ARL-TR-9927 ● JUNE 2024

DEVCOM ARMY RESEARCH LABORATORY

# Neuro-Symbolic Learning for Context-Aware Real-Time Human–Agent Interaction

by Vinod K Mishra, Julian de Gortari Briseno, and Mani Srivastava

DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

## 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-9927 ● JUNE 2024


# Neuro-Symbolic Learning for Context-Aware Real-Time Human–Agent Interaction

**Vinod K Mishra**
*DEVCOM Army Research Laboratory*

**Julian de Gortari Briseno and Mani Srivastava**
*University of California*

**DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.**

# REPORT DOCUMENTATION PAGE

| 1. REPORT DATE | 2. REPORT TYPE | 3. DATES COVERED | |
|---|---|---|---|
| | | START DATE | END DATE |
| June 2024 | Technical Report | 1 February 2024 | 30 April 2024 |

**4. TITLE AND SUBTITLE**
Neuro-Symbolic Learning for Context-Aware Real-Time Human–Agent Interaction

| 5a. CONTRACT NUMBER | 5b. GRANT NUMBER | 5c. PROGRAM ELEMENT NUMBER |
|---|---|---|
| W911NF-17-S-0003 | | |
| **5d. PROJECT NUMBER** | **5e. TASK NUMBER** | **5f. WORK UNIT NUMBER** |
| | | |

**6. AUTHOR(S)**
Vinod K Mishra, Julian de Gortari Briseno, and Mani Srivastava

| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) | 8. PERFORMING ORGANIZATION REPORT NUMBER |
|---|---|
| DEVCOM Army Research Laboratory<br>ATTN: FCDD-RLA-NC<br>Aberdeen Proving Ground, MD 21005 | ARL-TR-9927 |

| 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**
DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**
Robotic agents have been developed to perform a number of tasks under human command and sometimes independently as well. Human–agent teams can extend the capabilities of both humans and robots. Multi-Agent Reinforcement Learning (MARL) is a natural approach for such teams. The interaction within a team composed of human and robot agents considered so far ignores the role of physical context in MARL. Here, we present a distributed online MARL in such a scenario incorporating wireless communication and physical movement supplemented with intra-team communication. We present the system, its mathematical analysis, and some initial experimental results herein.

**15. SUBJECT TERMS**
symbolism; connectionism; neuro-symbolic; human–agent teaming; reinforcement learning; Network, Cyber, and Computational Sciences

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES |
|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | C. THIS PAGE | | |
| UNCLASSIFIED | UNCLASSIFIED | UNCLASSIFIED | UU | 20 |

| 19a. NAME OF RESPONSIBLE PERSON | 19b. PHONE NUMBER *(Include area code)* |
|---|---|
| Vinod K Mishra | (410) 278-0114 |

**STANDARD FORM 298 (REV. 5/2020)**
*Prescribed by ANSI Std. Z39.18*

## Contents

## List of Tables

# 1. Introduction and Background

In the historical development of machine learning (ML), two different ways to model human intelligence (i.e., symbolic and connectionist approaches) have been emphasized. In this work, we have combined both and applied them to a simple task involving multi-agent coordination with the Perception-Cognition-Communication-Action (PCCA) loop.

## 1.1 Approaches Focused on Symbolism

The approach focused on symbolism uses rules of logic to work with objects represented as symbols.

We start with two sets: (1) Set A (collection of propositions), and (2) Set B (general principles). Then, there are three different types of reasoning verified by logicians.

- Deductive: One derives A from B only when A is a formal logical consequence of B.
- Inductive: Given A, one infers B.
- Abductive: One infers A as an explanation of B without rigorous logical analysis, thus allowing preconditions from consequences. This is opposite to induction in direction.

All types involve working with symbols. They also require relatively few input symbols for representing knowledge of the target system and internal functioning of the programs are transparent.

It was discovered that this approach does not work well with noisy and ambiguous real-world data.

## 1.2 Approaches Focused on Connectionism

When cognition is represented by neural networks (NNs), it captures that as interconnected networks of uniform “neurons” like units. This led to the following observations.

- Cognitive processes (attention, problem-solving, memory, learning, decision-making, language, perception, imagination, and logic reasoning) arise from neurons and their connections.
- Learning occurs through weight modification minimizing cumulative error and with discovery of statistical patterns in the input data.

Despite many successes, this approach also has some shortcomings (e.g., lack of compositional generalization and a verifiable train of logic, and no understanding of why a decision was made). Application of this approach to critical areas like medical diagnosis, autonomous driving, and mathematical reasoning has proved very problematic.

