ARL-TR-9318 ● SEP 2021

DEVCOM ARMY RESEARCH LABORATORY

Analysis and Visualization of Battlefield Simulations Based on Deep Reinforcement Learning

by Vinod K Mishra, Cleon Anderson, and Adam Childs

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

ARL-TR-9318 ● SEP 2021

Analysis and Visualization of Battlefield Simulations Based on Deep Reinforcement Learning

Vinod K Mishra
Computational and Information Sciences Directorate,
DEVCOM Army Research Laboratory

Cleon Anderson and Adam Childs
Parsons Corporation

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED (From - To)**
September 2021	Technical Report	June 2020–May 2021

4. TITLE AND SUBTITLE
Analysis and Visualization of Battlefield Simulations Based on Deep Reinforcement Learning

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Vinod K Mishra, Cleon Anderson, and Adam Childs

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
DEVCOM Army Research Laboratory
ATTN: FCDD-RLC-NC
Aberdeen Proving Ground, MD 21005

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-TR-9318

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release: distribution unlimited.

13. SUPPLEMENTARY NOTES
ORCID ID: Vinod K Mishra, 0000-0001-9432-9082

14. ABSTRACT
The extraction of useful information from large, disparate, and heterogeneous data sets requires a good set of tools and techniques. We report the details of such an exercise involving battlefield simulation data and development of the tools for the statistical analysis and visualization.

15. SUBJECT TERMS
simulation, visualization, OpSim, statistical analysis, deep reinforcement learning, DRL

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Vinod K Mishra
Unclassified	Unclassified	Unclassified	UU	22	**19b. TELEPHONE NUMBER (Include area code)** (410) 278-0114

Contents

List of Figures

1. Introduction

Recent advances in technology have made the need for very fast command and control (C2) decisions in battlefield very important. Traditionally, subject-matter experts (SME), using their accumulated knowledge, experience, and expertise resulting in courses of action, have handled these tasks for achieving the desired goals. Now the technology has advanced and the pace of actions in the tactical arena has accelerated. Commanders need to make decisions for a given situation in a very short time, sometimes virtually a split-second. This requires the introduction of automation in these tasks, leading to use of algorithms based on an artificial intelligence (AI) and machine learning (ML) framework.

ML techniques are classified into two broad areas, one approach needs a large amount of real situation data for training (e.g., supervised learning) and other works with the concept of actions and rewards (e.g., reinforcement learning [RL]). There are also many hybrid techniques combining both approaches in many different ways. Deep RL (DRL) is one such technique. In this report, we describe tools developed for comparative analysis and visualization of battlefield simulation data using both SMEs and DRL algorithms.

2. Generation of Simulation Data

We give a brief description of the components of simulation based on work by Narayanan et al.[1]

2.1 Scenario

Operation Tiger Claw, developed by the Maneuver Center of Excellence (Fort Benning, Georgia), generates predefined realistic combat scenarios consisting of Blue and Red forces within a simulation environment. The scenarios also include battle plans developed by expert military SMEs and generated using doctrinal force employment in accordance with the Operation Order and corresponding threat tactics.

The two simulation environments incorporating the Tiger Claw C2 scenario are the following:

1) StarCraft II Learning Environment
2) Operation Simulation (OpSim)

2.2 Simulation Process

2.2.1 OpSim and SitaWare

We worked with the simulation data generated by the decision-support tool OpSim, which provides the following:

1) planning support,
2) mission execution monitoring,
3) mission rehearsal,
4) embedded training, and
5) re-planning.

OpSim integrates with SitaWare's command, control, communications, computers and intelligence (C4I) program allowing all command levels to share situational awareness. It provides C2 and battle management capabilities including interoperability for exchanging battlespace information with coalition partners. There are different versions specific to three different environments: Headquarters, Frontline, and Edge. OpSim coordinates all operational actions, thus making it an embedded simulation connecting directly to the operational mission command. It uses service-oriented architecture–based simulation and is capable of running faster than many current state-of-the-art simulation environments.

2.2.2 OpenAI Gym

OpSim does not support AI applications, so an interface to OpenAI Gym (an open-source program) was developed to expose the simulation state and offer simulation control to the external agents. Gym is a toolkit for developing and comparing RL algorithms. It makes no assumptions about the agent's structure and is compatible with any numerical computation library, such as TensorFlow or Theano. The Gym library has a collection of environments that share the same interface for writing and executing general RL algorithms. It provides a collection of environments with a common interface for RL development and testing. It also gives the ability to select 1) altered actions for chosen entities within the simulation and 2) the amount of time to simulate before responding back to the interface.

