMENT LEARNING AND TOPOLOGY", Technical Disclosure Commons, (January 06, 2021)
https://www.tdcommons.org/dpubs_series/3946

This work is licensed under a Creative Commons Attribution 4.0 License.
This Article is brought to you for free and open access by Technical Disclosure Commons. It has been accepted for inclusion in Defensive Publications Series by an authorized administrator of Technical Disclosure Commons.

SELF-ADAPTIVE ANOMALY DETECTION WITH DEEP REINFORCEMENT LEARNING AND TOPOLOGY

AUTHORS:
Qihong Shao
Carlos M. Pignataro

ABSTRACT

In the networking field, network topology is one of the most important perspectives as it can bring additional insights to the modeling process. Existing anomaly detection approaches do not take topology information into consideration. To address such limitations techniques are presented herein that support deep Convolutional Neural Network (CNN) modeling with Reinforcement Learning (RL) employing, for example, an Advantage Actor Critic (A2C) algorithm. Additionally, aspects of the techniques presented herein support an innovative new way to model a "customer profile" that leverages topology information.

DETAILED DESCRIPTION

Anomaly detection is an important part of any system, from an online system to a network. It can be a very challenging task due to, for example, the lack of abnormal data points and insufficient knowledge. Even with large amount of data, it is still difficult to identify and label an "abnormal" data point. In traditional machine learning (ML) and deep learning (DL) models for anomaly detection it is difficult to reach the desirable level of accuracy and frequent false alarms may arise. Such models are limited by a range of conditions, including:

- They are typically based on a strong assumption about the underlying mechanism of anomaly patterns. Both ML and DL methods assume it has either a distribution or latent space and try to find the underlying models. It may not produce satisfactory results under scenarios where the assumptions do not hold.
- They are typically tedious for threshold setting. In some instances, hundreds or thousands of parameters may need to be tuned, and in practice it is not feasible for a customer to properly setup and customize their solutions.

- Significant human effort is typically needed to customize such models and they can be difficult to scale up and access. Such systems can be very complex and engineers may not only need understand all of the components but also comprehend the set of methods to be able to tune the parameters for each of component.
- They typically lack self-adaptive learning. Anomaly detection approaches can accumulate more and more data, which should dynamically improve the performance. However, existing anomaly detectors are typically assumed to be static and often need human involvement to make changes.
- They are typically tailored to network specific features. A network has, among other things, a topology which provides useful and additional information for anomaly detection models. All of this useful information is not properly leveraged in existing approaches.

Reinforcement learning (RL) is emerging as a hot topic with better computational resources becoming available. It does not require human involvement and it can learn an optimal policy by interaction within an environment. It is a natural fit for resolving the challenges in anomaly detection, including self-adapted learning. The objective of RL is to make decisions, from external behavior data, for an agent based on the underlying reward function. As shown in Figure 1, below, a reward function provides an agent incentives to improve its strategy hence thus receiving as many rewards as possible. The agent's normal behavior is then understood by the reward function which is inferred via RL. Using a learned reward function, one may evaluate whether a new observation from the target agent follows a normal pattern. In other words, if the new observation generates a low reward then it implies that the observation is not explained by the preferences of the agent that have been learned thus far and that observation may be considered as a potential anomaly.