## 1.3 Approach Based on Neuro-symbolic Integration

Recently, researchers have tried to combine the first two approaches as "Neuro-Symbolic" (NeSy) approach to AI. There are six broad types in which both neural (N) and symbolic (S) components can be combined:

- Symbols as both input and output with respect to NNs (denoted as S-N-S).
- Neural as subroutine inside overall symbolic approach (denoted as S[N]).
- Neural and symbolic both at the same co-routine level (denoted as N|S).
- Symbolic rules integrated with NN's architecture or training (denoted as N:S->N).
- Symbolic as soft constraint on loss function in training NN (denoted as N_S).
- Symbolic engine directly embedded inside an NN engine, logical reasoning as tensor calculus (denoted as N[S]).

The environment and goals of a given problem leads to one or another to be chosen as all of them have strengths and weaknesses.

## 1.4 Knowledge Representations

In NeSy-centric approach, a strong basis of logic is a necessity. Some common terms capturing this approach is described in this section.

*Propositional logic*, sometimes called zeroth-order logic, deals with propositions, their relations, logical connectives, and arguments based on them. It is the foundation on which other higher order logics are built.

Propositional logic has the following formal components.

- A countably infinite set of elements called proposition symbols or proposition variables (typically the letters *p*, *q*, *r,* etc.).
- A finite set of elements called operator symbols or logical connectives.
- A finite set of transformation rules or inference rules.

- A countable *set of initial points or axioms.*

*First-order logic,* also known as predicate or quantificational logic, contains propositional logic structure and adds a few more features. It contains non-logical objects and uses quantified variables related to them. Some examples are set theory, group theory, and formal theory of arithmetic.

*Knowledge graph* is a data model structured as a graph used to store interlinked objects, concepts, and so forth. Additionally, they also indicate relationships among the stored entities. Recently their use has extended to Graph Neural Networks (GNNs) as well.

## 2. Human–Agent Teaming

Robotic agents have been developed to perform a great number of tasks under human command and sometimes independently as well. The next step in this evolution is to form human–agent teams (HATs) to extend the capabilities of both humans and robots. Multi-Agent Reinforcement Learning (MARL) is a natural ML approach for such teams.

### 2.1 Problems with the Current MARL Approaches to HAT

So far MARL applications to HATs have faced many problems as shown in the following.

- In HAT, most of the agent actions (communication, sensing, motion, manipulation, etc.) are influenced by the physical context and potentially influence each other through physical world pathways. This awareness is also needed for intra-team communication. Existing work in MARL ignores such physical couplings.
- Currently, HAT applications do not incorporate considerations of wireless communication and physical movement into distributed online reinforcement learning (RL). Many of the existing methods
    - target problems lacking any physical environment,[1]
    - simulate physical environment by parallelizing data collection and exploration by agents, and train for the policy from the collected data,[2] and
    - target such use cases with distributed RL packages, such as RLlib.[3]
- For the artificial intelligence (AI) agents, moving in real time through physical space, the network topology and capacity change as a function of

both relative positions of agents (e.g., longer links will have lower rates) and absolute agents of agents (e.g., an agent inside a building will have poorer quality link). For efficient operation, the learning algorithm must therefore consider the current network context when distributing the tasks.

- As the mobility of AI agents and humans is not random, there would be predictability both in (1) the short time horizon due to continuity of motion and constraints of the terrain, and in (2) a medium time-horizon due to mission tactics and procedures influencing movement patterns. This should be reflected in the HAT algorithm.

## 2.2 Our General Approach

Synchronous algorithms cause all agents to work in lock step, which is very inefficient. The following changes were made in approaching this problem:

- Using asynchronous algorithms sacrifices some across-agent consistency of policy but is more responsive.[4]
- Use of physical context to attach weights to agents.[5]

Predictability is a feature of this approach.

- It occurs directly via the learnt policy for AI agents, and indirectly for human agents.
- It is used to
    - optimize the distributed learning by intelligently organizing computational subtasks across the network nodes,
    - co-optimize the computation and movement in a more sophisticated version, and
    - explore movement coordination and task communication involving sensing of the physical world and targeting of actuation changing the physical environment.

These different objectives may often be at odds (e.g., the best location for sensing may be different from the one for communication) and may benefit from cooperation. This will require a multi-objective RL approach.

## 2.3 Mathematical Formulation

The problem agents face in the simulated environment can be defined as a decentralized partially observable Markov decision process (Dec-POMDP)

augmented with communication, formalized by the tuple ($S$, $\{A_i\}$, $T$, $R$, $\{\Omega_i\}$, $O$, $\gamma$) as given Table 1.