2.3 RL and DRL

Formally, RL is a Markov decision process consisting of the following:

1) $\{s\}$ = A set of states

2) $\{a\}$ = A set of actions

3) $P_a(s, s')$ = A state transition probability function (probability of moving from current state s to the next state s'due to the chosen action a)

4) $R_a(s, s')$ = A reward function accrued to the decision maker

The Markov property implies the next state s' depends only on the current state s and decision maker's action a, and is independent of all previous states and actions.

The RL process tries to find an optimal action that maximizes the expected, cumulative sum of discounted rewards. DRL integrates RL with a deep neural network (DNN) architecture for approximating optimal actions for states within an environment using the components 1–4 and a DNN.

Two types of commanders were developed for the OpSim simulation environment:

1) Rule-engine based commander: it uses an expert designed rule engine.

2) DRL-trained commander: it uses a multi-input and multi-output long short-term memory (LSTM) neural network trained with the Advantage Actor Critic (A2C) algorithm[2] (see the Appendix).

OpSim's RL interface supports multi-agent training where each force can be either a rule-based or AI commander. RLlib (a library that provides scalable software primitives for RL) is used to scale the learning on a high-performance computing system.

2.4 Simulation Components

Tiger Claw is run 10–30K times in the OpSim environment incorporating the following:

- different random number seeds and
- slight variations to the initial conditions.

These runs provide a diverse set of state-action pairs used for training the DRL agent. The output from the OpSim simulation resulted in development of two software modules, whose details follow in Fig. 1.

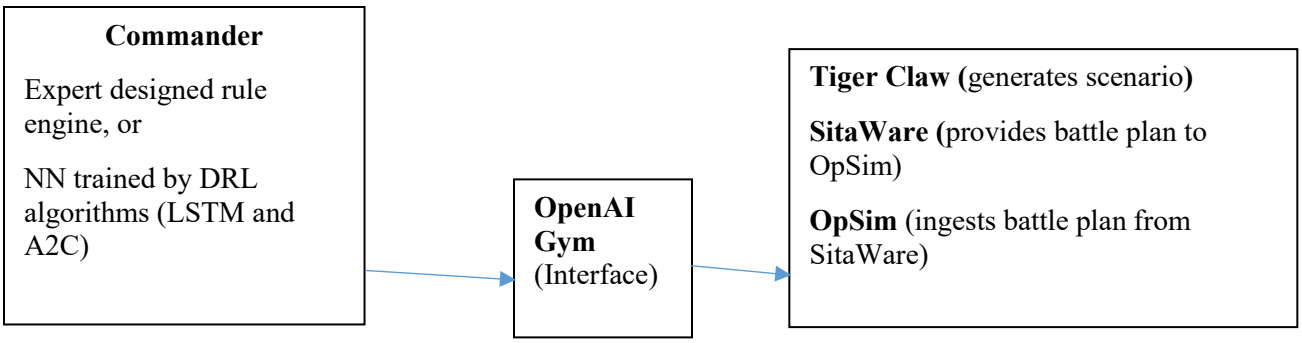

Fig. 1 Components of OpSim simulation

3. Analysis Module

The analysis module takes the simulation data as input and runs statistical analysis algorithms to find their properties. It includes the following elements:

- *User interface*: It is based on Plotly Dash. The user selects variables and the kind of analysis, which displays plots with which the user can interact to choose the desired information.
- *Packages*:
    - Python and R: development work
    - Python Dask library and PyArrow: transport and handling of the data
    - Scikit-learn, pandas, sklearn, matplotlib and seaborn: analysis and visualization
    - R was embedded into Python to use libraries available in R but not in Python: This allowed R's easy-to-use libraries (e.g., logit models) to be incorporated into analysis.
- *Properties*:
    - The iPython package is used to ingest OpSim simulation data into the PostgreSQL database: High-dimensional and noisy data transformed into two-dimensional tables (using pandas and vanilla Python) and offloaded into appropriate databases (used PostgreSQL) for data analysis using PySpark (provides interface between Python and Spark).