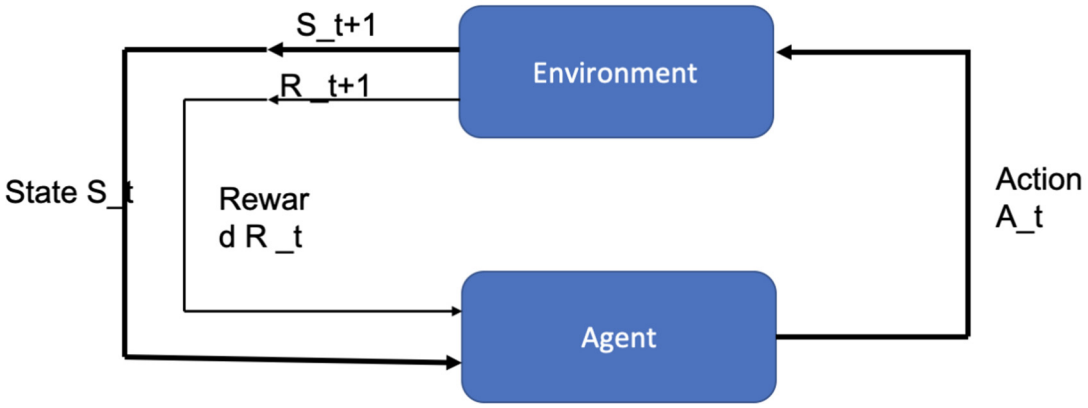

Figure 1: Reinforcement Learning (RL)

A network has a very unique feature, topology information. Existing network anomaly detection approaches typically focus on a single device and single feature detection, without considering a network's topology. However, network topology is very important for a number of reasons.

As will be described below, aspects of the techniques presented herein have broad applicability since the RL is applied to the abstraction of a topology.

First, a topology is a layer-network construct, which may apply to:

- Network topology, which in the most pragmatic sense is a link-state database. This is, for example, an Open Shortest Path First (OSPF) link-state database (LSDB) topology or an Intermediate System - Intermediate System (IS-IS) topology. As well, for a general case any topology such as, for example, a network analysis toolkit, collectors, etc. are also relevant.
- Service topology, which is usually a graph of an ordered application of services. The most direct application is a service graph, as described in, for example, Internet Engineering Task Force (IETF) Request for Comments (RFC) 7665 (see, https://tools.ietf.org/html/rfc7665#section-2.1) and RFC 8300 (see, https://tools.ietf.org/html/rfc8300#section-6.4).

It is important to note that aspects of the techniques presented herein may be well applied to a cross-layer topology, mapping and correlating between a network topology and a service topology.

In summary, in the network field, topology is one of the most important perspectives, which can, among other things, bring additional insights to the modeling process. Existing approaches do not take topology information into consideration.

In light of the this, techniques are presented herein that support the application of deep CNN modeling and RL (with, for example, an A2C algorithm) to the challenges of anomaly detection. Aspects of the presented techniques support a new way to model a "customer profile" by leveraging topology information.

In support of the discussion that will be presented below it will be helpful to briefly describe the problem setting under which RL may be applied through the techniques that are presented herein.

Aspects of the techniques presented herein support a CNN-based anomaly detector that is trained consistently through RL to provide a self-adapted anomaly detection system.

Following the framework of RL, the environment is modeled as follows:

- S is the finite set of states.
- A is the finite set of actions. A = {0, 1} in which 1 means the given state is anomalous and 0 otherwise.
- P(s,a,s') is a dynamic/transition model for each action, the state transition probability of changing to state s' from state s when action a is taken, according to the following formula:

$$P = (S_{t+1=}s' | s_t = s, a_t = a)$$

- R (s,a) is reward of executing action "a" in state "s".
- v[0, 1) is a discount factor, which weighs immediate and future rewards.
- Policy determines how the agent chooses actions : S -> A, mapping from states to actions.

A state value function determines the expected sum of future rewards under a particular policy which specifies what is good in the long run:

$$V^{\pi}(s) = E_{\pi}[G_t|s_t = s] = E_{\pi}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \cdots |s_t = s]$$

For the problem setting under aspects of the techniques presented herein "Anomaly Detector" is defined as the policy. The optimal anomaly detector is the detector that satisfies the following constraint:

$$\pi^*((s) = \arg max\ V^{\pi}(s)$$

The Experience E is a set of tuples that are defined as <s, a, r, s'>. The variables "s" and " s' " in S indicate the states of the target system before and after the action a. In an anomaly detector, actions are selected by the anomaly detector. Therefore, the experience records all of the behaviors of the anomaly detector. Considering a deterministic optimal anomaly detector, it should maximize the performance and, in fact, is fully determined by the accumulated reward function Q(s, a). Q(s, a) represents the accumulated reward started from state "s" with action "a," which is the average accumulated reward in anomaly detection following the anomaly detector .