**Table 1 Elements of POMDP**

| Parameter | Element |
|---|---|
| $S$ | Set of states |
| $A_i$ | Set of actions for agent $i$ |
| $T$ | Set of conditional transition probabilities between states |
| $R$ | Reward function |
| $\Omega_i$ | Set of observations for agent $i$ |
| $O$ | Set of conditional observation probabilities |
| $\gamma$ | Discount factor |

We define several metrics for characterizing the HAT operations.

**Individual Quality of Work (IQW)**. IQW is a team score for evaluating the team performance in accordance with the objective encoded into the scene. It consists of a quality factor and a speed factor. For each agent $i$, it is calculated as follows:

$$IQW[i] = \max\left(0, \frac{O_{d_{Col}}[i] - O_{nd_{Col}}[i] - O_{d_{Drop}}[i]}{N_{d_{obj}}}\right)$$

where

$O_{d_{Col}}$ = the truly dangerous objects collected,

$O_{nd_{Col}}$ = the truly non-dangerous objects collected,

$O_{d_{Drop}}$ = the truly dangerous objects dropped accidentally, and

$N_{d_{obj}}$ = the total number of truly dangerous objects in the scene.

The agents are penalized for not carefully choosing the type of objects being collected, as well as for not coordinating correctly with their teammates.

**Team Quality of Work (TQW)**. TQW is the sum over the IQW of all agents and gives an understanding of the aggregate quality of actions taken by all agents.

**Team Speed of Work (TSW)**. TSQ characterizes the speed factor.

$$TSW = \frac{T_{timeout}}{\max\{T_{timeout}/10, \min(T, T_{timeout})\}}$$

where

$T_{timeout}$= a predetermined timeout value after which the session ends automatically, 20 min in our experiments.

T = the actual ending time.

Agents are not penalized for using the whole $T_{timeout}$, and any improvement in time completion over it gets a corresponding bonus. Additionally, a limit is imposed on how fast a session may end and its corresponding award.

**TeamScore**. It is calculated using the work quality and speed and it compares the algorithms with each other:

$$TeamScore = TQW * TSW$$

Some other similar metrics include

- the (dangerous objects collected/the total number of dangerous objects existing in the scene) fraction and
- the (dangerous objects collected/the total number of objects collected) fraction.

Both metrics together let us understand better the factors influencing the $TeamScore$. Also, the average number of times sensors are activated measures the effectiveness of coordination between agents based on their need to individually collect redundant measurements.

## 3. Heuristic Approach

The ideas outlined previously were used in setting up a "toy" model as outlined here. A simple setup was used to put together a model demonstrating the basic ideas.

### 3.1 Simulation Setup and Process

**Setup**. The simulation experiments were setup in the following way:

- No. of agents in a team = 5, No. of sessions = 40, Duration of each session = 20 min
- No. of objects in each session = 20, Area of scenario = 20 × 20 m, No. of objects assigned in each room = 5
- Randomly chosen parameters are (1) starting positions of both agents and objects within a room, (2) confidence over each agents' sensors, and (3) the true danger status and weight of each object

**Initial Conditions**. The starting conditions and values are as follows:

- A dangerous status is assigned to an object with a probability of 0.3.
- A weight of 1 is assigned with a probability of 0.5. Larger weights are assigned with decreasing probability.
- The maximum weight an object can be assigned = the total number of agents in the scene.
- Cell size for the occupancy map = 1 $m^2$.
- Sensing radius = 2 m.
- Radius within which agents need to be to increase the strength of another agent = 3 m.
- Communication radius = 5 m.
- Goal radius = 3 m.

**Process**. With respect to the heuristic planning, three variations were evaluated:

1) Centralized planning with assignable specialized roles (Centralized). The coordinator agent drops its privilege in the end to be able to carry dangerous objects that demand the effort of all agents in the scene if such objects exist. This approach is more structured and able to reduce the number of total sensor activations per session without necessarily impacting the overall team score. Thus, it shows the benefits of having a single point of information fusion.
2) Decentralized planning with specialized roles (DecSpecialized). The role of scouts is assigned to two agents, role of lifters assigned to three agents, and the scouts may become lifters after having sensed all objects.
3) Decentralized planning with no specialized role*s* (DecGeneral). The agents roam freely around the scene wasting time in redundant sensing. It is opposite to the more structured Centralized method. The unstructured nature of this method allows its agents to maximize collection of objects in detriment of their true danger status. This is contrary to both Centralized and DecSpecialized methods as they place more emphasis on sensing and fusion of results but lose on the total amount of collected dangerous objects.