- Custom classes used to access the data; it is extracted, cleaned, and variable values are standardized.
- Hadoop pipeline is used for their storage and retrieval.
- Tables created in the PostgreSQL database and pandas data-frame written into them dynamically.
- JavaScript Object Notation key-value pair are compared with extracted two-dimensional data, columns with all missing values are removed, missing variables are resolved, and columns are standardized.
- Tables queried and optimized display of plots created in R within the Jupyter environment:
    - The associative classification model is used to create a network of relations for association between select variables, and probabilistic models are applied to measure the odds of outcome.
    - An ML pipeline with semi-structured raw textual data was created and natural language processing was used. Meaning extracted from the text was used for classification and prediction. Addresses mentioned in raw text were converted into geographical coordinates for data analysis, prediction, and visualization.
    - It incorporated autocorrelation, log-linear analysis, descriptive statistics, analysis of variance, a logit model (for removing artifacts due to Poisson distribution), an associative classifier, random forest (for feature reduction), and other statistical tools.
- Data plots displayed the results of statistical analysis created.

The statistical analysis of OpSim data determines the variables influencing various outcomes.

Data analysis uses many algorithms; here we give two examples.

Example 1: Jarque–Bera (JB) test

The **JB test** determines whether the sample data have the skewness and kurtosis matching a normal distribution. If it is far from zero, it signals that the data do not

have a normal distribution and so cannot be treated as a random variable in the simulations. The JB statistic is

$$JB = \frac{n}{6}\left[S^2 + \frac{1}{4}(K-3)^2\right] \quad (1)$$

Here n = number of data points, S = skewness, and K = kurtosis of the given data. JB is distributed asymptotically as chi-squared (χ^2) with 2 degrees of freedom because JB is the sum of squares of two asymptotically independent standardized normal variables. In the simulation phase a normal variable means that it is random.

The plot in Fig. 2 shows that the data cannot be represented by normal variates.

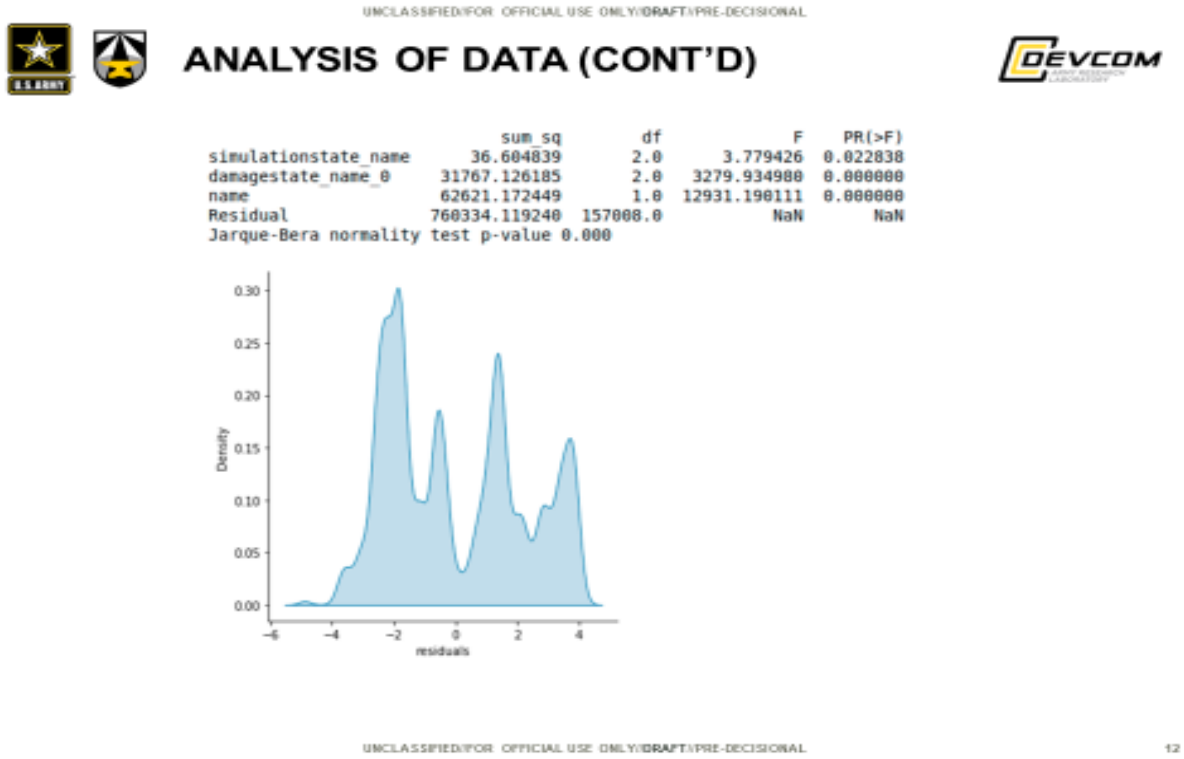

Fig. 2 Analysis example: JB test

Example 2: Tukey Honestly Significance Difference (HSD) test

This test determines if the means are significantly different from one another. It is given by

$$q = \frac{\mu_L - \mu_S}{\sigma/\sqrt{n}} \quad (2)$$

Here μ_L is the larger and μ_S is the smaller of the two means being compared, and σ is the standard deviation of the sum of the means.