The goal of an anomaly detector is to consistently improve the policy by learning from the experience to gain a better estimation of Q(s, a). This can be achieved by learning from the state and action history. Figure 2, below, shows the learning process of anomaly detection, s in blue represents the "state" and "a" in orange represents the "action." Starting state "s_0" takes an action "a_0" and it is possible to reach a next state, e.g. {s_11, s_12, s_13....}. It is a stochastic process and decides which action to take. Continuously, once state s_1x is reached it takes some other actions continuously to improve the policy. There are many different learning options including, for example, dynamic programming, Monte Carlo methods, temporal difference (TD), etc. The best option may be selected based on, for example, different customer data.

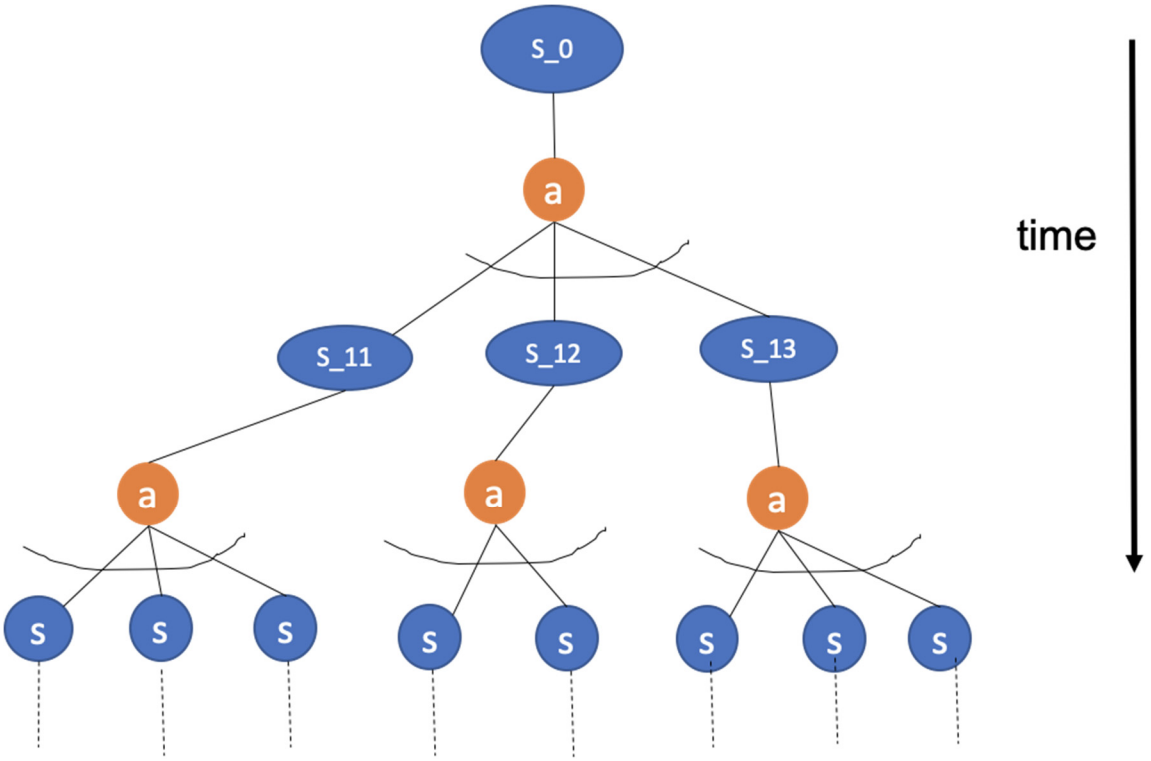

Figure 2: Learning Optional Policy from "s" (state) and "a" (action)

Further in support of the discussion that will be presented below it will be helpful to briefly discuss CNN with RL. As shown in Figure 3, below, customer device data and their topology information may be converted into "state frames." Each time series may be transformed into a set of multi-dimensional data instances using the sliding window method.

The "state" image that is generated from topology information would be a high dimension data, as there are hundreds or thousands of devices in a customer network and each device may have hundreds features. Hence, a CNN may be leveraged to reduce the dimensionality and extract the lower level and high level features out of the original "state" image in addition to finding the pattern and correlations.