Team scores were similar for all these high-level strategies. It begs the question of whether the problem is instead with the shared low-level strategies that are also derived from heuristics. Moreover, given the dynamic nature of the simulated scenario, one cannot assume a single heuristic will be valid at all points in time.

### 3.2 Simulation Results

Our simulations show the following results:

- The *TeamScore* for the heuristic strategies was less than or equal to 0.3, independent of the different planning methods.
- Even though the heuristic methods are optimized to reduce completion time and the amount of idle time a particular agent experiences, they fail to make effective use of agents' collective potential.

Among the more structured approaches:

- The Centralized method reduces the number of total sensor activations per session without impacting the *TeamScore*, thus showing the benefits of having a single point of information fusion.
- The DecGeneral method does not show this effect, which may be due to the agents roaming freely around the scene wasting time in redundant sensing. They maximize collection of objects in detriment of their true danger status. This is contrary to both Centralized and DecSpecialized, which emphasize sensing and fusion of results more but lose on the total number of collected dangerous objects.

The similarity in *TeamScore* values even when using different high-level strategies lead to the question: Are the shared low-level strategies the problems as they are also derived from heuristics? Given the dynamic nature of the simulated scenario, one cannot assume a single heuristic will be valid at all points in time.

## 4. Reward Machines Experiments

The heuristic strategies presented in Section 3 have hard-coded logic. This limits teams of agents' generalization capabilities and scalability to dynamic environments. It is a human bias but can be changed by incorporating a learning mechanism into agents' planning strategies through RL. The agents can learn from their experiences and ideally adapt to any environment.

The RL approach of reward machines (RMs) can further accelerate the learning process by mixing it with human knowledge. RM defines finite-state automatons representing the reward function of a task in terms of high-level events. This benefits agent cooperation by providing the abstraction and the interpretability typically lacking in canonical RL approaches. It has shown to be effective in the single agent setting for modeling non-Markovian rewards, for learning policies that

generalize across different instances of a task,[6] and for supporting re-usability of learned policies across tasks of increased complexity.[7]

In multi-agent cooperative teams, RM has shown promising results in grid word domains.[8,9] Here, the global (team-level) reward function is decomposed into agent-specific RMs. They capture the respective agent's subtask and relevant teammate communication for successful cooperation.

The effectiveness of RMs for the multi-agent PCCA coordination problem was explored using a simplified version of the simulated environment. This version does not incorporate the physics engine, allows for automatic sensing, is fully visible, and contains two agents and one object of weight two.

Each agent was trained separately using the RM algorithm in Ardon et al.[9] and then the performance of the learned agents' policies in the physics-enabled simulated environment was evaluated. It demonstrated how the joint execution of the agents' policies leads to a team-level solution of the global task.

## 4.1 System Setup

Table 2 presents the properties of the RL agent used for this investigation.

**Table 2 RL agent**

| Parameters | Purpose |
|---|---|
| $r_k$ | Per-timestep reward obtained by the team of agents |
| $T$ | Maximum allocated time to solve the task |
| $s$ | State containing environment information (e.g., the position of each agent and object in the scene, and the sensing results of objects close to an agent) |
| $r_k$ | Immediate reward at a $k$-th timestep, it is 10 if a dangerous object is dropped in the goal area at that step and $-0.01$ otherwise. |
| R | $R = \sum_{k=0}^{T} \gamma r_k$ = the $\gamma$-discounted total return of an episode |

Notes: The actions agent can perform = *do nothing*, *up*, *left*, *right*, *down*, *pick up*, *drop*
Goal = find the policy $\pi(a|s)$ to maximize R

The individual agents are given predefined RMs that model the subtasks needed to complete the overall high-level task. Each agent is trained using the Q-Learning algorithm for Reward Machines (QRM).[10]

## 4.2 Experimental Results

The preliminary results show the potential benefits of using RL with RMs for coordination of PCCA loops within multi-agent teams. The abstraction of representation and subtasks decomposition by RMs has one great advantage. It minimizes the problem of transferring the policies trained in the simplified

environment to the physics-enabled simulated environment while keeping an acceptable level of performance.

Table 3 shows the experimental parameters, and it was found that heuristic strategies converge around an average team score value of 0.3. This was independent of the team structures designed to reduce completion time and achieve a good team score. It may indicate that the problem may lie with heuristic-based low-level strategies. Finally, a TSW of 1 was obtained in all cases, demonstrating that teams were unable to finish before the time limit in all cases.