The test determines if there is a strong chance that an observed numerical change in one variable is causally related to an observed change in another variable.

The test in Fig. 3 shows that most of the pairs fail. The analysis module can be extended by incorporating new tests if needed for a new problem domain.

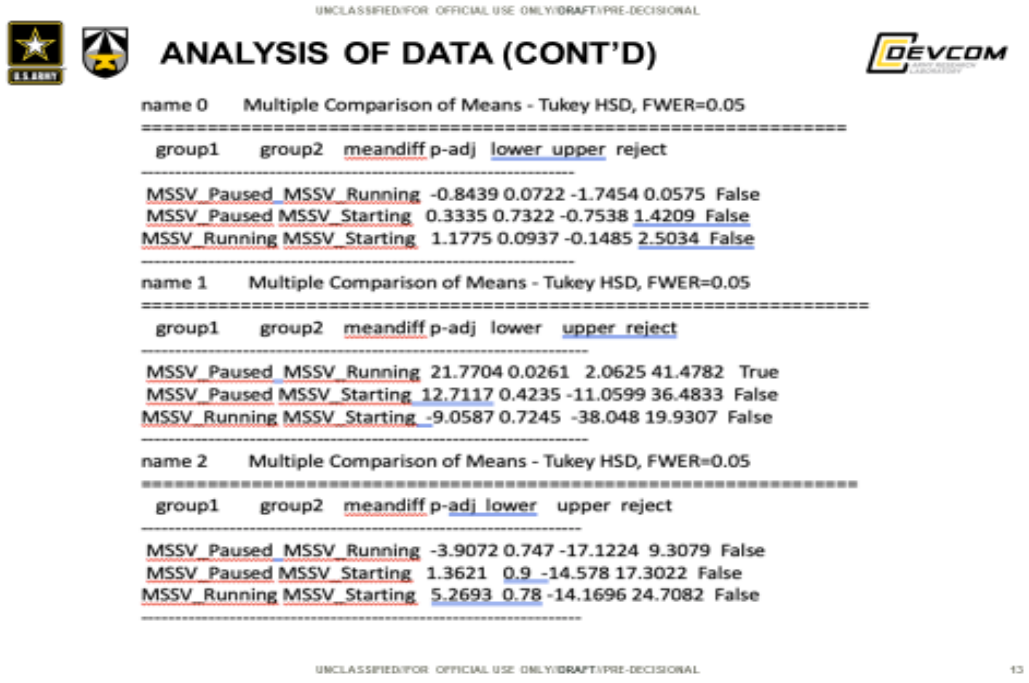

Fig. 3 Analysis example: Tukey HSD

4. Visualization Module

The visualization module presents the data as video streams. It includes the following elements:

- *User interface*: It has Flask web user interface (UI) for data visualization.
- *Packages used*: Slurm (resource manager), Flask/Python, PostgreSQL, Apache Hive, PyArrow, Python libraries (Selenium, Folium, Bokeh, Dask, R, and many others)
- *Properties*:
    - A bash script for submitting work to the Slurm resource manager (ingests entity location data and generates video clips displaying entity movement and damage state, and uses lat/long coordinates)
    - The Flask Python framework creates a user-friendly interface to the Magpie (Lawrence Livermore National Laboratory) Hadoop

Distributed File System (HDFS) stack and also to the PostgreSQL back-ends.

- The Magpie stack functions as a virtualized HDFS running on the General Parallel File System and allows one to extract, transfer, load, and analyse large and disparate data sets such as Common Hardware System (CHS) data, which are burdensome for typical desktop apps (Excel or Access) and also more standardized ones such as Expedient Leader/Follower Test Incident Report (ExLF TIR) data as well.
- PyArrow integration used to provide external-to-internal data transfer (and vice versa) within the stack.
- The user interface can be tailored to customer requirements (for example CHS), since updated Flask templates can be created (with Jinja) by reusing the code with minimal updates. This UI combined with other supported Python libraries (Selenium, Folium, Bokeh, Dask, R, and many others) enables us to customize the analysis, reporting, and visualization libraries to provide useful insight into the data. All of these technologies are commercially available and open source, and are used to provide solutions to customers.
- Flask/Python framework integrating multiple analytical libraries
- Data storage in PostgreSQL and Apache Hive, also PyArrow integrated for Python interoperability
- Containerization of analysis environment for portability to multiple systems
- Video creation functionality integrated to run during the DRL process itself

Figure 4 shows an example visualization of the terrain and Blue Force and Opposition Force positions with equipment.