Under aspects of the techniques presented herein, a deep CNN takes a stack of state frames as an input, which pass through its network and output a vector of actions for each action that is possible in the given state.

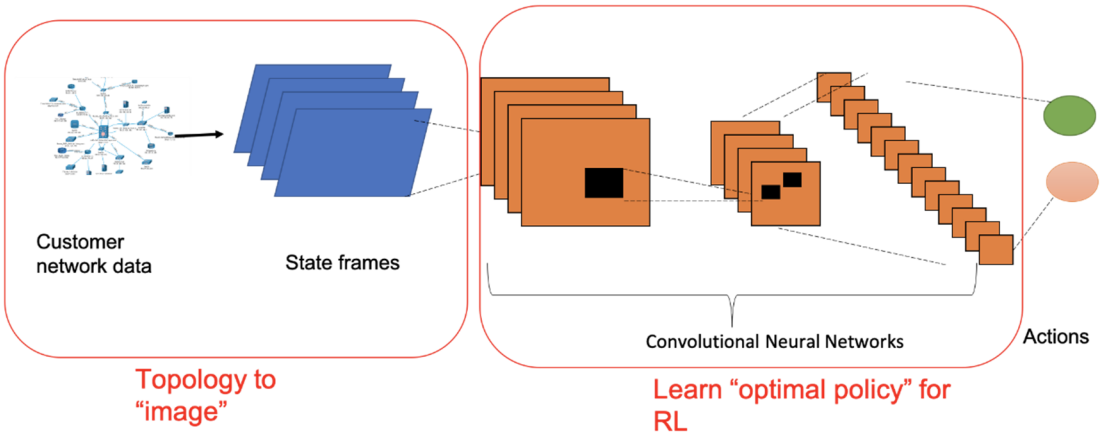

Figure 3: Architecture for Anomaly Detection using RL with Topology

Aspects of the techniques presented herein encompass a number of elements that are of particular interest and note. Various of those elements will be described and discussed below.

A first element of interest and note concerns converting "topology" into "state image." There are different methods and embodiments for how topology, and specifically the connections, may be represented. While various of the more complex ideas may be described, one of the most practical approaches is to simply use the existing concept of Topology Aggregation. For example, as illustrated in Figure 4, below, Private Network-to-Network Interface (PNNI) and link-state:

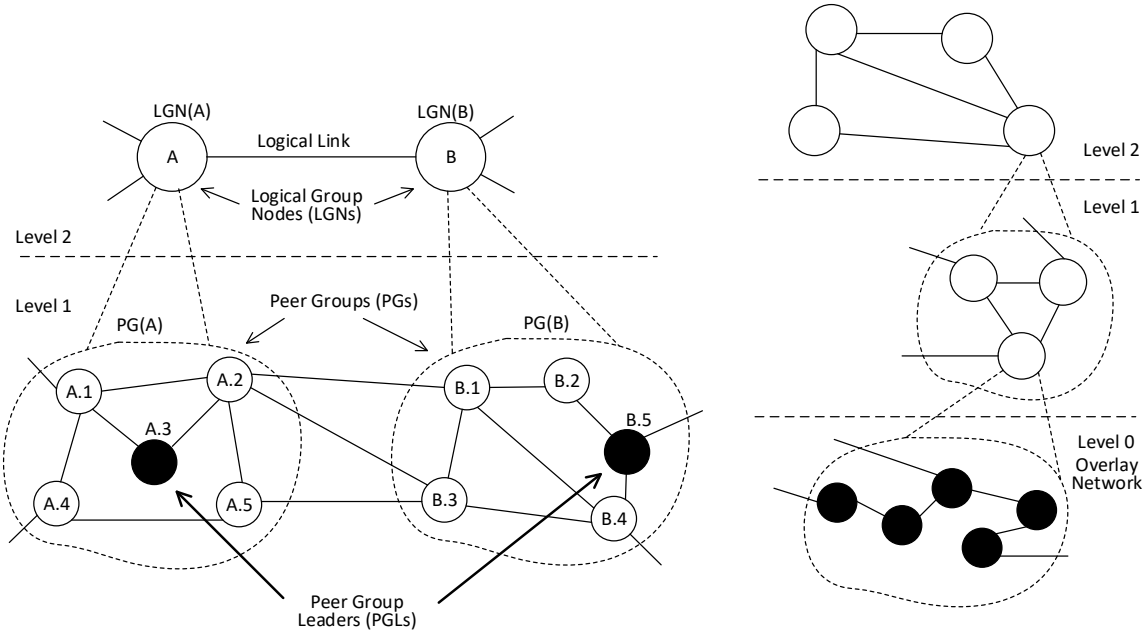

Figure 4: Illustrative Topology Aggregations

The following narrative will present a novel approach for performing topology aggregation via aspects of the techniques presented herein. It is important to note that for each "node" at the lowest aggregation level a set of time series will be held which will be analyzed as, for example, a central processing unit (CPU), backplane, memory, etc. Part of the challenge is also to decide the aggregation.

One important element under aspects of the techniques presented herein encompasses converting “topology" into a "state image.” A main objective is to convert customer device information (e.g., multi-variance time series data) and their network topology information into a "state image."

A first step may encode "time series" as "images." Time series data can be encoded into different types of images, namely, for example, a Gramian Angular Field (GAF) and a Markov Transition Fields (MTF). See Figure 5, below. This enables the use of techniques from computer vision, such as CNN as applied under aspects of the techniques presented herein.

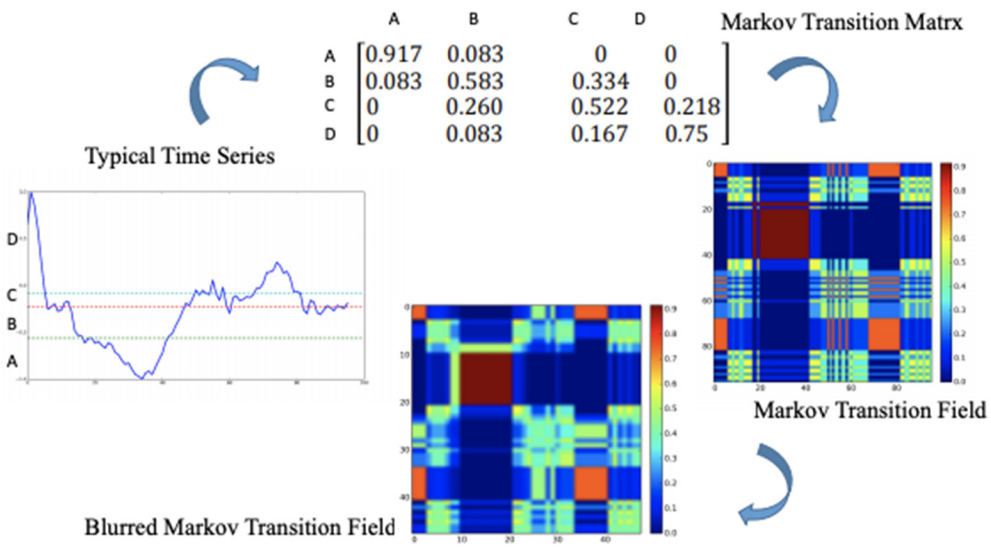

Figure 5: Illustrative Data Encoding

A second step may generate a "state" image with topology. For purposes of exposition, consider the five devices {A,B,C,D,E} as shown in Figure 6, below.

