Continual and Real-time Learning for Modeling Combat Identification in a Tactical Environment

Ying Zhao
Naval Postgraduate School, Monterey, CA

Nate Derbinsky
Northeastern University, Boston, MA

Lydia Y. Wong, Jonah Sonnenshein
Metron, Inc., San Diego, CA

Tony Kendall
Naval Postgraduate School, Monterey, CA

Abstract

This paper presents data from a working ML/AI system being applied to simulated combat identification (CID) scenarios. The application, which integrates continual and real-time reinforcement learning with the Soar cognitive architecture, learns from online experience to accurately predict hostile airborne objects based on kinematic characteristics. Our results suggest that this integration framework can scale to big data scenarios to learn tactically in real-time.

1 Introduction

The application discussed in this paper concerns two concepts for Naval defense, as shown in Figure 1: common tactical air picture (CTAP; left) and combat identification (CID; right). The CTAP process collects, processes, and analyzes data from a vast network of sensors, platforms, and decision makers and provide situational awareness to decision makers. The CID process locates and labels critical airborne objects (as friendly, hostile, or neutral) with high precision and efficiency based on a CTAP as part of the core *kill chain* process.

The existing methods of CTAP and CID involve wide ranges of participating platforms, participating sensors, networks and system. Challenges in the CID process include (1) extremely short time for fusion, decision-making, and targeting; (2) uncertain and/or missing data outside sensor (e.g. radar, radio) ranges; (3) manual decision-making; (4) heterogeneous data sources for fusion and decision-making; and (5) multiple decision-makers in the loop.

Specifically, the CTAP and CID problems can be seen as both big data and no data. On one hand, the data used for CID comes from a combination of massively cooperative and non-cooperative sensors (where, typically, each sensor collects certain attributes). The big CID data then needs to be fused over time and space since they are collected in a distributed, continuous and real-time fashion. There is limited computation and storage available for sailors in a tactical environment, therefore, using and consuming this big data in a continuous fashion provides an extra challenge. On the other hand, adversaries often conceal and change their true intentions, therefore rare or no data are observed for analyzing anomalous and adversarial behavior. Therefore, the CTAP, CID, and kill chain problems are challenging application areas for analytic, machine learning (ML), or artificial intelligence (AI) methods.

2 Tactical Decision Making and Online, Continual, Adaptive, and Real-time Learning Required in CID

The focus of this work is on decision-making in the CID process. Figure 1 (right) shows watch stations involved in a CID decision-making process for a Combat Information Center (CIC). CID

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

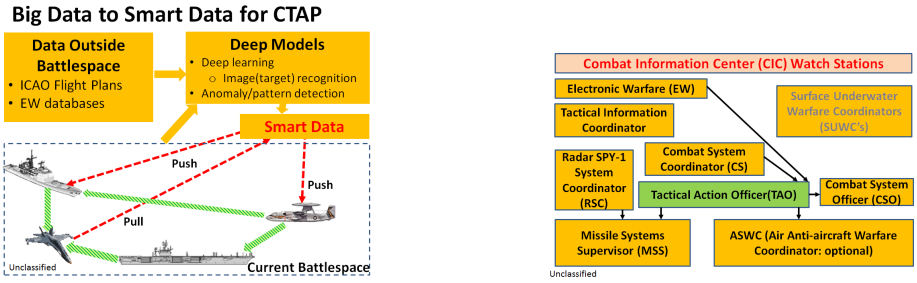

Figure 1: Left: The Common Tactical Air Picture (CTAP) process collects, processes, and analyzes data from a vast network of sensors, platforms, and decision-makers and provides situational awareness to sailors. Right: a Tactical Action Officer (TAO) is a core decision-maker in a CID process with a very high cognitive burden.

decision-making can be very manual, requiring heterogeneous data sources and involving multiple stake-holders (e.g. watch stations). Decision makers such as Tactical Action Officers (TAOs) and Air Defense Officers (ADOs) are constantly overwhelmed with the cognitive reasoning required [5].

Our goal is to apply ML/AI methods to assist sailors by reducing their cognitive burden and improve timely CID decision-making in a tactical environment, where a tactical action is a conceptual action aimed at achieving a specific goal (e.g. deciding if an airborne object is a friend or foe).

In a tactical environment, a CID application needs to be trained initially offline and then continually improved and adapted from a stream of new data, since decision makers continuously receive new data and need to continuously update their decisions. The big data is updated in real-time, therefore, machine learning has to be performed in an online, continuous, adaptive, and real-time fashion as each and every new datum is streamed into the system. A real-time learning algorithm needs to update its parameters for each new data point so learning occurs at any moment. Machine learning in this context gives decision makers a definite tactical edge.