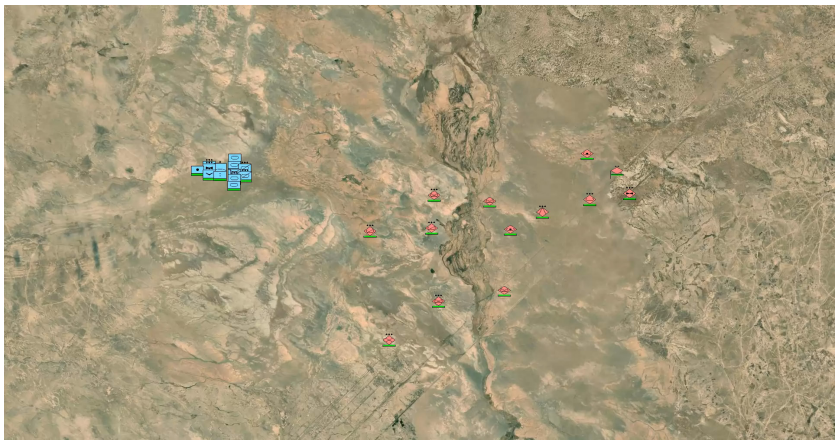

Fig. 4 Visualization example: Tiger Claw scenario

5. Conclusion and Next Steps

The tools developed for artificial intelligence for C2 in Multi-Domain Operations – Director's Strategic Initiative project (C2DSI) are very versatile. They are adaptable to other kinds of data sources with appropriate modifications. In particular, the data should be time-dependent for utilizing the full capability of visualization module. In future, the modules will be made even more versatile for working with other scenarios.

6. References

1. Narayanan P, Vindiola M, Park S, Logie A, Waytowich N, Mittrick M, Richardson J, Asher D, Kott A. First-year report of ARL Director's Strategic Initiative (FY20–23): artificial intelligence (AI) for command and control (C2) of Multi-Domain Operations (MDO). DEVCOM Army Research Laboratory; 2021 May. Report No: ARL-TR-9192.

2. Karagiannakos S. The idea behind Actor-Critics and how A2C and A3C improve them. AI Summer; 2018 Nov 17 [accessed 2021 July]. https://theaisummer.com/Actor_critics/.

3. Ciuiu D. On Jarque-Bera normality test. ResearchGate; 2008 [accessed 2021 July]. https://www.researchgate.net/publication/23645608.

Appendix. Long Short-Term Memory (LSTM) and A2C

General formulation of RL

1. The given collection of entities $(S, A, R, P, \gamma, s_0)$ is the starting set of RL, where

   (i) (S) = the set of all states S

   (ii) (A) = the set of all actions A

   (iii) R = reward (given as a function)

   (iv) P = the transition probability

   (v) γ = the discount factor, and

   (vi) s_0 = the initial state,

2. Any one of the following approaches control the RL agent's behavior:

   (i) *Policy function* $\boldsymbol{\pi}$**:** It represents an action A taken by the agent in a given state S and can be one of the following: (a) Deterministic policy $\pi_\theta(s) = a$, or (b) Stochastic policy with P as the probability of action a in state s having parameters θ: $\pi_\theta(s_t, a_t) = P[a_t|s_t, Q]$

   (ii) *Value function as Q-function* $Q^\pi(s_t, a_t)$**:** It predicts the expected Total Future Reward (TFR) for given (s_t, a_t) pair under policy π with discount factor γ

$$Q^\pi(s_t, a_t) = E(r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots | s_t, a_t)$$

Here r_{t+1} is the reward at time-step t+1 and so on.

   (iii) *Value function as V-function* $V^\pi(a_t)$ is the value of an action a under all the states

$$V^\pi(a_t) = \sum_{s_t} Q^\pi(s_t, a_t)\, \pi_\theta(s_t, a_t)$$

   (iv) *Value function as advantage or A-function* $A^\pi(s_t, a_t)$ *is the difference between Q-function and V-function* for other actions that Agent could have taken.

$$A^\pi(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(a_t)$$

   (v) *Value function* measuring the Model or agent's representation of the environment

Deep reinforcement learning uses a deep neural network to represent the value or policy function and optimizes the loss function by stochastic gradient descent method.