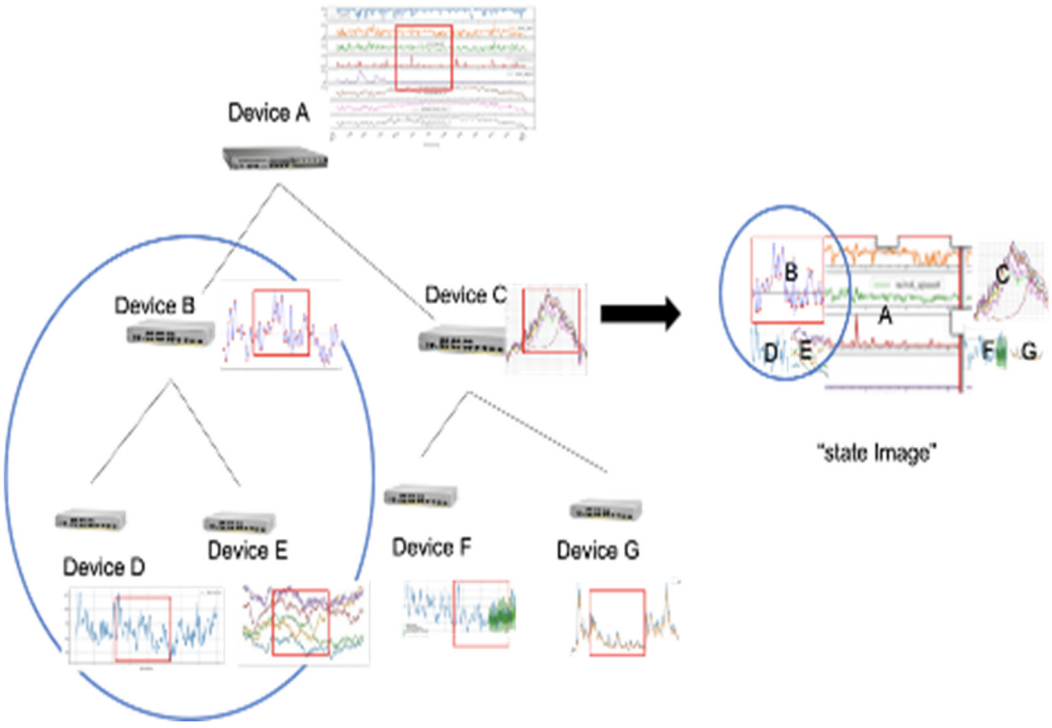

Figure 6: Converting Topology Information into "state" Images

Each of the devices has multiple features (including, for example, CPU, memory, etc.) as time series data. For a given time span T: [t1, t2]:

- Single device "snapshot" extraction is performed. For each device, its sub-image during the time span T is obtained using the encoding method (e.g., GAF, MTF, etc., as described previously). In the instant example, a snapshot/sub-image may be obtained of all devices B, C, D, E, F, and G in time span T (marked with a red rectangle).
- A "snapshot" is merged from multiple devices by topology. Based on their importance and relevant topology, the image snapshots can be merged into a "state image" (as shown on the right side of Figure 6, above).

The images that were discussed above may be merged through a series of steps, including:

- Capturing the "importance" of different sub-image by their sizes.
    - A core node will received a larger size, while an edge node receives less size. For example, A is the "core" node, which have the largest size (e.g. size 4). Further, B and C are middle tier nodes, which have middle size (e.g. size 2). Finally, {D,E,F,G} might be the edge nodes, which have smallest size(e.g. size = 1).
- Determining relevant distance through preserving the "connectivity" by "relevant distance" in the image.
    - For example, B and {D,E} are in the same cluster (blue circle), as they are connected to each other in the topology. Additionally, C is far away from {D,E} in the merged image, as they are not connected.

Following a merge operation one single "state image" is available for time span [t1,t2]. Once all "state frames" are generated for the CNN, it contains the "spatial" / topology information (e.g., sub-image close to each other for B with {C,D} in the same cluster). The CNN will take all the inputs to generate a final, optimal policy in order to maximize the reward function.

Continuing with a discussion of the process under aspects of the techniques presented herein, a sliding window may be moved to a next span [t1', t2'] resulting in the

generation of another "state image." It is important to note that the two windows [t1,t2] and [t1', t2'] may be overlapping or non-overlapping depending upon the "step change."

As described above, under aspects of the techniques presented herein topology information may be leveraged during the generation of "state images" for CNN. Among other things:

- CNN is good at finding the high level and low level features. CNN can easily detect that A is the most important node (e.g. if A has anomaly, its impact is huge).
- CNN can also detect the relevant distance. For example, B and {D,E} are close to each other and implicitly present their topology connectivity.