In this paper, we investigate the efficacy of one such AI system for using, fusing, and improving existing knowledge models with the goal of timely and automatic decision-making and reduced cognitive burden in the operational environment. The system builds on an existing methodology [7], combining unsupervised and reinforcement learning (RL) with the Soar cognitive architecture [3, 4]. The contribution of the paper is that we present and demonstrate a working implementation using a simulation scenario to accurately predict hostile airborne objects based on kinematic characteristics. The system can potentially scale up to big data and large knowledge bases related to CID to fuse and learn from data at a symbolic and tactical level in real-time.

3 Continual and Real-time Learning via Soar Reinforcement Learning (Soar-RL) for CID

Soar [4] is a cognitive architecture that scalably integrates a rule-based AI system with many other capabilities, including RL and long-term memory. The main decision cycle involves rules that propose new operators (e.g. internal decision or external actions), as well as preferences for selecting amongst them; an architectural operator-selection process; and application rules that modify agent state. The reinforcement-learning module (Soar-RL) modifies numeric preferences for selecting operators based on a reward signal, either via internal or external source(s) – importantly, Soar-RL learns in an online, incremental fashion and thus does not require batch processing of (potentially big) data. Soar has been used in modeling large-scale complex cognitive functions for processes such as those in a kill chain [2].

A CTAP/CID application typically includes "track" data for airborne objects (each to be classified as either friendly, neutral, or hostile). A track is a time series of information for the object, including data observed by sensors and needed to be predicted as follows:

- Inputs: Longitude, Latitude, Altitude, Speed, Acceleration, Heading (predicted compass direction), IFF (Identification, friend or foe – an identification system designed for identifying

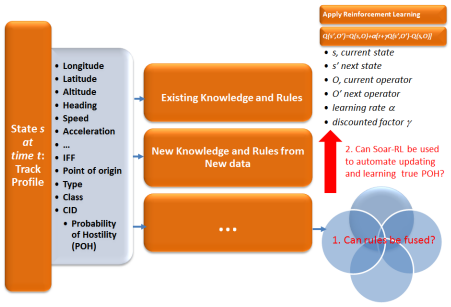

Figure 2: Soar-RL applied to CID

friends), Point of origin (where the airborne object came from), Aircraft type, Aircraft class, etc.

- Outputs: CID (probability of hostility (POH), where 1 is a friend and 0 is an enemy)

Figure 2 illustrates how a track state is mapped into Soar-RL. Each state goes through multiple production rules, which are matched either from the existing knowledge and rules or those discovered from new data. Soar-RL fuses the rules together as evidence for decision-making. There are two decisions or classes considered in this problem: "hostile" or "not hostile" and the input data for classification/decision making are the kinematic characteristics such as speed, altitude, heading, and the changes of these kinematic characteristics.

As shown in Equation (1), Soar-RL is implemented in a typical RL implementation involving a recursive formula that is widely accepted in the RL research and literature [6][8]. Since we only consider an on-policy setting or SARSA, $Q(s_{t+1}, a) = 0$ in Equation (1). Therefore, $Q(s_{t+1}, a_{t+1})$ is updated continuously for each time point and immediate reward r.

$$Q(s_{t+1}, a_{t+1}) = Q(s_t, a_t) + \alpha[r + \gamma \max_{a \in A} Q(s_{t+1}, a) - Q(s_t, a_t)] \tag{1}$$

In Soar-RL, each rule is associated with a preference on which an operator decision is made. The goal for Soar-RL is to learn and adjust the preferences of the initial rules dynamically and in real-time when a new data point is presented. The preferences of the rules are related to the POHs but not restricted to the value ranges of probabilities. The preferences are adjusted/learned/trained based Equation (1). The actions for the reinforcement learning are the two decisions (hostile or not) for an airborne object based on the observable kinematics for the object at time t. The preference of a decision (i.e. hostile or not) is the expected total preference for each decision (action) at time t, computed recursively.

One advantage of applying Soar-RL to CTAP/CID applications is the relative ease of incorporating existing knowledge via English-like production rules that may originate via a variety of sources, including knowledge bases or learning/discovery by ML/AI algorithms.

4 Simulation Results

To illustrate the methods, we used the Naval Simulation System (NSS) [10], which employs Monte Carlo simulation to model Naval platforms and missions, including sensors, tactical pictures, and the command and control of assets in a tactical scenario.

Each scenario run was generated according to a different random seed number specified by an NSS user. There were about 2690 tracks in the simulation, 449 (16.7%) tracks were hostile. Each different random seed generates a slightly different set of track data. Four random seeds $(10, 11, 1256, 23576)$ with corresponding track data were reported in this paper. The goal is to classify airborne object hostility based on the kinematic features.