**Table 3 Simulation parameters**

| Parameters | Value |
|---|---|
| Team size | 5 |
| Number of objects | 20 |
| Session duration | 20 min |
| Occupancy map cell size | 1 $m^2$ |
| Sensing radius | 2 m |
| Strength contributing radius | 3 m |
| Communication radius | 5 m |
| Goal radius | 3 m |

## 5. Conclusions and Future Work

In this work, a simulated environment and a set of initial strategies toward solving the problem of multi-agent PCCA loop coordination was presented. However, this work has taken only an initial step toward the solution of such a complex problem.

Solving more challenging tasks beyond the toy example considered here requires further work to address the observed limitations, which include the following.

- This approach needs to be extended to span the single agent RM learning with those of their associated RL policies. At the same time, the agents must achieve the goals of the global cooperative task.[9]
- The RM representations need to be extended to allow the learning of subgoals (RM transitions) as probabilistic first-order formulae. This will make the RM methods more scalable and applicable to more complex tasks including the level of uncertainty of sensing observations.
- To allow better generalization and scalability, a function approximation method, such as Deep Deterministic Policy Gradient or Deep Q-Networking[10] could be used with RMs for learning policies.

- A better understanding of the underlying physics of the environment, a richer scheme of communication between agents, and accounts for better explainability will be also necessary for going forward.
- A better handling of the uncertainty of sensing measurements will be needed to coordinate sensing actions more effectively.
- Finally, the issues arising with mixed human–AI teams, such as the asymmetry in perception and cognition, the barriers of natural language communication, and the problems of trust calibration need to be explored.

## 6. References


1. Hoffman M, Shahriari B, Aslanides J, Barth-Maron G, Behbahani F, Norman T, Abdolmaleki A, et al. Acme: a research framework for distributed reinforcement learning. arXiv preprint arXiv:2006.00979(2020).

2. Balaji B, Mallya S, Genc S, Gupta S, Dirac L, Khare V, Roy G, et al. Deepracer: autonomous racing platform for experimentation with sim2real reinforcement learning. 2020 IEEE International Conference on Robotics and Automation (ICRA). 2020;2746–2754. IEEE.

3. Liang E, Liaw R, Nishihara R, Moritz P, Fox R, Goldberg K, Gonzalez J, Jordan M, Stoica I. RLlib: abstractions for distributed reinforcement learning. International Conference on Machine Learning. 2018;3053–3062. PMLR.

4. Mnih V, Puigdomenech Badia A, Mirza M, Graves A, Lillicrap T, Harley T, Silver D, Kavukcuoglu K. Asynchronous methods for deep reinforcement learning. International Conference on Machine Learning. 2016;1928–1937. PMLR.

5. Espeholt L, Soyer H, Munos R, Simonyan K, Mnih V, Ward T, Doron Y, et al. Impala: scalable distributed deep-RL with importance weighted actor-learner architectures. International Conference on Machine Learning. 2018;1407-1416. PMLR.

6. Wu SA, Wang RE, Evans JA, Tenenbaum JB, Parkes DC, Kleiman-Weiner M. Too many cooks: Bayesian inference for coordinating multi-agent collaboration. Topics in Cognitive Science. 2021;13(2):414–432.

7. Nguyen D, Venkatesh S, Nguyen P, Tran T. Theory of mind with guilt aversion facilitates cooperative reinforcement learning. Proceedings of the Asian Conference on Machine Learning (ACML); 2020, 33–48.

8. Neary C, Xu Z, Wu B, Topcu U. Reward machines for cooperative multi-agent reinforcement learning. Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS); 2021, 934–942.

9. Ardon L, Furelos-Blanco D, Russo A. Learning reward machines in cooperative multi-agent tasks. Proceedings of Neuro-Symbolic AI for Agent and Multi-Agent Systems (NeSyMAS) Workshop at the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 2023.

10. Toro Icarte R, Klassen T, Valenzano R, McIlraith S. Using reward machines for high-level task specification and decomposition in reinforcement learning. Proceedings of the International Conference on Machine Learning (ICML); 2018, 2107–2116.

## List of Symbols, Abbreviations, and Acronyms

| | |
|---|---|
| AI | artificial intelligence |
| Dec-POMDP | decentralized partially observable Markov decision process |
| GNN | Graph Neural Network |
| HAT | human–agent team |
| IQW | Individual Quality of Work |
| MARL | Multi-Agent Reinforcement Learning |
| ML | machine learning |
| N | neural |
| NeSy | Neuro-Symbolic |
| NN | neural network |
| PCCA | Perception-Cognition-Communication-Action |
| QRM | Q-Learning algorithm for Reward Machines |
| RL | reinforcement learning |
| RM | reward machine |
| S | symbolic |
| TQW | Team Quality of Work |
| TSW | Team Speed of Work |