It is important to note that there may be other ways to convert the network topology into "image state frames." For simplicity of exposition the above discussion of aspects of the techniques presented herein presented just one possible approach. A generated "state image" may have a high dimension and may be fed into a CNN for further processing.

Another important element under aspects of the techniques presented herein encompasses learning an optimal policy using an A2C algorithm. To find an optimal policy, an A2C algorithm, which is one of the best state of the art "on-policy" algorithms, may be employed. The algorithm learns the value function for one policy while following itself, not following another policy. The term "actor-critic" is an algorithm that satisfies various criteria, such as:

- The actor updates the policy distribution in the direction suggested by the critic, which conducts actions in an environment.
- The critic computes value functions to help assist the actor in learning.

To update the network, experiences may be collected and processed immediately. For example:

- The environment may be interacted with and state transitions collected.
- After n-steps, or at the end of the episode, updates may be calculated and then applied.
- The data may be discarded.

As shown in Figure 7, below, the Asynchronous Advantage Actor Critic (A3C) algorithm is the asynchronous version where each agent talks to the global parameters

independently. Due to the thread-specific issue agents would play with policies with different versions and the aggregate update is not optimal. To resolve this issue, A2C adds a "coordinator" which will wait for all parallel actors to complete before updating the global parameters.

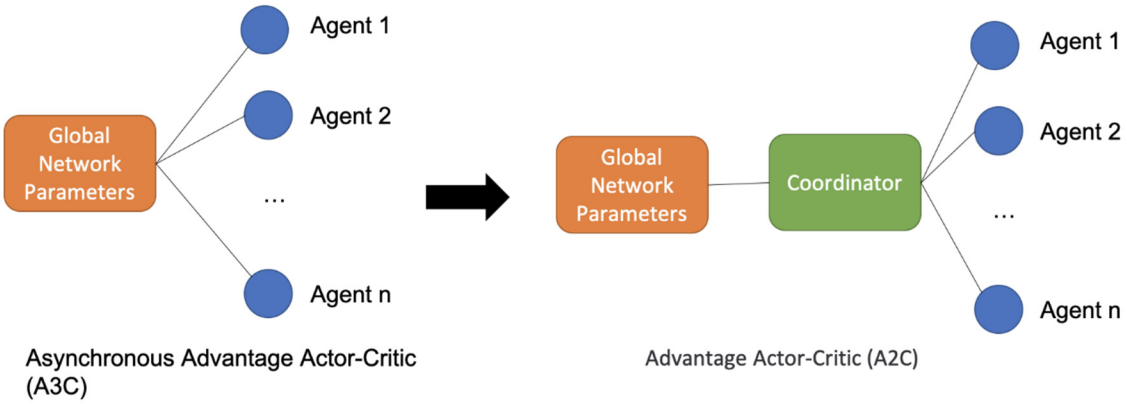

Figure 7: Illustrative A3C and A2C Algorithm

In summary, as was described above under aspects of the techniques presented herein, the anomaly detector makes no assumption for underlying distribution as well as the threshold. Additionally, it can consistently improve its performance without human interaction. The presented self-adaptive anomaly detection techniques using deep RL with topology can fully release humans from the tedious parameter setting process and learn by the agent itself.

Various aspects of the techniques presented herein are of interest and note, including, for example:

- A model can be utilized to handle multiple time series, which can leverage customer network "topology" information.
- Customer network data may be converted into images for subsequent processing. CNN may be applied to process customer network data and topology information. And, importantly, CNN with RL may be used to train an agent.
- As depicted in Figure 6, above, customer device data and topology information may be converted into "state frames." Each time series may be transformed into a set of multi-dimensional data instances using the sliding window method.

A deep CNN may take a stack of state frames as an input, pass such data through its network, and output a vector for each action possible in the given state.

- Real time customer performance counters may be handled.
- In the network field, topology is one of the most importance perspectives that one may consider as it can bring additional insights to the modeling process.
- An innovative way of modeling a "customer profile" is provided that leverages network topology information.