Table 1 summarizes and compares classification error rates for two Soar-RL settings (1. initial preferences were computed from big data method [9]; 2. initial preferences are zeros) with a few

other supervised batch learning methods such as decision trees (i.e. J48), logistic regression (LR), and Naïve Bayes (NB) from the tool Weka [1]. The online Soar-RL performed comparatively well with the other batch machine learning methods.

Table 1: Classification Errors Comparison

	J48	LR	NB	Soar-RL 1	Soar-RL 2
Train (seed=10)	0.26%	0.93%	1.19%	2.8%	1.9%
Test (seed=11)	0.24%	0.75%	0.92%	1.4%	0.6%
Test (seed=1256)	1.24%	1.02%	1.17%	1.9%	1.8%
Test (seed=23576)	0.63%	1.32%	1.56%	2.2%	1.2%

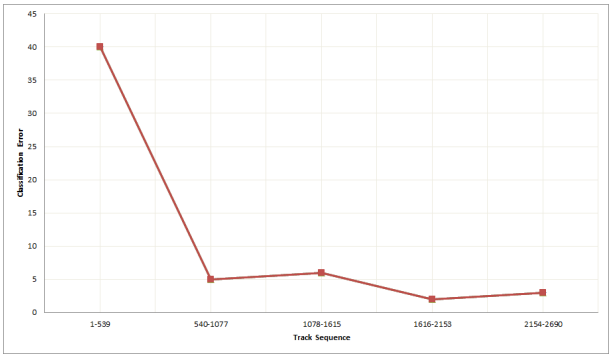

Figure 3: On-line learning table.

5 Conclusion

Soar-RL/LLA is a continual learning method for CID; CID applications can train an ML/AI assistant in a tactical environment and must continue to learn and adapt to sailors' feedback or new data. Figure 3 evidently shows that Soar-RL learns (and the error rate decreases) as the scenario progresses from the beginning set of track points (1-539) to the last set of track points (2154-2690). The Soar-RL ensures the following aspects of continual learning requirements:

- Bounded system size – the Soar-RL model's capacity is fixed as reflected in the number of preferences is fixed, the system uses its capacity intelligently and converges to ensure maximizing future reward;
- No direct access to previous experience – while the model can remember a limited amount of experience, Soar-RL does not have direct access to past tasks or be able to rewind the environment;

These conditions of Soar-RL are also studied in the theory and practice of the alpha decay and convergence of Soar-RL [6]. In summary, Soar-RL provides a continual and real-time machine learning for a potential CTAP and CID application. Soar-RL provides a real-time, scalable learning mechanism that enables systems of systems to gradually improve performance and adapt to new environments.

6 Acknowledgements

Thanks to the Naval Research Program at the Naval Postgraduate School for the support of the research project. Special thanks to LCDR Khoa H. Nguyen and Mr. William A.Treadway at OPNAV for the support of the research. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied of the U.S. Government.

References

[1] Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; and Witten, I. H. 2009. The WEKA Data Mining Software: An Update. SIGKDD Explorations 11(1):10–18.

[2] Jones, R. M.; Laird, J. E.; Nielsen, P. E.; Coulter, K. J.; Kenny, P.; and Koss, F. V. 1999. Automated intelligent pilots for combat flight simulation. AI Magazine 20(1):27.

[3] Laird, J. E.; Derbinsky, N.; and Tinkerhess, M. 2012. Online determination of value-function structure and action-value estimates for reinforcement learning in a cognitive architecture. Advances in Cognitive Systems 2:221–238.

[4] Laird, J. E. 2012. The Soar Cognitive Architecture. MIT Press.

[5] Scruggs, V. A. 2009. Combat identification training in the navy. Technical Report D0020254.A3, Center for Naval Analyses.

[6] Sutton, R. S. and Barto, A. G. 1998, 2014. Reinforcement Learning: An Introduction. MIT Press.

[7] Zhao, Y.; Mooren, E.; and Derbinsky, N. 2017. Reinforcement learning for modeling large-scale cognitive reasoning. In Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, 233–238.

[8] Soar Manual 2018. https://soar.eecs.umich.edu/downloads/Documentation/SoarManual.pdf: Page 135

[9] Zhao, Y.; Gallup, S. P.; and Mackinnon, D. J. 2011. System self-awareness and related methods for improving the use and understanding of data within DoD. Software Quality Professional, 13(4), 19–31. Retrieved from https://core.ac.uk/download/pdf/36720183.pdf

[10] Metron. 2018. Naval Simulation System (NSS).http://www.metsci.com.