A.1 Advantage Actor Critic (A2C) Algorithm

The reinforcement learning methods belong to two broad classes of methods:

1) Based on value functions that assign each state-action pair to a value

2) Based or optimizing the policies directly without using value functions

Actor critic methods combines both approaches. The critic is a value-based neural network and it measures how good the action is. The actor is a policy-based neural network and it controls the RL agent's action. Both run in parallel and the real-time feedback from critic improves the actor. Figure A-1 shows a diagram of the A2C method.

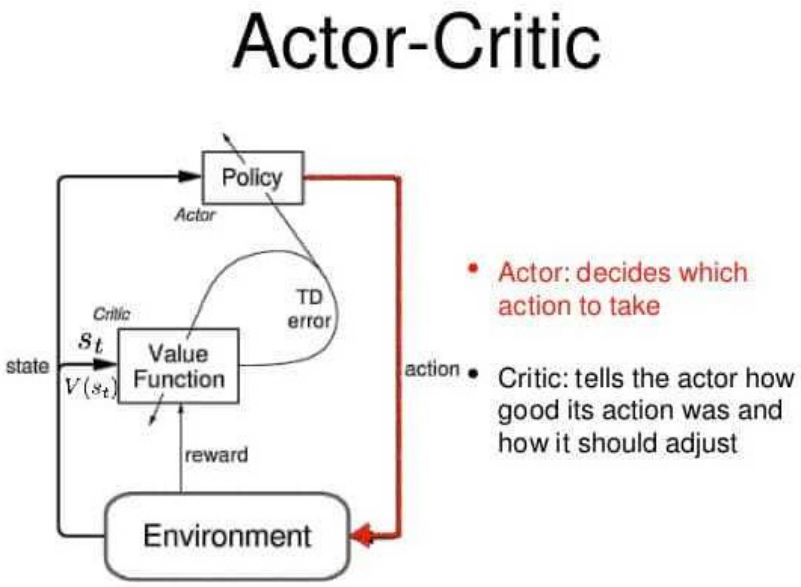

Fig A-1 A2C diagram (from Sutton and Barlo[1])

In A2C, the value function is the advantage function

$$A(s_t, a_t) = Q(s_t, a_t) - V(s_t) \tag{A-1}$$

Here $Q(s_t, a_t)$ is the Q-value for the action in that state (or maximum future reward in that state) and $V(s_t)$ is the average value of that state. A positive advantage pushes the gradient in that direction and vice versa.

[1] Sutton RS, Barlow AC. Reinforcement learning: an introduction. MIT Press; 1998.

A.2 Long Short-Term Memory (LSTM) Algorithm

The general structure of recurrent neural network includes input, output, and hidden layers. The last one contains neurons with memory so it allows information to persist. In general, their information content does not persist for a long time due to vanishing or blowing up of the gradients of the parameters.

The LSTM solves this problem by using special type of cells consisting of three gates:

1) Forget gate: It chooses whether to keep information of previous timestamp needs to be kept or forgotten

2) Input gate: It adds or updates information.

3) Output gate: It passes the updated information to the next timestamp.

In addition, it has two states:

1) Hidden state or short-term memory with information at current timestamp as $H(t)$ and at previous timestamp as $H(t-1)$.

2) Cell state or long-term memory with information at current timestamp as $C(t)$ and at previous timestamp as $C(t-1)$

The updating for different states is done using sigmoid, tanh, and other similar functions.

List of Symbols, Abbreviations, and Acronyms

A2C	Advantage Actor Critic
AI	artificial intelligence
C2	command and control
C2DSI	artificial intelligence for C2 in Multi-Domain Operations – Director's Strategic Initiative project (also known as AIC2MDO – DSI)
C4I	command, control, communications, computers and intelligence
CHS	Common Hardware System
CoA	course of action
DNN	deep neural network
DRL	deep RL
ExLF TIR	Expedient Leader/Follower Test Incident Report
HDFS	Hadoop Distributed File System
LSTM	long short-term memory
ML	machine learning
NN	neural network
RL	reinforcement learning
SME	subject-matter expert
UI	user interface

1 (PDF)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA
1 (PDF)	DEVCOM ARL FCDD RLD DCI TECH LIB
3 (PDF)	DEVCOM ARL FCDD RLC NC V MISHRA C ANDERSON A CHILDS